Degradation Models and Optimizations for CMOS Circuits

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Dipl. Inform. Victor M. van Santen
geb. in Houten (Niederlande)

Tag der mündlichen Prüfung:	21.06.2022
Hauptreferent:	Prof. Dr.-Ing. Jörg Henkel Karlsruher Institut für Technologie (KIT)
Korreferent:	J.-Prof. Dr.-Ing. Hussam Amrouch Universität Stuttgart
Korreferent:	Prof. Dr.-Ing. Ulf Schlichtmann Technische Universität München (TUM)

Kurzfassung

Die Gewährleistung der Zuverlässigkeit von CMOS-Schaltungen ist derzeit eines der größten Herausforderungen beim Chip- und Schaltungsentwurf. Mit dem Ende der Dennard-Skalierung erhöht jede neue Generation der Halbleitertechnologie die elektrischen Felder innerhalb der Transistoren. Dieses stärkere elektrische Feld stimuliert die Degradationsphänomene (Alterung der Transistoren, Selbsterhitzung, Rauschen, usw.), was zu einer immer stärkeren Degradation (Verschlechterung) der Transistoren führt. Daher erleiden die Transistoren in jeder neuen Technologiegeneration immer stärkere Verschlechterungen ihrer elektrischen Parameter. Um die Funktionalität und Zuverlässigkeit der Schaltung zu wahren, wird es daher unerlässlich, die Auswirkungen der geschwächten Transistoren auf die Schaltung präzise zu bestimmen.

Die beiden wichtigsten Auswirkungen der Verschlechterungen sind ein verlangsamtes Schalten, sowie eine erhöhte Leistungsaufnahme der Schaltung. Bleiben diese Auswirkungen unberücksichtigt, kann die verlangsamte Schaltgeschwindigkeit zu Timing-Verletzungen führen (d.h. die Schaltung kann die Berechnung nicht rechtzeitig vor Beginn der nächsten Operation abschließen) und die Funktionalität der Schaltung beeinträchtigen (fehlerhafte Ausgabe, verfälschte Daten, usw.). Um diesen Verschlechterungen der Transistorparameter im Laufe der Zeit Rechnung zu tragen, werden Sicherheitstoleranzen eingeführt. So wird beispielsweise die Taktperiode der Schaltung künstlich verlängert, um ein langsameres Schaltverhalten zu tolerieren und somit Fehler zu vermeiden. Dies geht jedoch auf Kosten der Performanz, da eine längere Taktperiode eine niedrigere Taktfrequenz bedeutet. Die Ermittlung der richtigen Sicherheitstoleranz ist entscheidend. Wird die Sicherheitstoleranz zu klein bestimmt, führt dies in der Schaltung zu Fehlern, eine zu große Toleranz führt zu unnötigen Performanzseinbußen.

Derzeit verlässt sich die Industrie bei der Zuverlässigkeitsbestimmung auf den schlimmstmöglichen Fall (maximal gealterter Schaltkreis, maximale Betriebstemperatur bei minimaler Spannung, ungünstigste Fertigung, etc.). Diese Annahme des schlimmsten Falls garantiert, dass der Chip (oder integrierte Schaltung) unter allen auftretenden Betriebsbedingungen funktionsfähig bleibt. Darüber hinaus ermöglicht die Betrachtung des schlimmsten Falles viele Vereinfachungen. Zum Beispiel muss die eigentliche Betriebstemperatur nicht bestimmt werden, sondern es kann einfach die schlimmstmögliche (sehr hohe) Betriebstemperatur angenommen werden.

Leider lässt sich diese etablierte Praxis der Berücksichtigung des schlimmsten Falls (experimentell oder simulationsbasiert) nicht mehr aufrechterhalten. Diese Berücksichtigung bedingt solch harsche Betriebsbedingungen (maximale Temperatur, etc.) und Anforderungen (z.B. 25 Jahre Betrieb), dass die Transistoren unter den immer stärkeren elektrischen Felder enorme Verschlechterungen erleiden. Denn durch die Kombination an hoher Temperatur, Spannung und den steigenden elektrischen Feldern bei jeder Generation, nehmen die Degradationphänomene stetig zu. Das bedeutet, dass die unter dem schlimmsten Fall bestimmte Sicherheitstoleranz enorm pessimistisch ist und somit deutlich zu hoch ausfällt. Dieses Maß an Pessimismus führt zu erheblichen Performanzseinbußen, die unnötig und demnach vermeidbar sind. Während beispielsweise militärische Schaltungen 25 Jahre lang unter harschen Bedingungen arbeiten müssen, wird Unterhaltungselektronik bei niedrigeren Temperaturen betrieben und muss ihre Funktionalität nur für die Dauer der zweijährigen Garantie aufrechterhalten. Für letzteres können die Sicherheitstoleranzen also deutlich kleiner ausfallen, um die Performanz deutlich zu erhöhen, die zuvor im Namen der Zuverlässigkeit aufgegeben wurde.

Diese Arbeit zielt darauf ab, maßgeschneiderte Sicherheitstoleranzen für die einzelnen Anwendungsszenarien einer Schaltung bereitzustellen. Für fordernde Umgebungen wie Weltraumanwendungen (wo eine Reparatur unmöglich ist) ist weiterhin der schlimmstmögliche Fall relevant. In den meisten Anwendungen, herrschen weniger harsche Betriebssbedingungen (z.B. sorgen Kühlsysteme für niedrigere Temperaturen). Hier können Sicherheitstoleranzen maßgeschneidert und anwendungsspezifisch bestimmt werden, sodass Verschlechterungen exakt toleriert werden können und somit die Zuverlässigkeit zu minimalen Kosten (Performanz, etc.) gewahrt wird.

Leider sind die derzeitigen Standardentwurfswerkzeuge für diese anwendungsspezifische Bestimmung der Sicherheitstoleranz nicht gut gerüstet. Diese Arbeit zielt darauf ab, Standardentwurfswerkzeuge in die Lage zu versetzen, diesen Bedarf an Zuverlässigkeitsbestimmungen für beliebige Schaltungen unter beliebigen Betriebsbedingungen zu erfüllen. Zu diesem Zweck stellen wir unsere Forschungsbeiträge als vier Schritte auf dem Weg zu anwendungsspezifischen Sicherheitstoleranzen vor:

Schritt 1 verbessert die Modellierung der Degradationsphänomene (Transistor-Alterung, -Selbsterhitzung, -Rauschen, etc.). Das Ziel von Schritt 1 ist es, ein umfassendes, einheitliches Modell für die Degradationsphänomene zu erstellen. Durch die Verwendung von materialwissenschaftlichen Defektmodellierungen werden die zugrundeliegenden physikalischen Prozesse der Degradationsphänomena modelliert, um ihre Wechselwirkungen zu berücksichtigen (z.B. Phänomen A kann Phänomen B beschleunigen) und ein einheitliches Modell für die simultane Modellierung verschiedener Phänomene zu erzeugen. Weiterhin werden die jüngst entdeckten Phänomene ebenfalls modelliert und berücksichtigt. In Summe, erlaubt dies eine genaue Degradationsmodellierung von Transistoren unter gleichzeitiger Berücksichtigung aller essenziellen Phänomene.

Schritt 2 beschleunigt diese Degradationsmodelle von mehreren Minuten pro Transistor (Modelle der Physiker zielen auf Genauigkeit statt Performanz) auf wenige Millisekunden pro Transistor. Die Forschungsbeiträge dieser Dissertation beschleunigen die Modelle um ein Vielfaches, indem sie zuerst die Berechnungen so weit wie möglich vereinfachen (z.B. sind nur die Spitzenwerte der Degradation erforderlich und nicht alle Werte über einem zeitlichen Verlauf) und anschließend die Parallelität heutiger Computerhardware nutzen. Beide Ansätze erhöhen die Auswertungsgeschwindigkeit, ohne die Genauigkeit der Berechnung zu beeinflussen.

In Schritt 3 werden diese beschleunigte Degradationsmodelle in die Standardwerkzeuge integriert. Die Standardwerkzeuge berücksichtigen derzeit nur die bestmöglichen, typischen und schlechtestmöglichen Standardzellen (digital) oder Transistoren (analog). Diese drei Typen von Zellen/Transistoren werden von der Foundry (Halbleiterhersteller) aufwendig experimentell bestimmt. Da nur diese drei Typen bestimmt werden, nehmen die Werkzeuge keine Zuverlässigkeitsbestimmung für eine spezifische Anwendung (Temperatur, Spannung, Aktivität) vor. Simulationen mit Degradationsmodellen ermöglichen eine Bestimmung für spezifische Anwendungen, jedoch muss diese Fähigkeit erst integriert werden. Diese Integration ist eines der Beiträge dieser Dissertation.

Schritt 4 beschleunigt die Standardwerkzeuge. Digitale Schaltungsentwürfe, die nicht auf Standardzellen basieren, sowie komplexe analoge Schaltungen können derzeit nicht mit analogen Schaltungssimulatoren ausgewertet werden. Ihre Performanz reicht für solch umfangreiche Simulationen nicht aus. Diese Dissertation stellt Techniken vor, um diese Werkzeuge zu beschleunigen und somit diese umfangreichen Schaltungen simulieren zu können.

Diese Forschungsbeiträge, die sich jeweils über mehrere Veröffentlichungen erstrecken, ermöglichen es Standardwerkzeugen, die Sicherheitstoleranz für kundenspezifische Anwendungsszenarien zu bestimmen. Für eine gegebene Schaltungslebensdauer, Temperatur, Spannung und Aktivität (Schaltverhalten durch Software-Applikationen) können die Auswirkungen der Transistordegradation ausgewertet werden und somit die erforderliche (weder unter- noch überschätzte) Sicherheitstoleranz bestimmt werden. Diese anwendungsspezifische Sicherheitstoleranz, garantiert die Zuverlässigkeit und Funktionalität der Schaltung für genau diese Anwendung bei minimalen Performanzeinbußen.

Abstract

Ensuring reliability in CMOS circuits is currently one of the key challenges in chip design. With the discontinuation of Dennard scaling, each new generation of semiconductor technology increases the electric fields of transistors. This stronger electric field stimulates the degradation phenomena (circuit aging, self-heating, etc.) by accelerating their physical processes. This, in turn, causes ever-increasing degradations to the transistors. Therefore, in each new technology generation the transistors feature higher shifts (degradations) to their nominal electrical parameters. Thus, the impact of this increasing transistor degradation on the circuit must be determined.

The two key impacts are elongated propagation delays (i.e., the circuit operates slower) and increased power consumption. If unaccounted for, the elongated delay can lead to timing violations (i.e., the circuit cannot finish the calculation in time before the start of the next operation) and hamper the functionality of the circuit (erroneous output, corrupted data, etc.). To account for these shifts in transistor parameters over time, guardbands are introduced. For example, increasing the clock period of the circuit ensures elongated delays can be tolerated. Finding the correct guardband is crucial. Underestimating the guardband results in errors (such as timing violations), while overestimating it results in unnecessary performance losses.

Currently, industry relies on worst-case conditions when estimating reliability, i.e. industry employs pessimistic overestimations of the guardband (and thus induced degradations) to guarantee the functionality of their products. It is imperative for them, to ensure that under any condition – including unfavorable high temperatures and elevated voltages – their product (e.g., chip or circuit) maintains reliability and functionality. Additionally, considering the worst-case allows for many simplifications. For example, instead of carefully understanding and modeling each phenomenon, industry can just take an unlucky sample (bad sample with lots of manufacturing variability) and experimentally determine its delay/power under the toughest conditions (extreme temperature and voltage) to ensure that any degradation experienced by the end-user must be less than this worst-case experiment.

Unfortunately, this practice of worst-case estimations (experimental or simulation-based) can no longer be sustained. With continuous miniaturization due to geometry scaling, the rising electric fields stimulates degradation phenomena and manufacturing becomes harder (see introduction of EUV-lithography). Therefore, the worst case is a tremendous amount of detrimental manufacturing variability in addition to considerable degradations due to stimulated degradation phenomena. This level of pessimism leads to substantial performance decreases, which are not necessary for each customer. For example, while military or space circuits might operate for 25 years under harsh conditions, consumer electronics (e.g., smartphones, video-streaming devices) operate at lower temperatures and only have to maintain their functionality across their 2 year warranty period.

This work aims to provide custom reliability estimations for the individual use-case scenarios of the circuit. For challenging environments like the space applications (where repair is impossible), worst-case estimations remain relevant. However, typical applications do not operate under such harsh conditions for such a long time. Consumer electronics include cooling systems (e.g., the fan of a notebook/server) or other systems (e.g., thermal throttling, power scaling) to guarantee less harsh environmental conditions (e.g., lower peak temperature). In such cases, custom reliability estimations can regain performance lost to otherwise overly pessimistic guardbands.

Unfortunately, current standard design tools are not well-equipped to deal with these custom reliability estimations. The current procedures rely on the information from the foundry (the semiconductor manufacturer) about their worst-case transistors and standard cells (logic gates, arithmetic units, etc.). The tools then integrate the worst case into the design steps (e.g., synthesis, static timing analysis). There is merely rudimentary support in individual tools, but contrary to worst-case estimations (via process corners), there is no interconnected reliability tool flow.

This work aims to enable standard design tools to deal with this demand for reliability estimations of any circuit under any condition. Instead of developing our own solution, this work enhances standard design tools to leverage their maturity. We extend their functionality to enable the tools to consider custom reliability estimations. For this purpose, we present four steps as contributions towards custom reliability estimations:

1. Improve degradation models to enable accurate transistor degradation modeling.
    - Unify the major degradation phenomena into a joint physical degradation model, which considers their interactions and dependencies on the operating conditions (voltage, temperature, activity). This model estimates the degradation based on the individual conditions of the use-case scenario.
    - Consider recently uncovered phenomena (self-heating in FinFET transistors) to enable tools to estimate reliability in current state-of-the-art FinFET transistors.
2. Accelerate the models to provide large-scale modeling capabilities. High-performance modeling is required for integration of degradation models into large-scale digital reliability estimations.
    - Speed up modeling by providing upper bounds for a given operating condition.
    - Parallelize the degradation models to leverage the parallel computing hardware for large-scale transistor reliability estimations.
3. Incorporate custom reliability modeling into standard tools
    - Estimate aging in standard cells to provide custom reliability estimation for large-scale digital circuits.
    - Integrate reliability into analogue/mixed-signal simulator tool chain for custom reliability estimations of analogue/mixed-signal circuits.
4. Accelerate the standard tools
    - Massively parallel implementation of circuit simulators to enable reliability estimations of large-scale circuits (non-standard-cell digital designs, large analogue designs).

Step 1 enhances the degradation modeling. The goal of step 1 is to provide a single unified model for the degradation effects (combining the work of the material scientists) and consider their interactions (phenomena A might accelerate phenomena B). Combined with newly discovered key phenomena, this enables accurate degradation modeling of transistors according to multiple phenomena simultaneously.

Step 2 accelerates these degradation models, as the original implementation of physicists require minutes to model a single transistor. This work accelerates these models *without any loss in accuracy* by levering the parallelism found in the computation hardware of today and by simplifying calculation where possible (e.g., guardbands require only peak degradation levels and not degradation over time).

Step 3 then incorporates these degradation models into the standard tools. Standard tools are currently only aware of the best, typical and worst possible standard cells (digital) or transistors (analogue) with all three process corner provided by foundries. These tools do not estimate for a given condition (temperature, voltage, activity) and hence integration work is necessary.

Step 4 accelerates the standard tools to provide insights in the current large-scale circuits. Non-standard cell digital designs and large analogue designs are currently not supported by existing tools, as the existing analogue reliability estimation tools are not compatible with large-scale analogue simulators (e.g., FastSPICE). This work provides high-performance implementations of these analogue circuit simulators.

These contributions (each spanning multiple publications) enable standard tools to estimate reliability for custom use-case scenarios. For a given circuit lifetime, temperature (peak temperature or temperature over time), voltage (peak voltage or voltage over time) and activity (workload, applications) this enhanced standard tool flow can estimate the impact of these degradations and thus provide the required (neither under- nor overestimated) guardband.

Acknowlegdment

First and foremost, I want to thank my mentor J.-Prof. Dr.-Ing. Hussam Amrouch. His support over the years was invaluable and without it, I would not have made it. During various hardships, he was there both with advice and actions to overcome those situations. Scientifically, he developed my skills during countless discussions on ideas, writing and implementations. Together we supervised many master and bachelor theses, which helped me to learn supervise people in such a way, that they grow and flourish. Over our many years of working together, we shared business trips, various tough situations and also good experiences, which ultimately lead to a strong bond and great friendship.

I also want to thank Prof. Dr.-Ing. Jörg Henkel for his guidance, especially in the early years of my PhD. I was given freedom to work on my own topics, so that I could explore and develop my own ideas. He frequently challenged me and my ideas, to hone my scientific understanding and paper writing. Various other obstacles, such as the impossible deadlines, made me stronger and grow as a person. Additionally, I want to thank Prof. Dr.-Ing. Ulf Schlichtmann. In various discussions (e.g., during meetings at conferences), he gave feedback to my work and gave advice on how to succeed as a scientist. Lastly, I want to thank him for being a Korreferent of my PhD defense.

My colleagues at the chair of embedded systems at KIT also helped me many times during presentation rehearsals, technical problems and travel issues. Especially Sami Salamin, Sajjad Hussain and Volker Wenzel, who all shared an office with me, had a great impact on my PhD. Lengthy discussions helped me to develop as a scientist, guide me in student supervision and were just plain fun. It was a joy to share the office with you across these years. Lastly, I want to thank Martin Buchty, our technician to solve countless technical issues and always try his best to solve my own and my student's PC, server and license issues. You were a great help and pleasant person to deal with.

In the last years of my PhD, I moved to the university of Stuttgart to the newly formed research group of Prof. Amrouch. My colleagues Paul Genssler, Florian Klemme, Tim Bücher and Simon Thomann moved with me from the KIT to Stuttgart. I want to thank all of them for their help in implementations, scientific ideas, constructive feedback and just being all-around great guys to work with, both for the time in KIT and at the university of Stuttgart. It was joy to share that time with you and your scientific contributions and implementations helped me greatly in my research.

Additionally, my PhD would not be as successful without the work and input by my colleagues across various collaborations. Initially, this work was co-developed with the Team of Prof. Montserrat Nafria Maqueda from Universitat Autonoma de Barcelona (UAB). Javier Martin-Martinez and Montserrat Nafria tought me the physical understanding of Bias Temperature Instability modeling and transistor characterization (wafer measurements). Later, the collaboration with Prof. Souvik Mahapatra from Indian Institute of Technology Bombay (IITB) and his team, Narendra Parihar, Nilesh Goel, Uma Sharma, Subrat Mishra, Chaitanya Pasupuleti deepened my Bias Temperature Instability modeling knowledge, especially with respect of defect modeling and defect properties. The collaboration with Sevilla and the teams of Prof. Francisco V. Fernandez from IMSE-CNM (University of Seville and CSIC) and Prof. Elisendra Roca, i.e. Rafael Castro-Lopez and Juan Nunez helped with silicon measurements of reliability mechanisms and further deepened my understand of reliability physics and the challenges of circuit measurements and measurement equipment. Prof. Yogesh S. Chauhan from Indian Institute of Technology Kanpur (IITK) and his team taught me how transistor modelcards (parameter lists for transistor models) are created and how transistor models are calibrated. Prof. Preeti Ranjan Panda from Indian Institute of Technology Delhi (IITD) and Divya Praneetha Ravipati collaborated on cache modeling and emerging technologies in multi-core compute systems.

To the diploma, bachelor and master thesis students who I supervised during my time at KIT and Stuttgart, I want to express my gratitude. Your implementations were the base of my work and you frequently put your heart and soul in your thesis. I am grateful that you chose me as your supervisor and all of you were really friendly and pleasant to work with. I am proud of your achievements and happy that each thesis ended with both your growth as a scientist and engineer as well as my growth as a supervisor and educator.

In terms of funding, this was in part supported by SPP1500, the priority project (“Schwerpunktprojekt”) from the national german research foundation (Deutsche Forschungs Gesellschaft DFG) “Dependable Embedded Systems“. The project within SPP 1500 to which this work is linked, is “VirTherm-3D” and is collaboration with Technical University Munich (TUM).

To all my friends who have been there for me over the years. Thank you for being awesome, listening to my many stories about work and giving me great advice. When I came back from my frequent business trips, you always made Karlsruhe a true home for me and I fondly remember the time with all of you during my PhD.

Lastly, I want to thank my family and especially my parents Maarten and Yvonne for their unwavering support. You taught me everything in life and I am eternally grateful that I could always count on you. Any time, any place and for any purpose you were there for me. I would not be the person today without your guidance, support and simply your many acts of kindness and encouragement during my upbringing, education and PhD. Thank you for being awesome parents and your undying support.

Publications

This work is based on the following publications:

First Author Journals

1. Victor M. van Santen, Fu Florian Diep, Jörg Henkel and Hussam Amrouch
   Massively Parallel Circuit Setup in GPU-SPICE
   in IEEE Transactions on Computers (TC), 2020.
2. Victor M. van Santen, Hussam Amrouch, Poja Sharma and Jörg Henkel
   On the Workload Dependence of Self-Heating in FinFET Circuits
   in IEEE Transactions on Circuits and Systems II (TCAS-II), 2019.
3. Victor M. van Santen, Hussam Amrouch and Jörg Henkel
   Modeling and Mitigating Time-Dependent Variability from the Physical Level to the Circuit Level
   in IEEE Transactions on Circuits and Systems I: Regular Papers (TCAS-I), 2019.
4. Victor M. van Santen, Hussam Amrouch and Jörg Henkel
   New Worst-Case Timing for Standard Cells under Aging Effects
   in IEEE Transactions on Device and Materials Reliability (TDMR), 2019.
5. Victor M. van Santen, Hussam Amrouch and Jörg Henkel
   Modeling and Evaluating the Gate Length Dependence of BTI
   in IEEE Transactions on Circuits and Systems II (TCAS-II), 2018.
6. Victor M. van Santen, Javier Martin-Martinez, Hussam Amrouch, Montserrat Nafria and Jörg Henkel
   Reliability in Super- and Near-Threshold Computing: A Unified Model of RTN, BTI and PV
   in IEEE Transactions on Circuits and Systems I (TCAS-I), 2018.

Other Journals

7. Divya Praneetha Ravipati, Rajesh Kedia, Victor M. van Santen, Jörg Henkel, Preeti Ranjan Panda and Hussam Amrouch
   FN-CACTI: Advanced CACTI for FinFET and NC-FinFET Technologies
   in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2021

8. Paul R. Genssler, Victor M. van Santen, Jörg Henkel and Hussam Amrouch
   On the Reliability of FeFET On-Chip Memory
   in IEEE Transactions on Computers (TC), 2021.

9. Sami Salamin, Victor M. van Santen, Martin Rapp, Jörg Henkel and Hussam Amrouch
   Minimizing Excess Timing Guard Banding Under Transistor Self-Heating Through Biasing at Zero-Temperature Coefficient
   in IEEE Access, Feb 2021.

10. Sami Salamin, Victor M. van Santen, Hussam Amrouch, Narendra Parihar, Souvik Mahapatra and Jörg Henkel
    Modeling the Interdependencies between Voltage Fluctuation and BTI Aging
    in IEEE Transactions on Very Large Scale Integration (TVLSI) Systems, 27(7), 1652-1665, 2019

11. Hussam Amrouch, Victor M. van Santen and Jörg Henkel
    Interdependencies of Degradation Effects and their Impact on Computing
    in IEEE Design & Test (D&T), 2017.

First Author Conferences

1. Victor M. van Santen, Simon Thomann, Yogesh S. Chauhan, Jörg Henkel and Hussam Amrouch
   Reliability-Driven Voltage Optimization for NCFET-based SRAM Memory Banks
   in Proceedings of IEEE VLSI Test Symposium (VTS'21), 2021.

2. Victor M. van Santen, Linda Schillinger and Hussam Amrouch
   Impact of Transistor Self-Heating on Logic Gates (Special Session)
   In IEEE International Symposium on VLSI Design, Automation and Test (VLSI-DAT'21), Virtual, 2021.

3. Victor M. van Santen, Simon Thomann, Chaitanya Pasupuleti, Paul R. Genssler, Narendra Gangwar, Uma Sharma, Jörg Henkel, Souvik Mahapatra and Hussam Amrouch
   BTI and HCI Degradation in a Complete 32X64 bit SRAM Array including Sense Amplifiers and Write Drivers under Processor Activity
   in Proceedings of the IEEE 58th International Reliability Physics Symposium (IRPS'20), 2020.

4. Victor M. van Santen, Paul R. Genssler, Om Prakash, Simon Thomann, Jörg Henkel and Hussam Amrouch
   Impact of Self-Heating on Performance, Power and Reliability in FinFET Technology (Special Session)
   in 25th Asia and South Pacific Design Automation Conference, ASP-DAC, 2020.

5. Victor M. van Santen, Hussam Amrouch and Jörg Henkel
   Reliability Estimations of Large Circuits in Massively-Parallel GPU-SPICE (Special Session)
   in 24th IEEE International Symposium on On-Line Testing and Robust System Design (IOLTS), 2018.

6. Victor M. van Santen, Javier Diaz-Fortuny, Hussam Amrouch, Javier Martin-Martinez, Rosana Rodriguez, Rafael Castro-Lopez, Elisenda Roca, Francisco V. Fernandez, Jörg Henkel and Montserrat Nafria
   Weighted Time Lag Plot Defect Parameter Extraction and GPU-based BTI Modeling for BTI Variability
   in IEEE 55th International Reliability Physics Symposium (IRPS), 2018.

7. Victor M. van Santen, Hussam Amrouch, Javier Martin-Martinez, Montserrat Nafria and Jörg Henkel
   Designing Guardbands for Instantaneous Aging Effects
   in ACM/EDAC/IEEE 53rd Design Automation Conference (DAC), 2016.

8. Victor M. van Santen, Hussam Amrouch, Narendra Parihar, Souvik Mahapatra and Jörg Henkel
   Aging-Aware Voltage Scaling
   in IEEE/ACM 19th Design, Automation and Test in Europe Conference (DATE'16), 2016.

Other Conferences

9. Hussam Amrouch, Animesh Basak Chowdhury, Wentian Jin, Ramesh Karri, Khorrami Farshad, Prashanth Krishnamurthy, Ilia Polian, Victor M. van Santen, Benjamin Tan and Sheldon Tan.
Machine Learning for Semiconductor Test and Reliability (Special Session)
In IEEE VLSI Test Symposium (VTS'21), Virtual, 2021.

10. Florian Klemme, Jannik Prinz, Victor M. van Santen, Jörg Henkel and Hussam Amrouch
Modeling Emerging Technologies using Machine Learning: Challenges and Opportunities
in IEEE/ACM 39th International Conference On Computer Aided Design (ICCAD), 2020.

11. Hussam Amrouch, Victor M. van Santen, Girish Pahwa, Yogesh Chauhan and Jörg Henkel
NCFET to Rescue Technology Scaling: Opportunities and Challenges (Special Session)
in 25th Asia and South Pacific Design Automation Conference (ASP-DAC), 2020.

12. Hussam Amrouch, Victor M. van Santen, Om Prakash, Hammam Kattan, Sami Salamin, Simon Thomann, and Jörg Henkel
Reliability Challenges with Self-Heating in FinFET Technology (Special Session)
in IEEE 25th International Symposium on On-Line Testing And Robust System Design (IOLTS'19), 2019.

13. Hussam Amrouch, Victor M. van Santen and Jörg Henkel
Estimating and Optimizing BTI Aging Effects: From Physics to CAD (Special Session)
in IEEE/ACM 37th International Conference on Computer-Aided Design (ICCAD), 2018.

14. Hussam Amrouch, Subrat Mishra, Victor M. van Santen, Souvik Mahapatra and Jörg Henkel
Impact of BTI on Dynamic and Static Power: From the Physical to Circuit Level
in IEEE 55th International Reliability Physics Symposium (IRPS), 2017.

15. Hussam Amrouch, Javier Martin-Martinez, Victor M. van Santen, Miquel Moras, Rosana Rodriguez, Montserrat Nafria and Jörg Henkel
Connecting the Physical and Application Level Towards Grasping Aging Effects
in IEEE 53rd International Reliability Physics Symposium (IRPS), 2015.

16. Hussam Amrouch, Victor M. van Santen, Thomas Ebi, Volker Wenzel and Jörg Henkel
Towards Interdependencies of Aging Mechanisms
in IEEE/ACM 33rd International Conference on Computer-Aided Design (ICCAD), 2014.

Book Chapters

1. Victor M van Santen, Florian Klemme, Hussam Amrouch
   The Vital Role of Machine Learning in Developing Emerging Technologies
   Chapter in "Recent Findings in Boolean Techniques: Selected Papers from the 14th International Workshop on Boolean Problems", Springer Nature, 2021

2. Victor M. van Santen, Hussam Amrouch, Thomas Wild, Jörg Henkel and Andreas Herkersdorf
   Thermal Management and Communication Virtualization for Reliability Optimization in MPSoCs
   Chapter in "Dependable Embedded Systems", Springer International Publishing, 2020.

Workshops

1. Victor M. van Santen, Hussam Amrouch and Jörg Henkel
   Modeling Short and Long-term Effects of Aging from the Defect to Application Level (Invited presentation)
   in Workshop on System-to-Silicon Performance Modeling and Analysis
   at the ACM/EDAC/IEEE 53rd Design Automation Conference (DAC), 2016.

Collaborations

The following long-term collaborations supported this work with technology data, semiconductor measurements and degradation models:

1. Prof. Montserrat Nafria Maqueda from Universitat Autonoma de Barcelona (UAB)
   Professor of electrical engineering with focus on "Engineering of electron devices for the Internet of Things"

2. Prof. Souvik Mahapatra from Indian Institute of Technology Bombay (IITB)
   Professor of electrical engineering with focus on "CMOS transistor gate stack reliability"

Other collaborations in the scope of this work:

3. Prof. Francisco V. Fernandez from IMSE-CNM (University of Seville and CSIC)
   Professor of electrical engineering

4. Prof. Yogesh S. Chauhan from Indian Institute of Technology Kanpur (IITK)
   Professor of electrical engineering

5. Prof. Preeti Ranjan Panda from Indian Institute of Technology Delhi (IITD)
   Professor of computer science and engineering

Student Theses

The following student theses contributed to the implementation and prototypes of this work:

1. The Impact of Self-Heating on Circuits
   Linda Schillinger, Master Thesis, December 2020

2. Improved Aging Modeling for Circuit Reliability Evaluations
   Leon Felix List, Bachelor Thesis, November 2020

3. Inferring Transistor Compact Models with Neural Networks
   Jannik Prinz, Master Thesis, May 2020

4. Just-In-Time Compilation for the Circuit Simulator GPU-SPICE on the CUDA platform
   Albert Walner, Master Thesis, September 2019

5. Reliability Analysis of SRAM Circuits in Conventional and Emerging Technologies
   Simon Thomann, Bachelor Thesis, September 2019

6. Modeling Self-Heating Dependencies in FinFET Transistors
   Michael Meinschäfer, Master Thesis, May 2019

7. Modeling Self-Heating Effects in FinFET Technology
   Florian Banscher, Bachelor Thesis, February 2019

8. High Performance Circuit Simulations
   Islam Hamada, Bachelor Thesis, August 2018

9. Reliability Analysis of Circuits under Variability Effects
   Florian Klemme, Master Thesis, May 2018

10. High Performance Reliability Estimation of Circuits with CUDA
    Fu Lam Florian Diep, Diploma Thesis, September 2017

11. A Novel Approach to Standard Cell Simulation in SPICE using Equivalence Classes
    Sebastian Roether, Diploma Thesis, September 2017

12. Fast, yet Accurate BTI Variability Evaluation at the Physical and Device Levels
    Sven Brinkmann, Diploma Thesis, July 2017

13. Measuring Aging-induced Degradation in Microprocessors
    Tim Bücher, Bachelor Thesis, May 2017

14. GPU-based Implementation for State-of-the-Art MOSFET Compact Modelling
    Dimitar Milev, Master Thesis, Mai 2017

15. Graphical User Interface for Reliability Estimation Tool
    Daniela Jacinta Rostek, Student Research Project, September 2016

16. Fast, yet Accurate Implementation for Advanced Reliability Modeling
    Max Camillo Eisele, Bachelor Thesis, August 2016

17. Evaluating the Mutual Influence between the Application and Physical Level on Reliability
    Michael Skinder, Master Thesis, March 2016

18. Evaluating the Impact of Applications on Simulating Aging Effects
    Christian List, Diploma Thesis, March 2016

Contents

Acronyms and symbols

Acronyms

AES	Advanced Encryption Standard
AMS	Analogue/Mixed-Signal
API	Application Programming Interface
BAT	BTI Analysis Tool (a BTI aging model based on RD)
BL	Bit line in a SRAM array
BTI	Bias Temperature Instability
BSIM	Berkeley Short-channel IGFET Model
BSIM-CMG	Berkeley Short-channel IGFET Model for Common Multi-Gate transistors (FinFET)
CES	Chair of Embedded Systems
CMOS	Complementary Metal-Oxide-Semiconductor
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture (API for general purpose computing on GPUs)
DCT	Discrete Cosine Transformation
DSP	Digital Signal Processor
EDA	Electronic Design Automation
EPFL	Ecole Polytechnique Federale de Lausanne
EM	Electromigration
FF	Fast-fast process corner (best possible transistors)
FinFET	Fin Field-Effect Transistor (a multi-gate non-planar MOSFET)
FPGA	Field-Programmable Gate Array
GND	Ground potential
GPU	Graphic Processing Unit
HCD	Hot-Carrier induced Degradation
HT	Hole Traps
ID	Identifier
IT	Interface Traps
KIT	Karlsruhe Institute for Technology

LU	Lower and Upper triangular matrix
MAC	Multiply and Accumulate operation
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
MOSRA	MOSFET Reliability Analysis (reliability software tool)
NCFET	Negative Capacitance Field-Effect Transistor
nMOS	n-type (negative-channel) MOSFET
NTC	Near-Threshold Computation (selecting V_{dd} near the technology's V_{th})
OT	Oxide Traps
PDK	Process Design Kit (CMOS technology information for circuit design)
PDO	Probabilistic Defect Occupancy model (a BTI aging model based on TD)
PDP	Power Delay Product
pMOS	p-type (positive-channel) MOSFET
PTM	Predictive Technology Model
PV	Process Variation (manufacturing variability)
RAT	Read Access Time
RD	Reaction-Diffusion theory (a BTI aging theory)
RDF	Random Dopant Fluctuation
RTN	Random Telegraph Noise
SA	Sense Amplifier
SILC	Stress-Induced Leakage Current
SNM	Static Noise Margin
SPICE	Simulation Program with Integrated Circuit Emphasis (a circuit simulator)
SQL	Standard Query Language
SRAM	Static Random-Access Memory
SS	Slow-slow process corner (worst possible transistors)
STA	Static timing analysis
STC	Super-Threshold Computation (selecting V_{dd} far above the technology's V_{th})
TCAD	Technology Computer-Aided Design
TD	Trapping-Detrapping theory (a BTI aging theory)
TDDB	Time-Dependent Dielectric Breakdown
TDP	Thermal Design Power
TSMC	Taiwan Semiconductor Manufacturing Company
TT	Typical-typical process corner (nominal transistors)

TTOM	Transient Trap Occupancy Model (part of BAT)
WD	Write Driver
WL	Word line in a SRAM array

Constants

k_B	Boltzmann constant $8.617 \times 10^{-5}\,\mathrm{eV\,K^{-1}}$

Latin symbols and variables

C_{gg}	Transistor gate capacitance
C_{th}	Transistor thermal capacitance
$D(\tau_e, \tau_c)$	Defect distribution
f_{clk}	Clock (operating) frequency of a circuit
f_{sw}	Switching (toggle) frequency of a transistor
g_m	Transistor transconductance
I_d	Transistor drain current (total current flowing through the drain)
I_{ds}	Transistor drain-to-source current
L	Length of a transistor (MOSFET and FinFET)
N_{IT}	Number of Interface Traps
N_{OT}	Number of Oxide Traps
N_{HT}	Number of Hole Traps
P_{occ}	Occupancy probability of a defect
$P_{capture}$	Carrier capture probability of a defect
$P_{emission}$	Carrier emission probability of a defect
R_{th}	Transistor thermal resistance
T_C	Transistor channel temperature
T_{chip}	Chip temperature
t_{delay}	Propagation delay of a circuit
t_{GB}	Reliability timing guardband
t_{rec}	Time a circuit is in recovery (i.e., no or lower voltage applied)
t_{stress}	Time a circuit is under stress (i.e., voltage applied or operating)
V_b	Transistor bulk voltage or transistor body voltage
V_d	Transistor drain voltage
V_{dd}	Positive circuit supply voltage

V$_g$	Transistor gate voltage
V$_{GB}$	Reliability voltage guardband
V$_s$	Transistor source voltage
V$_{ss}$	Negative circuit supply voltage (often identical to GND)
V$_{th}$	Transistor threshold voltage
W	Width of planar MOSFET transistor

Greek symbols and variables

β	Voltage acceleration factor
η	Impact of a occupied defect (in ΔV_{th})
λ	Duty cycle (transistor on-/off-ratio)
μ	Transistor carrier mobility
τ_c	Capture time of a carrier in a defect
τ_e	Emission time of a carrier in a defect

Operators and math symbols

$\vec{a}$	vector
$\overline{A}$	complementary circuit line (e.g., BL to $\overline{BL}$ in a SRAM array)

1 Challenges in Traditional Design for Reliability

Maintaining reliability in semiconductor circuits is a critical concern in current nano-scale CMOS circuits [1]. To explore why reliability became critical in recent years, this section provides an overview over the key issues with maintaining reliability and their origin.

Semiconductor manufacturers employ geometry scaling to improve the performance of transistors in each new generation. With smaller geometry, transistors become more efficient and faster. Since the capacitances (transistor gate capacitance, parasitic capacitances in the wires, etc.) scale with geometry, the circuit can switch faster (charging smaller capacitances with same driving current) and consume less power (current is flowing for less time, since capacitance is charged faster). At the same time, geometry scaling also allows for more transistors in the same (chip) area and thus enables more processing power (e.g., more processing cores or higher logic complexity).

For years, this geometry scaling governed the entire technology scaling, this means if the geometry was reduced by 30% then also all voltages were reduced by 30% (this is called a "scaling factor"). Scaling the voltage with the same scaling factor has two distinct reasons. First, reducing the supply voltage (V_{dd}) saves power per component and by increasing transistor density (since smaller geometry enables higher logic density), the power density can remain constant [2]. Secondly, scaling voltage with the same factor ensures that the electric field over the gate dielectric of a transistor remains the same, which is crucial to not put more (electric) stress on the materials within the dielectric (an insulator only insulates up to a given electric field, compare arcing of electricity in air). This scaling of geometry and voltage in tandem is called "Dennard Scaling" [2].

1.1 Discontinuation of Dennard Scaling

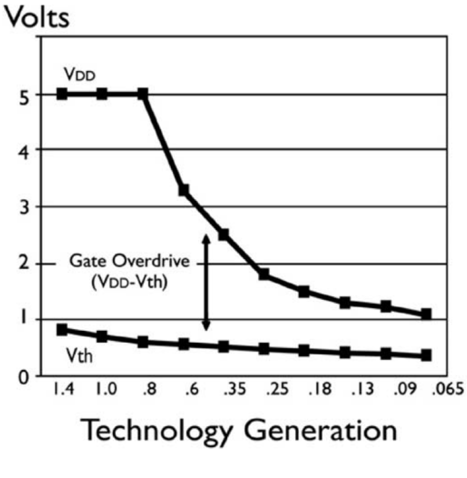


Figure 1.1: Supply voltage (V_{DD}) approaching threshold voltage (V_{th}) across older technology generations. Src: P. Paukan, Intel IEDM 2007

Unfortunately, continuing Dennard Scaling is impossible in recent years, since V_{dd} cannot be further reduced. Transistors feature a threshold voltage (V_{th}), i.e. the voltage value at which they switch from the OFF to the ON state (i.e., become conductive with the formation of channel). Should the V_{dd} drop below the V_{th}, the transistor cannot fully form a conductive channel and the transistor does not fully turn ON. Even when the V_{dd} is approaching

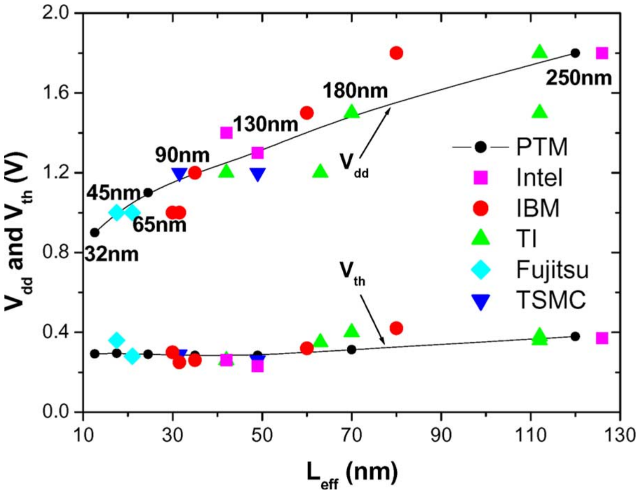


Figure 1.2: Supply voltage (V_{DD}) versus threshold voltage (V_{th}) for newer technology generations. Note, that supply voltage cannot be scaled much further, as it approaches the threshold voltage. Src: [3]

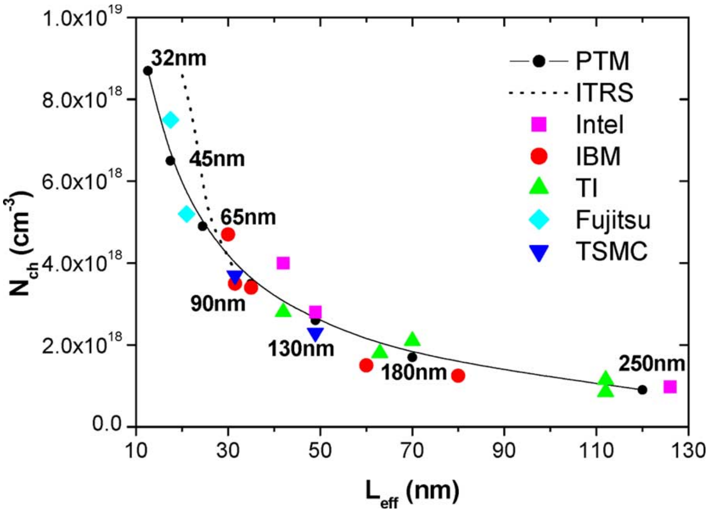


Figure 1.3: Threshold voltage (V_{th}) cannot be scaled, as channel doping concentrations (N_{ch}) cannot be further increased to prevent too high variability (e.g., RDF). Src: [3]

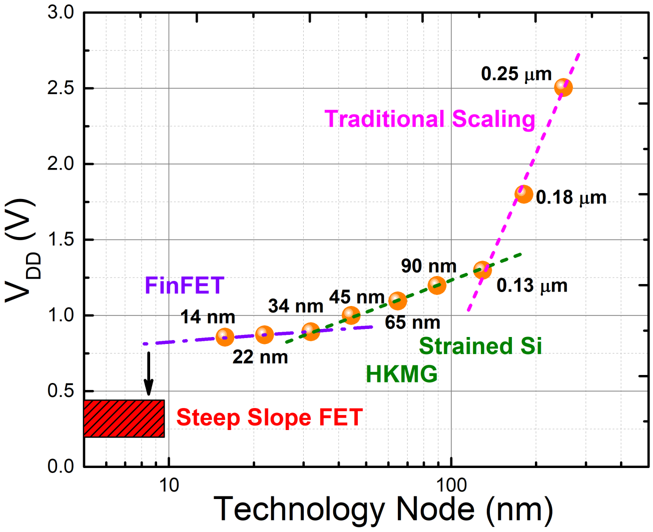


Figure 1.4: Discontinuation of Voltage Scaling towards newer technologies. Traditional scaling marked is ”Dennard Scaling“ and could not be sustained (the lines flatten). Src: [4]

V_{th}, the channel formation becomes ever weaker and thus decreases the strength (current flowing across the transistor) of the transistor.

Supply voltages (V_{dd}) already approached the threshold voltage (V_{th}) for older technologies with multiple Volts as shown in Fig. 1.1. However, for recent technologies the V_{dd} is still approaching V_{th} (see Fig. 1.2). V_{dd} cannot be reduced with same factor as the geometry, to ensure that the V_{dd} is always well above V_{th}, resulting in insufficient V_{dd}-scaling as observed in Fig. 1.4. The V_{th} itself cannot be reduced further, since the dopant concentrations within the transistor become too high (see Fig. 1.3) and then would introduce too much variability (e.g., Random Dopant Fluctuation (RDF)).

In summary, V_{dd} is approaching V_{th} narrowing the safety margins for degradations, which in turn limits V_{dd} scaling. Therefore, current generations barely scale V_{dd}.

1.2 Insufficient Voltage Scaling stimulates Degradation Phenomena

Since V_{dd} is now insufficiently reduced/scaled compared to geometry, the electric fields within the transistors increase. To illustrate with typical numbers in recent years, if geometry is scaled with a factor of 0.7 and voltage is scaled with a factor of 0.9, then the electric field increases as follows. Area of the gate of the transistor is 0.49 (both width and length are scaled by 0.7, i.e. $0.7^2 = 0.49$), yet the voltage is scaled with 0.9. This results in an increase of the electric field of $83\,\%$ ($\frac{0.9}{0.49} \approx 1.83$).

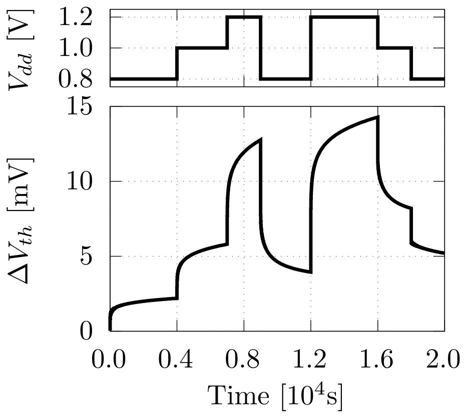


Figure 1.5: Aging-induced degradation (bottom) follows the voltage (top) as it is stimulated by it.

This elevated electric field imparts a higher (electric) stress on the transistor. Degradation phenomena like Bias Temperature Instability (BTI), Hot-Carrier induced Degradation (HCD) and Time Dependent Dielectric Breakdown (TDDB) all are stimulated by the electric field (see Fig. 1.5). A stronger electric field accelerates and stimulates the underlying physical degradation processes (e.g., defect formation and electrical activation within the gate dielectric) and thus leads to increased degradation (lower ON current of transistors, threshold voltage increase, etc.).

Therefore, with every technology scaling step, the degradation phenomena receive more stimulation and thus induce ever higher degradations. This explains, why reliability became critical in recent years forcing designers have to protect the circuit against ever-increasing degradations by degradation phenomena.

1.3 Design for Reliability

Design for reliability is the process to consider reliability in circuit design. The circuit is over-designed, so that it can tolerate degradation in its components (subcircuits, transistors, etc.) while still operating within its specification. Design for reliability can be performed with traditional worst-case estimations and the modern custom reliability estimations.

1.3.1 Traditional Worst-Case Reliability Estimations

Designing a reliable circuit mandates the consideration of non-ideal (e.g., manufacturing variability, aging) circuit components (transistors, subcircuit, etc.). As mentioned in the Section 1.1, due to the discontinuation of Dennard scaling, technologies feature ever-increasing degradations with each new generation.

Traditionally, circuit designers consider reliability by abstracting it to the worst-case scenario. This traditional design for reliability process can be broken down to three steps:

1. Create initial design and ensure functionality under ideal/typical conditions
2. Evaluate design under worst-case condition
3. Harden design to maintain functionality under worst-case condition

First, the circuit functionality (e.g., bandwidth, delay, processing power) is designed with ideal transistors. Each transistor features no manufacturing variability and operates at the ideal temperature (typically room temperature), i.e. every transistor operates exactly as specified. Nominal transistors under nominal conditions are called the "typical-typical" process corner, abbreviated TT. The second step is to evaluate the circuit with non-ideal transistors. Now, transistors should operate in their worst capacity. Hence, we use a process corner called "slow-slow" (abbreviated SS):

- Worst possible manufacturing variability:
  smallest geometry, lowest dopant concentrations, etc.
- Worst possible degradation/aging:
  strongest parameter shifts (e.g., 10 year operation at elevated temperatures)
- Worst possible temperature:
  highest temperature, e.g. $125\,^{\circ}\mathrm{C}$

Under this condition, the circuit usually cannot maintain its functionality (e.g., insufficient bandwidth, delay too long, insufficient processing per second) and has to be re-designed to also maintain its functionality in this worst-case condition.

This re-design for the SS corner is the third step and called "hardening" the circuit. The designer selectively employs stronger components to compensate for the degradations introduced by the non-ideal transistors. For example, stronger transistors (e.g., wider transistors in MOSFET, more fins in FinFET) provide higher currents and thus can supply sufficient current even after degradations occurred. The goal is to harden selectively to keep silicon area and power consumption at bay (larger transistors occupy more area and consume more power).

1.3.2 Reliability Guardband

The circuit hardening approach ensures reliability of the circuit at the cost of area and power, while maintaining performance. An alternative to protect against degradations, is to keep the circuit design exactly as is and apply a guardband. Employing a guardbands is less effort than re-designing the circuit and hence a frequently used method to tolerate degradations and variability within the circuit.

The two most common guardbands are timing guardbands and voltage guardbands. A timing guardband is operating the circuit at a lower clock frequency, i.e. a prolonged clock period to allow for a degraded slower circuit. This prolonged period allows the circuit to settle before the next data needs to be processed in the next clock period. For example, if the propagation delay of the circuit is $0.8\,\mathrm{ns}$ at the TT corner (ideal condition) and $1.0\,\mathrm{ns}$ at the SS corner (worst-case degradation) then operating the circuit at a clock period of $1.0\,\mathrm{ns}$ ensures that no timing violations occur within the circuit. Hence, the timing guardband in traditional worst-case estimations is defined as:

$$t_{GB} = t_{delay}(SS) - t_{delay}(TT) \tag{1.1}$$

with t_{GB} being the timing guardband, $t_{delay}(SS)$ the propagation delay of the circuit in the SS corner (worst-case degradation) and $t_{delay}(TT)$ the propagation delay in the TT corner (ideal condition). Conceptionally, the guardband is a deliberately under-performing circuit to ensure that degradations within the circuit cannot induce sufficiently long delay shifts to incur timing violations. A larger guardband allows for more degradations (slower transistors), but also further slows down the processing speed of the circuit. In the example from earlier, operating at the TT clock period of $0.8\,\mathrm{ns}$ would result in a clock frequency of $1.25\,\mathrm{GHz}$, while the SS period of $1.0\,\mathrm{ns}$ results in slower $1.0\,\mathrm{GHz}$ clock frequency. Hence, applying a t_{GB} of $0.2\,\mathrm{ns}$ induces a $0.25\,\mathrm{GHz}$ or $20\,\%$ decrease in processing speed.

Similarly, to the timing guardband, the voltage guardband allows the circuit to tolerate degradations. However, in contrast to the timing guardband, the cost is not performance, but power. In our earlier example, the delay increase from $0.8\,\mathrm{ns}$ to $1.0\,\mathrm{ns}$ forced the designers to decrease the clock frequency to match the prolonged delay with a prolonged clock period. Now, with applying a voltage guardband, the supply voltage V_{dd} is elevated above the nominal value to compensate the transistor degradations. In other words, in the SS corner V_{dd} is increased until the following condition is met:

$$t_{delay}(SS) \text{ at } \mathrm{V}_{dd}(SS) = t_{delay}(TT) \text{ at } \mathrm{V}_{dd}(TT) \tag{1.2}$$

This results in an increase of supply voltage $\mathrm{V}_{dd}(SS) > \mathrm{V}_{dd}(TT)$, which is the voltage guardband definition:

$$\mathrm{V}_{GB} = \mathrm{V}_{dd}(SS) - \mathrm{V}_{dd}(TT) \tag{1.3}$$

with V_{GB} as the voltage guardband and $\mathrm{V}_{dd}(SS)$ the supply voltage in the SS corner as well as $\mathrm{V}_{dd}(TT)$ for the supply voltage in the TT corner. For example, with a voltage guardband of $\mathrm{V}_{GB} = 0.2\,\mathrm{V}$ the circuit would operate at elevated $\mathrm{V}_{dd}(SS) = 1.0\,\mathrm{V}$ from the nominal $\mathrm{V}_{dd}(TT) = 0.8\,\mathrm{V}$ to maintain $t_{delay} = 0.8\,\mathrm{ns}$.

Increasing the voltage by V_{GB} ensures that all transistor degradations are compensated by stronger supply voltage. Hence, performance (bandwidth, processing speed, etc.) remains as desired. The only cost for a voltage guardband is the higher power consumption. If the supply voltage is higher, then the leakage current flowing the transistors is higher (contributing to static power consumption) and the (parasitic) capacitances of the circuits require more carriers to be charged to a higher current, result in larger currents (i.e., higher dynamic power consumption).

Therefore, to tolerate degradations in a CMOS circuit, the designer can either sacrifice performance with a timing guardband t_{GB} or increase power consumption with a voltage guardband V_{GB}. Typically, for simplicity only one of the two methods is employed, but technically a mixture of both guardbands could be used.

To simplify the discussion, the rest of this work focuses on the timing guardband, as both can be used interchangeably. Should the reader prefer voltage guardbands, then qualitatively all results apply analogously. Quantitatively the results do not translate 1 to 1, as the impact of elevated voltage differs significantly from operating at lower clock frequencies. For example, elevating voltage is vastly different if V_{dd} is close to or far away from the V_{th} (near-threshold (NTC) or super-threshold computation (STC)). Despite these differences, the voltage guardband could be estimated in a similar fashion with the same tools (e.g., circuit simulations). Therefore, rest of this document establishes the cost of reliability as performance (timing guardband). However, it is again emphasized that equivalently the cost of reliability could be power (voltage guardband) or silicon area (circuit hardening) or any combination of all three and that analogous approaches and tools would be used.

1.3.3 Pessimism in Worst-Case Reliability Estimations

Employing the SS corner, allows the designer to consider just a single degraded state instead of testing under all possible combinations (e.g., $0.5\,\mathrm{V}$ / $110\,^{\circ}\mathrm{C}$, $0.7\,\mathrm{V}$ / $60\,^{\circ}\mathrm{C}$). Evaluating the circuit under each combination of manufacturing variability, aging and operating conditions (voltage, temperature) would be unfeasible as too many combinations are possible.

However, a disadvantage of the worst-case evaluation of the circuit is severe pessimism. In reality, the circuit will not consist of solely ”unlucky“ transistors with the worst manufacturing variability, operating at the worst operating conditions and with uniform worst-case aging. Therefore, employing the SS corner provides an overly pessimistic, yet safe design.

To illustrate, how severe this pessimism is, a couple of simple examples. First, in a CMOS inverter, only one of the two transistors can conduct a current at a time. Since only one transistor is ON, the other transistor is currently OFF, which means the inherent recovery of aging can take place and defects within the gate dielectric are healing.

Another example is manufacturing variability. Geometry and dopant concentrations follow distributions, hence it is statistically highly improbable that all transistors within a circuit feature smaller than specified geometries and unfavorable dopant concentrations. Since both geometry and dopant follows normal distribution [5], these distribution dictate that some transistors are actually stronger than average (wider geometry, favorable dopant concentrations). Typically, 3- to 6-sigma estimations[1] are used [6], i.e. the SS corner is considered to be three to six times its variance. Mathematically this results in transistors which are worse than $93.3\,\%$ (3 sigma) or $99.999\,66\,\%$ (6 sigma) of the total population.

These conceptional examples, should illustrate how severely pessimistic considering the SS corner is. Actual quantitative explorations are featured in the evaluation of this work, especially in step two and three.

1.3.4 Custom Reliability Estimation to reduce Pessimism

With the ever-increasing degradations due to the discontinuation of Dennard scaling (see Section 1.2), evaluating with the SS corner (i.e., the worst case) becomes ever-more expensive in terms of guardbands and hence performance (see Section 1.3.2). A goal of reliability engineers has been to reduce this pessimism by considering the actual degradation a circuit would encounter (i.e., degradation according to its use-case scenario) instead of the SS corner.

For example, if a circuit is operated with a cooling system, which guarantees a peak temperature of $80\,^{\circ}\mathrm{C}$ (e.g., by increasing fan speed if temperature rises), then evaluating at SS corner with $125\,^{\circ}\mathrm{C}$ is overly pessimistic and provides no benefit with respect to maintaining reliability.

For an actual reliability evaluation, the following information must be gathered (called the use-case of a circuit):

[1] Sigma refers to standard deviation σ of a normal distribution.

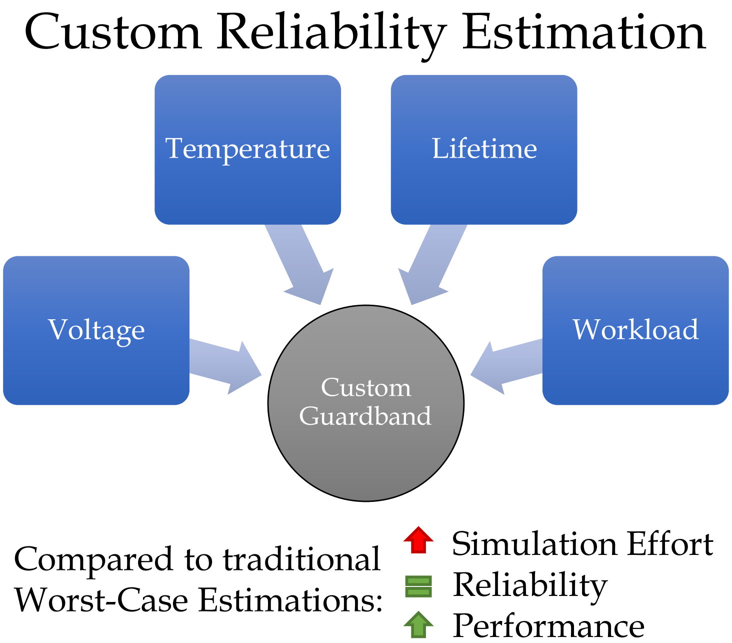


Figure 1.6: Custom reliability evaluation based on the voltage, temperature, lifetime and workload according to a use-case scenario. Compared to traditional worst-case estimations this results in a higher simulation effort, same reliability and better performance.

- Operating conditions of the circuit (e.g., temperature)
- Supply voltage of the circuit
- Workload of the circuit (transistor activity, i.e. transistor duty cycles and switching frequency)
- Target lifetime of the circuit (e.g., 10 years in service or 2 years warranty)

With workload being the application (e.g., software or calculations) run on the circuit. The workload governs the transistor activity, i.e. duty cycle defined as the on-/off-ratio (ON-time divided by total time) and switching frequency as the frequency at which the transistor switches ON to OFF (and vice versa). Note, that each of these values (temperature, voltage, activity) can be taken with their peak value over the entire lifetime or values over time. For example, temperature either peaks at 93 °C or the reliability is evaluated with the actual temperature curve (temperature trace) over time (77 °C for 1 sec, then 79 °C at the next sec, etc.).

With the use-case known for a circuit, variability and aging models can be used to estimate the variability- and aging-induced degradations for this particular data (temperature, lifetime, workload, etc.). This results in a much less pessimistic reliability estimation with much lower degradation levels. These lower degradations result in less required guardbands and thus higher performance.

The key challenge is to determine the degradations correctly and thus determine the required guardband accurately. If the degradation/guardband is underestimated, then the actual degradation might overcome the guardband and the circuit exhibits timing violations (e.g., resulting in data corruption). On the other hand, if the degradation/guardband is overestimated, then the circuit operates slower than necessary and hence performance is lost. However, as perfect estimations are not possible (every simulation features inaccuracies), every uncertainty should result in a slight overestimation of degradation (i.e., a slightly too high guardband). It is crucial to maintain reliability while a slightly sub-optimal performance is tolerable.

1.4 Complexity is the Key Challenge for Reliability Estimations

Complexity is the key reason, why the industry relies on the SS corner (worst-case conditions) for reliability estimations. With the SS corner, all degradation phenomena can be gathered (e.g., experimentally determined) jointly, without identifying the underlying sources of the degradations. Ultimately, the designer does not need to understand the individual phenomena, their dependencies (e.g., phenomenon X rises with temperature but phenomenon Y drops with temperature) or their interactions (e.g., phenomenon X increases in the presence of phenomenon Y).

All that matters for the circuit design is the peak degradation of each transistor (or circuit component) in his circuit in the specified use-case. In this section, we explore three aspects of complexity, which proxy as reasons why custom reliability estimations are not employed by industry:

- Reliability is a quantitatively complex issue
- Degradation phenomena are complex physical processes
- Software tools for custom reliability estimations are complex

1.4.1 Quantitative Exploration of Complexity

To illustrate the complexity in today's complex circuits, we present the following first approximation:

$$\begin{aligned}&1\,000\,000\,000 \text{ transistors with } 2\,\mathrm{GHz} \text{ for } 10 \text{ years}\\&1\,000\,000\,000 \text{ transistors with } 2\,000\,000\,000 \text{ operations per second for } 31\,536\,000\,\mathrm{s}\\&\approx 63\cdot 10^{24} \text{ operations in the circuit}\end{aligned} \tag{1.4}$$

Processors today easily feature 1 billion transistors, which operate at $2\,\mathrm{GHz}$. Assuming a desired lifetime of 10 years, which is $365 \times 24 \times 60 \times 60 = 31\,536\,000\,\mathrm{s}$, this results in 63×10^{24} operations (potential transistor switches) over the processors lifetime (see Eq. 1.4). To put this number in perspective, the number of grains of sand on earth are commonly estimated to be 7.5×10^{21} and thus our complexity (63×10^{24}) is more than a thousand times larger than all grains of sand on earth.

Therefore, simulating an entire microprocessor for 10 years in detail (e.g., using analogue simulators simulating voltage and current flows through each transistor) is absolutely unfeasible. Therefore, reliability estimations always feature abstractions, simplifications and approximations. The challenge is to identify the suitable abstractions, simplifications and approximations to enable these custom reliability estimations despite the overwhelming quantitative complexity, yet limit the accuracy loss of the resulting reliability estimation. This is one of the goals of this work.

1.4.2 Degradation Phenomena are Complex

Reliability estimations are not just complex in terms of the observed components and operation. The degradation phenomena themselves are additionally quite complex physical phenomena. While detailed physical background is explained in Section 2.2, this section provides a high-level overview and motivation.

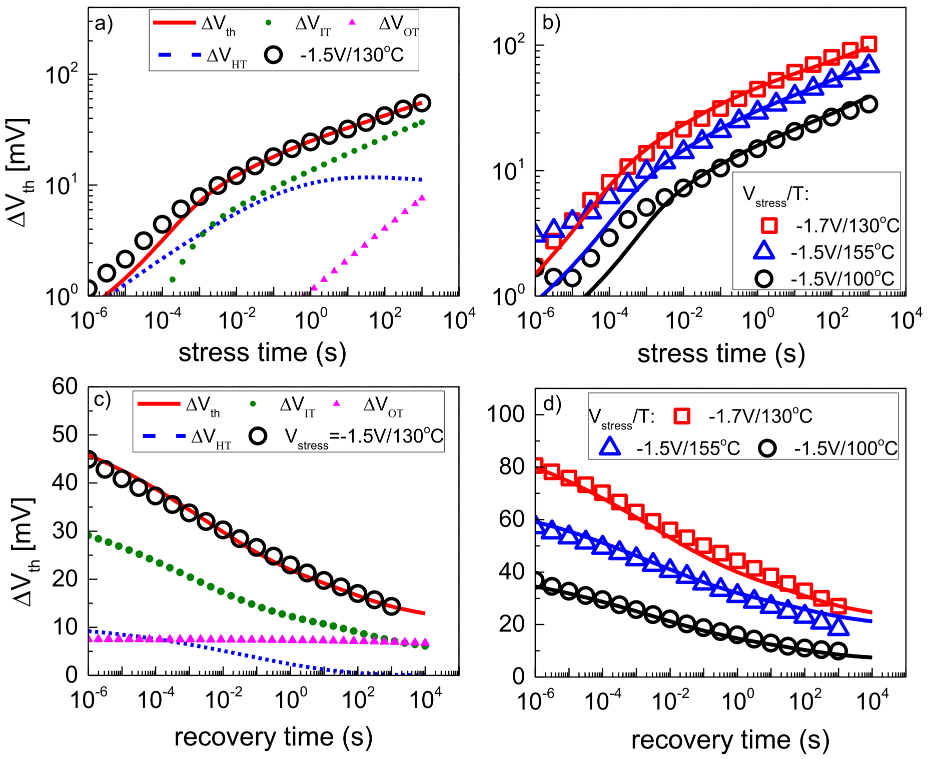


Figure 1.7: Threshold voltage shift due to OT (ΔV_{OT}), HT (ΔV_{HT}) and IT (ΔV_{IT}) individually add up to the total perceived degradation ΔV_{th} at $V_{dd} = 1.5\,\mathrm{V}$ and $T_C = 130\,°\mathrm{C}$. Src: [7]

1.4.2.1 Different Defects in Single Phenomenon

BTI is a phenomenon, which degrades transistors under the application of an electric potential difference (voltage) across the gate dielectric of a transistor. When this potential difference induces an electric field, defects within the gate dielectric are activated (from electrically neutral to electrically charged) and generated (new defect formation).

The problem in modeling BTI is that not a just a single defect type is generated. A gate dielectric is an oxide layer on top of the mono-crystalline silicon substrate. Hence three defect types emerge:

- Oxide Traps (OT), which are defects (e.g., oxide vacancies) within the oxide (SiO_2 and HfO_2) layer.
- Interface traps (IT) which are defects (e.g., dangling valence bonds) from the passivation layer at the interfaces (transition). First, the interface from the oxide dielectric (SiO_2) to the high-κ dielectric (HfO_2) and then from the oxide dielectric (SiO_2) to the substrate (Si)).
- Hole Traps (HT), which are missing electrons within the lattice of the oxide dielectric (both SiO_2 and HfO_2).

Each of these defect types is physically different and hence emits vastly different behavior (see IT, OT and HT degradation in Fig. 1.7a). For example, since HT are carriers, these defects are active in the nano- to millisecond domain. Contrary, the interface traps degrade over months of operation. As these three defect types are fundamentally different, their dependencies with respect to temperature, time, electric field (voltage) and manufacturing processes differ vastly.

Therefore, instead of modeling BTI, in reality three separate models are necessary, where each model models a defect type individually. Then, the induced degradation of each defect type is estimated and combined to obtain the final degradation, macroscopically observed as a shift in the electrical parameters of the transistor.

The different defects also appear in other phenomena like HCD and TDDB. Therefore, each phenomenon is complex to model with individual physical models for each defect type instead of just the empirical macroscopic observations (e.g., some fitting function over experimentally observed transistor parameter shifts).

1.4.2.2 Phenomena are Non-Monotonic (Recovery)

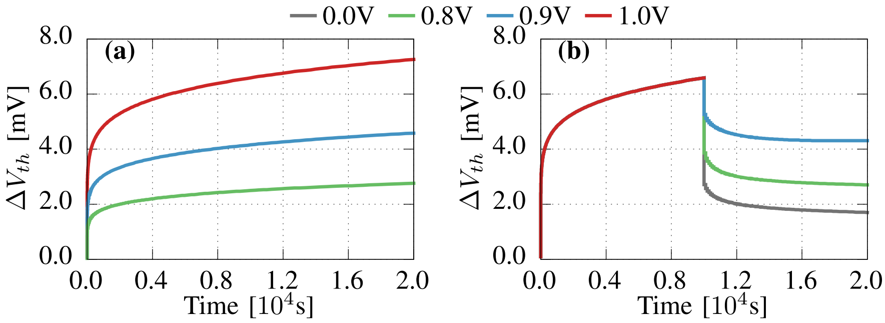


Figure 1.8: (a) Aging-induced degradation is stimulated by the voltage V_{dd}, i.e. higher V_{dd} leads to higher ΔV_{th}. (b) Recovery of aging-induced degradation even if the voltage is above 0 V.

Degradation phenomena become even more complex due to the feature of recovery. Instead of a monotonic increase in degradation (e.g., a monotonic increase in transistor parameters), these phenomena exhibit recovery (healing) [8]. When the stress is reduced (temperature falls, voltage decreases, etc.), the defects which were generated or activated can recover (see Fig. 1.7d) and Fig. 1.8. This means that the physical processes are (partially) reversible, i.e. the lattices of the materials can heal and the defects can be passivated again (turn from electrically charged to electrically neutral).

Therefore, instead of solely modeling the rate of degradation, additionally the rate of recovery must be modeled. As these processes are reversible, induced degradation can be lowered and if this is not considered then the impact (degradation) of these phenomena is severely overestimated. In fact, degradation and recovery are the opposing forces and influence each other, furthering the complexity of the modeling.

As recovery is different for each defect type (with respect to dependencies on temperature, electric field and time), also recovery is individually modeled for each defect type (see IT, HT and OT degradation in Fig. 1.7c) [9]. Lastly, recovery is solely partially reversible.

Some defect types feature irreversible processes, where a reaction is triggered which creates volatile products which can leave the transistor and thus are irreversibly damaged (e.g., hydrogen diffusion out of the chip in IT). This partial reversibility further complicated the modeling and makes reliability estimations even more challenging.

1.4.2.3 Phenomena have a History

Typically, an experiment is reproducible, i.e. applying the same conditions yields the same result. Unfortunately, this is not true for degradation phenomena like aging phenomena.

For example, if the transistor experiences a pulse 2 ms at 1 V and then 2 ms at 0 V, followed by another pulse of 1 s at 1 V, then the response of the transistor will differ (see Fig. 1.9). Before the first pulse, the transistor is pristine and hence only manufacturing-induced defects are present and all defects are electrically neutral. After the first pulse, a certain number of defects is generated and electrically activated (charged). Now the transistor recovers until all reversible degradation has been reversed.

When the second pulse hits the transistor, more defects are present (manufacture-induced and induced by the last pulse) but most will be passive. Therefore, the second pulse might generate less defects (as the easiest locations

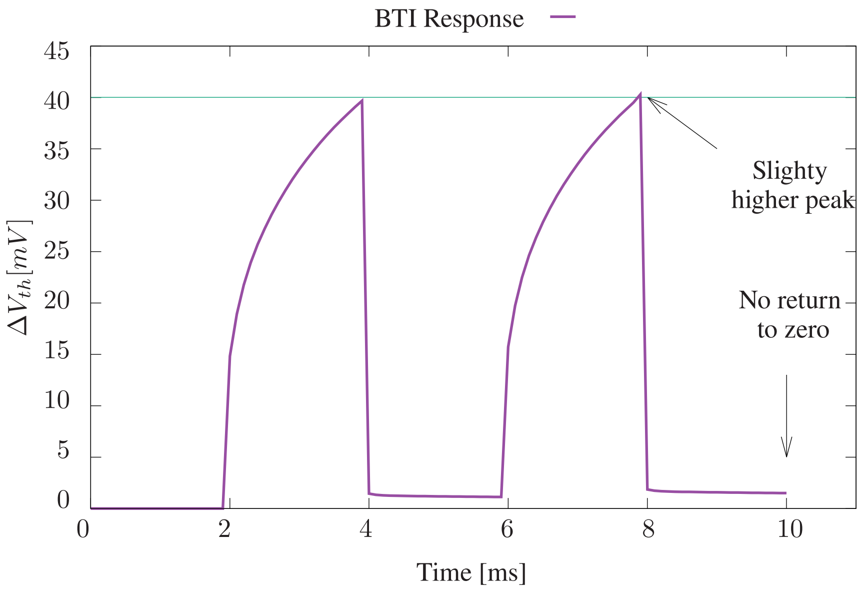


Figure 1.9: The aging phenomenon BTI features history, i.e. the same stimuli (a repeating voltage pulse) results in different responses (as the curve does not return to 0, but instead increases). This difference accumulates over many pulses (not shown here).

(lowest activation energy) to break the lattice already were broken by the first pulse) but activate more defects (as more electrically neutral defects are now present).

Hence, transistors alter over time. Irreversible degradation starts to accumulate and even if all degradation would be reversible, additional defects formed within the lattice create a stronger parameter shift despite the same conditions. In other words, transistors remember their entire history in terms of operating conditions (experienced voltages and temperatures over time).

In the degradation models, these internal states of the transistors need to be tracked (e.g., by calculating the number of defects at all times and tracking irreversible degradation) and considered when calculating the response to a particular condition (e.g., a voltage pulse at a given temperature).

1.4.2.4 Phenomena change constantly with manufacturing

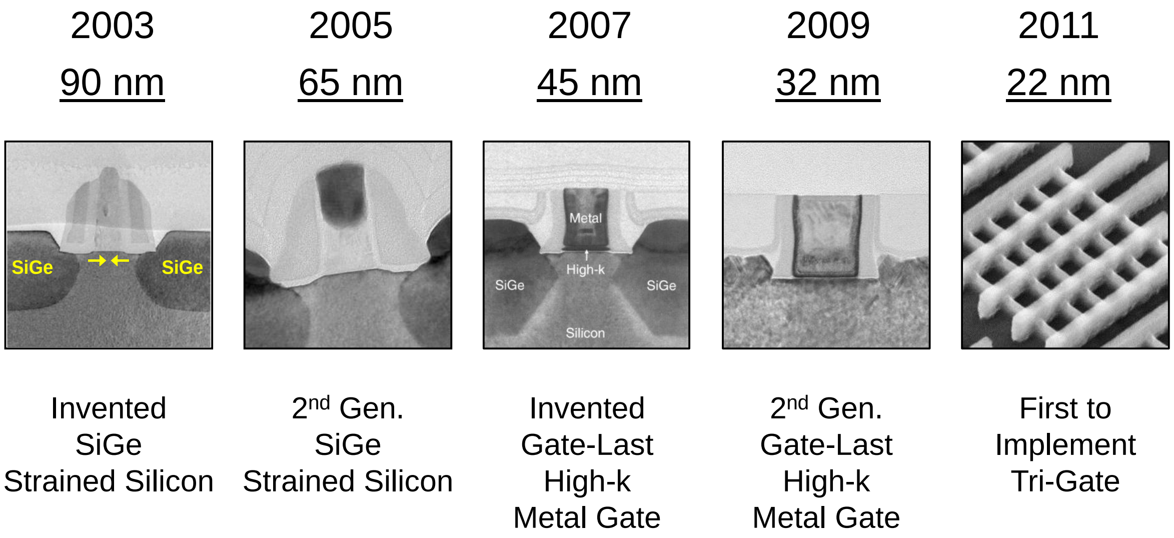

2003
90 nm
2005
65 nm
2007
45 nm
2009
32 nm
2011
22 nm
SiGe
SiGe
Metal
High-k
SiGe
SiGe
Silicon
Invented SiGe Strained Silicon
2nd Gen. SiGe Strained Silicon
Invented Gate-Last High-k Metal Gate
2nd Gen. Gate-Last High-k Metal Gate
First to Implement Tri-Gate

Figure 1.10: Innovations in transistor manufacturing with different materials. These material changes have an impact on degradation phenmomena. Src: Mark Bohr, and Kaizad Mistry. "Intel's revolutionary 22 nm transistor technology." Intel website (2011).

Degradation phenomena are highly sensitive to changes in the manufacturing process and materials of transistors, which are constantly changing (see Fig. 1.10). The phenomena depend on the process as the number of generated defects. These defects are generated during manufacturing and their number is governed by the quality of the material and its lattice (poly-crystalline with grain boundaries, i.e. number and size of grains), which varies with many variables during the manufacturing. For instance, process temperatures (higher temperature means more defects) and annealing steps (healing processes) vary the quality of the crystalline lattice.

Additionally, the choice of materials matters. The oxide layer is not purely a SiO_2 oxide. It is a nitrated (nitrogen-infused) oxide SiO_xN_y With the introduction of high-κ dielectrics, the HfO_2 high-κ layer was added on top of the SiO_2 dielectric to improve the electrostatics of the transistor (reduce channel leakage through higher dielectric constant and gate leakage through thicker gate material).

With these innovations, the addition of N and Hf into the gate dielectric the susceptibility of the material to the manufacturing process as well as susceptibility to the operation with electric field changes [10]. For example, while HfO_2 is electrically superior to SiO_2 it could not entirely replace it as it is electrically and mechanically weaker. In other words, the introduction of HfO_2 worsened BTI as the electric field imposes more degradation in the weaker material. The designers tried to combat the mechanical inferiority by maintaining a thin layer of SiO_2 on the bottom, but the susceptibility to the electric field is unavoidable, since the HfO_2 was introduced to improve the electrostatics and hence the electric field must pass through the HfO_2.

Therefore, the models must be re-calibrated with each change (i.e., each generation) in manufacturing. If temperatures during manufacturing are altered, then the manufacturing-induced defects change and the phenomena react to this change. Similarly, if the materials change, the physical origins of these phenomena alter and hence also their induced degradation. Fortunately, the models do not need to be re-developed as the physical principles and processes do not change. Solely a re-calibration (update the parameters) is necessary. Still, this further complicates degradation modeling within transistors as each new transistor generation now needs to be re-calibrated.

1.4.2.5 New Phenomena appears in recent Transistor Types: Self Heating

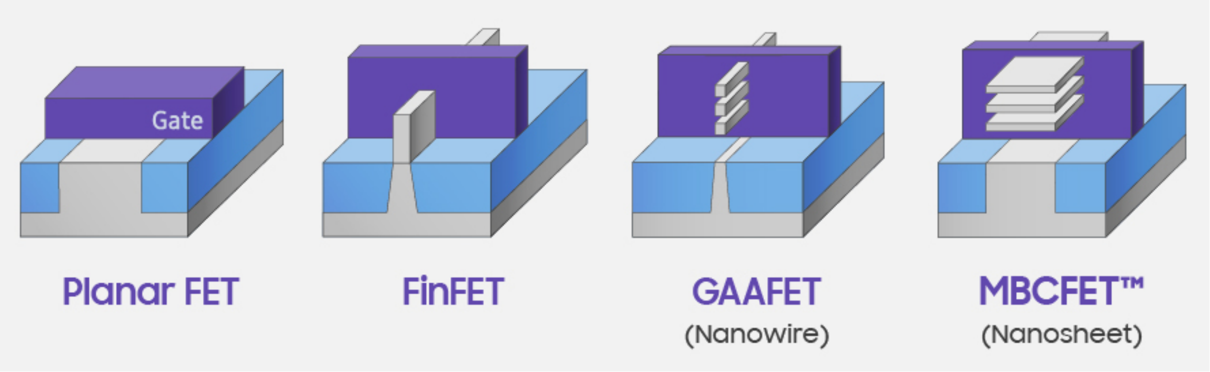


Figure 1.11: Innovations in transistor manufacturing. New generations introduce new materials, manufacturing steps and techniques. All these changes have an impact on degradation phenmomena. Src: Samsung Foundry Forum 2019

With the ever-changing transistors (materials, geometry, doping concentrations) new degradation phenomena may appear. With the introduction of the FinFET transistors and beyond (GAAFET, etc.), the conductive channel is encapsulated in the gate dielectric on three sides (previously in planar transistors only on the top, see Fig. 1.11). This encapsulation improves the electrostatics of the transistor (e.g., decreases leakage), but also insulates thermally. Combined with considerable Joule heating due to the high current density of the channel (a semi-conductor), this elevates the channel temperature. This heating is intrinsic to the transistor operation with encapsulated channels and thus called “self heating” (or Self Heating Effect - SHE).

Self-heating is this new phenomenon, which elevates channel temperatures and thus directly as well a indirectly worsens the performance of the transistor. The direct impact is the reduced carrier mobility through the elevated lattice temperature, which decreases the current flowing through the channel (i.e., lowering the driving strength of the transistor). On top of this direct impact, the elevated temperature of the channel also accelerates the underlying physical processes of the other degradation phenomena (e.g., BTI). Hence, the presence of self-heating worsens the degradations induced by these other phenomena.

This new self-heating was not present in the planar transistors of the last decades. Only recently, with the introduction of the 3D structure of FinFET transistors, this new phenomenon was introduced. Therefore, changes to the transistor technology may introduce new phenomena, which have to be considered in reliability estimations. Worst-case scenarios (like the SS process corner) might not be forced to consider each new phenomenon, as their considerable pessimism ensures that new phenomena are hidden within their extensive safety margins (guardbands). However, for custom reliability estimations, the guardband is chosen tightly (to preserve as much performance as possible) and thus must consider every mayor phenomenon.

1.4.3 Limited EDA Tool Support

Current EDA tools are only in their infancy with respect to custom reliability evaluations. Since the customers were used to and preferred the worst-case estimations via the SS process corner (for simplicity and safety), that is what the EDA vendors chose to support.

For analogue/mixed-signal (AMS) simulations limited support exists in the form of MOSRA (Synopsys) [11], RelXpert by Cadence and UDRM from Mentor [12]. Yet, these tools are very limited in their use. For example, these tools are hard to calibrate against foundry provided reliability data, have a large impact on the run-time and since some (e.g., MOSRA) solely support the use of the build-in aging models, these are outdated (e.g., provide no support for self-heating or time-dependent variability).

For large-scale digital circuits, there is no option beyond the provided SS process corner during synthesis, layout and sign-off. Hence, it is impossible for the chip designer to specify a temperature, a supply voltage, a desired chip lifetime and a given workload and then to design a chip to these specifications (and not unnecessarily robust with SS corners).

In principle, there is no inherit obstacle preventing the integration of degradations beyond the SS process corner into digital or AMS simulation, synthesis and sign-off. The integration is a question of considering additional input data (typically the workload) and then using the existing tool chain to create custom degraded process corners. This integration for both large-scale digital circuit and AMS (with modern reliability models) is the goal of this work.

1.5 Four Steps for Custom Reliability Estimations in EDA Tools

The contributions of this thesis are grouped into four steps. Each step is a step closer to custom reliability estimations in standard tools for large-scale digital and AMS circuits.

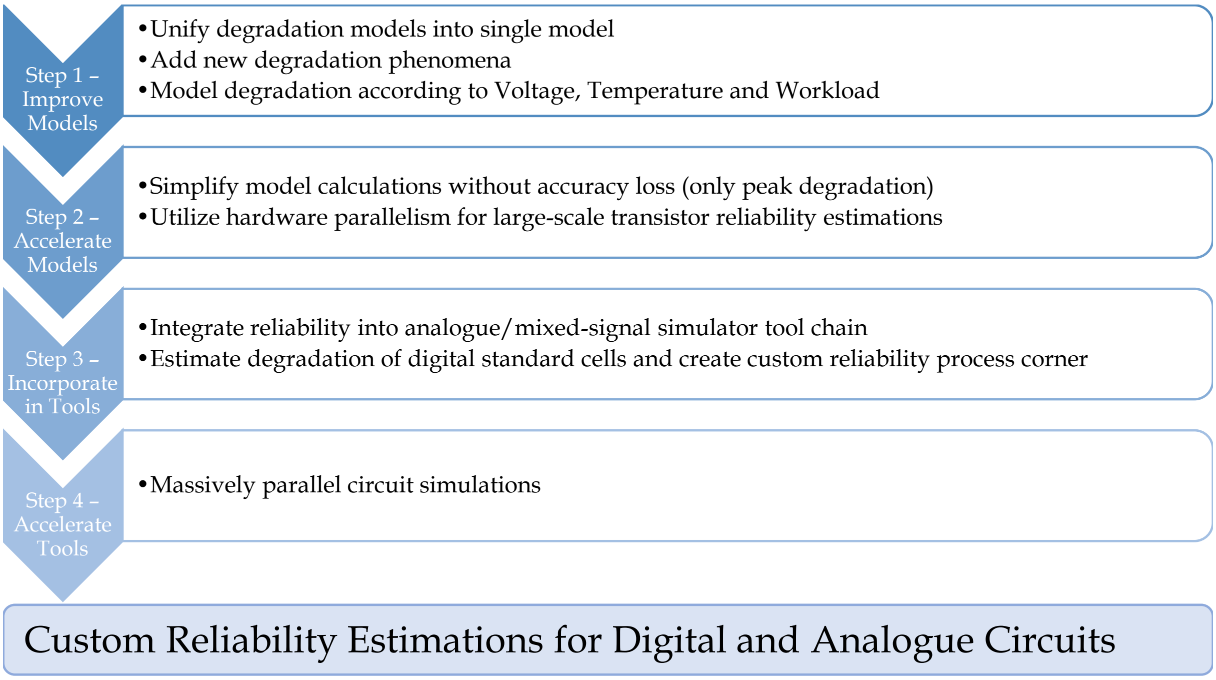

Step 1 – Improve Models
•Unify degradation models into single model
•Add new degradation phenomena
•Model degradation according to Voltage, Temperature and Workload
Step 2 – Accelerate Models
•Simplify model calculations without accuracy loss (only peak degradation)
•Utilize hardware parallelism for large-scale transistor reliability estimations
Step 3 – Incorporate in Tools
•Integrate reliability into analogue/mixed-signal simulator tool chain
•Estimate degradation of digital standard cells and create custom reliability process corner
Step 4 – Accelerate Tools
•Massively parallel circuit simulations
Custom Reliability Estimations for Digital and Analogue Circuits

Figure 1.12: The four steps which group the contributions of this thesis towards custom reliability estimations in analoue and digital circuits.

1.5.1 Step One - Improving Degradation Models

Step one prepares the degradation models (aging, self-heating, etc.) for custom reliability estimations. As discussed in Section 1.4.2, degradation phenomena are quite complex and thus the simplistic modeling offered by EDA vendors is insufficient. To estimate the guardband correctly, a high degree of certainty in the results is required and hence the models have to represent the current state of the art.

This step models new phenomena and prepares the models designed by reliability physicists for the integration into the EDA tools:

- Unify the major degradation phenomena into a joint physical degradation model, which considers their interactions and dependencies on the operating conditions (voltage, temperature, activity). This model estimates the degradation based on the individual conditions of the use-case scenario.
- Consider recently uncovered phenomena (variability in transistor defects, self-heating in FinFET transistors) to enable tools to estimate reliability in current state-of-the-art FinFET transistors.

1.5.2 Step Two - Accelerating Degradation Models

As outlined in Section 1.4.1, reliability estimations are the modeling and estimation of a humongous amount of transistors and operations. In order to make these estimations feasible, the degradation models directly from the reliability physicists cannot be used. These models are optimized to be as accurate as possible and, in the physicists work, estimating a single or only a handful of transistors is sufficient.

However, for our purposes of integrating the degradation models into EDA tools (for large-scale circuits), a much higher performance is required. For these purposes the degradation models are accelerated:

- Speed up modeling by providing upper-bounds for a given operating condition.
- Parallelize the degradation models to leverage the parallel computing hardware for large-scale transistor reliability estimations.

1.5.3 Step Three - Incorporating Degradations into Standard Tools

Once the degradation models are up to date (step one) and sufficiently fast (step two), they can be incorporated into the standard tools. In this work, we highlight how this integration can performed for both large-scale digital circuits as well as AMS circuits:

- Estimate aging in digital standard cells to provide custom reliability estimation for large-scale digital circuits.
- Integrate reliability into analogue/mixed-signal simulator tool chain for custom reliability estimations of analogue/mixed-signal circuits.

1.5.4 Step Four - Accelerating Standard Tools

Section 1.4.1 showed how performance is crucial for reliability estimations in circuits. Therefore, this work presents options to further increase the performance of the standard tools:

- Massively parallel implementation of circuit simulators to enable reliability estimations of large-scale circuits (non-standard-cell digital designs, large analogue designs).

1.6 Custom Reliability Estimation for Analogue and Digital Circuits

The goal of this thesis is the custom reliability evaluation for analogue and digital circuits. For a given use-case, i.e. voltage, temperature, lifetime and workload, the designers should be able to determine the correct guardband (as discussed in Section 1.3.2).

Sufficient timing guardband to tolerate induced degradations in the transistors. This estimation with high certainty leads to less pessimistic guardbands and hence higher performance (via higher clock frequencies).

This certainty in the estimations stems from the state-of-the-art aging and degradation modeling (step one), which provides accurate degradation estimations based on a given temperature, voltage, lifetime and workload for each transistor. Then, since these models are feasible even in large-scale simulations due to their high performance (step two), even large analogue circuits can be simulated in that degraded state (high-performance due to step four), to obtain the new circuit parameters (amplifier bandwidth, noise margins, etc.).

For digital circuits, it is crucial to obtain a custom degraded process corner, i.e. standard cells exactly degraded to the specified voltage, temperature, lifetime and workload. Our integration into the standard tools (step three) allows us to provide degraded process corners for each individual use-case (temperature, voltage, lifetime and workload).

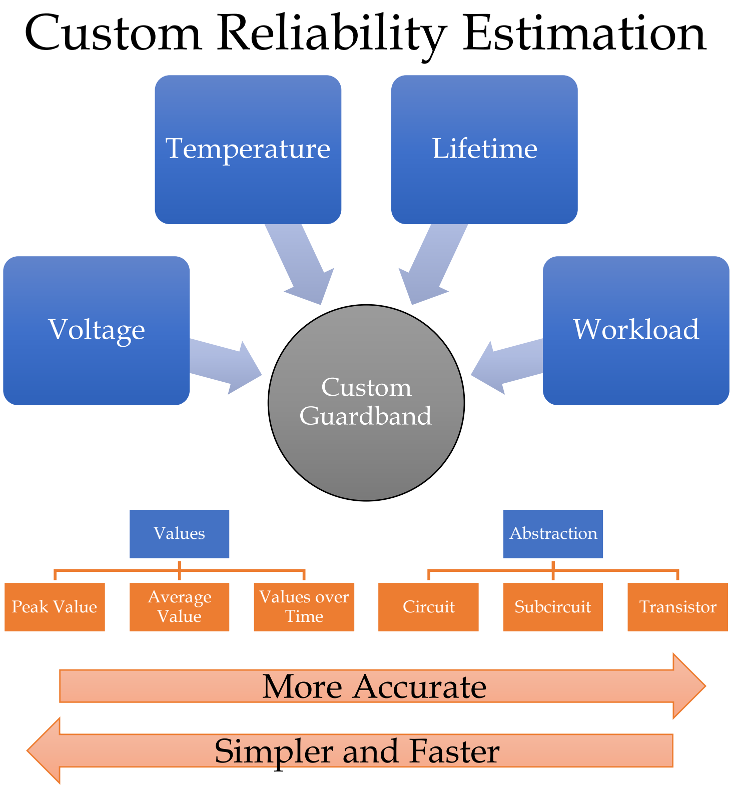


Figure 1.13: Custom reliability evaluation based on the voltage, temperature, lifetime and workload according to a use-case scenario.

1.6.1 Defining a Use-Case

The circuit designer has to define a use-case, which consists of two dimensions:

- Abstraction level
  - Transistor: Voltage, Current, Power, Delay and Degradation is calculated individually per transistor. This represents a fully analogue simulation with the highest level of detail.
  - Standard Cells / Subcircuit: Voltage, Current, Power, Delay and Degradation is estimation for entire subcircuits or subcomponents (e.g., ALU and register file of a CPU). This is typical for digital designs, which employ standard cells instead of transistors for their granularity.
  - Circuit: Voltage, Current, Power, Delay and Degradation is estimated for the entire circuit at once. High inaccuracies for the estimation, but allows for large circuit estimations.
- Value level (for each Temperature, Voltage and Workload individually)
  - Peak Value: A single worst-case value for the entire lifetime.
  - Values over Time: A value (e.g., temperature) for each time step

Most AMS circuit designers are used to detailed use-cases. They employ transistors as the abstraction level and values over time for voltage and workload. Hence, each transistor receives their own unique voltage over time and duty cycles over time. Temperature is typically abstracted to just one peak temperature for the entire circuit (abstraction: circuit, value: peak).

Digital circuit designers are used to standard cell designs. The circuit consists of standard cells, which have their timing (delay) and power information pre-determined from a standard cell library. Voltage and temperature is typically per subcircuit (e.g., higher temperature for the register file in a CPU), higher voltage for the memory controller in a CPU) but as values over time. As temperature and voltage (voltage scaling to increase power efficiency) fluctuate heavily across large digital circuits, many simulation tools (e.g., HotSpot [13]) are employed to determine temperature and voltage over time per subcircuit (many thousands of standard cells each).

An example of a use-case for a digital circuit would be

- Temperature: Peak $125\,^{\circ}\mathrm{C}$ in register file, Peak $85\,^{\circ}\mathrm{C}$ in memory controller, Peak $105\,^{\circ}\mathrm{C}$ in data cache, etc.
- Voltage: $0.9\,\mathrm{V}$ in register file, $1.2\,\mathrm{V}$ in memory controller and $0.9\,\mathrm{V}$ in data cache, etc. All these values over time, with power gating (set $V_{dd} = 0\,\mathrm{V}$) when CPU is idle (not in use).
- Degradation: 10 years at conditions above
- Workload: Worst-case workload (high degradation in each standard cell) as processor is general purpose.

Note, that average values (temperature, voltage, activity) are not useful, as reliability estimations require an upper bound to guarantee reliability. Hence, if a single value is used, it is always the worst value of a given range. Also note, that peak values are not the same as a worst-case estimations. If cooling maintains a peak temperature of $80\,^{\circ}\mathrm{C}$, then this estimation assumes a constant temperature of $80\,^{\circ}\mathrm{C}$, which is much lower than the worst case of constant $125\,^{\circ}\mathrm{C}$.

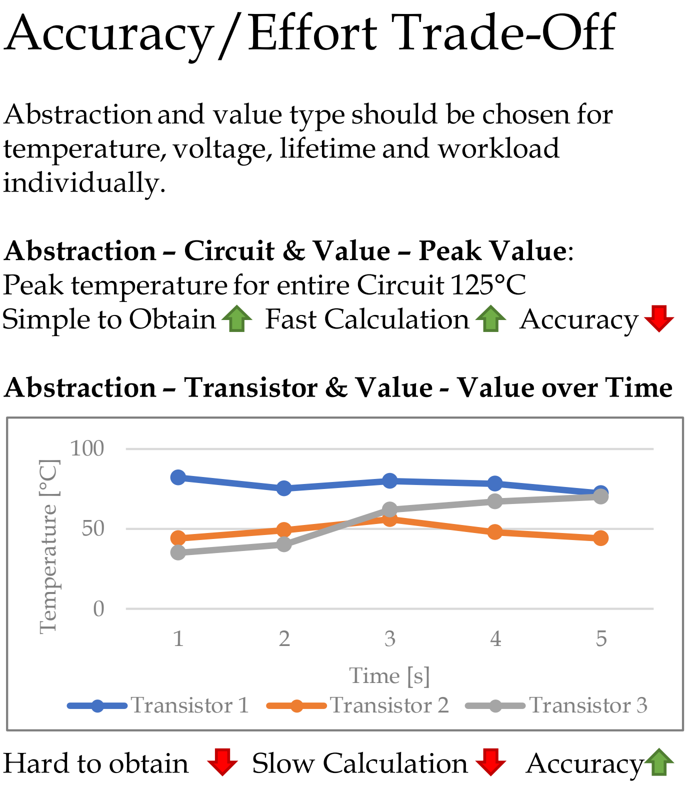
1.6.2 Accuracy versus Effort Trade-Off

During the definition of a use-case, the circuit designer has to balance accuracy vs effort, as shown in Fig. 1.14. For example, for smaller digital circuits, one might rely on a transistor abstraction instead of a standard cell abstraction. Each standard cell is then simulated as a set of transistors compared to an atomic unit in a standard cell library. This imposes a higher simulation effort, yet also provides a more accurate result.

A couple examples to illustrate this trade-off:

- Considering the effect of power gating (set $V_{dd} = 0\,\mathrm{V}$) when circuit is idle to allow degradation phenomena to recover in these power gating cycles. This is in contrast to the simplification of simply considering a constant voltage of $1.0\,\mathrm{V}$ applied throughout the lifetime of the circuit (ignoring power gating).
- Considering the thermal distribution of a chip versus a uniform temperature distribution. Employing a heat map of a large circuit (such as an CPU) allows to define a temperature (peak or over time) per CPU component. As caches (e.g., L2 cache) are typically much colder, than the logic (e.g., ALU) with high power densities, this will reduce the pessimism of the reliability simulation.
- Considering the workload in an embedded system versus a general purpose application. Embedded systems frequently perform the same task over and over again. For instance, a security camera CPU will process an image every $20\,\mathrm{ms}$ (50 frames per second). This means that for the entire lifetime of that processor it will perform the same task every $20\,\mathrm{ms}$ and the activity (duty cycle, switching frequency) of its transistors can be known. Compare this to a smartphone, where the end-user can use any application in the app store of its manufacturer, i.e. where the workload cannot be assumed by the circuit designer.

In each of these examples, considering the more granular information (either spatially or temporal) provides a higher accuracy in the estimated reliability at the cost of more simulation effort. Each additional piece of information reduces pessimism from the estimation as now less and less worst-case assumptions (e.g., worst-case workload versus known workload) need to be made.



Accuracy/Effort Trade-Off

Abstraction and value type should be chosen for temperature, voltage, lifetime and workload individually.

Abstraction – Circuit & Value – Peak Value:
Peak temperature for entire Circuit 125°C
Simple to Obtain ⬆ Fast Calculation ⬆ Accuracy ⬇

Abstraction – Transistor & Value - Value over Time

Hard to obtain ⬇ Slow Calculation ⬇ Accuracy ⬆

Figure 1.14: In a custom reliability evaluation, for each input: voltage, temperature, lifetime and workload a trade-off between accuracy of the resulting guardband and information/simulation effort needs to be found.

1.6.3 Custom Reliability per Use-Case

With the use-case defined, the circuit designer employs the custom reliability estimation to determine a guardband in his circuit which barely maintains reliability. For exactly the temperature, voltage, workload and lifetime specified, the designer receives a custom reliability guardband. As mentioned in Section 1.3.2, this lower guardband either safes performance (timing guardband), power (voltage guardband) or silicon area (circuit hardening).

This regained performance is not just an optimization step, but also allows designers to further benefit from ongoing technology scaling. Each new generation might stimulate the degradation phenomena more (see Section 1.2) and thus with worst-case estimations (SS process corner) there might be no benefit to employ the newer CMOS technology. This is already evident by the reluctance of industries like the automotive industry to employ the newest CMOS technologies. However, with custom reliability estimations reliability can be guaranteed while still benefiting from the power and performance benefits of a new CMOS technology as the pessimism in reliability estimations is significantly reduced.

2 Background in Degradation Phenomena and Models

Degradation phenomena split into two categories. Aging phenomena and degradation effects. First an overview over the key aging phenomena:

Bias Temperature Instability (BTI)
Electrical activation and generation of defects within the gate dielectric of a transistor.

Hot-Carrier Degradation (HCD)
Kinetic activation and generation of defects within the gate dielectric of a transistor, localized near the drain of the transistor.

Time-Depedent Dieelectric Breakdown (TDDB)
Formation of a conductive path through defects within the gate dielectric of the transistor.

Electromigration (EM)
Kinetic transport of material within the conductive wires of a circuit.

Aging phenomena degrade a circuit (transistors or wires) over time, i.e. their degradation is ever-increasing if the same conditions are held over time. Now an overview over degradation effects, which are effects which degrade the transistor without a dependency on time. Either they act immediately (e.g., responses of the transistor to influences like elevated temperature) or are active all the time (e.g., manufacturing variability).

Manufacturing Variability
Tolerances of manufacturing processes induce fluctuation of parameters in semiconductor structures. Manufacturing variability is composed of many smaller effects such as (not extensive list):

↪ Random Dopant Fluctuation (RDF)
The imprecision in the dopant implantation, both in terms of dopant concentrations and dopant distribution across the channel of a transistor and its drain/source regions.

↪ Process Variation (PV)
The imprecision of the lithographic processes, which results in tolerances the transistors geometry (e.g., transistor length (L) and width (W)).

↪ Line Edge Roughness (LER)
The roughness of the edges of features in semiconductor manufacturing.

Thermal Transistor Degradation
Lower carrier mobility and lower threshold voltage if the lattice temperature of the transistor channel increases. Other effects, which also change transistor parameters, are also included see [14].

Self Heating Effect (SHE)
Elevated channel temperature of a transistor due to trapped Joule heating in encapsulated transistor channels (e.g., in FinFET).

Random Telegraph Noise (RTN)
Random capture and emission of charges in defects in the lattice of the gate dielectric of a transistor.

This work does not cover all degradation phenomena, but instead focuses on two types of phenomena. The first type are the currently dominant phenomena, i.e. the phenomena with the strongest impact on transistor performance (e.g., highest induced parameter shifts). Secondly, phenomena which interact with a dominant phenomenon. For example, BTI and HCD share similar physical processes. Even if one phenomenon would not induce strong degradations (even though in current CMOS technologies both BTI and HCD do), we would still have to consider both phenomena, as they strongly interact with each other (e.g., BTI can amplify or mitigate HCD). The defects of one phenomenon can be electrically activated by the other, as explained in [15]. In contrast, the dopant concentration of the channel does not interact with BTI, i.e. RDF and BTI are two completely orthogonal processes and the guardband of a circuit can be simply the sum of the individual guardbands of the orthogonal degradation phenomena. Therefore, this work focuses on BTI, HCD, RTN since they share similar physical processes as well as PV and SHE since these two modulate the degradation imparted by BTI, HCD and RTN. How the considered phenomena interact with each other is explained in the following Sections along with a brief explanation of the phenomena themselves.

For background on the not discussed phenomena, please refer to the following references. Electromigration is explained historically in [16] and with the impact on reliable circuit design in [17]. For TDDB please refer to [18–20] with a special focus on TDDB in high-k dielectrics in [21, 22].

2.1 Empirical and Physics-Based Models

In this work, we frequently distinguish between empirical and physics-based models. Empirical models are models which describe the macroscopic behavior of a phenomenon, without providing a detailed understanding or modeling of the underlying physical processes and mechanisms. For example, when a transistor is experimentally tested one could describe a BTI-induced degradation curve via a power-law equation [23, 24]:

$$\Delta V_{th} = 0.05 \cdot e^{-1500/T_C} \cdot {V_{dd}}^4 \cdot t^{\frac{1}{6}} \cdot \lambda^{\frac{1}{6}} \tag{2.1}$$

with T_C as the channel temperature, V_{dd} as the supply voltage, t as the operation time of the transistor and λ as the duty cycle (on-/off-ratio) of the transistor. Essentially, this model follows a power-law over time with an exponent of $\frac{1}{6}$ and correction factors for temperature, voltage and duty cycle. Therefore, this model is an mathematical fit to experimental data of a transistor with fitting parameters such as 0.05 and $-1\,500$ as well as all exponents to fit the mathematical expression against the experimentally observed data points.

These empirical models feature simple mathematical expressions (see Eq. 2.1) and are hence very useful for simulating large numbers of transistors as each transistor estimation takes little effort (resolve and compute the simple expression). Hence, they are popular in circuit reliability estimations [23, 25]. In fact, before the underlying physical process of a phenomenon are understood, they are the only way to model a phenomenon. Perform an experiment with the transistor or circuit, let it operate under different conditions (in this case different V_{dd}, T_C and λ), measure the degradation and fit the mathematical expression. Therefore, empirical models are the first generation of degradation models for any degradation phenomenon.

Physics-based degradation models take a different approach. Instead of fitting a mathematical expression to experimentally observed data, they try to model the underlying physical processes. Parameters in physics-based models are no arbitrary fitting parameters, but instead natural constants and concepts. For example, the elementary charge of carriers or activation energies to break specific chemical bonds. For the same phenomenon (BTI), a physics-based model (the reaction-diffusion model from [8]) is structured as follows:

Parameter	Value	Description
t	-	Current time
N_{IT}	-	Interface trap density
N_0	-	Initial Bond density
N_H	-	Atomic H density near interface
N_{H_2}	-	Molecular H$_2$ density near interface
C_H	-	Atomic H concentration
C_{H_2}	-	Molecular H$_2$ concentration
k_f	-	Si$-$H Bond breaking constant
k_r	-	Si$-$H Bond annealing constant
k	$8.617 \cdot 10^{-5} eV/K$	Boltzmann constant
δ	$3\,\text{nm}$	Interfacial thickness
E_{AKF}	$0.175\,\text{eV}$	Activation Energy
E_{AKR}	$0.2\,\text{eV}$	Activation Energy
E_{AKH}	$0.3\,\text{eV}$	Activation Energy
E_{AKH_2}	$0.3\,\text{eV}$	Activation Energy
E_{ADH}	$0.2\,\text{eV}$	Activation Energy
E_{ADH_2}	$0.58\,\text{eV}$	Activation Energy
k_{f0}	Device Dependent	Si$-$H Bond breaking constant
k_{r0}	$9.9 \times 10^{-7}\,\text{cm}^3/\text{s}$	Si$-$H Bond breaking constant
k_{H0}	$8.56\,\text{cm}^3/\text{s}$	H dimerization rate
k_{H_20}	$5.7 \times 10^{-5}\,\text{cm}^3/\text{s}$	H$_2$ dissociation rate
D_H	$9.56 \times 10^{-8}\,\text{cm}^2/\text{s}$	Diffusivity atomic H
D_{H_2}	$3.5 \times 10^{-5}\,\text{cm}^2/\text{s}$	Diffusivity atomic H$_2$

Table 2.1: Parameter list of reaction diffusion equations.

$$\frac{\partial N_{IT}}{\partial t} = \overbrace{k_f \cdot (N_0 - N_{IT})}^{\text{Defect Generation}} - \overbrace{k_r \cdot N_{IT} N_H}^{\text{Defect Healing}} \tag{2.2}$$

$$k_f = k_{f0} \cdot (V_g - V_{th0})^{\frac{3}{2}\Gamma_{IT}} \cdot e^{-\frac{E_{AKF}}{k \cdot T}}$$

$$k_r = k_{r0} \cdot e^{-\frac{E_{AKR}}{k \cdot T}}$$

$$\frac{\delta}{2} \cdot \frac{\partial N_H}{\partial t} = D_H \cdot \frac{\partial N_H}{\partial x} + \frac{\partial N_{IT}}{\partial t} - \delta \cdot k_H [N_H]^2 + \delta K_{H_2} N_{H_2} \tag{2.3}$$

$$k_H = k_{H0} \cdot e^{-\frac{E_{AKH}}{k \cdot T}}$$

$$k_{H_2} = k_{H_20} \cdot e^{-\frac{E_{AKH_2}}{k \cdot T}}$$

$$D_H = D_{H0} \cdot e^{-\frac{E_{ADH}}{k \cdot T}}$$

$$\frac{\delta}{2} \cdot \frac{\partial N_{H_2}}{\partial t} = D_{H_2} \cdot \frac{\partial N_{H_2}}{\partial x} + \frac{\partial N_{IT}}{\partial t} - \frac{\delta}{2} \cdot k_H [N_H]^2 + \frac{\delta}{2} K_{H_2} N_{H_2} \tag{2.4}$$

$$D_{H_2} = D_{H_20} \cdot e^{-\frac{E_{ADH_2}}{k \cdot T}}$$

$$\frac{\partial C_H}{\partial t} = D_H \cdot \frac{\partial^2 C_H}{\partial x^2} - k_H C_H^2 + k_{H_2} C_{H_2} \tag{2.5}$$

$$\frac{\partial C_{H_2}}{\partial t} = D_{H_2} \cdot \frac{\partial^2 C_{H_2}}{\partial x^2} - \frac{1}{2} k_H C_H^2 + k_{H_2} C_{H_2} \tag{2.6}$$

The reaction-diffusion model [8] of BTI provides a set of differential equations to model the mechanisms explained in 2.2.1.1.1 and 2.2.1.1.2. Note, that these equations are physics-based equations, which aim at understanding and modeling the underlying physical processes of BTI. The parameters in Table 2.1 are physical constants or experimentally determined properties of materials. To briefly explain the individual equations: Equation 2.2 models the reaction in equation 2.7 with the k_f-term as the bond dissociation term and the k_r term as the bond annealing term. Equation 2.3 describes the dimerization of atomic H into molecular H_2 like the reaction described in equation 2.8. The differential equation takes the diffusion of atomic H and the number of interface traps as additional factors into account. The dissociation of H_2 is described by equation 2.4. Equations 2.5 and 2.6 model the diffusion of atomic or molecular hydrogen respectively.

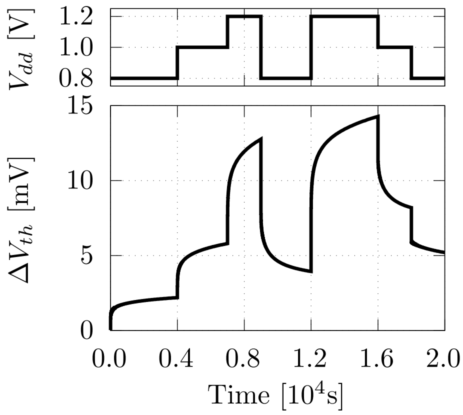


Figure 2.1: BTI-induced degradation (bottom) follows the voltage (top) as it is stimulated by it. Only physics-based models can determine BTI under different dynamic V_{dd}.

Importantly, physics-based equations are not complex for the sake of complexity. By modeling the underlying physical properties, these models are capable of modeling the phenomena with much higher fidelity. Not just higher accuracy (for example, a more accurate dependency on λ or V_{dd}), but also with additional features. For example, in the empirical model the voltage must remain constant over the entire operation of the transistor (i.e. its entire lifetime (e.g., 10 years)), while in the physic-based model the voltage can be changed over time. Therefore, when a circuit scales its V_{dd} to save power, this particular empirical model could not determine the BTI-induced degradation, while the physics-based model can (see Fig. 2.1). Similarly, temperature and other inputs can be dynamic (changing over time) in physics-based models, while most empirical models require static values (identical over time lifetime).

Unfortunately, physics-based degradation models require high computational efforts to obtain a result. Instead of resolving and computing a simple mathematical expression as in Eq. 2.1, an entire set of differential equations must be solved. Therefore, the accuracy and fidelity of the physics-based degradation models come at the cost of complexity and effort.

Nevertheless, whenever possible (not all phenomena are well-enough understood), this work relies on physics-based models as their accuracy and fidelity are necessary for custom reliability estimations. For the defined use-case, dynamic voltage and temperature as well as high confidence in the results is absolutely mandatory to guarantee suitable guardbands. Hence physics-based, are the only choice (if available for that particular phenomenon).

2.2 Aging Phenomena

Aging phenomena haven been a challenge for CMOS reliability for decades and hence are also researched extensively over the same time period. During these decades, various CMOS technologies have come and gone. Technology scaling has introduced new transistor designs and materials, which altered the strength of the aging phenomena. Hence, over the years different phenomena were the dominant aging phenomena in a given CMOS technology as shown in Fig. 2.2.

Note, that aging phenomena had different names over the years. BTI is frequently called NBTI or PBTI if it occurs in pMOS and nMOS (opposite transistor to phenomena naming) transistors. HCD was called Hot-Carrier Injection (HCI), Hot-Carrier Induced Degradation (HCID) or Channel Hot Carriers (CHC). In this work we use the general terms, hence BTI as the overall name for both NBTI and PBTI as well as HCD for hot carriers.

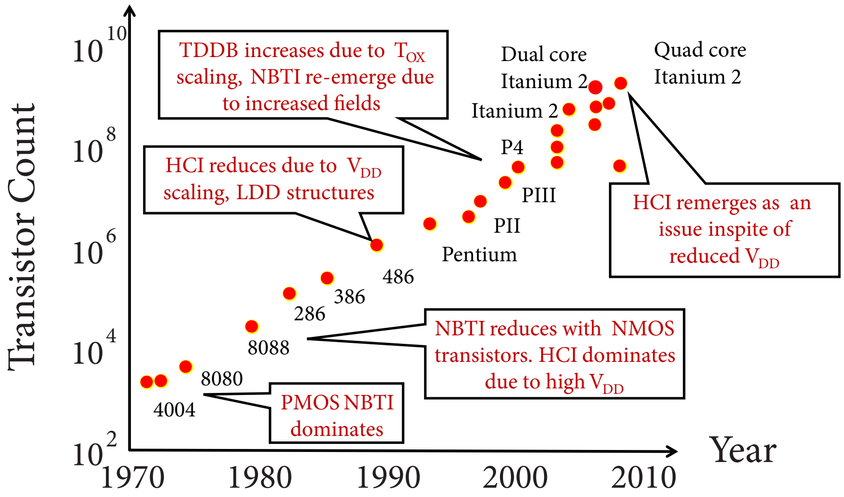


Figure 2.2: History of aging phenomena according to [26] with emerging and subsiding aging phenomena over time. Src: [26].

Although aging has been studied for so long and the rise and decline of the aging phenomena made understanding the key aging phenomenon important at a point in time, the mechanisms behind the aging phenomena are still not fully understood to this day. CMOS technology alters too quickly (new materials, new transistor geometries, smaller geometries introduce quantum effects, etc.) to fully understand and explain the intricacies of the underlying physical processes.

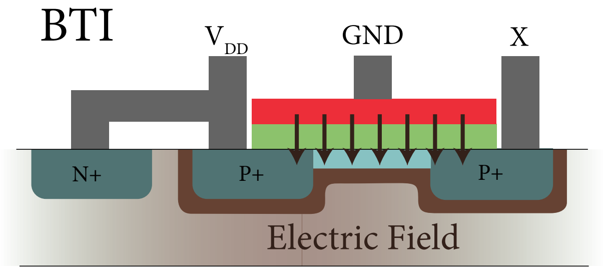


Figure 2.3: BTI occurs due to the vertical electric field over the gate dielectric of a transistors, which generates and activated defects within the gate dielectric.

2.2.1 Bias Temperature Instability

Bias Temperature Instability is the degradation of a transistor (e.g., a shift in V_{th}) due to the generation and activation of defects in the gate dielectric of a transistor. The generation and activation of these defects is due to the electric field over the gate dielectric, which imposes electrical stress on the materials within the gate dielectric.

Two main theories attempt to explain the underlying physical processes. First, the reaction-diffusion theory [9] and secondly the trapping-detrapping theory [27]. Each theory has many different models with many names. For example, the trapping-detrapping theory is implemented as atomistic trap-based BTI model [28], just trapping-detrapping model [29], Comphy (Compact Physics-based model) [30] or Probabilistic Defect Occupancy (PDO) model [31]. In this work, we rely on our collaborators, i.e. Souvik Mahapatra from IIT Bombay with the BTI Analysis Tool (BAT) [9] representing the reaction-diffusion model and Monteserrat Nafria with PDO model [31] for trapping-detrapping. While the details of other model implementations differ, the general explanations within models of the same theory are comparable.

The two theories disagree on various underlying physical processes. A direct discussion between the proponents of the two leaders behind each theory is provided in a joint publication [32]. The publication clearly outlines despite 50 years of research, that certain physical processes and mechanisms of BTI are agreed upon, while other physical processes are disagreed on. This work takes no stance on this discussion and simply employs one model of each theory. The theories have different strengths and weaknesses and this governs our selection in the individual contributions. However, it is important to note, that the other theory (or even other models within the same theory) could be employed with comparable results. For a deeper explanation of the differences in circuit design, refer to [28]. The work in [28] aims to highlight the differences in an actual simulation study of reliability in standard cells.

A short and simplified overview of the two theories is given in Table 2.2. The following sections provide a explanation of the individual theories and more differences will become apparent, but these differences are the key differences.

Property	BTI Theory	
	Reaction-Diffusion (RD) theory	Trapping-Detrapping (TD) theory
Degradation saturates due to	Diffusion limited	Reaction limited
Defect Types	IT, HT and OT	IT and HT. No OT.
Permanent Component	Partial degradation is irrecoverable	All BTI degradation is recoverable

Table 2.2: Comparison between RD and TD theories for BTI.

2.2.1.1 Reaction-Diffusion Theory

This section provides an overview for an explanation of BTI according to the reaction-diffusion theory. For a more detailed explanation about each physical process and mechanism please refer to the BTI book [33].

According to RD, there are three defect types:

- Interface traps (IT) which are defects (e.g., dangling valence bonds) from the passivation layer at the interfaces (transition). First, the interface from the oxide dielectric (SiO_2) to the high-κ dielectric (HfO_2) and then from the oxide dielectric (SiO_2) to the substrate (Si)).
- Oxide Traps (OT), which are defects (e.g., oxide vacancies) within the oxide (SiO_2 and HfO_2) layer.
- Hole Traps (HT), which are missing electrons within the lattice of the oxide dielectric (both SiO_2 and HfO_2).

In the following Section, we explain each defect type in detail.

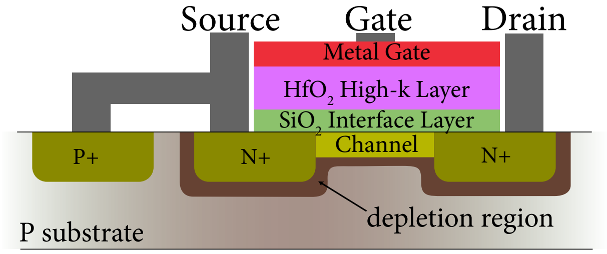


Figure 2.4: Two interfaces (material transitions) exist in a MOSFET. The first interface occurs from the oxide dielectric (SiO_2) to the high-κ dielectric (HfO_2) and then the second interface from the oxide dielectric (SiO_2) to the substrate (Si).

2.2.1.1.1 Interface Traps Interface Traps (IT) are generated at the interfaces (transition between materials). In a high-κ MOSFET, two interfaces exist as shown in Fig. 2.4. The first interface is the transition from the oxide dielectric (SiO_2) to the high-κ dielectric (HfO_2) and then the second interface from the oxide dielectric (SiO_2) to the substrate (Si). At each interface, the materials are unaligned, i.e. not every atom has a partner to bond to. For example, at the SiO_2 to Si interface, the large O atoms sit between the Si atoms and thus have a lower lattice density then the dense Si lattice. Therefore, not every Si atom in the Si-substrate has a Si atom in the SiO_2-dielectric as they are further apart.

Note, everything discussed in this section is from the perspective of a PMOS transistor but works analogously in a NMOS transistor with opposite carrier charges and bias voltages.

Hydrogen passivation As this would result in dangling bonds (electrons not part of an atomic bond), hydrogen (H_2) is introduced during manufacturing to passivate the surface. Hydrogen molecules can separate into hydrogen atoms and then bond to the Si atoms in the Si-substrate who could not find a partner in the SiO_2-dielectric. This creates Si$-$H bonds and thus passivates (removes) all the dangling bonds at the interface. In other words, each Si atoms in the Si-substrate is now either bonded to a matching Si atom in the SiO_2-dielectric or to an H atom. Therefore, removing the negative charges (electrons from the dangling bonds) at the interface.

$$\text{Interface Traps:}$$

$$\mathrm{Si-H+h^+} \rightleftharpoons \mathrm{Si^+ + H} \tag{2.7}$$

$$\text{Molecular Hydrogen:}$$

$$\mathrm{H+H} \rightleftharpoons \mathrm{H_2} \tag{2.8}$$

$$\text{Interface Trap Generation:}$$

$$\mathrm{Si-H} \xrightarrow{h+} \mathrm{Si{\cdots}H} \tag{2.9}$$

$$\mathrm{Si{\cdots}H} \xrightarrow{E_A\ thermal} \mathrm{Si^+ + H} \tag{2.10}$$

$$\text{Interface Trap Recovery:}$$

$$\mathrm{Si^+ + H} \longrightarrow \mathrm{Si{\cdots}H} \tag{2.11}$$

$$\mathrm{Si{\cdots}H} \longrightarrow \mathrm{Si{-}H + h^+} \tag{2.12}$$

Breaking of Si-H Bonds The Si$-$H bonds at the interface can be broken with the application of a vertical electric field across the gate dielectric, which provides the stimulus to attract minority carriers to the Si-substrate. This attraction of minority carriers is the desired formation of the conductive channel in a MOSFET. However, these carriers may also break the Si−H bonds at the interface. This physical process is illustrated in the left side of Fig. 2.5 and follows the following steps:

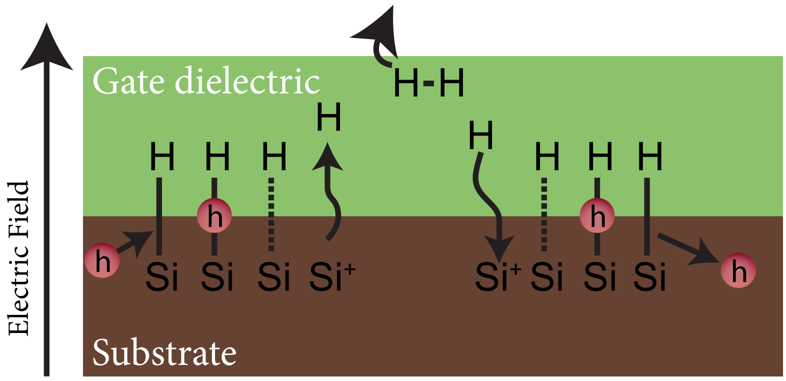


Figure 2.5: Steps of the interface trap generation (5 steps on the left) and recovery (5 steps on the right) due to BTI according to the RD model.

1. The gate of the transistor has potential lower than the substrate, which will generate an electric field over the gate dielectric. Due to the electric field, minority carriers are attracted towards the gate dielectric. This forms a channel directly beneath the gate dielectric. For a pMOS holes are the minority carriers and thus are gathering near the gate dielectric.

2. Some of these carriers (holes) from the channel can use quantum tunneling to reach a Si−H bond. There, the hole recombines with electrons in the Si−H bond. Now the Si−H bond which started at 2 electrons, has only 1 electron remaining, which weakens the bond (dashed line in Fig. 2.5). This weaker bond has a reduced activation energy of both bonding partners.

3. Through thermal activation (kinetic "wiggling" of atoms due to temperature) the H atom obtains sufficient kinetic energy to reach its activation energy. The H atom rips the remaining valence electron to bonds it to an orbit around its nucleus.

4. This Si−H bond is thus broken and the remaining Si becomes positively charged Si^+, as its valence electron was stolen by the H atom.

5. The neutral H atom diffuses slowly through the SiO_2 lattice in the gate dielectric via random hopping.

6. If 2 neutral diffusing H atoms meet, they bond into molecular H_2, which is less reactive and thus diffuses faster.

These steps explain the breaking of the Si−H bonds. However, this reaction saturates, as the following steps limit the reaction rate (diffusion limited):

7. Slow diffusion of H_2 via random hopping through the SiO_2 and HfO_2 dielectrics.

8. The H_2 concentration increases, as H_2 diffusion is limited. This results in $H + H \rightleftharpoons H_2$ reaction reaching a equilibrium.

9. With the dimerization reaction reaching an equilibrium, the atomic H concentration rises, due to the continues generation of H atoms in the $Si{-}H + h^+ \rightleftharpoons Si{\cdots}H$ reaction. This now limits the breaking of Si−H bonds due to $Si{-}H + h^+ \rightleftharpoons Si{\cdots}H$ reaction.

10. The H_2-diffusion front (area of high concentration) moves through the gate dielectric. As soon as the H_2-diffusion front reaches the poly-crystalline Si, H_2 escapes and the damage is irreversible.

11. H_2 concentration reaches an equilibrium as diffusion out of the system and H_2 generation finds a steady state.

Healing of Si-H Bonds Healing the broken Si−H bonds is the reverse mechanism to the breaking of the Si−H bonds. Breaking and healing are exact opposites and the damage due to breaking of the bonds can be recovered by the healing process. Healing occurs when the electric field over the gate dielectric is reduced (it then heals for a while, until a new steady state of degradation is found) or entirely removed (i.e., the transistor can heal with defects still being generated). A full recovery is impossible due to the irreversible nature of the removal of H_2 from the system (as it escapes in Step 10 above). Once H_2 reaches the poly-crystalline gate, it is forever lost and the corresponding interface traps cannot be healed, due to the lack of a bonding partner. Healing follows this reaction equation:

$$2\,\mathrm{Si}^{+} + \mathrm{H}_2 \longrightarrow 2(\mathrm{Si}-\mathrm{H}) + 2\,\mathrm{h}^{+} \tag{2.13}$$

The process is illustrated in the right side of Fig. 2.5 and follows the following steps:

1. Creation of atomic H due to splitting of molecular H_2: $\mathrm{H}_2 \longrightarrow \mathrm{H}+\mathrm{H}$
2. Diffusion of atomic H via random hopping.
3. An atomic H atom finds a reactive Si^+ with an unsatisfied bond (Si ion), i.e. a missing electron.
4. Both partners form a weak bond (dashed line) with the reaction: $\mathrm{Si}^+ + \mathrm{H} \longrightarrow \mathrm{Si}-\mathrm{H}+\mathrm{h}^+$
5. The bond generates a hole (reverse recombination) and strengthens itself (increases activation energy).
6. The generated holes use quantum tunneling through gate dielectric either towards the channel or more likely (due to the vertical electric field) through the gate dielectric towards the poly-crystalline gate electrode.

Most broken Si−H bonds heal immediately, as the atomic H is still near the unsatisfied Si^+. Therefore, there is a constant battle between defect generation and healing. This ratio between defect (bond) breaking k_f and defect (bond) healing k_r, which is defined as $\frac{k_f}{k_r}$, will determine the number of IT defects N_{IT}.

Temperature and Voltage If the strength of the electric field increases (higher V_{gs}), then more minority carriers are in the channel and more Si−H bonds are broken, thus generating IT defects (positively charged Si^+ ions). Similarly, if the temperature T_C is increased, then the achieving the required thermal activation (overcoming activation energy E_A) is more likely and more weakened Si−H bonds are broken: $\mathrm{Si}\cdots\mathrm{H} \xrightarrow{E_A\ thermal} \mathrm{Si}^+ + \mathrm{H}$.

In summary, IT generation is accelerated if either T_C or V_{gs} is increased.

2.2.1.1.2 Oxide Traps Oxide Traps (OT) separate into two types, which are physically and conceptually very similar, but heavily debated [32]:

- Capture and emission of carrier in pre-existing (generated during manufacturing) defects (e.g., oxygen vacancies) in the SiO_2 and HfO_2 lattice
- Generation of new defects (e.g., oxygen vacancies) in the SiO_2 and HfO_2 lattice

The authors of [30] and [31] assume that the generation of new defects within the SiO_2 and HfO_2 lattices are not possible. The Si−O and Hf−O bonds are too strong (too high activation energy E_A) to be broken during the operation of the transistor. Instead, solely pre-existing defects (due to imperfect manufacturing) can be electrically activated (they are electrically neutral after manufacturing).

On the other hand, [9, 33] assumes the capture and emission in pre-existing defects as Hole Traps (HT) and generating of new defects as Bulk Traps (abbreviated as OT, historically Oxide Traps). For the sake of clarity, we follow this notation in this work. Hence, Oxide Traps (OT; Bulk Traps in [9]) are newly generated defects in the SiO_2 and HfO_2 lattices and Hole Traps (HT) are pre-existing (manufacturing) defects in the SiO_2 and HfO_2 lattices. This notation allows us to explain both theories, since TD just assumes OT to be non-existent.

Note, that the regular SiO_2 is in fact nitrided during manufacturing (i.e., Nitrogen is introduced in the lattice) to create a SiON lattice. However, for the purpose of explaining the basics of BTI mechanisms, this is ignored. Additionally, the differences between Si and Hf are also ignored, as the discussed mechanisms are identical for Si and Hf. With respect to BTI, the introduction of the HfO_2 high-κ layer only added a second interface to the gate stack. For the sake of simplicity, consider HfO_2 just electrically superior and mechanically/chemically inferior material to SiO_2. Therefore, SiO_2 still remains a part of the high-κ gate stack (see Fig. 2.4) to protect the HfO_2 from the heat and channel carriers within the channel (as HfO_2 would otherwise deteriorate).

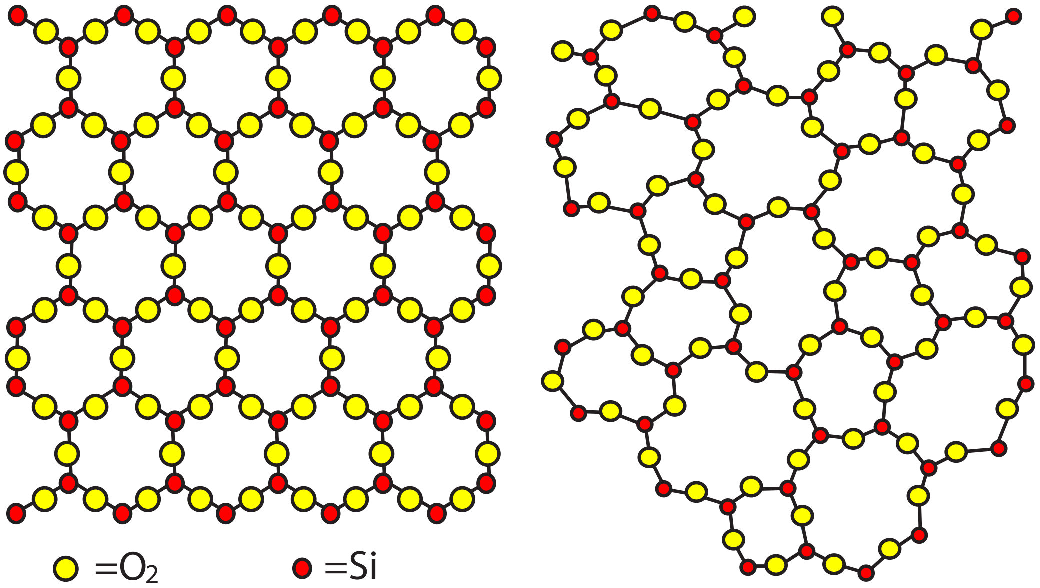
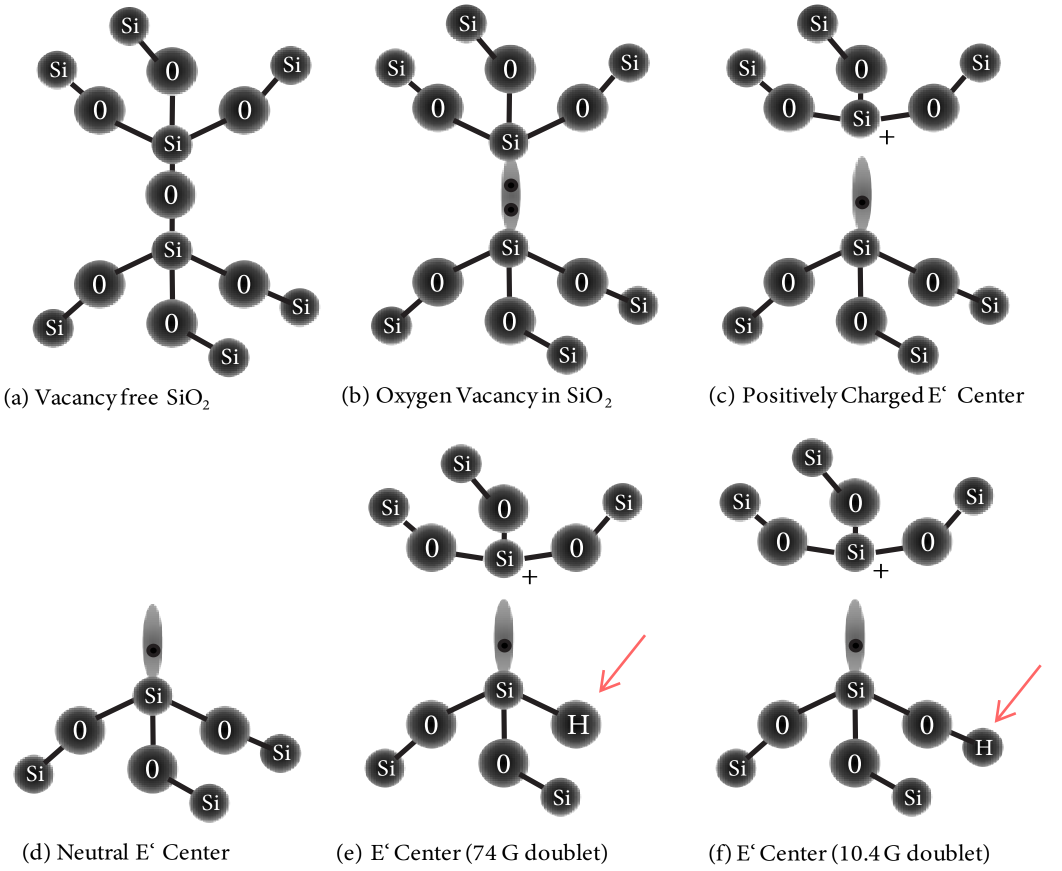


Figure 2.6: Crystalline and amorphous SiO_2. Src: [26]

Amorphous materials are irregular structures compared to crystalline lattices, but even in their irregular structure all atoms have sufficient partners to bond with. Figure 2.6 shows how O_2 and Si bond in a regular crystalline and irregular amorphous lattice. In both lattices, SiO_2 forms rings (visible in hexagons in the left side of Fig. 2.6). In the crystalline lattice, each ring is a hexagon consisting of six Si and six O atoms. In the amorphous form, some SiO_2 rings are bigger (seven or more Si and O) and some SiO_2 rings are smaller (down to 4 Si and O) than in the crystalline structure. These variations from the optimal alignment have a lower activation energy and thus can be split more easily.

Pre-Existing Oxide Vacancies (Hole Traps) During the manufacturing process, these bonds can break due to high temperatures involved and since entropy dictates that some missing O is energetically better [26]. Essentially, it is thermo-dynamically beneficial to miss some oxygen atoms to reach a lower energy state. It is important to note, that tuning manufacturing processes might increase or decrease the number of defects within the lattice. For example, a higher temperature dictates a higher number of missing oxygen as the lowest energy state. Therefore, keeping temperatures as low as possible after growing/depositing the SiO_2 and HfO_2 oxides, reduces BTI in a CMOS technology. These missing oxygen atoms are called oxygen vacancies and shown in Fig. 2.7. Figure 2.7(a) illustrates a healthy lattice opposed to a lattice with missing oxygen in the middle of 2.7(b).



(a) Vacancy free SiO_2

(b) Oxygen Vacancy in SiO_2

(c) Positively Charged E‘ Center

(d) Neutral E‘ Center

(e) E‘ Center (74 G doublet)

(f) E‘ Center (10.4 G doublet)

Figure 2.7: Oxide traps as electrically activated oxide vacancies. Red arrows indicate passivated bonds with hydrogen instead of an oxygen or silicon atom. Src: [26]

Generation of Oxide Vacancies (Oxide Traps) The generation of OT is transforming 2.7(a) → (b), i.e. the generation of a oxide vacancy in a SiO_2 lattice. The physical process follows the following steps:

1. Oxygen will reach its activation energy, due to temperature and the electric field applied over the dielectric. This Oxygen will be located in one of the sub-optimal SiO_2 rings as these have slightly lower activation energies E_A.
2. With the activation Oxygen will break its bonds with the 2 neighboring Si atoms: $\mathrm{Si{-}O{-}Si} \longrightarrow 2\mathrm{Si} + \mathrm{O}$
3. Oxygen will dimerize to molecular oxygen $\mathrm{O} + \mathrm{O} \longrightarrow \mathrm{O}_2$ and diffuse through the system and escape from the system.

According to [9,33] this generation of OT is irreversible and always leads to permanent degradation of the transistor. However, note that the resulting oxide vacancy is electrically neutral in Fig. 2.7(b).

According to [30], OT do not exist, as Si−O bonds have too high of an E_A to be broken by an electric field or temperatures during operation. During manufacturing the transistor is exposed to 600 °C and more, while during operation the transistor is expoed to 125 °C and lower.

Capture and Emission of Carriers within Oxide and Hole Traps A neutral oxide vacancy does not have any impact on the channel of the transistor. To activate such a defect (oxide vacancy), a carrier must be captured. To deactivate (return to neutral) the defect, the carrier is emitted and moves away from the defect. The electrical activation of these oxide vacancies creates charges located deep within the gate dielectric. The two states are shown in Figure 2.7(b) → (c). The activation itself is as follows:

1. A carrier from the channel uses quantum tunneling through the gate dielectric and reaches an oxygen vacancy
2. The carrier recombines with one of the valence electrons of the Si
3. A positively charged trap is generated deep within the gate dielectric

Figure 2.7(d, e, f) show different possibilities on how the electrical properties of the oxide vacancy can be affected. In (d) a neutral oxide vacancy is shown, which has no silicon partner. This could be at the edge of the material (like in the Si$-$SiO$_2$-interface) or when a ring inside the amorphous material is broken. (e) and (f) show modifications where an oxygen or silicon atom is replaced by a hydrogen atom. Hydrogen is used to passivate dangling bonds at the Si$-$SiO$_2$-interface, but it can also passivate dangling bonds in the dielectric itself. When a hydrogen atom is near the oxide vacancy the properties of the electrically activated trap are slightly altered.

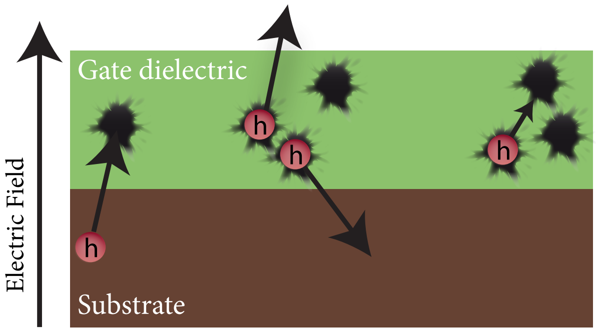


Figure 2.8: Left: Capture (Activation) of carriers in oxygen vacancies. This turns an electrically neutral defect to a positively charged defect, degrading the transistor.
Middle: Emission (Deactivation) of carriers in oxygen vacancies back to the channel or gate electrode.
Right: Moving the carrier within the gate dielectric from one defect to another.

Figure 2.8 shows how tunneling of carriers can be necessary for the electrical activation & deactivation of oxide traps. Carriers can activate an oxide trap by tunneling into the oxide vacancy and recombining with the electron. The second tunneling shown is tunneling from the oxide vacancy towards the channel or gate of the transistor and thus deactivating the oxide vacancy. The last tunneling option is tunneling from oxide vacancy to oxide vacancy, which will deactivate the original oxide vacancy and will activate another one.

2.2.1.2 Transistor Degradation

Interface traps (IT), oxide traps (OT) and hole traps (HT) cause a parameters shift in the MOSFET. Most notable is the weakening of the electric field over the gate dielectric. As positive charge builds up at the gate dielectric (IT, OT and HT produce Si$^+$ ions), the electric field will attract less and less holes to form a channel. Each Si$^+$ ion at the interface (IT) or deep within dielectric (OT, HT) is one less hole in the conductive channel, as they have the same charge and thus repel each other. This manifests itself as a threshold voltage (V_{th}) shift in the transistor. A higher gate voltage is necessary to obtain the same conductive channel strength.

IT also features a second parameter shift due to the positive charge buildup at the SiO$_2$$-$Si interface is a reduction in carrier mobility μ. The channel forms as close as possible to the SiO$_2$$-$Si interface and as positive charges are repelling each other, the carriers within channel experience more Rutherford scattering: The holes in the channel are repelled by the Si$^+$ ion and thus strike the lattice more (resulting in a lower mean free path), thus lowering carrier mobility. OT and HT are deep within the gate dielectric and hence do not lead to increased Rutherford scattering within the channel. Therefore, OT and HT do not lead to a carrier mobility μ shift.

Both V_{th} and μ degradation have an impact on the most important metric of a transistor, the drain current I_D as both V_{th} and μ affect the formation of a channel below the gate dielectric and thus also reduce the current through said channel.

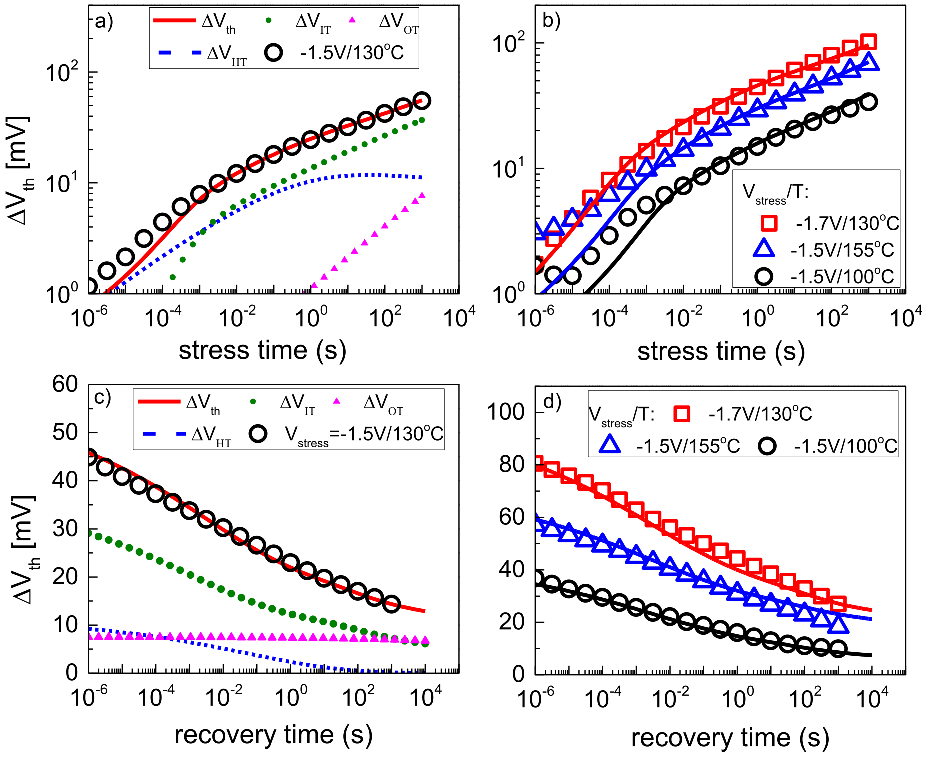


Figure 2.9: BAT models the induced overall ΔV_{th} as a sum of uncorrelated OT (ΔV_{OT}), HT (ΔV_{HT}) and IT (ΔV_{IT}). Note the different time domains of defects. HT are really fast and saturate (when all OT are occupied), IT are quite fast and do not separate (IT feature defect generation) and slow OT generation after seconds. Src: [7]

Note that some BTI models, model the entire I_D degradation solely with a shift in V_{th}. Hence, there is the shift in V_{th} due to BTI directly and an additional representative V_{th}-shift to mimic the impact of lower μ (and other minor factors). While this is physically incorrect (e.g., V_{th} and μ have different thermal dependencies) it simplifies the integration in circuit simulators (changing one easily accessible parameter compared to changing many). Therefore, it is popular to either ignore the parameter shifts beside V_{th} (wrong, as the impact of BTI is underestimated) or to model all degradation with a representative V_{th}-shift (conceptually wrong, but in practice quite accurate).

2.2.1.3 BTI Models

Two BTI models are employed in this work. Both models stem from our collaborators, i.e. Souvik Mahapatra from IIT Bombay with the BTI Analysis Tool (BAT) [9] representing the reaction-diffusion (RD) models and Monteserrat Nafria with Probabilistic Defect Occupancy (PDO) model [31] representing trapping-detrapping (TD) models.

2.2.1.3.1 BTI Analysis Tool The BTI Analysis Tool (BAT) is a reaction-diffusion (RD) model [8,9]. It models BTI degradation as the sum of individual degradation models per defect type as shown in Fig. 2.10. Each defect type is considered to be uncorrelated to the other defect types [9] and as such there are no interactions between these distinct physical mechanisms. These underlying mechanisms for the different defect types are explained in 2.2.1.1.1 and 2.2.1.1.2.

IT modeling is split by generation of recovery of the traps themselves by a set of differential equations (shown in Fig. 2.11 and similar to the older equations shown in Section 2.1). Additionally, the occupancy of these defects (if the defect is electrically charged/active by capturing a carrier) is determined by the IT occupancy modeling. HT are considered to be pre-existing due to manufacturing and thus no generation or recovery of the traps themselves is considered. Instead, solely HT occupancy in pre-existing defects is estimated. For OT the generation of defects

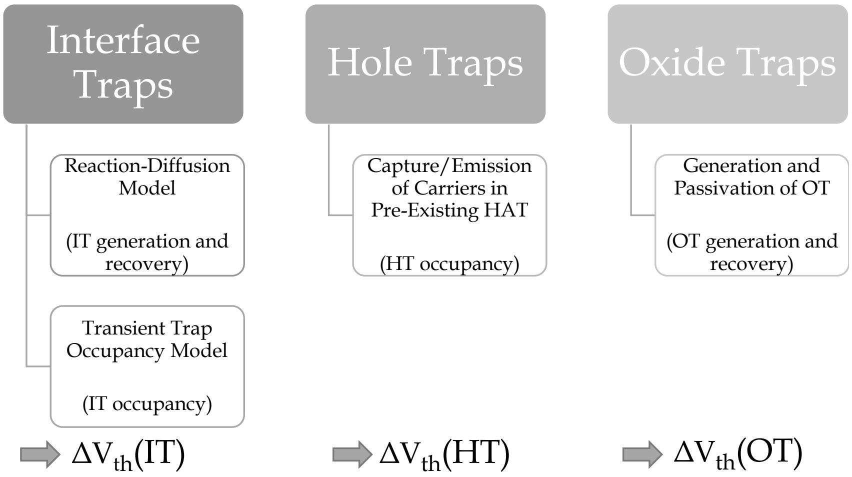

Interface Traps
Hole Traps
Oxide Traps
Reaction-Diffusion Model
(IT generation and recovery)
Transient Trap Occupancy Model
(IT occupancy)
Capture/Emission of Carriers in Pre-Existing HAT
(HT occupancy)
Generation and Passivation of OT
(OT generation and recovery)
ΔV_th(IT)
ΔV_th(HT)
ΔV_th(OT)

Figure 2.10: Structure of the BTI Analysis Tool (BAT) BTI model with its individual modeling per defect type.

is considered. This OT generation is almost permanent as under typical operation conditions (voltages in the 1.0 V range) the recovery of those defects is negligible [33].

The key difference between the original RD equations 2.2 and Fig. 2.11 is that the current RD models take two interfaces into account (SiO_2 and HfO_2 layers in gate stack).

$$\frac{dN_{IT(1)}}{dt} = k_{F(1)}(N_1 - N_{IT(1)}) - k_{R(1)} N_{IT(1)} N_H^{(1)} \quad (1)\,[1^{st}\text{ interface}]$$

$$\frac{dN_{IT(2)}}{dt} = k_{F(2)}(N_2 - N_{IT(2)}) N_H^{(2)} - k_{R(2)} N_{IT(2)} N_{H2}^{(2)} \quad (2)\,[2^{nd}\text{ interface}]$$

$$\frac{\delta}{2}\frac{dN_H^{(1)}}{dt} = D_H \frac{dN_H^{(1)}}{dx} + \frac{dN_{IT}}{dt} \quad (3); \frac{\delta}{2}\frac{dN_{H2}^{(1)}}{dt} = D_{H2}\frac{dN_{H2}^{(1)}}{dx} \quad (4)$$

$$\frac{dN_H}{dt} = D_H \frac{d^2 N_H}{dx^2} \quad (5); \quad \frac{dN_{H2}}{dt} = D_{H2}\frac{d^2 N_{H2}}{dx^2} \quad (6)$$

$$k_{F(1)} = k_{FIT} * E_{ox} * e^{(\Gamma_E E_{ox})} * e^{-\frac{E_{AKF1}}{kT}} \quad (7a)$$

$$\Gamma_E = \Gamma_0 + \alpha / kT \;(7b)$$

$$D_{H2} = D_{H2\_Stress} / (1 + (S * (t / t_{STR}))) \quad (8)$$

$k_{F(1)}, k_{F(2)}, k_{R(1)}, k_{R(2)}$: forward and reverse reaction rate constants

N_1, N_2 : initial H passivated bond density

$N_{IT(1)}, N_{IT(2)}$: interface trap density

$N_H^{(1)}, N_H^{(2)}, N_{H2}^{(1)}, N_{H2}^{(2)}$: near interface H and H_2 density

D_H, D_{H2} : H and H_2 diffusivity

δ : interfacial thickness

Γ_E, Γ_0 : field acceleration, a : polarization factor

S = parameter for stocastic hopping

t_{STR} = stress time

Figure 2.11: Reaction-diffusion equations for two interfaces (SiO_2 and HfO_2 interface) according to [9]. Src: [9]

Stress and Recovery BTI models frequently consider a transistor to be under stress or under recovery. In this context, “stress” means that a transistor experiences a higher stimulation of its degradation mechanisms. This can stem from elevating the voltage or temperature. Similarly, if the voltage drops (either to zero or simply a lower value than before) or temperature drops, then the physical mechanisms reverse and defects physically heal or emit their carrier to electrically deactivate. This process is called “recovery” and reduces the degradation imposed on the transistor.

For Stress :

$$\Delta V_{IT}(t) = \Delta V_{IT}(EOPR) + \frac{qf_{Slow}}{C_{ox}}\left[\Delta N_{IT}(t) - \Delta N_{IT}(EOPR)\right]$$

$$+ \frac{qf_{Fast\_PR}}{C_{ox}}\left[f_{Slow\_PS}\left\{ \Delta N_{IT}(EOPR) - \Delta N_{IT}(EOPS) e^{-\left(\frac{t_{PR}}{\tau_{EC}}\right)^{\beta_{EC}}} \right\}\right]\left[1 - e^{-\left(\frac{t_1}{\tau_{EE}}\right)^{\beta_{EE}}}\right]$$

For Recovery :

$$\Delta V_{IT}(t) = \frac{qf_{Slow}}{C_{ox}}\left[f_{Slow\_PS}\Delta N_{IT}(t)\right] + \frac{qf_{Fast}}{C_{ox}}\left[f_{Slow\_PS}\left\{\Delta N_{IT}(EOPS) - \Delta N_{IT}(EOPR)\right\}\right] e^{-\left(\frac{t_1}{\tau_{EC}}\right)^{\beta_{EC}}}$$

$$+ \frac{qf_{Fast}}{C_{ox}}\left[f_{Slow\_PPS}\left\{ \Delta N_{IT}(EOPR) - \Delta N_{IT}(EOPPS) e^{-\left(\frac{t_{PR}}{\tau_{EC}}\right)^{\beta_{EC}}} \right\}\right]\left[1 - e^{-\left(\frac{t_{PS}}{\tau_{EE}}\right)^{\beta_{EE}}}\right]\left[e^{-\left(\frac{t_1}{\tau_{EC}}\right)^{\beta_{EC}}}\right]$$

time stamp : t = total time, t_1 = time in current cycle

PR (PS) : previous recovery (stress) cycle

EOPR (EOPS) : end of previous recovery (stress) cycle

EOPPS : end of previous to previous stress cycle

f_{Slow} : trap fraction above Fermi level in current cycle

f_{Fast} $(= 1 - f_{Slow})$: trap fraction below Fermi level in current cycle

f_{Slow_PS} : trap fraction above Fermi level during previous stress cycle

f_{Fast_PR} : trap fraction below Fermi level during previous recovery cycle

τ_{EC} : electron capture time (during recovery)

τ_{EE} : electron emission time (during stress)

β_{EC}, β_{EE} : stretching parameters for electron capture and emission

$f_{Slow} = 0.3$ (Type - I) and 0.35 (Type - II) at $V_{GREC} = 0V$

$\tau_{EC} = 1ms$ (Type - I) and 0.08ms (Type - II) at $V_{GREC} = 0V$

$\tau_{EE} = 10ms, \beta_{EE} = 0.4$

$\beta_{EC} = 0.15 @ V_{GREC} = 0V$

Figure 2.12: Transient Trap Occupancy Model for IT defects. Src: [9]

Transient Trap Occupancy Model (TTOM) For IT the BAT actually borrows concepts from the TD models and considers capture and emission of carrier in the defects with the TTOM model [9, 34]. Following the applied voltage (gate-source voltage V_{gs}) of a transistor, the occupancy of the defects is estimated. If a defect is occupied, it is electrically charged and hence contributes to the degradation imposed on the transistor. If the defect is unoccupied, no carrier is present and the electrically neutral defect does not contribute to the degradation.

For Stress : $\Delta V_{HT}(t) = \Delta V_{HT}(EOPR)$

$$+ \left[\Delta V_{HT,MAX} - \Delta V_{HT}(EOPR)\right]\left[1 - e^{-\left(\frac{t_1}{\tau_{HT}}\right)^{\beta_{HT}}}\right]$$

$$\Delta V_{HT,MAX} = \frac{q}{C_{ox}} k_{NHT} e^{\Gamma_{HT} E_{OX}} e^{-\frac{E_{AHT}}{kT}}$$

For Recovery : $\Delta V_{HT}(t) = B + \left(\Delta V_{HT}(EOPS) - B\right) e^{-\left(\frac{t_1}{\tau_{DT}}\right)^{\beta_{DT}}}$

$$B = \frac{q}{C_{ox}} k_{FHT} e^{\Gamma_{HT} E_{OX}} e^{-\frac{E_{AHT}}{kT}}\left[1 - e^{-\left(\frac{t_{CR}}{\tau_{HT}}\right)^{\beta_{HT}}}\right]$$

k_{FHT} : pre factor for hole trapping

Γ_{HT} : field acceleration

E_{AHT} : Arrhenius T activation

$\Delta V_{HT,MAX}$: saturated trapping magnitude

τ_{HT}, τ_{DT} : trapping and detrapping time constants

β_{HT}, β_{DT} : stretching parameters for trapping and detrapping

k_{FHT} : 1e10 /cm^2 (Type - I) and 2.2e10 /cm^2 (Type - II)

$\Gamma_{HT} = 0.2$ cm/MV

$E_{AHT} = 0.055$ eV

$\tau_{HT} = 100ms$ (Type - I) and 50ms (Type - II)

$\tau_{DT} = 1ms$ (Type - I) and 0.08ms (Type - II)

Figure 2.13: BAT HT modeling. Src: [9]

Hole Trap Modeling The HT modeling (shown in Fig. 2.13) considers defects to be generated solely during manufacturing. During operation, therefore solely the occupancy of these defects changes. HT are very fast defects as their degradation can occur in microseconds (see Fig. 2.9). Additionally, as the number of HT is limited (no generation occurs), HT-induced ΔV_{HT} saturates when all HT are occupied with charges. Hence, traditionally HT degradation was ignored, as ultra-fast measurement could not uncover them [35] and they already saturated in measurements over a second.

$$\text{For Stress: } \Delta V_{OT}(t) = \frac{q}{C_{ox}} k_{FOT}(1 - e^{\left(-\left(\frac{t1\text{-}t_{PR}}{m}\right)^{\beta_S}\right)})$$

$$+\left(\Delta V_{OT}(EOPR) - \Delta V_{OT}(EOPPS)\right)$$

$$m = \eta * \left(V_{GSTR}\right)^{\frac{\Gamma_{OT}}{\beta_S}} e^{\frac{E_{AOT}}{kT^{\beta_S}}}$$

$$\Gamma_{OT} = \Gamma_{OT0} + \alpha_{OT} / kT$$

$$\text{For Recovery: } \Delta V_{OT}(t) = \Delta V_{OT}(EOPS)e^{-\left(\frac{t1}{\tau_R}\right)^{\beta_R}}$$

k_{NOT} : pre factor for bulk trap generation

E_{AOT} : Arrhenius T activation

τ_R : time constant for recovery

$\Gamma_{OT}, \Gamma_{OT0}$ = field acceleration

α_{OT} = polarization factor

β_S, β_R : stretching parameter for stress and recovery

η : disperson pre - factor

$k_{NOT} = 5.4e22\,/cm^2$ (Type - I) and $1e22\,/cm^2$ (Type - II)

$\beta_S = 0.3,\ E_{AOT} = 0.7eV, \eta = 5e15,\ \alpha_{OT} = 0.6, \Gamma_{OT0} = -10$

$\tau_R = 2e10s, \beta_R = 0.15$

Figure 2.14: BAT OT modeling. Src: [9]

Oxide Trap Modeling The OT modeling (shown in Fig. 2.14) describes the generation of bulk traps. These traps are responsible for the degradation of the oxide layers themselves and occur under severe conditions, such as high voltage and temperatures as well as long stress times. The OT defects are also responsible for the stress-induced leakage current (SILC) [9, 33]. As seen in Fig. 2.9 OT are really slow defects without saturation (as defects are continuously being generated).

Model Calibration The BAT model employs a wide variety of calibration or fitting parameters as seen in Fig. 2.11-2.14. These parameters depend on the properties of the technology like the used materials and manufacturing process. Thus, calibration of the model takes significant experimental effort, but in return an accurate model for BTI and that specific technology can be determined resulting in accurate reliability estimations.

It is important to note, that while each new (or different) technology requires an updated set of parameters, the actual equations do not change. This is due to the physics-based origin of this model, which describes the underlying mechanisms and not the macroscopic empirical observations (which have to change equations for each new technology).

2.2.1.3.2 Probabilistic Defect Occupation Model The Probabilistic Defect Occupancy (PDO) model is part of the trapping/detrapping (TD) models [32]. TD assumes that all defects (regardless of defect type) originate from the manufacturing process and that no defect generation occurs during operation, i.e. only considers pre-existing traps. Only the electrical activation and deactivation of pre-existing defects. Activating a defect means to capture a carrier from the channel ("trapping a carrier“) and deactivating the defect means to release/emit the carrier back to the channel or the gate ("detrapping a carrier"). Instead of the older trapping/detrapping nomenclature, this work employs the more modern capture/emission terms.

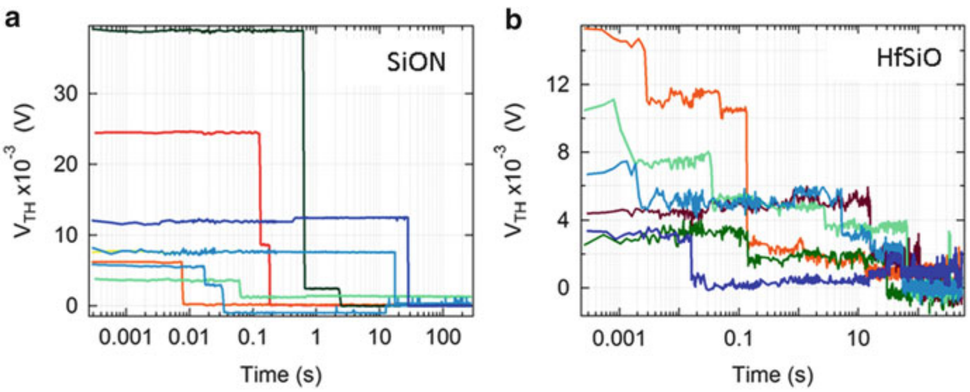


Figure 2.15: Reduction of threshold voltage shift due to carrier emission during recovery in a previously stressed transistor. In nano-scale transistors the stochastic discrete nature of BTI becomes evident with randomly occurring sudden drops in V_{th}.
a) BTI degradation on traditional devices with the sudden stochastic drops
b) Degradation for high-κ devices with noise on top of the stochastic drops. Src: [36]

Among the TD models [28, 37, 38] this work employs the probabilistic defect occupation (PDO) model [31] as the PDO model is easier to combine with other degradation phenomena like RTN. The PDO model assumes a stochastic relationship between the carrier capture/emission in the defects during stress/recovery of the transistor.

Stochastic BTI In current nano-era transistors the number of defects is so small, that individual captures or emissions can be observed, turning a continuous process into a discrete one as shown in Figure 2.15. As discrete capture/emission depends on random processes like carrier quantum tunneling, this discrete process has to be described as a stochastic process.

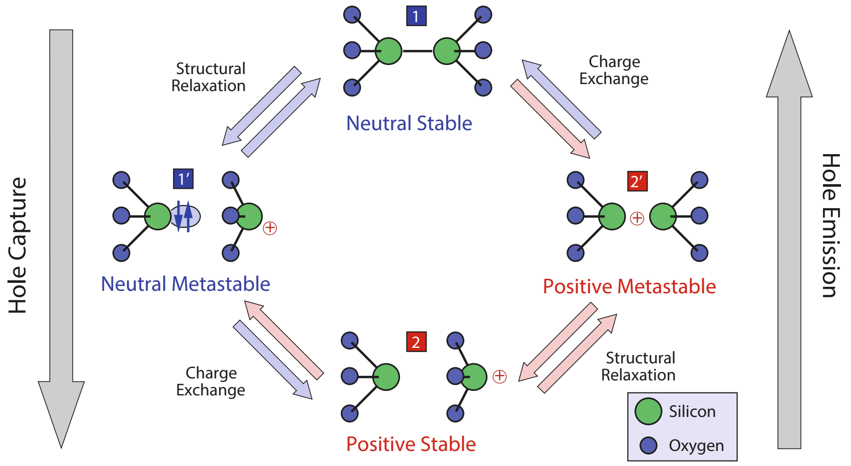


Figure 2.16: Oxide defect states according to [39]. Src: [39].

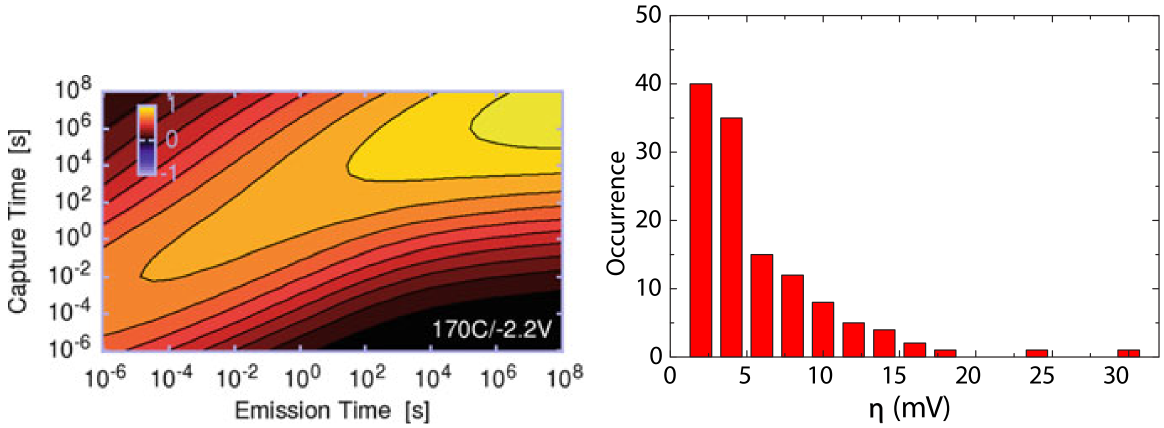
Capture and Emission of a Carrier in a Defect The underlying physical mechanisms of the PDO model are the capture/emission of carriers in defects via various states, as shown with OT in Figure 2.16 and explained in detail in [39]. OT are oxygen vacancies as explained in section 2.2.1.1.2. In oxygen vacancies a carrier can be captured and emitted by the 2 valence electrons between the silicon atoms. When an electric field is applied across the gate dielectric a hole is captured and one of the valence electrons recombines with the hole and a positive charge remains as shown as state 2 in Figure 2.16. If the electric field is switched off, the hole is emitted and both silicon atoms return into their initial state 1.

Note, that this capture and emission of carrier also occurs in the other defect types. While IT require a capture of a carrier to generate such a defect and emit a carrier to heal the IT (see Section 2.2.1.1.1), the IT itself can also capture and emit a carrier. When the Si−H bond is broken and the positive Si^{+} ion remains, it can emit that hole (or capture an electron, depending on perspective) back to the channel and become electrically neutral. In this manner, the IT is still a defect, but similarly to the electrically neutral oxide vacancies it does not contribute to transistor degradation while being inactive (electrically neutral). Hence, the Si ion in an IT can capture and emit a carrier. This is also considered as the TTOM part of IT modeling in BAT.

The key difference between PDO and BAT (or RD and TD models) is in the generation of defects. BAT considers that new IT and OT are generated, while PDO considers all IT and OT to be pre-existent. This is one of the key debates among BTI scientists, as explained in [32].

Location of a Defect The location of the defect within the gate dielectric determines how probable it is for a carrier to tunnel from the channel to the defect. Tunneling has a higher probability for shorter distances, so a defect far away from the channel is unlikely to receive a carrier, whereas a defect located in proximity to the channel is more likely to receive a carrier. Once the stress due to the electric field is released, an activated defect can release its carrier. The probability of the release however also depends on the location of the defect itself. A defect close to either the gate or the channel can get rid of its carrier relatively easy, whereas a trap in the middle of the gate dielectric is surrounded by an insulator.

Furthermore, the location of the defect also affects the degradation to the transistor as defects close to the source or drain of the transistor hamper the channel formation less than defects located in the middle of the channel [40]. The influence of the location on every defect is modeled by 3 parameters, the capture time (τ_c), emission time (τ_e) and the V_{th}-shift (η). The capture and emission times represent the average time it takes until a carrier uses quantum tunneling to be captured or emitted, i.e. the difficulty of tunneling to (capture) and from (emission) the defect. The V_{th}-shift represents the contribution to the threshold voltage shift of every defect, i.e. how much the transistor is hampered in the channel formation, if said defect is occupied.



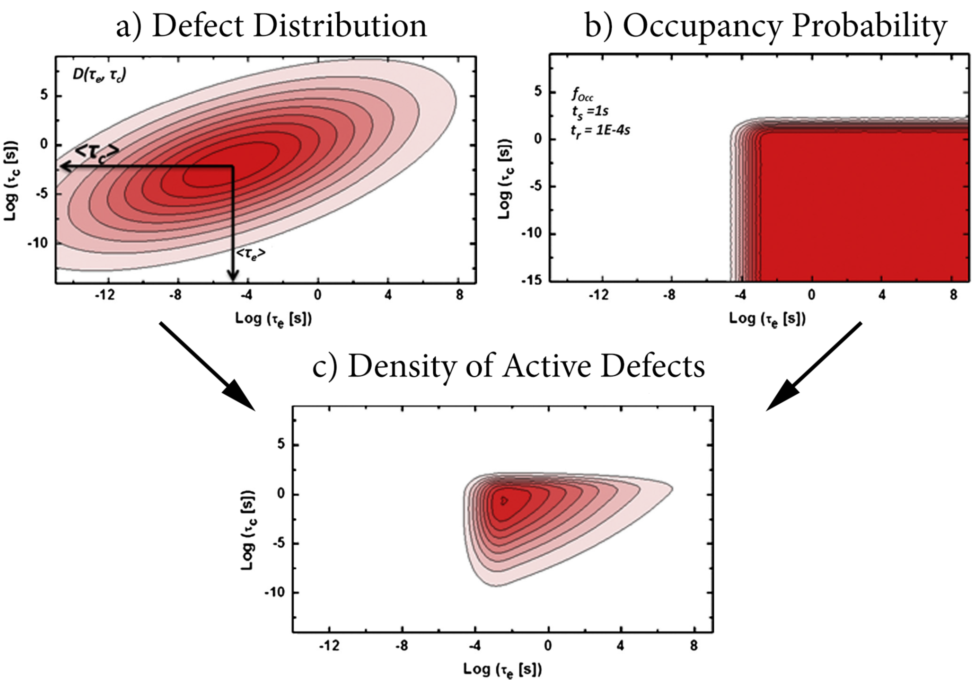
(a) Lognormal bi-variate distribution for capture and emission times. Src: [37]

(b) Exponential distribution shown as a histogram for a Monte Carlo simulation. Src: [31]

Figure 2.17: Distributions for the defect parameters in the PDO model.

According to [31, 37, 41] these 3 properties can be modeled as a log-normal bi-variate distribution for the capture and emission times and an exponential distribution for the threshold voltage shift. These 2 distributions are shown in Figure 2.17a and 2.17b.

A three-step process is necessary to calculate the threshold voltage shift with the PDO model. The defect distribution and occupancy probability must be determined and finally the density of occupied/active defects calculated based on the previous two results. The defect distribution tells us which defects (i.e. (τ_c, τ_e) pairs) are present within our transistor, while the occupancy probability tells us which defects are currently occupied/active due to the stress/relaxation history of the device. Folding the existing defect distribution with the occupancy probability returns density of active defects or in more general terms, the defects currently degrading the transistor.


a) Defect Distribution

b) Occupancy Probability

c) Density of Active Defects

Figure 2.18: a) Lognormal bi-variate defect distribution with the probability density shown as color in the capture-emission-time maps. Darker color indicates higher probability of a defect, i.e. most defects have a capture time (τ_c) around 1×10^{-3} s and emission time (τ_e) around 1×10^{-5} s
b) Shows the occupancy probability due to the current stress/relaxation of the device. With a stress time of $t_{stress} = 1$ s and recovery time of $t_{rec} = 1 \times 10^{-4}$ s the bottom right area has a high occupancy probability. For defects with longer times, for example $\tau_c \approx 10$ s the stress time is not long enough to be likely to be occupied.
c) Shows the mathematical folding of a) and b) and shows the occupied defects which contribute their $\eta = \Delta V_{th}$(defect) to the overall threshold voltage shift of the transistor. Src: [41]

Defect Distribution The defect distribution of the targeted technology has to be determined in experiments. From these experiments we extract technology dependent parameters, which determine the shape and location of the defect distribution in the capture-emission time map shown in Figure 2.18a. The color of the map indicates the density of the defects, i.e. the dark red area shows the capture and emission times which are the most frequent. Note, that this map alters with conditions. For example, if temperature or voltage increases it becomes more likely for carriers to tunnel and thus the τ_c and τ_e times becomes shorter (represents shifting the distribution to the lower left corner in Fig. 2.18a (concept) or 2.17a (measured))

Occupancy Probability Calculating the occupancy probability (P_{occ}) takes the current stress/recovery and past stress/recovery of the transistor into account. Graphically explained with Figure 2.18b: In a transistor under stress the occupancy probability is high in the area under a rising horizontal line. All defects with a $\tau_c < t_{stress}$ have captured a carrier with a high probability, while all defects with $\tau_c > t_{stress}$ are very unlikely to have captured a carrier. This is represented by the horizontal line at $\tau_c = t_{stress}$, where all defects below the line are likely to be occupied and thus encapsulating the rectangular area.

During recovery of the transistor, a vertical line $\tau_e = t_{rec}$ starting at the y-axis and moving towards the right indicates defects which start to release their carriers. All these defects can be characterized by $\tau_e > t_{rec}$ and thus had sufficient time to release their carriers. Every defect left from the moving vertical line will therefore likely have lost their carrier.

Over time, with $t_{stress} = 1$ s and recovery time of $t_{rec} = 1 \times 10^{-4}$ s the occupancy probability distribution shown in 2.18b starts to form, where a rectangle in the bottom right corner illustrates defects which are still occupied. These defects in the bottom right feature sufficiently short capture times τ_c to capture a carrier (bottom of the plot, which corresponds to short τ_c) but feature sufficiently long emission times τ_e to not yet release the carrier (right of the plot, which corresponds to long τ_e).

Occupied Defects The final step to determine the threshold voltage shift is to fold the occupancy probability with the defect distribution to obtain the number of electrically active occupied defects. The density of occupied defects is shown in 2.18c. Once the occupied defects are determined, the individual threshold voltage shifts of each defect (η) are summed up and the overall transistor threshold voltage shift (ΔV_{th}) is calculated. In a simplified manner this can be expressed as follows [31]:

$$\Delta V_{th} = N \cdot \eta_i \cdot \int_0^\infty \int_0^\infty D(\tau_c, \tau_e) \cdot P_{occ}(\tau_c, \tau_e; t_{stress}, t_{relax}) d\tau_e d\tau_c \tag{2.14}$$

The first term N is the number of defects in the defect distribution. The impact per defect η_i is the threshold voltage shift of i-th defect. $D(\tau_c, \tau_e)$ describes the defect distribution and $P_{occ}(\tau_c, \tau_e; t_{stress}, t_{rec})$ is the occupancy probability for given capture time τ_c, emission time τ_e, stress time t_{stress} and recovery time t_{rec}. Both defect distribution and occupancy probability are integrated for all capture times and for all emission times to obtain the currently occupied defects. Each defect then contributes with its η_i to the total threshold voltage in the transistor.

A more detailed and accurate explanation is provided in Section 4.1.5, in which this model is significantly extended as one of the contributions of this work.

2.2.2 Hot-Carrier Degradation

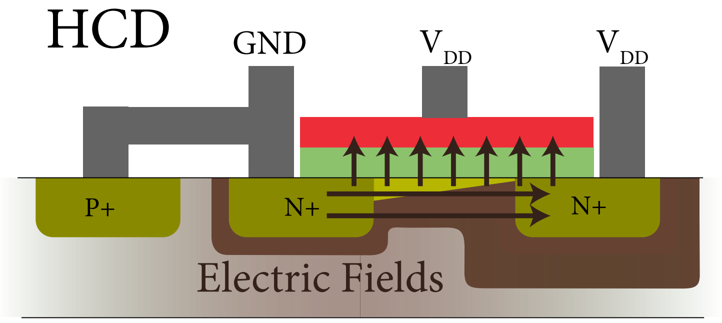


Figure 2.19: Tranistor voltages necessary to induce HCD with both a lateral and vertical electric field.

Hot-Carrier Degradation (HCD) occurs when a lateral and vertical electric field is applied to the transistor (see Fig. 2.19). Carriers form a channel below the gate dielectric due to the vertical electric field. The lateral electric field accelerates these carriers towards the drain of the transistors. Combined this results in a diagonal upward force for the carrier, slamming them into gate dielectric near the drain of the transistor. This deposition of kinetic energy into the gate dielectric can lead to the formation of IT near the drain of the transistor.

Since the formation of IT defects is explained in detail in 2.2.1.1.1 for BTI, in this section only the differences are highlighted. For HCD, the generation of IT differs as it is driven by kinetic energy. Once the defect is formed, the later steps in the process like diffusion are the same mechanisms and as such the reader is referred to the corresponding Section 2.2.1.1.1. The process is shown in Figure 2.20 and follows the following steps:

1. The transistor bias situation for HCD shown in Figure 2.19 is applied and generates two electric fields. One vertical field over the gate dielectric and one lateral field across the channel. Due to the electric field over the gate dielectric a channel forms directly beneath the dielectric. In MOSFETs the channel consists out of minority charges, so for our pMOS transistor, the minority carriers are holes which accumulate near the gate dielectric. The second electric field accelerates the holes towards the drain.

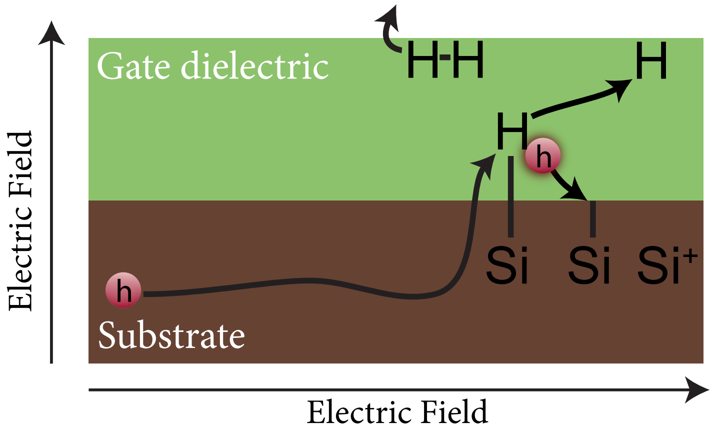


Figure 2.20: Steps of the interface trap generation due to HCID

2. The hole has gained sufficient kinetic energy through the acceleration that it reaches the activation energy of the H atom in the Si−H bond.
3. Due to the vertical electric field and Rutherford scattering the holes are hitting the interface between the channel and the gate dielectric. If a "hot" carrier (i.e., a hole with sufficiently high kinetic energy) hits a H atom, it receives considerable energy due to the deposition of the kinetic energy upon the H atom.
4. With this kinetic energy, the Si−H bond is broken due to the kinetic activation and the remaining Si becomes positively charged (Si^+ ion) as its electron recombines with the remaining hole. The neutral H atom diffuses into the gate dielectric.
5. Continue at step 5 for the BTI explanation in Section 2.2.1.1.1.

This formation of defects can be modeled with the following equation [42, 43]:

$$\Delta V_{th}(HCD) = P\left(1 - e^{-\frac{t_{stress}}{\tau}^{m}}\right) \text{ with} \tag{2.15}$$

$$m = m_0 \cdot e^{-\frac{t}{\tau_m}^{k}} \text{ and} \tag{2.16}$$

$$\tau = \frac{A}{e^{\Gamma \cdot A2(V_d)}} \cdot e^{\Gamma \cdot A1(V_d - \alpha V_g)} + \frac{B}{C} \cdot e^{\Gamma \cdot B1(V_d - \beta V_g)} \tag{2.17}$$

With as P a model parameter related to maximum degradation, t_{stress} is the stress time, time constant τ is related to bond dissociation rate, and m governs HCD modeling for short t_{stress} before the phenomena saturates. To model m for general times instead of just short t_{stress}, the equation 2.16 models m as a time-dependent parameter (depending on current time t). In this equation, m_0 is an adjustable parameter, while both time constant τ_m and parameter k are fixed parameters (1×10^5 s and 0.036, respectively).

The time constant modeling in equation 2.17 models the dependencies on the gate voltage V_g, drain voltage V_d and various technology constants A, A1, A2, B, B1 and C. All these constants can also be expressed as terms depending on temperature (e.g., $A = A1 \cdot e^{\frac{-E_a \cdot A}{k_B \cdot T_C}}$) or gate length L. For details about temperature and channel length modeling as well as the values for the technology parameters, please refer to [42].

Note, that this is an empirical model, as the underlying physics are not modeled. Instead, ΔV_{th} is modeled as a macroscopically shifting parameter according to stress time t_{stress}, current time t, channel temperature T_C, gate voltage V_g and drain voltage V_d. Nevertheless, this is not a simple mathematical fit to a single dimension (e.g., solely stress time t_{stress}), as models used in [23]) but instead considers time, voltage and temperature quite accurately.

Currently, HCD is less understood than BTI and as such no physics-based models are available for current nano-era transistors such as FinFETs. Therefore, we rely on these advanced empirical models in this work.

2.3 Degradation Effects

2.3.1 Process Variation

Semiconductor manufacturing consists of various lithographic processes involving optics, which operate at their resolution limits for current nanometer lithographic etching. These tolerances in the lithographic processes introduce variations in the manufacturing of the geometric structures, e.g. variations of the width W and length L of transistors, called Process Variation (PV). According to [44], PV can be modeled as Gaussian distributions. These distributions only depend on the precision of the manufacturing (e.g., alignment and sharpness of optics in lithographic process) itself and do not change during the operation of the transistor. Hence, the width W and length L of transistors can be defined with:

$$L = \frac{1}{\sigma_{PV}\sqrt{2\pi}} \cdot e^{-\frac{(L-\mu_L)^2}{2\sigma_{PV}^2}} \tag{2.18}$$

$$W = \frac{1}{\sigma_{PV}\sqrt{2\pi}} \cdot e^{-\frac{(W-\mu_W)^2}{2\sigma_{PV}^2}} \tag{2.19}$$

with μ_L, μ_W as the mean for L and W and σ_{PV} as the deviation in the distribution. Note, that both L and W feature identical deviation σ_{PV}, as this is given by manufacturing with the same uncertainty in both X- and Y-direction (i.e., regardless of the orientation of the transistor and the same for L and W).

Note that for FinFET transistors, the width is not continuous but instead discrete. Instead of widening an individual MOSFET to increase the driving strength of transistors (W from $4.8\,\mu\mathrm{m}$ to $6.2\,\mu\mathrm{m}$), FinFETs increase the number of fins and as such are discrete in nature (2 to 3 fins) . However, each fin still has a fin thickness, which is also governed by the manufacturing processes and its tolerances. As such fin thickness can equally be modeled by the same equations above (including identical deviation σ_{PV}) just with its own mean μ_{fin}. The impact of PV on FinFETs is still the same, as the volume/area of the channel (and gate) is altered by PV in both MOSFET and FinFETs.

2.3.2 Random Telegraph Noise

Random Telegraph Noise (RTN) is a noise phenomenon observed within transistors. Specifically, fluctuations in the threshold voltage (V_{th}) cause fluctuations in the driving current of the transistors. It physical origin lies in the probabilistic capturing/emitting of carriers in defects, i.e. the noise is not driven by the electric field. Since quantum tunneling is the underlying physical principle of capture and emission of carriers, these processes are stochastic, spontaneous and intrinsically random by their nature [45]. Spontaneously a defect may emit its carrier, despite the transistor still being on, i.e. with an applied vertical electric field. Similarly, spontaneous capturing may occur, despite the lack of an electric field. These unexpected spontaneous deviations are RTN. RTN charges - analogously to BTI - defects, and thus analogously manifests itself as $\Delta V_{th}(RTN)$ (see Fig. 2.21).

2.3.2.1 Delineation of BTI and RTN

There are various BTI and RTN definitions across various publications. Only recently, with advances in measurement techniques their shared physical origin was uncovered [45, 46]. This also lead to various models, which model the underlying physical principles and thus model both phenomena simultaneously [46–49]. Under this view, RTN and BTI are macroscopic manifestations of the same underlying physics. The stochastic capture and emission of carriers in defects (defects, which may or may not be generated during operation, see RD versus TD BTI model theories in

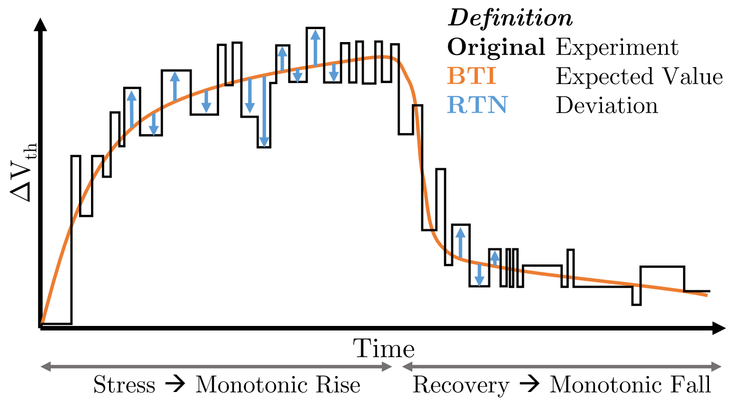


Figure 2.21: Conceptional explanation of BTI and RTN. BTI is monotonically increasing degradation during the on-state of a transistor and monotonic decreasing degradation during off-state of a transistor. The electric field ensures that defects which capture a carrier, are unlikely to release that carrier again. In contrast, RTN is random and spontaneous capturing/emission and represents the deviation from the expected value.

Section 2.2.1.3). To clearly delineate the two phenomena in this work, *we define BTI as a monotonic increasing function of ΔV_{th} when V_{gs} is increasing and monotonic decreasing function of ΔV_{th} when V_{gs} is decreasing.* BTI features inter-transistor variability, i.e. different transistors may age differently, but within a transistor ΔV_{th} is tightly coupled to V_{gs}. The deviation from the expected values (i.e., intra-transistor variability) is by definition RTN. See Fig. 2.21 for a graphical representation of this BTI/RTN definition.

For transistors within the nanometer scale, the expected values of BTI might not be a smooth continuous curve but instead a discrete function, due to the countable number of defects. In large transistors, thousands of defects captured/emitted carriers and thus created a seemingly continuous curve, but now in the nano-era, the countable number of defects manifest themselves as visible individual capture/emission events. Each of these events leads to an increase/decrease of V_{th} with their defects η. This is demonstrated in Fig. 2.22, where a couple of defects capture carriers leading to staircase visible on top for BTI. RTN, on the other hand occurs within a single defect, which randomly captures and emits and thus alters between two levels. Should instead multiple defects fluctuate between occupied (captured) and unoccupied (emitted) states, then 2^n ΔV_{th} levels would be apparent in the lower part of Fig 2.22.

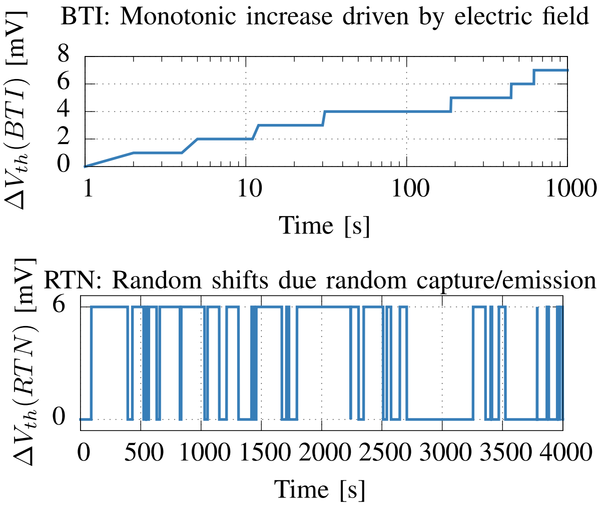


Figure 2.22: Top: Stress Phase of BTI in a modern (e.g. 22nm) transistor. The discrete behavior due to a countable number of defects is illustrated. Bottom: RTN does not increase its magnitude over time. In this plot, a single defect is capturing and emitting, i.e. switching between occupied and unoccupied states. n defects would lead to 2^n degradation levels as each defect can be either occupied or unoccupied.

Traditionally, BTI and RTN were mainly delineated by their impact over time. BTI degraded a circuit over the course of months, while RTN occurred in the micro-second domain. However, two phenomena sharing the same physical origin (capture and emission in defects) with two different time domains contradicts physics. Measurements by reliability physicists illustrate that time constants stretch from micro-seconds to years for both phenomena alike. The Section 5.1.2 highlights how BTI occurs at short time domains. Additionally, it explains how measurement delay alters the measurement, hiding the true fast nature of BTI. These results are in line with the time constants reported by [37] $\tau_c, \tau_e \in [10^{-6}s, 10^{15}s]$ in planar transistors and their results for high-κ FinFET structures in [35]. Further confirmation for fast BTI can be found in the measurements of [34, 50, 51]. Thus, both BTI and RTN occur in the same time domain, which can range from micro-seconds to years as this is simply the time domain the underlying physical phenomenon operates in.

2.3.3 Thermal Transistor Degradation

The MOSFET transistor is susceptible to parameter shifts when the temperature changes. For example, the two main changes in a transistor are the lower threshold voltage V_{th} and lower carrier mobility μ when the channel temperature T_C of a transistor increases. A lower V_{th} increases transistor performance, while a lower μ decreases performance. These opposite forces lead to different thermal behaviors and illustrate the complexity of thermal modeling of transistors.

Transistors exhibit different thermal behavior at different voltages (V_{gs} and V_{ds}, both approximately the supply voltage V_{dd} in digital circuits). At higher V_{dd} , transistors decrease their performance at high temperatures T_C, i.e. the hotter the transistor the more degradation the transistors (lower I_D) exhibits. This is due to the fact, that at higher V_{dd} the detrimental impact of the μ reduction is stronger than beneficial impact of the reduction in V_{th}. For lower V_{dd}, this trend can potentially reverse, i.e. the beneficial impact of lower V_{th} is stronger than the detrimental impact of lower μ. In short, at high V_{dd} high temperature worsens the transistor, while at low V_{dd} temperature might be beneficial.

As not just V_{th} and μ depend on temperature, the actual transistor modeling is more complex. Fortunately, all this complexity is covered by the transistor models. As we employ circuit simulations in this work, we intrinsically employ the transistor models (typically BSIM [14,52] for MOSFET and BSIM-CMG for FinFET [53]) which model the effect of temperature on various transistor parameters.

For example, the *simplified* impact of temperature on the threshold voltage within BSIM is:

$$V_{th} = V_{th0} + \Delta V_{th,all} \tag{2.20}$$

$$\begin{aligned} V_{th0} = {} & \frac{k_B \cdot T_C}{q} \cdot ln \left[\frac{C_{ox}\frac{k_B \cdot T_C}{q} \cdot (C_{ox}\frac{k_B \cdot T_C}{q} + 2Q_{bulk} + 5C_{si}\frac{k_B \cdot T_C}{q})}{2q \cdot n_i \cdot \epsilon_{sub} \cdot \frac{k_B \cdot T_C}{q}} \right] \\ & + V_{fb} + \phi_B + \Delta V_{th,QM} + \frac{k_B \cdot T_C}{q} + q_{bs} \end{aligned} \tag{2.21}$$

Note, that a whole range of parameters is affected by temperature (i.e., feature the term "$\frac{k_B \cdot T_C}{q}$"): C_{ox} is the oxide capacitance, C_{si} is the body capacitance, Q_{bulk} is the fixed depletion charge, $\Delta V_{th,QM}$ is the surface potential considering quantum mechanical effect, k_B is Boltzmann constant, q is the electronic charge, n_i is the intrinsic carrier concentration, T_C is the channel temperature, ϵ_{sub} is the dielectric constant, V_{fb} is the flatband voltage, ϕ_B is the body-effect voltage parameter and q_{bs} is the body doping.

In short, employing accurate transistor models allows for accurate, detailed and most importantly validated thermal modeling. The BSIM and BSIM-CMG models are used for post-silicon validation and sign-off in EDA tools and as such are an established industry standard model. These models are used to model commercial transistors in a production environment. Hence, the thermal modeling is calibrated and validated against experimental data for these commercial technologies by the semiconductor foundries. In fact, in each PDK (e.g., for analogue designs) the thermal parameters are provided to simulate the impact of temperature with BSIM before manufacturing.

In this work, BSIM and BSIM-CMG are used to model the beneficial or detrimental impact of temperature on transistors. Note, that this is the direct impact, i.e. temperature directly lowering or enhancing the performance of transistors. On top of this, temperature also stimulates other degradation phenomena (like BTI), which might hamper the operation and performance of a transistor.

2.3.4 Self-Heating Effect

The Self-Heating Effect (SHE) appeared with the introduction of FinFET transistors at 22nm technology and beyond [54–56]. Contrary to planar MOSFET transistors, FinFETs encapsulate the channel on three sides with gate dielectric to improve the electrostatics of the transistor. Therefore, instead of a strong heat conduction to the substrate of the transistor in planar MOSFETs, now in FinFETs the encapsulation of the channel traps the heat generated within the transistors channel (see Fig. 2.23). This elevates the channel temperature, which was previously only relevant in high-power MOSFETs due to their high current densities.

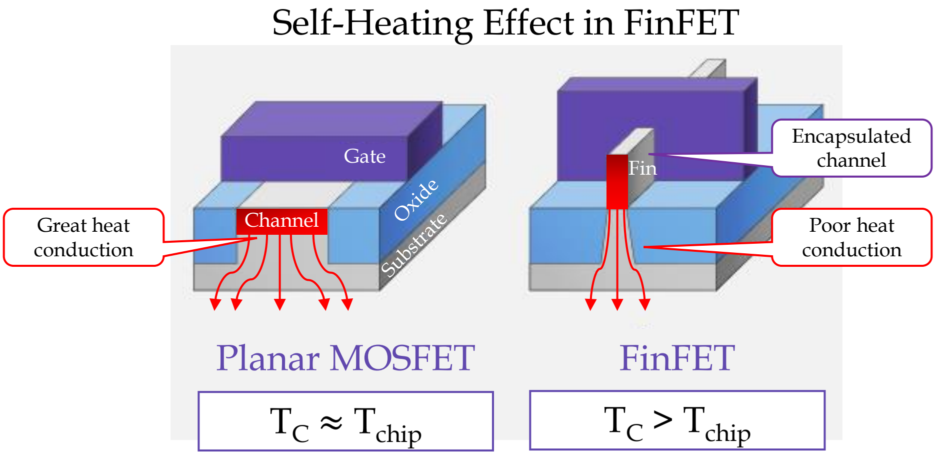


Figure 2.23: Encapsulation of the channel and limited heat conduction to substrate leads to elevated channel temperatures in FinFETs. This effect is the self-heating effect.

The SHE is the elevation of the temperature of the channel within the transistor. As the transistor's channel is a non-ideal semiconductor, the large current densities flowing through the channel result in strong Joule heating within. This generated heat (by the operation of the transistor) is trapped in FinFETs. This is due to the encapsulation of the channel on three sides with the gate dielectric (both an electrical and a thermal insulator) which increases the thermal resistance (R_{th}) of the channel significantly. Additionally, in FinFETs the channel features a tiny contact patch to the substrate below, as in FinFETs the channel is now a tall and thin fin compared to a low and broad rectangle on top of the substrate. Therefore, the generated heat within the channel cannot escape through the gate dielectric and only a limited heat flux is available towards the substrate (see Fig. 2.23).

This SHE results in elevated channel temperatures (see Fig. 2.24), which hampers the operation of the transistor as explained in Section 2.3.3. Different from regular thermal effects, SHE acts on a much faster time scale. For instance, in a FinFET the Fin can heat up and cool down within nanoseconds [58]. Typically, thermal effects are much slower, as the thermal capacitance of the structure is huge. For example, an entire processor needs seconds to heat up, as itself is a quite large structure and additionally it has a huge heat sink on top. Therefore, T_{chip} alters on the time scale of seconds. Yet, the thermal capacitance (C_{th}) of a transistor is minuscule as the fin occupies such a small volume (the Intel $14\,\text{nm}$ Fin is $42\,\text{nm}$ high and $8\,\text{nm}$ wide [57]). This fin is thermally insulated from

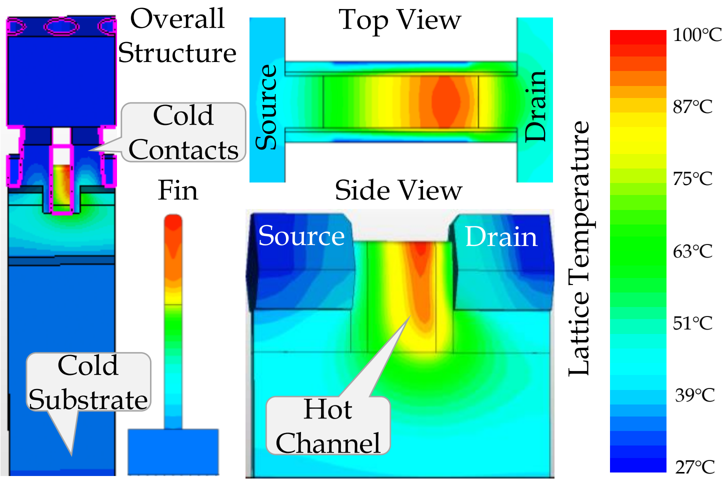


Figure 2.24: Material (TCAD) simulation of SHE heating the channel of a 14 nm FinFET transistor. Note the elevated channel temperature as heat cannot sufficiently be dissipated towards the substrate and source and drain contacts. Src: [57].

its surroundings (due to the encapsulation with the gate dielectric) and thus only its C_{th} matters. Therefore, the transistor can heat and cool within a couple clock cycles.

For the modeling of SHE and its impact on the transistor, we can rely on BSIM-CMG [14, 53]. As mentioned in Section 2.3.3, the model takes care of the impact of temperature on the transistor. Even better, it also features a SHE model (shown in Fig. 2.25) based on a RC-thermal model. RC-thermal models use the duality between thermal fluxes and electric currents to solve a thermal modeling problem in a circuit simulator. Thermal capacitances are electric capacitances, thermal resistances are electric resistances, thermal fluxes are currents and temperature is equivalent to a voltage. This RC-thermal modeling is also used in thermal simulators like HotSpot [13].

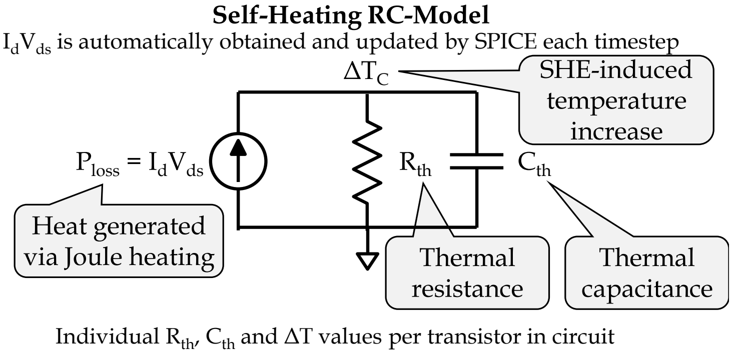

Self-Heating RC-Model
I_dV_ds is automatically obtained and updated by SPICE each timestep
ΔT_C
SHE-induced temperature increase
P_loss = I_dV_ds
R_th
C_th
Heat generated via Joule heating
Thermal resistance
Thermal capacitance
Individual R_th, C_th and ΔT values per transistor in circuit

Figure 2.25: Modeling of SHE in the BSIM-CMG compact transistor model as a RC-thermal model.

For SHE the heat generated is given by the Joule heating within the transistor, i.e. the power lost within the transistor $P_{loss} = I_d \cdot V_{ds}$. This electrical power acts as the heat generated, i.e. a heat flux in the RC-thermal model. Generated heat in a thermal model is represented by current source (as current is heat flux) and combined with thermal capacitance (C_{th}) and thermal resistance capacitance (R_{th}), this results in a simple circuit seen in Fig. 2.25. SPICE automatically provides I_d and V_{ds} to BSIM-CMG and C_{th}, R_{th} are provided by the semiconductor foundry. This calibration allows BSIM-CMG to then estimate $\Delta T_C(SHE)$ to increase the channel temperature (T_C) due to SHE and to apply the effect of the elevated temperature.

For the calibration of the CMOS technologies used in this work, refer for the IMEC 7 nm FinFET technology [59] to Section 4.3.4 and Table 4.2. For the Intel 14 nm FinFET SHE calibration, refer to [57].

3 Related Work

In this section, we present the related work of the individual contributions of this thesis.

3.1 Step One - Improving Degradation Models

Various degradation models are presented from different groups. First, reliability physicists have very accurate physics-based degradation models, which unfortunately are also very slow (e.g., solving differential equations to describe physical processes). These models are unsuitable for circuit simulations, as their target is single transistor modeling with the highest possible accuracy. As a second category, abstracted empirical models exist (describing observations (e.g., mathematical fitting function to measured degradation) without explaining them). These models are much faster (e.g., simple mathematical expressions), but typically cannot model dynamic stimuli (increasing or decreasing temperature, switching voltage, etc.) but instead assume static values (e.g., constant $85\,^{\circ}\mathrm{C}$). Additionally, these empirical models cannot generalize, i.e. their value ranges are limited by the experimental value ranges. For example, if the experiments is performed between $0.8\,\mathrm{V}$ and $1.2\,\mathrm{V}$ then the fitted empirical model cannot predict/determine the degradation at $0.7\,\mathrm{V}$, as the trend might differ significantly outside of the experimental value range.

Our approach in this work is to base ourselves on the physics-based models and to make them suitable for circuit simulations (instead of just single transistor estimations). Therefore, we employ the advantages of the physics-based modeling (accuracy, dynamic conditions, generalization) without their drawback (simulation time).

3.1.1 Unified Degradation Models

BTI and RTN are a frequent topic in the reliability community. Their shared physical origin is frequently reported [45, 51] and well-established. These works focus on the understanding of the physical origin and individual transistors. [46] experimentally characterized the parameters for the BTI model based upon RTN measurements, linking the two phenomena and modeled the impact on SRAM cells. However, neither guardbands, nor the impact of PV or a unified model are discussed. The Authors in [49] and [60] proposed a unified BTI, RTN model. Both did not consider the impact of PV on BTI nor RTN and did not target a wide voltage range including NTC. Including the near-threshold voltage range is important as the impact of PV on RTN/BTI and scaling of η is different in near-threshold compared to super-threshold (see Section 4.1.5.1). The unified model in [47] employed in circuit simulations in [48] has the same limitations.

SRAM cells are often employed to exemplify circuit reliability estimations. [15], [44] targeted multiple phenomena in SRAM cells, but did not consider RTN, BTI and PV jointly nor NTC in general. [61] targeted RTN, BTI and PV jointly with separate models instead of unified model nor targeted NTC (i.e. a wide voltage range). [62] and [63] considered SRAM reliability in NTC, but did not consider RTN respectively BTI.

RTN was targeted in [64] and [65] however they did not include BTI nor NTC in their work.

3.1.2 Integrating Self-Heating as a new Phenomena

The authors in [58,59,66,67] measured the frequency-dependence of FinFET transistors, but did not consider duty cycles or workloads. The duty cycle is only infrequently mentioned as a artifact during the characterization of SHE [68] and not seen as a key aspect, driven by the workload, that must be considered.

Temperature guardbanding for standard cells and large circuits (e.g. microprocessors) is discussed in detail in [69], but self-heating is not considered. A layout-level detailed temperature estimation for standard cells is presented in [70], but they do not capture SHE in transistors, i.e. induced ΔT_C.

Thus, in this work, we present the impact of duty cycle λ and actual workload-driven switching frequency f_{sw} on SHE for the first time. Additionally, this is the initial report for the impact of SHE on large circuits like microprocessors.

3.2 Step Two - Accelerating Degradation Models

Contrary to other works, the goal of this step is pure performance benefit without a loss in accuracy. In the reliability physics community, accuracy of physics-based degradation models is typically sacrificed to improve calculation speed. The abstracted empirical degradations are typically so fast, that there is no need for speed-up.

3.2.1 Estimating Peak Degradation with Longest Continuous Stress

Diverse approaches for BTI modeling exist ranging from the physical level [31] towards the micro-architecture level [23]. At the physical level BTI is measured based upon defect concentrations in transistors and its impact is expressed as induced shifts in transistor parameters (threshold voltage shift ΔV_{th}) [31]. These BTI models, which model the underlying physical processes of BTI to estimate it, are *physical BTI models*. In contrast, at the micro-architecture level, BTI is measured by observing failure rates of chips over time. Then BTI is expressed by simple equations fitted to mimic the observed failure rates in simulations which model shifts in transistor parameters (ΔV_{th}) [23]. BTI models with equations fitted to match chip failure behavior are called *empirical BTI models* in this work.

Interestingly, the *physical* and *empirical* approach differs significantly due to the direct (transistor degradation) and indirect (chip failure rates) calibration with measurements. Empirical models have a high degree of uncertainty due to the probabilistic nature of chip failures [23]. To ensure reliable designs, circuit designers must consider the worst samples of these distributions and design their guardbands accordingly.

Despite their inherent uncertainty, *empirical BTI models* are used as their simplicity and speed allows BTI estimations within complex circuitry. However, to carefully design *narrow guardbands*, the *physical models* are more suitable as their detailed modeling reduces uncertainty, providing results closer to the actually required guardbands. Therefore, commercial design tools like MOSRA from Synopsys employed *physical models* [71]. Since physical models are computational infeasible, MOSRA reduced the number of mathematical terms in their hot carrier model to limit computational and calibration complexity at the cost of compromising accuracy [71]. Despite those efforts simplifying *physical models*, MOSRA is only applicable to circuits with moderate complexity [72]. Academia also attempts to solve the performance problem, [73] reduced data which needs to be processed and [72] employs an offline look up table approach. Unfortunately, neither approach is sufficiently fast as [73] still calculates thousands of data points for a single transistor, while the look up tables for [72] can become unfeasible for complex circuitry evaluated over a wide range of operating conditions. Left without feasible *physical BTI modeling*, circuit designers are forced to employ *empirical models* despite their overestimation.

3.2.2 Massively Parallel PDO Model

To the best of our knowledge, no other degradation model is currently implemented on the GPU. Some models utilize multi-core CPU processing, but the vast majority of the degradation models are single-threaded processes as reliability physicists care the most about accuracy and matching experimental data instead of modeling performance.

Note, that most empirical degradation models are simple enough (single mathematical expressions), that they might be included in multi-core circuit simulations without their own executable/process. However, even in this context, we are not aware of GPU-based circuit simulations which include degradation models like BTI.

3.3 Step Three - Incorporating Aging into Standard Tools

The related work of this step is broken into two subsections for digital and analogue circuits, as these are very different approaches and thus feature very distinct related work.

3.3.1 Digital Circuits

For digital circuits, we can explore standard cell designs and (non-standard cell) transistor designs. We consider the transistor designs as analogue/mixed-signal designs, since both are developed in the SPICE family of circuit simulators (in different operating modes).

3.3.1.1 Worst-Case Aging in Standard Cells

MOSFETs are driven by the gate voltage [8,31]. Thus, input vectors, which determine gate voltages at the MOSFETs, govern BTI-induced degradation [8, 31]. At design-time, circuit designers must define the worst-case input vectors to determine worst-case timing, which guarantees that aging at runtime never causes timing violations. The impact of aging on standard cell delays is not explored with worst-case input vectors. Each paragraph corresponds to one mayor drawback of state of the art.

Existing work relies on the superposition of worst-case transistors, i.e. assumes that worst cell delay is obtained if all transistors are *uniformly* at peak degradation. This superposition seemed to lead to worst-case delay, as state of the art did not consider cell delay in circuit simulations. The authors in [74, 75] modeled cell delay with equations and thus fail to observe the opposition of pull-up to pull-down networks.

Other works consider activity as given (e.g., from a microprocessor simulator). They model the impact of activity on the timing of circuit (e.g., a microprocessor or standard cells) [28, 38, 76–79] [80]. These works target the workload-dependence of aging, i.e. estimate aging under specific input vectors or specific duty cycles per transistor. However, these works provide no worst-case estimation [28,38,76–78,80]. Or [74,79,81] assume that the worst-case estimation is again the uniform worst-case degradation per transistor, i.e. they miss the opposition of pull-down and pull-up network.

Even if circuit simulations are used to estimate delay, there is another challenge. Most related work uses simplified circuit simulations, hiding the behavior of the cell. In fact, a standard cell features parasitic capacitances and resistances, has input signals with different signal slews and drives a variety of load capacitances. All these factors significantly affect the standard cell delay [28, 79] [82] and are therefore considered in commercial cell characterization tools (e.g., Synopsys SiliconSmart, Cadence Liberate). However, [75–78, 83] do not consider these effects, which can lead to significant errors in the standard cell delay estimation [28, 82].

Finding the best-case input vectors for standard cells was the goal of [83], i.e. minimizing aging by maximizing aging recovery in standard cells. However, their approach relies upon simplified aging models, which cannot represent the complex dependency of aging on transistor activity. Additionally, their delay estimations feature neither parasitics nor slews or loads. This has an impact on the input vectors as delays are different under different load capacitances and signal slews (see Fig. 6.6).

3.3.2 Analogue/Mixed-Signal Circuits

For analogue/mixed-signal circuits, the following related work with respect to reliability estimations is considered.

3.3.3 SRAM Reliability Framework

Studying reliability can be performed at different abstraction levels and with different accuracies. At the higher abstraction level, reliability is studied of large circuits (exceeding 100k transistors), which is solely possible by breaking the circuits down to standard cells, but not down to the transistor level. Works like [82, 84, 85] have shown how large circuits like entire microprocessors can be studied by characterizing standard cell libraries under the effects of aging.

This works aims at lower abstraction levels (up to 100k transistors), as circuits are broken down to the transistor level. This allows us to study analogue and mixed-signal circuits, as these cannot be broken down to standard cells. Additionally, circuit simulation on the transistor level feature higher accuracy, as more of the transistor interactions (for example, pull-up versus pull-down networks as shown in [85]) are captured and circuits are not evaluated with abstracted delay and power tables for standard cells. Therefore, instead of taking the workload purely as signal probabilities of standard cell input pins [82, 85], we take individual voltage waveforms per transistor into account. So instead of duty cycle and switching frequency, we have full V_G, V_D, V_S, V_B waveforms which allow the aging models to more accurately consider recovery and thus provide a more accurate degradation value per transistor.

As a representative circuit, we study an SRAM array as SRAM are frequently studied in reliability [86]. However, the majority studies the SRAM memory cells in isolation missing the periphery with SA or WD [73, 86]. Reliability in periphery is mainly reported with isolated SA studies in [87, 88] and a single BTI report in WD with cells in [89]. For the SA, the work in [88] claims workloads from a processor simulator, yet solely records read frequency ("read activation") (f_{read}) and then simulates simple read 0 and read 1 patterns in SPICE to translate read operations to transistor duty cycle (λ_{tran}) and switching frequency f_{sw} *for transistors in the SA alone*.

3.4 Step Four - Accelerating Standard Tools

Accelerating the standard tools is typically performed by the EDA vendors. However, as a proof of concept, we highlight how the general-purpose computing of graphic cards can be utilized to accelerator analogue circuit simulators.

3.4.1 GPU-SPICE

Different approaches exist to evaluate circuit characteristics in large circuits. However, due to the focus on scenarios which demand full SPICE accuracy, no simplifications or structural changes are allowed which result in inaccuracies.

FastSPICE: FastSPICE is a family of SPICE implementations (e.g. commercial FineSim [90]), which simplifies transistor modeling and moves to hierarchical event-driven simulation to achieve large performance speedups and simulate large circuits. Many FastSPICE implementations are multi-threaded (e.g. FineSim [90]) to further enhance performance. However, these structural changes and simplifications result in inaccuracies up to 15% [91]. These inaccuracies are unacceptable in scenarios like security critical designs (e.g. for automotive ISO 26262).

Multi-Core SPICE Implementations: Commercial and public domain SPICE implementations exploit multiple CPU cores to achieve higher simulation speeds. Commercial HSPICE [92], Eldo [93], Spectre [94] and open-source NGSPICE [95] all are multi-threaded or multi-processing (Xyce [96]). Parallelized circuit setup was introduced in [97] and is widely used in commercial tools [92] [94]. These works employ message passing and other compute cluster techniques to employ the massive parallelism found across multiple machines. However, in a single PC the performance insufficient to enable simulations of large circuits (see Section 7.1.5). This work focuses on single desktop PC performance by employing Single-Instruction Multiple Data (SIMD) architectures like GPUs. GPUs are frequently used to accelerate SPICE phases (see the two following paragraphs), since they represent cheap off-the-shelf parallel hardware accelerators and are already available due to their integration into CPUs [98].

GPU SPICE Implementations: Various GPU implementations of SPICE are proposed. TinySpice [99] restructures the SPICE algorithm to optimize the repeated simulation of circuits like standard cells in Monte Carlo variability studies (e.g. for yield analysis). TinySpice breaks these circuits down to look-up tables for the GPU and provides large speed-ups for these small circuits, but is - by design - unsuitable for larger circuits [99].

$$A = L \cdot U \tag{3.1}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \tag{3.2}$$

The majority of state of the art focused on matrix LU-factorization (see Eq. 3.1), a time-consuming preprocessing necessary for circuit matrix solving. Agilent proposed solving the LU-factorization in SPICE on the GPU in 2009 [100]. Similarly, other works [101] [102] [103] [104] [105] focused solely on LU-factorization. The authors in [106] parallelized the device evaluation (a SPICE phase which includes the costly device linearization), as they considered that the bottleneck of SPICE. None of these works targeted circuit setup, which is the key focus of this work. However, all these works are fully compatible with our work and can be used to target the other computationally intense phases of a GPU-based SPICE implementation.

Distinction from CUSPICE: The basis of this work is the public domain simulator NGSPICE [95]. We rely on the initial attempt of a CUDA (parallel programming language for NVIDIA GPUs) implementation, called CUSPICE [107]. This implementation already features the fast and optimized KLU solver [108] [104] and parallel device evaluation [107] [109], so both bottlenecks are not present in our baseline. Both phases are outside of the scope of this work. We integrated our novel circuit setup into CUSPICE and restructured the code to improve performance (CUDA memory management) and solve the convergence issues for large circuits found in the current CUSPICE version (Apr. 2018) as a basis for our own GPU-SPICE [109]. Note, that we do not rely on the specifics of NGSPICE or CUSPICE (data structures, algorithms, etc.) and employ it solely as a proof of concept for our approach.

3.5 Custom Reliability Estimation of Circuits

Various commercial [11, 12] and academic [110–112] tools exist, which perform circuit reliability simulations with SPICE. In fact, for the EDA vendors, SPICE-based reliability analysis with updates in the transistor parameters is the de facto standard approach for analog circuits. These frameworks are comparable to our CARAT aging framework in Section 8.1.3.3, but with the following differences:

1. Our framework uses all voltages and temperatures of all transistors in our aging models (e.g., supporting V_{ds} dependencies).

2. Our framework supports dynamic voltage and frequency scaling, analogue circuits and other effects (e.g. IR-drops) which introduce dynamic voltage changes.
3. Our framework uses defect-centric physics-based degradation models [9] and thus supports recovery and short-term aging.
4. Aging estimation from nanoseconds (short-term aging) to 10 year lifetimes (long-term aging).
5. Modeling and consideration of self-heating
6. Works for arbitrary circuits (anything simulatable in SPICE) without any modifications to the netlist.

Some frameworks do consider a small subset of these points, but the commercial frameworks are severly limiting on the modeling side (simplistic and outdated degradation models) as the focus of EDA vendors is clearly on the SS process corners and static timing analysis for large-scale circuits (see Section 1.4.3). The academic frameworks provide physics-based modeling [112] and multiple phenomena simultaneously [111] but not both.

These frameworks are meant for a before-degradation/after-degradation comparison of smaller analog circuits and mainly study a single phenomenon or simplistic degradation as a whole (e.g., the impact of $\Delta V_{th} = 50\,\mathrm{mV}$ applied uniformly to all transistors). Individual degradation of transistors via multiple degradation phenomena simultaneously as well as dynamic voltage/temperature waveforms or extrapolation towards lifetimes are novel in this work.

4 Step One - Improving Degradation Models

This first step consists of three contributions, which all aim to improve degradation models. Degradation models are typically designed by reliability physicists with the aim to provide accurate modeling (match experimental data as closely as possible) and to consider all dependencies (dynamic temperature/voltage, transistor manufacturing, etc.). These models are not suited for custom reliability estimations as they are developed separately for each phenomenon. However, during operation transistors experience all phenomena simultaneously. Therefore, the first contribution is a unified model, which models BTI, RTN and PV across a wide range of V_{dd} at the same time. This combines the key degradation phenomena to enable modeling of multiple phenomena simultaneously.

Secondly, the recently uncovered variability in degradation phenomena is studied. In addition to manufacturing variability, transistors also experience variability from the degradation phenomena like BTI and RTN. Defect generation and activation is not a deterministic process. Instead, inherit variability both from one transistor to the next, but also within a transistor is a – widely ignored – property of degradation phenomena like BTI. This second contribution models the variability found in defects from the transistor all the way to the circuit level.

Lastly, as a third contribution, the new phenomena self-heating is included in large-scale circuit simulations for the first time. This introduces this newly occurring but absolutely critical phenomenon to large-scale circuit simulations and evaluates its impact on circuit reliability.

4.1 Unified Model of Bias Temperature Instability, Random Telegraph Noise and Process Variation

This section is based on my publication [113].

4.1.1 Reliability is key in Near-Threshold Computing

Advances in semi-conductor technology were achieved with ideal Dennard Scaling, i.e. scaling all technology parameters (e.g. V_{dd}, geometry, dopant concentrations) with the same scaling factor S resulting in higher performance, lower power consumption and lower cost per transistor. But, as the decreasing supply voltage V_{dd} approaches the threshold voltage V_{th} (that does not scale [64]) the channel formation in MOSFETs is weakened, leading to severe performance penalties. Hence, Dennard Scaling was discontinued and voltage was scaled with a smaller scaling factor than used in geometry to maintain performance.

However, recent trends like Internet of Things (IoT) and wearable devices demand energy-efficient on-chip systems in which energy instead of performance is the highest priority. There, maximizing operation times within an energy budget (e.g. a battery) is more important than obtaining high performance, as their tasks have low performance demands (e.g. monitoring heart beats). Intuitively, lowering V_{dd} to reduce dynamic and static power is the best approach. However, lowering V_{dd} results in a severe performance penalty and thus prolonged execution times, ultimately consuming more energy via static power. Fig. 4.1 highlights how the energy per operation is minimal when the on-chip system operates in near-threshold ($V_{dd} \approx V_{th}$). Lowering V_{dd} beyond V_{th} to sub-threshold ($V_{dd} < V_{th}$) prolongs execution times enough to compensate lower dynamic power and increases the overall energy consumption via static power. In super-threshold ($V_{dd} > V_{th}$), quadratic dependence of dynamic power on V_{dd} increases dynamic power resulting in high energy consumption.

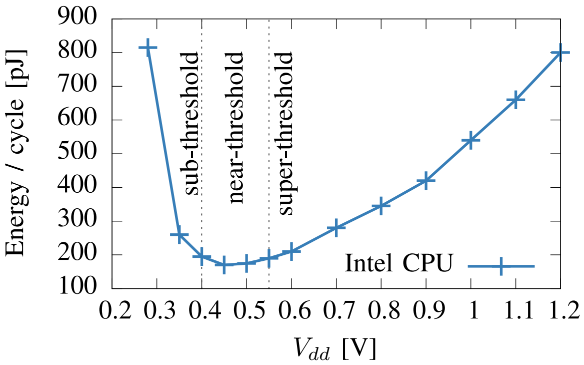


Figure 4.1: Measured energy per operation in Intel CPU, based on [114]. Near-threshold is up to 4.7x more energy efficient per operation compared to super-threshold and sub-threshold.

Near-Threshold Computing (NTC) offers a trade-off between short execution times to limit static power and low V_{dd} to limit dynamic power for maximum energy efficiency (see Fig. 4.1) [114]. As a direct result, the semiconductor industry started designing NTC processors, e.g. Intel [114, 115] and ARM [116] highlighting the advent of NTC.

While the energy efficiency is the key advantage of NTC, decreased reliability is its key disadvantage. Lowering V_{dd} decreases the resiliency of the on-chip system, magnifying the importance of keeping reliability degradations at bay. In fact, Intel despite employing hardened memory cells (10T instead of 6T SRAM) in its near- and sub-threshold processor, they still could not lower $V_{dd}(memory)$ as low as $V_{dd}(logic)$ as otherwise the data could not be stored reliably [114].

Maintaining reliability is more challenging in near-threshold than in super-threshold due to the lowered V_{dd}. In short, aging phenomena, like Bias Temperature Instability (BTI) scale down with V_{dd}, while manufacturing variability or noise phenomena, like Process Variation (PV) or Random Telegraph Noise (RTN) do not scale with V_{dd}. PV & RTN degradations in STC, which do not result in errors, occur equally in NTC and may now be sufficient to introduce errors, due to lower resiliency of the circuit at lower V_{dd}. Therefore, PV & RTN are key phenomena in NTC which must be taken into account.

NTC circuits do not exclusively operate in near-threshold. In fact, if the performance demand increases the V_{dd} can be increased in order to boost performance. For example, in Internet of Things, data is continuously collected, but periodically the data must be processed (e.g. interpreted & classified) and the result transmitted to the server. While near-threshold is sufficient for data collection to conserve energy, data processing has high performance demands and hence must be operated in super-threshold. Both Intel [114, 115] and ARM [116] designed their processors for operations across a wide V_{dd} range from super- to near-threshold (i.e. *STC-to-NTC circuits*), to be able to react to changing performance demands. *Therefore, we do not focus solely on NTC. Rather, we aim to model reliability for STC-to-NTC circuits, i.e. to consider a wide* V_{dd} *range from STC to NTC, to ensure reliability across the operation modes.*

Our novel Contributions within this section are:

1. A unified RTN & BTI model, modeling the shared physical processes behind RTN & BTI in order to model both phenomena simultaneously. It is capable of modeling both phenomena from super-threshold to near-threshold due to experimental calibration of the model across the entire voltage range.

2. Estimating reliability for *STC-to-NTC circuits* linking the unified model to a PV model, to capture the impact of PV on RTN & BTI jointly. It enables circuit designers to employ *probabilistic* guardbands to deal with *joint impact* of the key reliability degradations of STC and NTC: BTI, RTN and PV.

4.1.2 Reliability Degradation: PV, BTI and RTN

In the following, we briefly introduce the key reliability degradation phenomena in STC and NTC along with their underlying physical causes.

4.1.2.1 Process Variation (PV)

Semiconductor manufacturing consists of various lithographic processes involving optics, which operate at their resolution limits in current nanometer technologies. This introduces variations in the manufacturing of the geometric structures, e.g. variations of the width W and length L of transistors, called Process Variation (PV). According to [44], PV can be modeled as Gaussian distributions. These distributions only depend on the precision of the lithographic process itself and not on any run-/design-time parameters.

Other variability phenomena like random dopant fluctuation (RDF) or variations in the interconnects are not within the scope of this work. They do not physically interact (amplify/mitigate) with any of our targeted phenomena (RTN, BTI & PV) and therefore can be independently modeled, e.g. as distribution of initial conditions or a static offset (average- or worst-case) to consider them in our *STC-to-NTC circuits* reliability estimation. For instance, in [61] we showed how to model the impact of RDF on BTI, while a separate RTN model was employed. Unlike our previous work, here we employ a unified BTI/RTN model and analogously the impact of RDF on both BTI & RTN can be modeled through a distribution of the initial threshold voltage V_{th} of the transistor. Our model takes this different initial V_{th} into account (τ_c and τ_e modeling details in [37] used in eq. 4.2) and will amplify/mitigate BTI & RTN accordingly.

4.1.2.2 Bias Temperature Instability (BTI)

For more details on BTI in general, please refer to the BTI background Section 2.2.1.

Even if manufacturing (PV) is well-controlled, aging phenomena pose another threat to the reliability of a circuit. Aging phenomena are physical processes which induce degradations during run-time of a circuit. The dominant aging phenomena is called Bias Temperature Instability (BTI) [8] and originates from charging *material defects* (e.g. oxygen vacancies in SiO_2 / HfON/ HfO_2 lattice as well as broken Si-H bonds near the dielectric-channel-interface) in the gate dielectric of a transistor. If a vertical electric field is applied over the gate dielectric, these defects are excited to capture a carrier, gaining an electrical charge. Capturing a carrier changes the electrically neutral defects into electrically charged locations in the gate dielectric and thus weakens the electric field reaching the channel. This weakens the formation of the channel below the gate and thus *manifests* itself as a threshold voltage shift $\Delta V_{th}(BTI)$. As soon as the electric field is reduced or switched off, defects are excited to emit their carrier and thus regain electrical neutrality reducing the BTI-induced ΔV_{th}. Note that BTI defects may influence other MOSFET's parameters like carrier mobility and sub-threshold slope [117]. However, we focus in this work on ΔV_{th} as it is the most dominant degradation.

Theories: Traditionally two theories are used to explain BTI based upon different interpretations of the underlying physical origin of BTI. These models are traditionally called reaction-diffusion (RD) [8], [34] and trapping/detrapping (TD) [31], [37]. In recent years, both RD & TD started to incorporate each other's physical mechanisms in their models, e.g. by the incorporation of ultra-fast trapping in oxide vacancies into the RD model [34]. Our unified model combines RTN & BTI based upon their shared physical origin in charging/discharging defects and is agnostic to defect types (oxide vacancies, interface traps, etc.) as all defects capture and emit carriers equally via quantum tunneling. Therefore, our model is applicable to both RD and TD theories in their respective state-of-the-art.

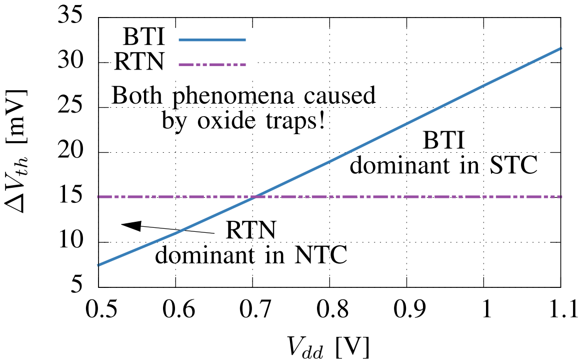


Figure 4.2: BTI and RTN degradation at different V_{dd} viewed as separated phenomena. Note the lowered degradation of BTI at lower V_{dd}, while RTN is unaffected by V_{dd}. RTN is the dominant phenomena in NTC, while BTI is the dominant phenomena in STC. As both phenomena are modeling the same physical origin, defect modeling across a wide voltage range becomes indispensable for the design of reliable NTC-to-STC circuits. Simulation is performed under $T = 125°C, t_{on} = 1000s, W = 150nm, L = 45nm$

4.1.2.3 Random Telegraph Noise (RTN)

See background Section 2.3.2 for details about RTN and its delineation from BTI.

4.1.2.4 Defect Modeling

Shared Stimuli: Traditionally, BTI and RTN were modeled and studied separately, which lead to different focusses from the respective research. We studied the stimuli of BTI and reported in descending strength on gate-source voltage V_{gs}, temperature T, transistor on-/off-ratio d and switching frequency f_{switch} [31], [15], [118]. Increasing these stimuli, i.e. higher voltage, higher temperature, larger on-/off-ratio and lower switching frequency increases ΔV_{th}.

For RTN, the magnitude of RTN-induced degradation is reported to depend on transistor width W and transistor length L [119], [120]. Larger transistors (in either or both dimensions) will lead to smaller ΔV_{th}.

Both BTI and RTN describe behavior of defects, so in order to model them as accurately as possible, we take all these stimuli simultaneously into account. From a physical perspective, RTN stimuli also apply to BTI and vice versa. The impact of defects (i.e. both RTN and BTI) thereby depends on gate-source voltage V_{gs}, temperature T, transistor on-/off-ratio d, switching frequency f_{switch}, transistor width W and transistor length L. In section 4.1.5.1 we describe the modeling in detail.

Impact PV on RTN/BTI: PV shifts W and L and the magnitude of RTN-induced degradation $\Delta V_{th} \propto \frac{1}{W \cdot \sqrt[3]{L}}$ [120]. Therefore, the induced shifts of PV have a direct impact on ΔV_{th} with a stronger impact in near-threshold than in super-threshold [119].

4.1.3 Reliability in STC and NTC

In the beginning of this section, we discuss reliability in NTC, followed by a brief discussion of well-known reliability challenges in STC. Operating in NTC instead of STC reduced degradations of reliability phenomena which scale with V_{dd}. Yet, at the same time, resiliency of a circuit against any induced degradations decreases as it also scales with V_{dd}.

Lower degradation at low V_{dd}: BTI is one of the key phenomena at super-threshold [31], [8], yet BTI depends strongly on V_{dd} (e.g. $\Delta V_{th}(BTI) \propto (V_{dd})^{4.3}$ [8]), i.e. lowering V_{dd} significantly lowers BTI-induced degradation. In contrast, PV induces the same variation in W and L regardless of V_{dd} and similarly the magnitude of RTN-induced degradation does not depend on V_{dd}.

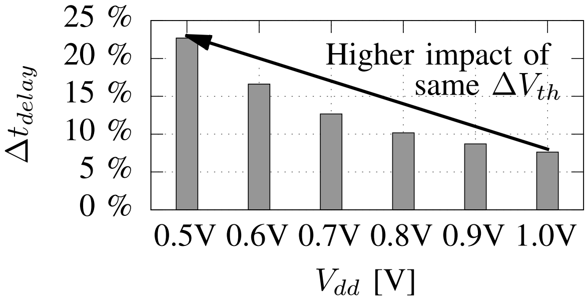


Figure 4.3: SPICE simulations of a ring oscillator with $\Delta V_{th} = 10mV$ applied to each transistor. The shift in propagation delay increases up to 4x as V_{dd} scales down from STC (1.0 V) to NTC (0.5 V). This demonstrates that the susceptibility to degradations is higher at low V_{dd} as the same degradation has higher impact.

Lower resiliency at low V_{dd}: The first-order approximation of the delay of a transistor (t_{delay}), i.e. equation 4.1 shows, that lowering V_{dd} has a severe impact on t_{delay}:

$$t_{delay} \propto \frac{1}{I_D} \text{ with } I_D \propto (V_{dd} - V_{th} - \Delta V_{th})^2 \tag{4.1}$$

In circuit simulations, Fig. 4.3 highlights how the same degradation ($\Delta V_{th} = 10mV$ in all transistors) has a larger impact at lower voltages. At just 7% delay increase at $V_{dd} = 1.0V$ the impact reaches 23% at $V_{dd} = 0.5V$, which represents a 4x increase. This confirms how the trends from the first-order approximation still hold in simulations based upon the BSIM compact model [14].

Reliability challenges in NTC: The degradations induced by PV, RTN and BTI have a considerably higher impact on the reliability in near-threshold than in super-threshold due to aforementioned lower resiliency in NTC. BTI scales down with ${V_{dd}}^{4.3}$ [8], mitigating its higher impact at lower voltages which scales with ${V_{dd}}^2$. In contrast to BTI, PV & RTN exhibit no reduction and thus lowering V_{dd} magnifies their overall impact on reliability by up to 4x (see Fig. 4.3), which makes maintaining reliability significantly more challenging.

Reliability challenges in STC: In super-threshold, BTI may temporarily induce high levels of degradations due to the high V_{dd}, which we showed in our work [118]. RTN may worsen this degradation by randomly increasing already high levels of degradations. Furthermore, PV may create weak (small) transistors with low resiliency against these degradations. Therefore, at super-threshold it is important to model the deterministic (BTI) and random (RTN) behavior of defects within PV-shifted transistors.

Reliability challenges in STC-to-NTC: In our work [121] we showed how the transition between different V_{dd}'s can result in transient errors, as the high degradation levels from super-threshold may meet the low resiliency of near-threshold. This implies to switch from STC → NTC slow enough to let BTI recover [121] and potentially employ our presented A-GEAR technique [121] to ensure reliability. NTC → STC switches (increasing V_{dd}) cannot result in transient errors, as the resiliency immediately increases while degradations need some time to increase [121].

In all three operation modes, i.e. STC, NTC and STC → NTC switches accurate degradation models are indispensable to estimate the degradations. They should be able to model the entire voltage range from STC to NTC as well as sub-microsecond V_{dd} transitions to enable the design of reliable circuits. For PV this is given, as it does not scale with V_{dd}, however no defect modeling is currently capable of modeling STC, NTC and STC → NTC switches.

4.1.4 Distinction from existing work:

1. Considering the impact of geometry (PV/technology scaling) on the unified BTI/RTN model.
2. Reliability modeling and guardband estimation suited from super-threshold to near-threshold, i.e. a wide voltage range.
3. Probabilistic guardbands for circuits with error correction and transistor hardening.
4. Evaluating reliability of current design (per transistor/entire circuit) for given x_j, i.e. tolerable ΔV_{th} per transistor.

This work partially bases itself on two of our own previous works [31] and [15]. New contributions on top of previous work in [31] includes considering the probabilistic nature of defects (i.e. RTN) in an unified RTN/BTI model. As our target is Internet-of-Things in which the operation can switch from STC and NTC and vice versa, the model is calibrated across the corresponding wide voltage range (e.g. from 1.2V to 0.4V). Additionally, the impact of PV is now taken into account, as it is one of the major reliability challenges in NTC. Furthermore, instead of being able to model solely digital circuits, our new implementation within algorithm 1 can model dynamic behavior for both voltage and temperature, which is imperative for STC/NTC switches, in which voltage and temperature vary significantly. Our SRAM evaluation is based upon [15], yet features probabilistic guardbands due to the probabilistic estimations of our unified model. Furthermore, an industrial guardband is used as a reference to indicate if a design is reliable or not. Lastly, the impact of the design choices of a designer, namely the transistor size (W, L) or employment of ECC in SRAM were added to the SRAM evaluation.

4.1.5 Our Unified BTI & RTN Model

In order to model RTN & BTI in a unified model, we discuss defects in detail, as these are the underlying physical processes of both phenomena.

4.1.5.1 Defects: Physics of BTI and RTN

Capturing/emitting carriers in defects are the underlying processes behind RTN and BTI. Each defect (regardless if oxide trap/interface trap/other) can be characterized by experimentally observing three macroscopic parameters [37]:
a) its capture time τ_c, i.e. the longest time to capture a carrier from the channel in a unoccupied defect.
b) the emission time τ_e, the longest time until the carrier is emitted back to the channel from an occupied defect.
c) η, the induced shift in threshold voltage if that specific defect has captured a carrier and hence electrically charged. While η solely depends on the horizontal location of the defect within the gate dielectric [51], capture and emission times are functions of temperature T and voltage V_{gs} [37]. Thus, η is not correlated to τ_e and τ_e, as it has been established in [122]. For example, if T rises, both capture and emission times are decreased due to more thermal energy for carriers to reach an defect. Hence, a defect o can be defined as:

$$o \in D(\tau_e, \tau_c) : (\tau_{c,o}(T, V_{gs}), \quad \tau_{e,o}(T, V_{gs}), \quad \eta_o) \tag{4.2}$$

$D(\tau_e, \tau_c)$ is the defect distribution within the transistor, i.e. the defect map with τ_e as x-axis and τ_c as y-axis. Mathematically it can be expressed as log-normal bi-variant distribution:

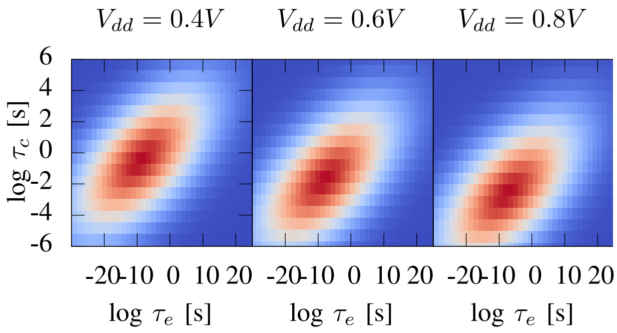


Figure 4.4: Our measured defect distribution at $V_{dd} = 0.4V/0.6V/0.8V$ and $25°C$. A darker red corresponds to a higher defect concentration. The distributions shift down (shorter capture times (τ_c)) when increasing the voltage. Shorter τ_c result in higher ΔV_{th} for a given stress time as more defects will capture a carrier in that time-frame. Emission time τ_e is barely affected, i.e. the probability to emit carriers $P_{emission}$ remains identical.

$$\begin{aligned} D(\tau_e, \tau_c) &= \frac{\exp\left(-\frac{1}{2(1-\rho^2)}(A^2 - 2\rho AB + B^2)\right)}{2\pi\tau_e\tau_c\sigma_{\tau_e}\sigma_{\tau_c}\sqrt{1-\rho^2}} \\ \text{with } \rho &= \frac{\exp\left(\rho_N\sigma_{\tau_e}\sigma_{\tau_c}\right) - 1}{\sqrt{(\exp\left(\sigma^2_{\tau_e}\right) - 1)(\exp\left(\sigma^2_{\tau_c}\right) - 1)}} \\ A &= \frac{\ln(\tau_e) - \mu_{\tau_e}}{\sigma_{\tau_e}} \\ B &= \frac{\ln(\tau_c) - \mu_{\tau_c}}{\sigma_{\tau_c}} \end{aligned} \tag{4.3}$$

Fig. 4.4 shows an example of an experimentally extracted defect distribution and how the distributions shift with rising voltage V_{gs}.

In order to determine the occurring ΔV_{th} in a transistor, the occupancy of each defect needs to be determined. As capturing and emitting carriers are probabilistic processes, we define occupancy probabilities P_{occ} for each defect. P_{occ} depends on the on- and off-states of a transistor, which can be conceptionally expressed as:

$$\begin{aligned} \text{BTI: } \tau_c \leq t_{on} &: P_{occ} \rightarrow 1 \\ \tau_c > t_{on} &: P_{occ} \rightarrow \text{ previous state} \\ \tau_e \leq t_{off} &: P_{occ} \rightarrow 0 \\ \tau_e > t_{off} &: P_{occ} \rightarrow \text{ previous state} \end{aligned} \tag{4.4}$$

BTI: BTI is the monotonic behavior of defects, which is shown in Fig. 2.21. If the on-time t_{on} is longer than the capture time τ_c for a defect, than that defect has a high probability that it is occupied, i.e. P_{occ} tends towards ($\rightarrow$) 1. Either the defect was unoccupied and captured a carrier, since $\tau_c \leq t_{on}$ or it was already occupied and did not emit the carrier. Similarly, if the off-time t_{off} is longer than emission time τ_e, the defect is likely unoccupied, i.e $P_{occ} \rightarrow 0$. For all defects with $\tau_c > t_{on}$ or $\tau_e > t_{off}$ the occupancy probability has not yet changed and remains in their previous state. The full description for P_{occ} in the context of BTI in digital circuits, according to [31], is as follows:

$$\begin{aligned}
&\text{Stress:}\\
&P_{occ}(t) = P_{occ}(t_i) + \left(\frac{\tau_e}{\tau_e + \tau_c} - P_{occ}(t_i)\right) \cdot \left(1 - e^{\frac{t-t_i}{\tau_{sr}}}\right) && (4.5)\\
&\text{Recovery:}\\
&P_{occ}(t) = \frac{\tau_e}{\tau_e + \tau_c} + \left(P_{occ}(t_i) - \frac{\tau_e}{\tau_e + \tau_c}\right) \cdot \left(e^{\frac{t-t_i}{\tau_{sr}}}\right) && (4.6)\\
&\text{with } \tau_{sr} = \frac{1}{\frac{1}{\tau_e} + \frac{1}{\tau_c}} \quad \tau_c = \tau_c(T,V) \quad \tau_e = \tau_e(T,V)
\end{aligned}$$

with t_i as the point in time of the i-th transition between on and off state and thus stress $\leftrightarrow$ recovery, resulting in a stress $t_{on} = t - t_i$ respectively recovery time $t_{off} = t - t_i$. Equations 4.5 and 4.6 do not depend on the time-step size Δt or induced degradation ΔV_{th}. They only depend on the stress time (t_{on}) and recovery time (t_{off}) and t the current point in time. Note, that only the previous occupancy probability P_{occ} at the transition from stress $\leftrightarrow$ recovery is used in the calculation, *but not the occupancy state itself.* Therefore, P_{occ} depends on (stress-/recovery-) time $t_i - t$, voltage V_{gs} and temperature T and none of the three stimuli are affected by previous ΔV_{th}, occupancy state $occ(o)$ or Δt.

Even though BTI is estimated with probabilistic models, we still consider it a deterministic monotonic behaviour, i.e. we expect a defect to be occupied if $\tau_c \leq t_{on}$. As $P_{occ} \to 1$, i.e. close to 1, this is very likely, but not guaranteed. By our previous definition, if our expectation of an occupied defect is not met, we call it RTN. So the difference in ΔV_{th} due to the expectation of an occupied defect and the actual unoccupied defect will be RTN.

RTN: RTN is the random component of the defect modeling, acting as noise on top of the monotonic BTI-induced shifts. Capturing and emitting carriers is a probabilistic process dependent on time, i.e. can be expressed as a probabilistic function of time. These probabilities are directly linked to capture and emission probabilities within a short time frame ($\Delta t \ll \tau_c, \tau_e$) [31]:

$$P_{capture} = \frac{\Delta t}{\tau_c} \tag{4.7}$$

$$P_{emission} = \frac{\Delta t}{\tau_e} \tag{4.8}$$

Defects which have similar capture and emission times (i.e. $\tau_c \approx \tau_e$) have occupancy probabilities near to 50% (i.e. $P_{occ} \approx 0.5$), as:

$$\begin{aligned}
&\tau_c \approx \tau_e\\
\Rightarrow &P_{capture} \approx P_{emission}\\
\Rightarrow &P_{occ} \approx 0.5
\end{aligned} \tag{4.9}$$

Defects which have higher capture than emission times (i.e. $\tau_c > \tau_e$) have low occupancy probabilities, as they capture rarely and emit frequently. Defects with lower capture than emission times (i.e. $\tau_c < \tau_e$) have high occupancy probabilities, as these defects will capture a carrier frequently but emit rarely. Capture τ_c and emission times τ_e modeling itself is completely independent of t_{on} or t_{off} and only depends on the temperature T and voltage V_{gs} during Δt.

Delineation of BTI and RTN: Conceptionally, BTI and RTN describe the same physical processes with eq. 4.5 & 4.6 for continuous time and eq. 4.7 & 4.8 for discrete time steps. Their delineation is deterministic vs. random/unpredictable behavior. If the transistor is on, i.e. V_{gs} is high, τ_c is low while τ_e is largely unaffected (see Fig. 4.4). This results in high $P_{capture}$ with low $P_{emission}$ and therefore high P_{occ} for each defect, i.e. we expect that most defects will be occupied. This expectation of high degradation is BTI, while the deviation from it (e.g. a lower occurring degradation) is RTN:

$$\begin{aligned}
\Delta V_{th} &= \Delta V_{th,expected.value} - \Delta V_{th,deviation} && (4.10)\\
&= \sum_{o \in D} P_{occ}(o) \cdot \eta_o - \left(\sum_{o \in D} (P_{occ}(o) - occ(o)) \cdot \eta_o \right) && (4.11)\\
&= \Delta V_{th}(BTI) - \Delta V_{th}(RTN) && (4.12)
\end{aligned}$$

A defect contributes with its η to $\Delta V_{th,total}$ if it is occupied:

$$\Delta V_{th,total} = \sum_{i=1}^{o} \Delta V_{th}(i) \qquad (4.13)$$

$$\begin{aligned}
\text{with} \quad & o \in D(\tau_e, \tau_c) \wedge occ(o) = 1 : V_{th}(o) = \eta_o\\
\text{and} \quad & o \in D(\tau_e, \tau_c) \wedge occ(o) = 0 : V_{th}(o) = 0
\end{aligned}$$

Impact of Geometry on Defects: As the geometry of a transistor shrinks, the number of defects decreases. Authors of [120] report the number of defects $|D(\tau_e, \tau_c)| \propto W \cdot L$. However, despite the reduction of the number of defects, the overall degradation ΔV_{th} increases. According to [119], in flash (floating gate) transistors $\Delta V_{th}(RTN) \propto \frac{1}{\sqrt{W \cdot L}}$ in super-threshold and $\Delta V_{th}(RTN) \propto \frac{1}{W \cdot \sqrt{L}}$ in near-threshold. Therefore, if the number of defects decreases while total degradation increases, the induced degradation per defect η must increase. The measured data in [123] provides more insight and reported for planar $45nm$ from super- to near-threshold the shift in the 95-percentile of RTN and the mean ($\overline{\eta_o}$):

$$\begin{aligned}
\text{95-percentile: } \Delta V_{th}(RTN) &\propto \frac{1}{W \cdot \sqrt[3]{L}} && (4.14)\\
\forall o \in D(\tau_e, \tau_c) : \quad \overline{\eta_o} &\propto \frac{1}{W \cdot \sqrt{L}} && (4.15)
\end{aligned}$$

PV: The circuit design defines W and L for each transistor, nonetheless both are also shifted randomly in manufacturing (PV). In fact, W and L are correlated distributions. Therefore, we can model the W and L as two normal distributions with different means (μ is the desired width and length the transistor) but the same variance (σ is shift due to manufacturing) as lithographic processes have a 1:1 correlation in X- and Y-direction. By modeling η_o as a distribution, by substituting W and L with their distributions in eq. 4.15, we can consider the impact of PV on the defect (η of each defect) and hence jointly on BTI and RTN. Thus each defect obtains its unique η_o from the exponential distribution shifted by the width and length of the transistor the defect is located in.

Defect Interactions: If a defect is occupied, it contributes its η to the total ΔV_{th} by reducing the electric field over the gate dielectric resulting in a weaker channel, as its formation is hindered. To explain if the reduced electric field E_{ox}, results in less stimuli for other defects, we employ our TCAD simulation in Atlas. As illustrated in Fig. 4.5, each occupied defect reduces E_{ox} *locally*, i.e. in a confined area, which resembles a butterfly in a cross-section view of a transistor. Therefore, only if two defects are spatially close to each other, they interact and reduce their local E_{ox} and η slightly, as their individual impact on the field is slightly masked by the other defect. However, these interactions occur infrequently with the few defects (< 10) in current nano-scale transistors and even in the rare case that two defects align perfectly these interactions are $< 1mV$. Additionally, these interactions become less likely

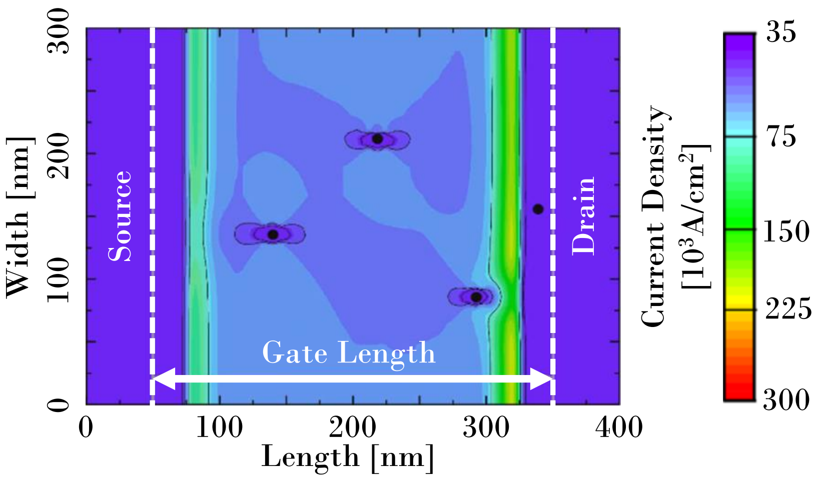


Figure 4.5: Our TCAD Simulation in Atlas with defects placed in gate dielectric to study defect interactions. The bottom left defect partially hinders the current flow to the bottom right defect, therefore reducing its η slightly ($< 1mV$), while other parameters like τ_c, τ_e are unaffected. For the few defects in current nano-scale transistors, interactions between defects can be neglected as it is unlikely they appear in proximity to each other, resulting in negligible interactions. In each following technology node, the number of defects will further decrease, which is why we neglected defect interactions in our unified BTI/RTN model.

in each new technology node as the number of defects further decreases, which is why defect interactions were neglected in our unified BTI/RTN model. *Therefore, occupied defects do not measurably influence the occupancy probability P_{occ} of other defects nor their impact η.*

4.1.5.2 Implementation of our Unified Model

In order to simulate BTI and RTN, we model defects and follow our previous definitions, i.e. either capture carriers randomly (RTN) or deterministically based upon the activity of the electric field (t_{on} and t_{off}) within the transistor (BTI). We described the modeling of a ΔV_{th}-waveform in detail in algorithm 1 and the following paragraph.

Model ΔV_{th}-Waveform (Algorithm 1): First the number of defects (n) is randomly determined based upon an scaled exponential distribution. Then each defect is placed within the defect distribution D map following the defect density (color saturation in Fig. 4.4) and we randomly assign a $\eta_o = \Delta V_{th}(o)$ for each defect o according to a scaled exponential distribution. The scaling factor is $\frac{1}{W \cdot \sqrt[3]{L}}$ with W and L normalized to their references (i.e. W_{ref} and L_{ref}, the width and length of the transistor the model was calibrated at) in order to incorporate the impact of transistor dimensions on the induced degradation due to charged defects. For every time-step t_m and every defect o we obtain a random number $rand$ following a uniform distribution. This number is compared against the capture and emission probability to model the random behavior of defects (RTN). However, an unoccupied defect cannot emit a carrier and an occupied defect cannot capture another carrier, so this comparison against the random number must be performed based upon the current occupancy of the defect ($occ(o_n, t_{m-1})$). If the defect was unoccupied and the $rand$ is above the capture probability $P_{capture}$, then the defect captures a carrier and thus contributes its $\eta_o = \Delta V_{th}$ to the overall degradation. If the defect was occupied and the $rand$ is above the emission probability $P_{emission}$, then the defect emits a carrier and thus does not further contribute its $\eta_o = \Delta V_{th}$ to the overall degradation. In all other cases, the occupancy does not change, i.e. still occupied defects contribute their η_o, while still unoccupied defects do not. After all occupancies are determined, overall degradation within a transistor is the sum of all current (at t_m) occupied defects($occ(o_n, t_{m-1}) = 1$), which contribute their η_o, i.e. their individual $\Delta V_{th}(o_n)$ to the overall $\Delta V_{th,total}$.

Note, that while the BTI model, which forms the foundation of our unified model, simulates the induced $\Delta V_{th,total}$ for an *average* transistor, i.e. average number of defects with average parameters (average τ_c, τ_e, induced V_{th}) [31], while our unified model calculates a random occurring degradation to mimic the actual probabilistic behavior of RTN & BTI.

Algorithm 1 Create ΔV_{th} waveform with unified model

```
Require: $W, L, T, V_{gs}$
Gamble $n$ from $P(X = k) = \frac{\lambda^k \cdot e^{-\lambda}}{k!} \cdot \frac{1}{W \cdot \sqrt{L}}$
Gamble $\tau_e, \tau_c$ from $D(\tau_e, \tau_c)$                          ▷ See eq. 4.3
Place $n$ defects $o_n$ in $D(\tau_e, \tau_c)$
Gamble $\eta_o$ for each $o \in D(\tau_e, \tau_c)$ with
$P(X = k) = \lambda \cdot e^{-\lambda \cdot k} \cdot \frac{1}{\frac{W}{W_{ref}} \cdot \sqrt[3]{\frac{L}{L_{ref}}}}$
for every time-step $t_m$ do
    for each $o_n$ in $D(\tau_e, \tau_c)$ do
        Gamble $rand$ for each $o \in D(\tau_e, \tau_c)$ based upon Uniform distribution
        if $occ(o_n, t_{m-1}) = 0$ then
            if $P_{occ}(o_n) \leq rand$ then                          ▷ See eq. 4.5 & 4.6
                Set $occ(o_n, t_m) = 1$                               ▷ Captured a carrier
                Set $\Delta V_{th}(o_n) = \eta_{o_n}$                 ▷ Contribute to $V_{th}$
            else
                Set $occ(o_n, t_m) = 0$                               ▷ Remain neutral
                Set $\Delta V_{th}(o_n) = 0$                          ▷ Do not contribute to $V_{th}$
            end if
        else
            if $P_{occ}(o_n) > rand$ then
                Set $occ(o_n, t_m) = 0$                               ▷ Emitted a carrier
                Set $\Delta V_{th}(o_n) = 0$                          ▷ Stop contributing to $V_{th}$
            else
                Set $occ(o_n, t_m) = 1$                               ▷ Stay charged
                Set $\Delta V_{th}(o_n) = \eta_{o_n}$                 ▷ Contribute to $V_{th}$
            end if
        end if
        Set $\Delta V_{th,total}(t_m) = \sum_{i=1}^{o_n} \Delta V_{th}(i)$
    end for
end for
```

Algorithm 2 Obtain $\Delta V_{th}(W, L, P_{func})$ with unified model

```
Require: $W, L, P_{func}, h$
for $h$ transistors $tran$ do
    Gamble $n$ from $P(X = k) = \frac{\lambda^k \cdot e^{-\lambda}}{k!} \cdot \frac{1}{W \cdot \sqrt{L}}$
    Place $n$ defects $o_n$ in $D(\tau_e, \tau_c)$
    Gamble $\eta_o$ for each $o \in D(\tau_e, \tau_c)$ with
    $P(X = k) = \lambda \cdot e^{-\lambda \cdot x} \cdot \frac{1}{\frac{W}{W_{ref}} \cdot \sqrt[3]{\frac{L}{L_{ref}}}}$
    Set $\Delta V_{th}(o_n) = \eta_{o_n}$                             ▷ Contribute to $V_{th}$
    Set $\Delta V_{th,total}(tran) = \sum_{i=1}^{o_n} \Delta V_{th}(i)$
end for
Calculate Cumulative density function $P(\Delta V_{th} = x)$ with $P \cdot \frac{h}{100} = |\{tran : \Delta V_{th,total}(tran) \geq x\}|$
Get $\Delta V_{th}(W, L, P_{func})$ with $P(\Delta V_{th}) \leq P_{func}$
```

Switches from occupied to unoccupied states and vice versa can only occur at simulation time steps t_m. However, intermediate state changes are *irrelevant* for the simulation, as these intermediate states are randomly chosen (depending on the comparison between P_{occ} and $rand$) and have no impact on the result of *next* time-step (i.e. occupancy state is not an input to the next time steps (see BTI section in 4.1.5.1)). Each t_m (or change in V_{gs}, T) P_{occ} is updated in order to correctly update the occupancy states of defects. New random numbers are compared against the updated P_{occ} resulting in different chances for occupancy and ultimately estimate ΔV_{th} at t_m.

In order to design reliable circuits, it is necessary to know the highest occurring degradation $max(\Delta V_{th,total})$. With $max(\Delta V_{th,total})$ known, a designer can provide sufficient resiliency to tolerate the induced $max(\Delta V_{th,total})$ in order to prevent errors. As defects are probabilistic, $max(\Delta V_{th,total})$ depends on the probabilistic reliability constraint, i.e. how much occurring BTI & RTN (in terms of %) must be tolerated (see Fig. 4.10). By definition

100% reliability is unobtainable, as in theory the degradation tends towards infinity. Algorithm 2 describes the modeling of $max(\Delta V_{th,total})$ in detail.

Obtain $\Delta V_{th}(W, L, P_{func})$ (Algorithm 2): Transistor width W, transistor length L, the probabilistic reliability constraints and the sample size (number of Monte Carlo samples) h are given. For h number of transistors we perform the following actions. First, we get the total number of defects n in that individual transistor based upon an exponential distribution. Then each defect o is placed based upon the density of defects in the τ_c-τ_e-space based upon the defect distribution D. Each defect o is then assigned a induced degradation η_o according to a scaled exponential distribution. Like before, the scaling factor is $\frac{1}{W \cdot \sqrt[3]{L}}$ based upon a reference in order to incorporate the impact of transistor dimensions on defects. In order to estimate the maximum degradation, all defects o are assumed to be occupied, i.e. $\forall o \in D(\tau_e, \tau_c) : occ(o_n) = 1$. This enforces the contribution of every defect to the overall degradation in the transistor ($\Delta V_{th}(o_n) = \eta_{o_n}$), i.e. the worst-case degradation. The contribution of all defects are then summed, to obtain the overall degradation for transistor $tran$. To estimate the cumulative density function, the probability is calculated, that a randomly chosen transistor (from the set of h transistors) has at least an induced ΔV_{th} of x. With the cumulative density function known, we can reverse the operation and look up which ΔV_{th} must be tolerated if a reliability constraint (P_{func}) is given, e.g. $\Delta V_{th} = 17mV$ for $P_{func} = 0.95$. As this is a Monte Carlo simulation, h should be reasonably large (i.e. $h \gg 100$) for accurate results.

The aforementioned model is a unified BTI and RTN model. The physical processes are identical (capturing/emitting carriers in defects) [45] and only the stimulus is different (on/off-times for BTI, randomness for RTN), which lends itself to a unified model [47,60]. By modeling the impact of transistor geometry and the random shifts in geometry on defects, the unified model incorporates the impact of PV on jointly BTI and RTN.

4.1.5.3 Parameter Extraction for Unified BTI/RTN Model

Our parameters for our defect modeling were obtained from a non-commercial high-κ semiconductor technology in our lab. We used the BTI parameters from our previous work [73,118], which can also be used to describe RTN as they model defects in general. In order to calibrate the randomness of RTN, we performed RTN measurements. However, in order to simulate current NTC circuits, we employed modeling parameters from [123] for newer $45nm$ technology in our unified model.

The parameters which need to be experimentally extracted are: Exponential distribution of n, Defect distribution D (per temperature T and voltage V_{gs}), exponential distribution of η (for different W, L) as well as randomness for RTN (i.e. the distribution type and range of the random variable $rand$). These parameters were extracted with our methodology explained in [124].

4.1.6 Designing Reliable Circuits

Designing reliable circuits incorporates estimating BTI/RTN/PV-induced degradations (e.g. with the unified model) and changing the design appropriately to meet a reliability constraint, e.g. a given probability of failure P_{fail}. While aging and noise occur at the physical level, a circuit designer cannot comprehend or handle details of reliability physics. Therefore, our model takes the transistor parameters (PTM model [3]) with the transistor stimuli of BTI/RTN (T, V_{gs}, on-/off-ratio) and returns a set of degraded transistor parameters (see [15,61] for details). These transistor parameters are then used by the compact model of the transistor (e.g. BSIM[1]) to alter all parameters, which depend on the degraded parameters (e.g., mobility as a function of ΔV_{th}). In this manner, all the low-level details of RTN, BTI and PV remain hidden, while the circuit designer can employ a fully spice compatible degraded parameter set in his circuit simulations to explore the impact of these degradation on his design.

[1] Note, that any compact transistor model can be used in which ΔV_{th} can be altered.

4.1.6.1 Methodology

Stimuli: First the stimuli for the degradation phenomena are determined with our previously published techniques. Software [15] or hardware-based stimuli extractions [73] can be employed to obtain the stimuli for each transistor within the circuit. These approaches extract the waveforms for temperature T and gate-source voltage V_{gs} over time depending on the use-case scenario for the circuit. For larger circuits, we employ the approach of [73] replacing V_{gs}-waveforms with equivalent, yet simpler waveforms and parameters.

Degradation: T and V_{gs} waveforms are then passed to the unified RTN/BTI model, which calculates $\Delta V_{th,total}$ based upon them. As RTN, BTI and PV are probabilistic processes, probability levels for $\Delta V_{th,total}$ are returned, e.g. for a $20mV$ shift a 7% probability and for a $30mV$ shift a probability of 5%. This is a direct result of the random nature of defects and their intrinsic variability, i.e. their results distributed as a probability density function.

Interpreting Probabilities: With the probabilities for degradation levels in each transistor known, the designer can analyze his circuit. The overall reliability of the circuit can be expressed as:

$$P_{fail}(\text{circuit}) = 1 - \prod_{j=1}^{m} (1 - \alpha_j \cdot P_{fail}(j, x_j)) \tag{4.16}$$

$$P_{func}(\text{circuit}) = \prod_{j=1}^{m} \alpha_j \cdot P_{func}(j, x_j) \tag{4.17}$$

with m transistors in the circuit and $P_{fail}(j, x_j)$ the probability of failure of transistor j exceeding $\Delta V_{th,total} = x_j$. If transistor j exceeds x_j, then errors can occur (e.g. timing violations), because j exceeded its budget and may become too slow to provide a stable signal to the following circuitry. The transistor weight α_j allows the consideration of varied impact of transistor degradations on the circuit. Some transistors are more important for the reliability of a circuit than others, e.g. if multiple transistors receive the output of a transistor as an input, amplifying its degradation. For large circuits, obtaining α_j may be challenging, so than it may be set to just 1 for each transistor. However, should the designer perform transistor sensitivity analyses to determine the weighting factors, then the model can incorporate the weights. If x_j is given (e.g. maximum tolerable degradation is $30mV$), then $P_{fail}(\text{circuit})$ can be determined. This enables the designer to judge the reliability of a given system. Vice-versa if $P_{fail}(\text{circuit})$ is given from a specification, x_j for each transistor can iteratively be found. If the circuit can tolerate $\Delta V_{th,total} = x_j$ in transistor j, than $P_{fail}(\text{circuit})$ is met.

For very large and/or complex circuits, we suggest that everything is broken down to the gate level and that synthesis and circuit analysis tools (e.g. static timing analysis) take care of the intrinsic complexity (e.g. handling switching critical paths, the sheer number of transistors, etc.). For details of this approach please refer to our work in [82]. In a nutshell, synthesis was made degradation-aware, i.e. standard cell libraries were degraded to allow synthesis algorithms to optimize the design automatically. After synthesis, the elaborate analysis tools can be employed in order to analyse the impact of the RTN, BTI & PV degradation on the circuit, even for very complex circuits. This allows for estimations of $P_{fail}(\text{circuit})$ even for very large and/or complex circuits (e.g. full microprocessors).

Tolerating Degradations: One possible approach to tolerate ΔV_{th} in transistors is transistor hardening through resizing, i.e. increasing the width of the transistor to increase its driving capabilities. Assuming the transistor must deliver a given drain current I_D after degradation (e.g. to switch fast enough or provide a stable signal), then W can be found based upon a first-order approximation for I_D:

$$I_D \approx \frac{\mu}{2} \cdot C_{ox} \cdot \frac{W}{L} \cdot (V_{dd} - V_{th})^2 \tag{4.18}$$

$$\Rightarrow W = \left(I_D \cdot \frac{2}{\mu \cdot C_{ox}} \cdot L \right) \cdot \frac{1}{(V_{gs} - V_{th})^2} \tag{4.19}$$

$$= A \cdot \frac{1}{(B - V_{th})^2} \tag{4.20}$$

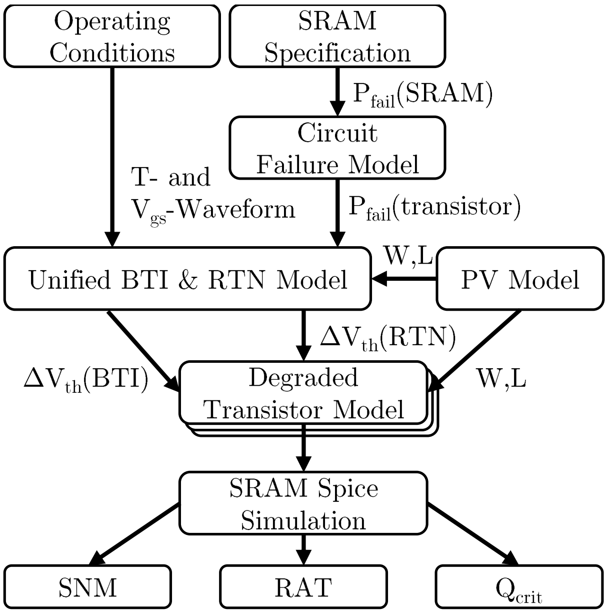


Figure 4.6: Process flow of our NTC circuit reliability estimation.

with A being defined by the transistor parameters and the desired I_D and B given by V_{gs}, which in most digital circuits is equal to V_{dd}. With equation 4.20 every $\Delta V_{th} = x_j$ can be translated to an increased W, which is able to deliver the desired I_D even if $\Delta V_{th} = x_j$ is applied.

As previously mentioned, W has an impact on ΔV_{th}, however W is always increased, which decreases ΔV_{th} and therefore results in safe design. An iterative approach is not feasible, as this iteration does not converge to a stable point (see Fig. 4.14).

4.1.6.2 SRAM Reliability Example

Intel was able to scale the logic of their ultra low power processor down to $V_{dd} = 0.28V$, while memory was kept at $V_{dd} = 0.55V$ [115] despite the selection of resilient 10T SRAM cells [114]. This indicates that the reliability at low V_{dd} was limited by the memory cells and therefore we chose SRAM cells in a processor register file as our target circuit for our NTC reliability estimation.

SRAM: SRAMs feature three key metrics, which describe the state of the memory cell. First metric is the Read Access Time (RAT), which is an indication for the performance of the SRAM cell. Secondly the Static Noise Margin (SNM), which determines the resiliency against voltage noise, preventing corruption of the stored data. Lastly, the critical charge (Q_{crit}), i.e. the minimal amount of charge deposited (e.g. by a particle strike) within the SRAM cell until the induced current is strong enough to corrupt the stored data.

SRAM Reliability: To protect against degradations like PV, RTN and BTI, manufacturers and designers over-design their circuits, i.e. they employ guardbands. Renesas technologies reports 3σ SNM guardband for SRAM cells [44] with σ as the standard deviation of PV. In this work we assume a similar guardband for RAT and Q_{crit}, i.e. $3\sigma(PV)$. If one of the metrics is degraded beyond 3 times the impact of PV, then the degradation within its transistors (ΔV_{th}) was too large to tolerate and the data within the SRAM is either corrupted (violated SNM/Q_{crit}) or arrived too late (violated RAT).

Reliability Modeling: In order to quantify SNM, RAT and Q_{crit} we employ SPICE simulations with BSIMv4.8 [14] as our transistor model parametrised by the $45nm$ predictive technology model [3]. We rely upon [125] for transistor sizing in reliable high-performance (HP-SRAM) and low-power (LP-SRAM) memory cells. An overview of our reliability modeling approach is shown in Fig. 4.6. The degradations to the transistors within the SRAM cell are calculated by our unified BTI & RTN model. The model is parametrised based upon [31, 118, 123]. The unified model obtains its input parameters from the stimuli extraction (similar to [73]), which extracts occurring T and the V_{gs} waveform for each transistor in the register file of a processor, the PV model, which provides ΔW,ΔL and the SRAM specification. Then, our tool creates a degraded transistor model[2], i.e. a file which characterizes the transistor, but with degraded V_{th} due to BTI & RTN and altered W,L due to PV. The degraded transistors are then employed within the aforementioned SPICE simulation of our HP-SRAM/LP-SRAM to determine the impact of the transistor degradation on the SRAM metrics RAT, SNM and Q_{crit}.

SRAM specification: As an example, we assume the probability $P_{fail}(SRAM) = 0.01$ corresponding to $P_{func}(SRAM) = 0.99$ as our reliability specification of our SRAM cells. With equation 4.16 this results in $P_{fail}(transistor) = 0.00167$ / $P_{func}(transistor) = 0.99832$ with $\alpha_j = 1$ for all transistors, as each transistor in a 6T SRAM cell is connected to 2 other transistor gates. With the employment of error correction codes (ECC), the reliability constraint can be softened. With SECDED Hamming codes, 1-bit errors are recoverable and 2 bit errors detectable. In a 64 bit register this means that 2 out of the 72 bit (64 data bits + parity) cells may fail, i.e. $P_{func}(SRAMwECC) = 0.99 - \frac{2}{72} = 0.962$ for detection and $P = 0.99 - \frac{1}{72} = 0.976$ for recovery. This softens the reliability constraint of transistors to $P_{fail}(tranECC) = 0.00473$ / $P_{func}(tranECC) = 0.99527$ for detection, i.e. 2.8x the tolerable P_{fail} compared to no employed ECC.

4.1.7 Evaluation

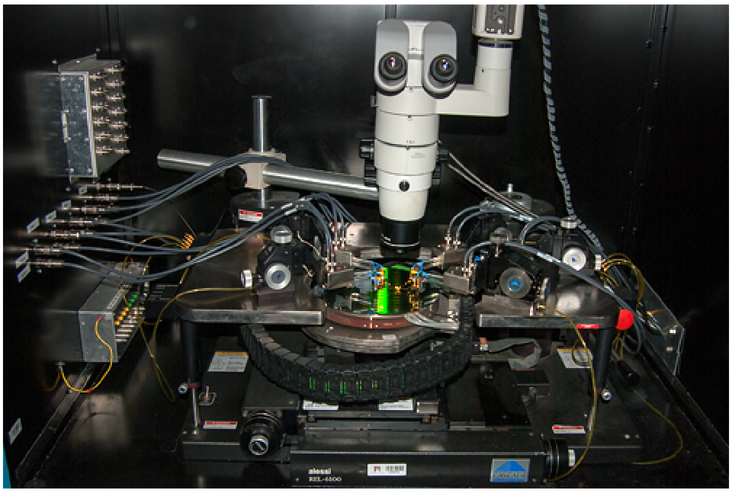


Figure 4.7: The semiconductor measurement setup of our collaborators (Montserrat Nafria from UAB) in a Faraday cage. The measurements are taken by ourselves at their setup.

Measurement Setup: Our collaborators Keithley SCS 5200 semiconductor measurement setup (see Fig. 4.7) which measures our non-commercial high-κ wafers in order to extract to the BTI and RTN parameters for our unified model. The image shows our microscope and our micro manipulators in contact with the wafer on an automated heatable chuck.

Model Validation: In order to validate that our unified model is capable of modeling RTN & BTI in NTC, we simulated waveforms and compared them against our measured waveforms in Fig. 4.8a) & b). For this purpose, we characterized RTN to parametrise our model to match our technology provided by our industrial partner. Fig. 4.8 illustrates that our model is capable of matching V_{th}-waveforms in demanding low-V_{th} single defect scenarios at very low voltages.

[2] Other parameters which depend on V_{th} are intrinsically altered by the compact transistor model (e.g. BSIM).

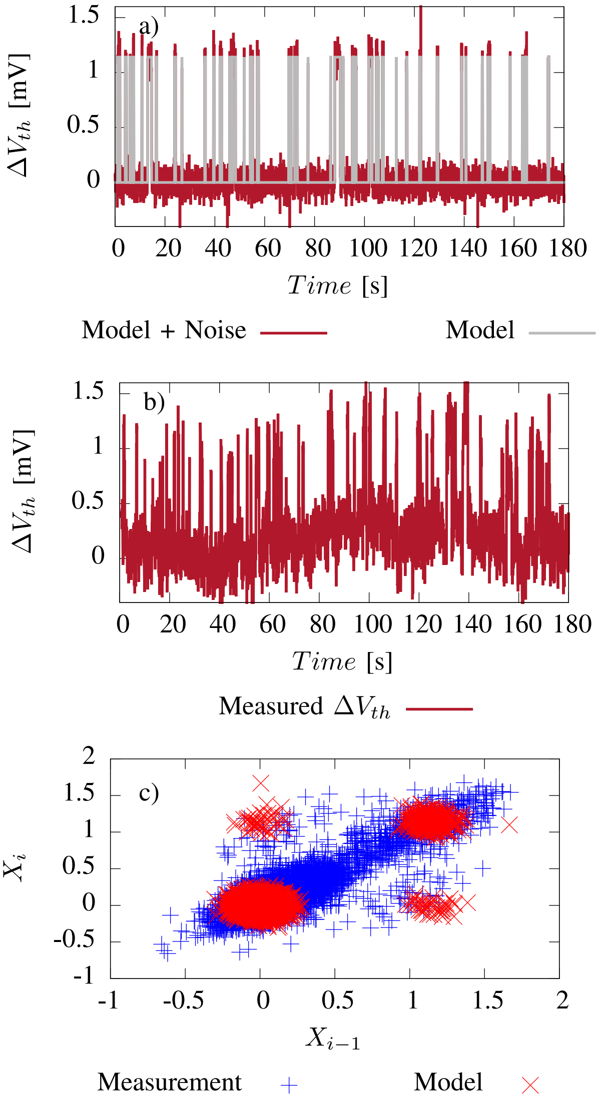


Figure 4.8: Comparison of modeled a) and measured b) V_{th} waveform for a $L = 135nm$, $W = 150nm$ transistor at $V_{dd} = 0.32V$ with a single $1.37mV$ defect. The model matches the measured waveform in magnitude and frequency, even when just a single unusually low V_{th} defect is selected, highlighting the accuracy of our modeling. Slight differences remain, as capture/emission of carriers is triggered randomly and low frequency noise occurs in our measurement setup, which warps the lower edge of the measured waveform. Therefore, in c) a time lag plot (explanation: [124]) is presented, which shows how the modeled single defect matches the auto-correlation of the measured waveform. The centers or the distributions match, while the wider distribution of the measured signal in blue is again due to the low frequency noise in the measured signal.

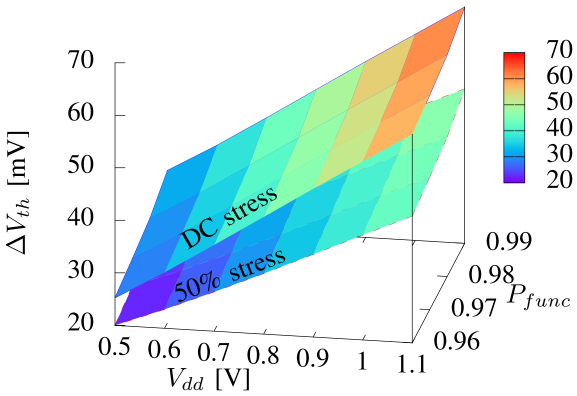


Figure 4.9: Unified model over various V_{dd} and reliability constraints (P_{func}) from NTC ($0.5V$) to STC ($1.1V$). Higher voltage, stress and more stringent reliability constraints result in higher degradations. The impact of V_{dd} is considerable, as is the difference between 50% on-/off-ratio and constant stress. The reliability constraints have a smaller impact on the degradation (see Fig. 4.10 for detailed evaluation).

Fig. 4.8c) shows a time lag plot of the measured and modeled waveform. Time lag plot are used to characterize RTN, especially its defect distribution [124]. As the locations of density of both distributions are similar, the model can predict RTN in NTC. While it is impossible to match the specific random pattern of the measurement (due to noise while measuring tiny I_D currents at 0.32V), the time constants within the defect distribution are matched.

At higher voltages in the STC domain we could rely upon our previously extracted defect distributions in [46], [31]. Especially in [46] we showed a good agreement between our defect distributions for BTI and RTN in STC.

Unified Model from NTC to STC: Our model is capable of modeling RTN & BTI across the wide voltage range from NTC to STC. In Fig. 4.9 the induced ΔV_{th} is shown for a constant DC stress and low stress (50% on-/off-ratio) scenario. On the x-axis the voltage, on the y-axis the reliability constraint and on the z-axis the induced shift are plotted. Higher voltage stimulates BTI and thus results in a higher voltage shift, while a tighter reliability constraint means including even rarer (higher degradation) RTN events.

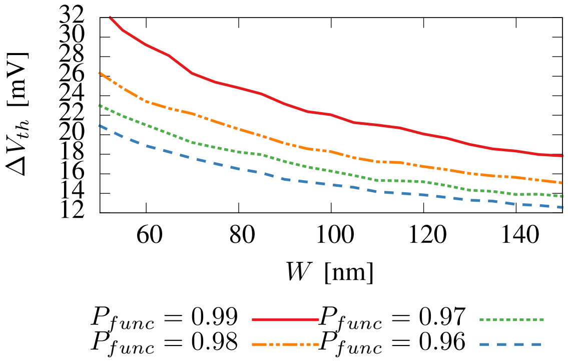


Figure 4.10: RTN-induced ΔV_{th} over different W and P_{func} at $L = 27nm$, $V_{dd} = 0.5V$. Larger transistors have less RTN-induced ΔV_{th}, e.g. $50mV$ at $W = 30nm$ compared to $19mV$ at $W = 130nm$ at $P_{func} = 0.99$.

Impact of Geometry on our Unified Model: Our unified model does depend on the transistor geometry, as shown in Fig. 4.10. Smaller transistors exhibit larger degradations, highlighting how RTN & BTI are worse with continuing technology scaling. As both are a probabilistic phenomenon, the reliability constraint plays an important role in the degradation which must be tolerated. For $P_{func} = 0.99$ $\Delta V_{th} = 33mV$ is reached for the smallest studied transistor, while for 0.96 just $20mV$ are reached.

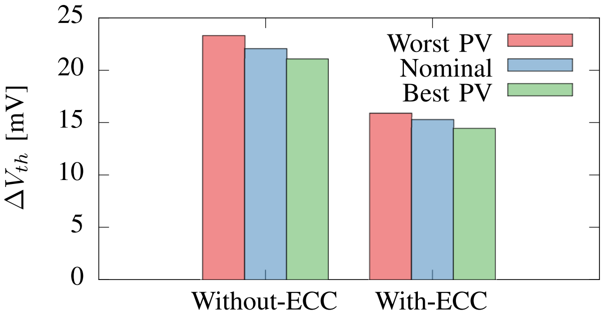


Figure 4.11: Impact of best ($L + 6\sigma$, $W + 6\sigma$), nominal (no shift) and worst-case ($L - 6\sigma$, $W - 6\sigma$) PV on $\Delta V_{th,total}$, i.e. impact of shifted W & L on BTI & RTN.

Impact of PV on Unified Model: PV randomly shifts W and L of a transistor. Fig. 4.11 shows the impact of these PV-induced shifts on our unified model. In our SRAM example, transistors in SRAM registers without ECC $P_{func} = 0.99$ must tolerate nominally $\Delta V_{th}(RTN) = 22.0mV$, while for the smallest device (i.e. high impact of individual defects) the tolerable degradation rises to $23.3mV$ and for the largest device (low impact of individual defects) drops to $21.0mV$. With ECC ($P_{func} = 0.962$), nominally $15.2mV$, worst-case $15.8mV$ and best-case $14.4mV$ respectively.

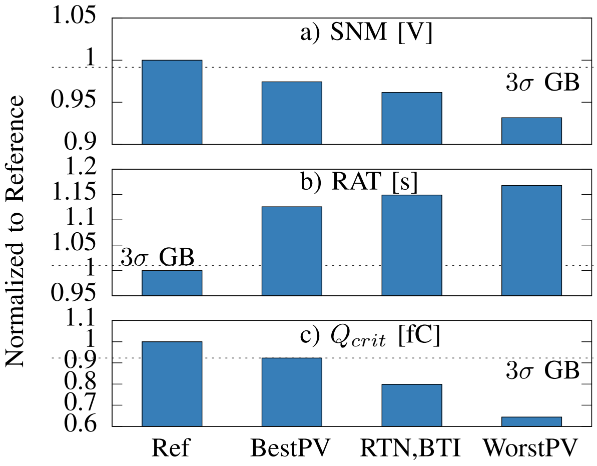


Figure 4.12: Impact of best $(L+6\sigma, W+6\sigma)$, nominal (no shift) and worst-case $(L-6\sigma, W-6\sigma)$ PV on BTI & RTN degrading a 45nm HP-SRAM cell at $V_{dd}=0.6V$ plotted with the 3σ guardband by [44]. There is a strong impact on Q_{crit}, followed by RAT and finally a weak impact on SNM.

SRAM Metrics: SNM, RAT and Q_{crit} are not equally important for the reliability of a SRAM cell. $P_{fail}(SRAM)$ due to soft errors is very low at sea-level [15], so even a major reduction in Q_{crit} has little impact on the overall reliability. RAT and SNM have a large impact, as increased delays lead to timing violations and low noise margins corrupt the data stored within the memory cell. Fig. 4.12 show the degradation to the individual metrics versus their guardbands according to Renesas [44]. All 3 metrics surpass their guardband, which is in line with our introduction, where [114] needed hardened SRAMs for a reliable operation at $V_{dd}=0.55V$. This highlights the challenge to maintain reliability in NTC due to the high impact of even slight degradations.

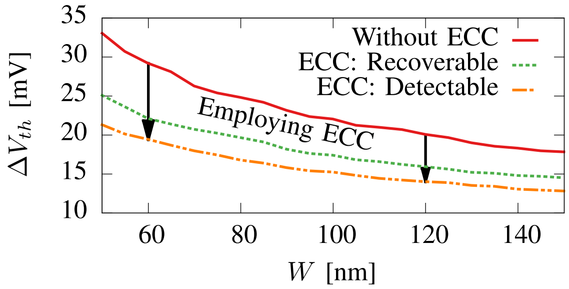


Figure 4.13: Employing ECC in a 64-bit memory cell results in lower reliability constraints (P_{func}) reducing the BTI & RTN guardband by up 36% at $W=50nm$ and 29% at $W=150nm$.

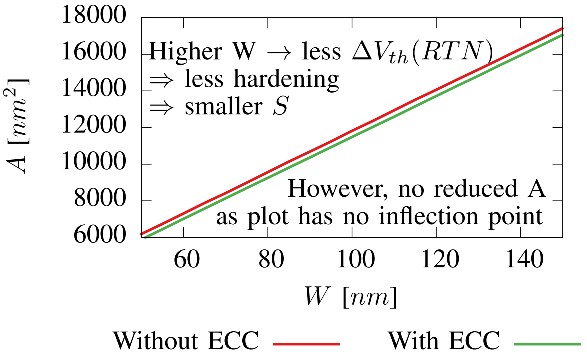


Figure 4.14: Area of hardened transistors, i.e. transistors with scaled widths to compensate for $\Delta V_{th,total}$. Even though smaller transistors produce a higher $\Delta V_{th,total}$ and thus are prone to stronger width scaling, they ultimately remain smaller after transistor hardening.

Tolerating Degradations: In order to tolerate the induced degradation by BTI, RTN & PV we proposed transistor hardening through resizing. Fig. 4.14 shows how the area of the transistors increase for the RTN-induced ΔV_{th}

according to Fig. 4.13 with and without ECC. It is noteworthy, that despite ECC providing protection for single bits only, while all bits are affected by BTI and RTN, ECC still remains effective and decreases necessary transistor hardening. Additionally, despite lower $\Delta V_{th}(RTN)$ in larger transistors, which results in smaller scaling factor S, the area $A = (S \cdot W) \cdot L$ still increases towards larger transistors.

Selecting the ideal W is an iterative approach:

1. Select W_1 for a transistor due to its driving strength (I_D) in the circuit design
2. Estimate the impact of BTI/RTN with the unified model based upon W (e.g. with algorithm 2): $\Delta V_{th}(W_n)$
3. Increase W to harden the transistor against BTI/RTN: $W_n = W_{n-1} + \Delta W(\Delta V_{th})$ with $\Delta W(\Delta V_{th})$ being the additional width to maintain I_D of the transistor and tolerate degradation (ΔV_{th}).
4. Go to step 2, resulting in the recursive function:
   $W_n = W_{n-1} \cdot A \cdot \frac{1}{(B - V_{th} - \Delta V_{th}(W_{n-1}))^2}$

However, this function does not converge towards an optimal solution in terms of area and I_D. Thus, despite stronger degradations (ΔV_{th}) in smaller transistors resulting in more $\Delta W(\Delta V_{th})$, it is still beneficial to employ the smallest transistors instead of choosing a larger W to begin with. Therefore, the pareto optimal point in which the product of reliability and area are minimal is always the smallest transistor, i.e. optimal W_1 is smallest width to reach given I_D.

4.2 Mitigating Defect Variability from the Transistor to the Circuit Level

This section is based on my publication [84].

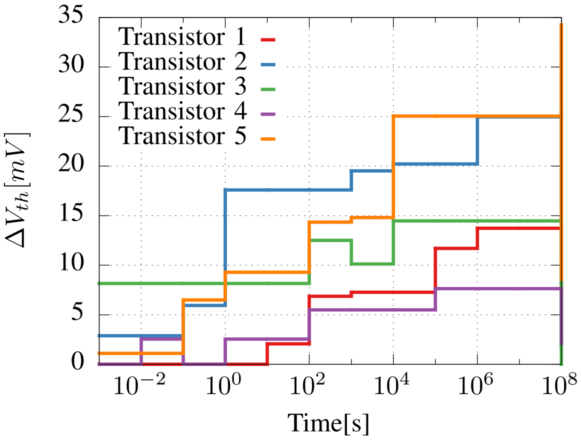


Figure 4.15: Transistors exhibit different degradations despite identical manufacturing and operation under the same stimuli. This highlights how transistors, which only differ in their defects can degrade differently and thus introduce a variability in their electrical parameters. This effect is called *defect variability*. The same colors will be used throughout the work, to indicate that the same pMOS transistors are used.

4.2.1 Defect Variability is CMOS Challenge

Variability is a big challenge for CMOS technology in the nano era. This challenge must be tackled during the circuit design. Two types of variability are known. First, Time-Dependent Variability (TDV) consisting of material imperfections - called defects - that are generated during manufacturing within the gate dielectric of each transistor [37]. In contrast to the second type, traditional Time-Zero Variability (TZV) like geometric, dopant fluctuations, etc., TDV consists of electrically neutral defects, that do not manifest themselves as any degradation immediately after manufacturing. Therefore, it only manifests itself after a stimuli (e.g. voltage) is applied. However, these defects cannot be ignored, as they can capture carriers during circuit operation, weaken the formation of a channel in a MOSFET and thus induce a threshold voltage shift ΔV_{th} in the transistor (see Fig. 4.15 and [31, 37]). In fact, ΔV_{th} degradation due to TDV is higher than TZV ΔV_{th} degradation consisting of work function, geometric, dopant fluctuations in current 10nm FinFET [126].

To protect the circuit against all types of variability, circuit designers typically employ a guardband. A guardband is purposefully over-designing the circuit, in which the deleterious effects of degradations can be tolerated. In this work, our guardband is a timing slack on top of circuit delay, which prevents timing violations by prolonging clock periods. Longer clock periods tolerate prolonged/degraded circuit propagation delays, caused by threshold voltage shifts ΔV_{th} in the transistors. Until circuit delay degrades beyond the safety margin provided by the guardband (i.e. $t_{delay}(current) < t_{delay}(nominal) + t_{delay}(guardband)$) reliable circuit operation is ensured by the guardband. The challenge is to select the guardband correctly. If the guardband is too low, the clock period is shorter than the propagation delay of the circuit and thus timing violations occur. If the guardband is too high, then the clock frequency is too low and circuit performance suffers unnecessarily. Therefore, peak degradations due to variability must be determined accurately, which is the goal of this work. In standard EDA tool flows, the guardband is determined based on the process corners. For Process, Voltage and Temperature (PVT) variations a best-case corner (fast-fast FF), a typical corner (typical-typical TT) and a worst-case corner (slow-slow SS) are provided by the semiconductor vendor. Each corner provides timing and power information for best, typical and worst-case operating conditions and manufacturing tolerances. Therefore, the SS corner features delay and power for standard cells at a) worst process (worst TZV sample & worst TDV sample) b) worst (highest) temperature c) worst (lowest) voltage. Designing the chip at TT and performing timing checks at SS is the typical approach. This is - by design - very pessimistic, which ensures reliable operation but severely harms performance.

To reduce pessimism, the circuit designer can de-rate the corners. De-rating is interpolating standard cell delay between corners, e.g. between SS and TT. By specifying known operating condition (e.g. 80C instead of 125C worst case temperature) the EDA tools derate (e.g. $t_{delay} = 0.9 \cdot T_{delay}(SS) + 0.1 \cdot T_{delay}(TT)$) or re-characterize the cells to obtain timing at these less pessimistic conditions. De-rating is available for temperature and voltage, yet for variability there is no such option. The impact of variability cannot be reduced. This fits time-zero variability, as the circuit designer has no control over the variability of semiconductor manufacturing. However, time-dependent variability is influenced by the circuit designer. As the name suggests, time-dependent variability depends on the duration of operation of the circuit. Even if the circuit is designed for 2 years operation (consumer warranty), variability values from more than 10 years operation (industrial warranty) are taken, introducing unnecessary pessimism. Additionally, TDV degrades more under high temperature (characterization at worst-case temperature of 110°C [127]) than at lower temperature (e.g., actual exhibited temperature) [113]. Therefore, currently TDV is severely overestimated in EDA tools. This work reduces pessimism by accurately estimating the upper bound for TDV depending on the conditions (time, voltage, temperature) specified by the circuit designer.

Estimating and lowering the upper bound for TDV is the goal of this work. We model TDV from the physical to the circuit level to estimate the upper bound as accurately as possible. Our approach is fully compatible with existing EDA tools, as we define TDV worst-case process corners. These process corners then are used to design safe but less pessimistic circuits. Then, optimization strategies are presented to mitigate TDV, i.e. reduce impact of ΔV_{th} on t_{delay} and thus reducing the upper bound further. This reduction results in even smaller guardbands, further reducing the area/power/performance overhead for TDV protection.

Our novel contributions within this manuscript are:

1. A novel fast, yet accurate approach to determine the upper bound for TDV in transistors called *reliability specification*. This physically motivated abstraction provides accurate upper bounds for TDV with minimal pessimism. This transistor upper bound is then used to create two worst-case cell libraries, which in turn provide upper bounds for TDV on the circuit level with minimal pessimism.

2. A novel variability-aware logic synthesis to mitigate the impact of TDV on circuit delay. The automatic selection of resilient cells (with respect to variability) by exploiting mature EDA synthesis algorithms introduces negligible area and power overheads, yet effectively reduces the impact of TDV. Hence, smaller timing guardbands can be employed without degrading reliability.

4.2.2 Definitions

To explain our TDV estimation, the physical origin of TDV is explained and the terms defined.

TDV consists of two phenomena, one aging phenomenon called Bias Temperature Instability (BTI) and the other is a noise phenomenon called Random Telegraph Noise (RTN). BTI is considered an aging phenomenon, which creates and activates defects in the gate dielectric manifesting itself in degraded electrical parameters in transistors [34, 37, 113]. In contrast, RTN is a noise phenomenon, which manifests itself in random fluctuations of the threshold voltage ΔV_{th} of transistors [113]. Both phenomena are part of Time-Dependent Variability (TDV).

BTI and RTN share a similar physical origin [45, 47, 113], i.e. the activation and passivation of defects within the gate dielectric.

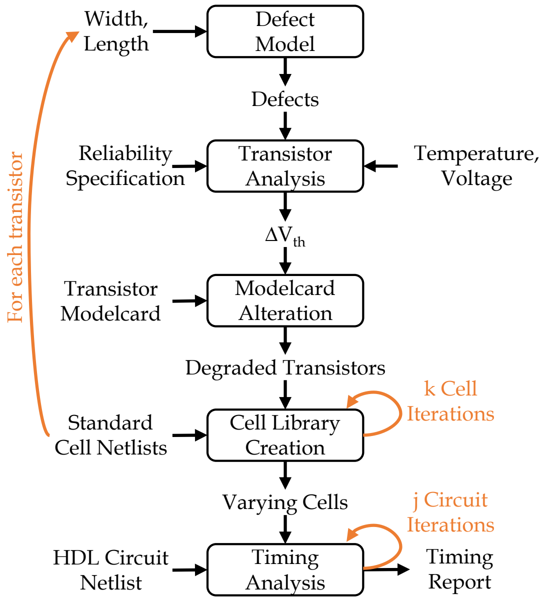


Figure 4.16: Overview of our proposed methodology. Cell iterations k is the number of different cells which are generated for each cell type (e.g., 1000 different AND2 cells, 1000 XOR2 cells, etc.). Circuit iterations j is the number of different annotated circuits which are generated (same structure/topology, just differently degraded cells) and the timing report is generated (e.g., $j = 5000$ DCTs with each cell picked from a variability cell library with $k = 1000$.

4.2.2.1 Inter- and Intra-Transistor Variability

Inter- and Intra-transistor variability are subsets of TDV. This section explains the two variability types within TDV.

Inter-transistor variability is the difference in electrical parameters (e.g., V_{th}, μ, I_{on}) *between two transistors*, which appears in both TZV (e.g., geometric) and TDV. For TDV, each transistor has a unique degradation waveform. This waveform is governed by the defects, which were introduced in that transistor during manufacturing. If the transistor features many defects, then the transistor degrades more and does so with more steps (as each defect captures a carrier). Therefore each transistor has a unique degradation waveform even for identical stimuli as shown in Fig. 4.15.

Intra-transistor variability describes the change of electrical parameters *within the same transistor*, which is a feature unique to TDV and not present in TZV. TDV can fluctuate randomly, as capture and emission of defects are both probabilistic events [37]. Exposing the transistor to the *same stimuli twice results in two different* ΔV_{th}, as the random physical processes occur slightly different each time. Therefore, even if the defects within the transistor are known, the response to operation can only be described statistically.

Unlike inter-transistor variability, which is widely explored in related work, intra-transistor variability is still not fully explored. Using our previous work with intra-transistor variability modeling [113] (model briefly explained in Section 4.2.4.1), this work is the first to explore the impact of intra-transistor variability on circuits.

4.2.3 Related work

4.2.3.1 Applicability of TZV Techniques to TDV

TZV estimation schemes on the transistor level are not applicable to TDV as they model variability as solely a time-zero statistical effect [128]. On the transistor level geometry, dopant concentrations, metal-gate work functions, etc. are modeled with random distributions, which do not change over time. However, TDV does not solely depend on manufacturing, but also on ever-changing stimuli (T, V_{gs}) and time. If the stimuli rise or time passes the average degradation increases. The traditional variability method is not able to account for these shifts in the random distribution over time, resulting in pessimistic approach for TDV to ensure reliability under any conditions at the cost of a high area/power/timing overhead.

On the circuit level various attempts have been made to abstract transistor variations into distributions or compact models for macroscopic metrics like delay [128–130]. Similarly, machine learning approaches like [131] can also be employed. However, in order to accurately estimate the impact of shifting transistor parameters on the delay of standard cells or circuits, full circuit simulations are required to take all necessary dependencies into account (cell topology, transistor geometry, load capacitance, input signal slew, etc. (details in Section 4.2.4.3).

4.2.3.2 Time-Dependent Variability

Next to research in the area of traditional variability, TDV has also been explored within BTI and RTN communities. The works [37,47,48] consider capture and emission within countable defects in nano-era transistors. These models are based on the foundations of defect-centric BTI modeling in [27, 132]. However, none of these works abstract towards the circuit level or beyond and none are compatible with standard EDA tool flows. These works were designed for maximum accuracy by modeling physical processes in detail and as such computational heavy. By design, they cannot be applied to complex circuits like microprocessors with their tremendous number of transistors. Their estimation is limited to transistors or individual cells like AND, INV and SRAM. Abstracted BTI circuit modeling is presented in works like [74] with the ability to evaluate complex circuits. However, they rely on simple empiric expressions as their aging models and as such do not consider TDV, neither inter- nor intra-transistor variability (details in Section 4.2.4.2).

Authors in [133] present work which is conceptionally similar. They explore BTI/RTN-variability from physical level to the circuit level with delay estimations in their temporal statistical static timing analysis. Even though their goals are similar, our work differs significantly from their design as follows: On the physical level, the two BTI reaction-diffusion and trapping-detrapping models are combined into a single model, even though they both rely on different physical phenomena (breaking of Si-H bonds versus trapping detrapping in hole traps) and thus cannot simply be combined [8]. First attempts to merge these two theories correctly just recently appeared [34]. The model in [31] circumvents this issue, as it is agnostic to defect types (interface traps, hole traps, etc.) as it relies solely on capture and emission, which occurs in *any* defect type, just with different parameters. In addition, a custom timing analysis based on Taylor approximations is used, while this work employs existing EDA timing tools. This means, we employ mature algorithms from EDA vendors, which allow us to incorporate circuit environments, consider complex topologies and shifting critical paths (see Section 4.2.4.3). These additional aspects are critical for accurate delay estimations [82].

The work in [134] uses a physical-level BTI model, but does not consider RTN, i.e. intra-transistor variability, which cannot be neglected in current nano-scale transistors (see Section 4.2.4.2). Additionally, their cell library characterization considers a single degradation for all transistors of a type (nMOS/pMOS) at discrete intervals and interpolates the resulting cell delay tables for intermediate values. Not taking unique transistor degradations into account does not provide the upper bound for path delay. Transistors may counteract each other (e.g., pull-up vs. pull-down transistors), i.e. uniform degradation in these transistors underestimates delay degradation. A non-degraded pull-down transistor counteracting a degraded pull-up transistor (e.g., in an inverter) results in the worst rise delay. This means they underestimate TDV and reliability cannot be guaranteed.

Section 4.1 presents the core of the BTI and RTN model employed here. This section is an extension of the unified model presented in Section 4.1. It extends the scope to modeling in both in near-threshold and super-threshold operation. Furthermore, it extends it to include TDV and reliability specification, worst-case cell libraries and optimization strategies (e.g., variability-aware synthesis).

4.2.3.3 Limited EDA Tool Variability Support

Detailed discussions about the limitations of commercial EDA tools are presented at the end of Sections 4.2.4.3 and 4.2.4.4.

4.2.3.4 Distinction from State of the Art

In summary, TDV is not explored in detail from the physical to the circuit level, due to the computational complexity. Physical defect-level models could only evaluate simple circuits and individual cells, while circuit level work had to simplify transistor/cell modeling to achieve computational feasibility. Our aim is to abstract physically motivated on the defect-level and to exploit mature algorithms higher up to achieve accurate estimations from the physical level to the circuit level. We do not simplify the defect modeling or abstract defects or even transistors away (e.g. by modeling cell delay with empirical equations directly). This is made possible by tackling the computational challenge with a novel fast approach to estimate transistor degradation on the transistor level. On the cell level, mature EDA tools provide sufficient performance, while on the circuit level the computational complexity is shifted to a one-time effort, by employing custom worst-case TDV cell libraries for chosen conditions (e.g. temperature $80\,^{\circ}\mathrm{C}$ for TDV-degradation). Additionally, designing for TDV is largely unexplored, therefore optimization strategies are provided at each abstraction level to reduce the induced ΔV_{th} and ultimately the guardband.

Distinction from state of the art:

1. Modeling time-dependant variability from defects on the physical to the circuit level without abstracting or simplifying either the low- or high-level estimations for the sake of computational feasibility. Instead we achieve the feasibility by a novel fast transistor reliability specification, maintaining compatibility to EDA tools and pushing computational effort to one-time characterization of worst-case cell libraries.
2. Modeling intra-transistor variability on the circuit level for the first time by taking it into account in our novel reliability specification.
3. Our two worst-case cell libraries are used in variability-aware synthesis (VAS) to optimize circuits automatically with synthesis tools to obtain smaller, yet sufficient guardbands.

4.2.4 Time-Dependent Variability

We explain our approach from the bottom up, starting with the physical level, over the transistor-, cell- all the way to the circuit-level, i.e. full microprocessors. On each level we present our methodology and investigate the impact of TDV at that level. Additionally, on each level optimization strategies are presented to reduce said impact.

4.2.4.1 Physical level

The exact cause for BTI/RTN is still part of a debate in the reliability physics community [32], however all agree and theorize that the observed degradations occur due to capture and emission of carriers within various defect types (e.g., interface traps, oxide traps). To become fully agnostic about the type of defects, we abstract defects to three universal defect parameters:

1. Capture time τ_c
2. Emission time τ_e.
3. Induced Threshold Voltage Shift η

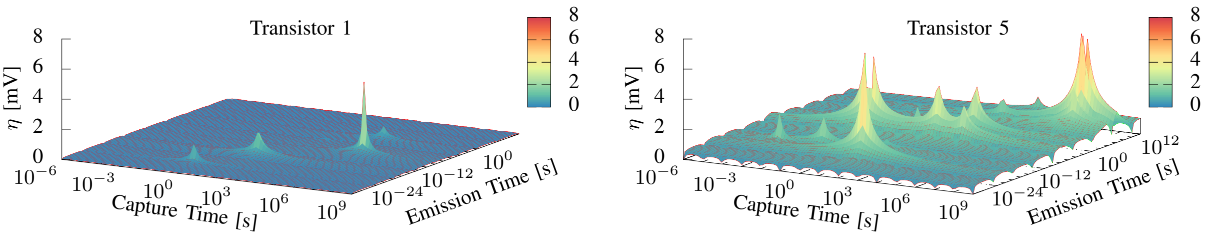


Figure 4.17: Defect maps for pMOS transistor 1 and 5 to illustrate that η is varying across the defects and does not depend on the location within the defect map. In terms of TDV, transistor 5 is a bad sample with many defects and high η in most defects, while transistor 1 has fewer defects and just 1 defect with high η. Therefore, applying the same stimuli results in higher ΔV_{th} in transistor 5 than in transistor 1, despite being indistinguishable (all defects neutral) immediately after manufacturing.

These three parameters allow us to model capture and emission in these defects (independent of defect type) by experimentally extracting these three parameters for each defect within a transistor (for example with the technique presented in [37]). The threshold voltage shift ΔV_{th} of a transistor can be then expressed as:

$$\Delta V_{th}(tran) = \sum_{d=0}^{m} \eta(d)$$

with m as the number of defects within the transistor and d the current defect. If the defect captured a carrier, the induced $\Delta V_{th}(d)$ is given by $\Delta V_{th}(d) = \eta(d)$ of that particular defect d, while other unoccupied defects remain electrically neutral, i.e. their $\Delta V_{th}(d) = 0$. The capture $\tau_c(d)$ and emission times $\tau_e(d)$ describe the latest/longest time for a defect d to capture and emit a carrier. In other words, how probable capture/emission for that defect is. Detailed explanation of our defect-centric model along with distributions for capture, emission times as well as η are available in Section 2.2.1.3.2 and 4.2.4.1.

Inter-Transistor Variability: Each transistor features a different number of defects, with unique parameters τ_c, τ_e and η per defect. These unique defect sets model DC in terms of inter-transistor variability. The number of defects m follows a Poisson distribution in which the mean λ is scaled to λ_1 in order to consider transistor geometry:

$$P(X = k) = \frac{\lambda_1^k \cdot e^{-\lambda_1}}{k!} \tag{4.21}$$

$$\lambda_1 = \lambda \cdot \frac{1}{W \cdot \sqrt{L}} \tag{4.22}$$

Each defect gets a random capture τ_c and emission time τ_e according to log-normal bi-variant distribution $D(\tau_e, \tau_c)$:

$$\begin{aligned} D(\tau_e, \tau_c) &= \frac{\exp\left(-\frac{1}{2(1-\rho^2)}(A^2 - 2\rho AB + B^2)\right)}{2\pi\tau_e\tau_c\sigma_{\tau_e}\sigma_{\tau_c}\sqrt{1-\rho^2}} \\ \text{with } \rho &= \frac{\exp\left(\rho_N\sigma_{\tau_e}\sigma_{\tau_c}\right) - 1}{\sqrt{(\exp\left(\sigma_{\tau_e}^2\right) - 1)(\exp\left(\sigma_{\tau_c}^2\right) - 1)}}, \\ A &= \frac{\ln(\tau_e) - \mu_{\tau_e}}{\sigma_{\tau_e}} \text{ and } B = \frac{\ln(\tau_c) - \mu_{\tau_c}}{\sigma_{\tau_c}} \end{aligned} \tag{4.23}$$

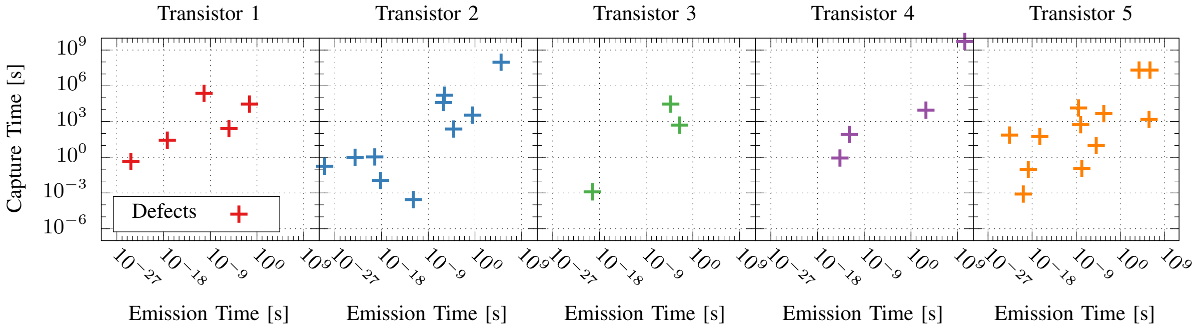


Figure 4.18: Defect maps for 5 pMOS transistors with the same geometry (W, L). Each transistor features a unique number of defects n with each defect featuring unique capture τ_c, emission time τ_e and degradation η. This illustrates how transistors, which are manufactured identically, later degrade differently due to different defects. Note, that the transistors match in color and parameters to Fig. (4.15, 4.17, 4.20 and 4.22).

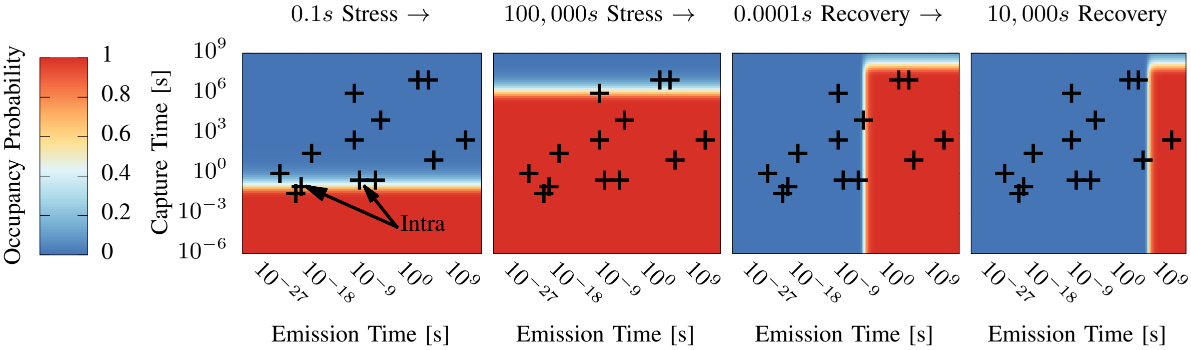


Figure 4.19: Occupancy probability P_{occ} for four consecutive (see arrow) stress/recovery phases, which act as stimuli to capture/emit carriers in defects of transistor 5. Defects in the white area have $P_{occ} \approx 0.5$ and thus their state is not certain compared to the defects in deep red ($P_{occ} = 1$) and blue ($P_{occ} = 0$). These defects are the responsible for intra-transistor variability, as these defects do not react deterministically to a stimulant.

and η according to exponential distribution (with scaled mean λ_2 to consider geometry):

$$P(X = k) = \lambda_2 \cdot e^{-\lambda_2 \cdot k} \tag{4.24}$$

$$\lambda_2 = \lambda \cdot \frac{1}{\frac{W}{W_{ref}} \cdot \sqrt[3]{\frac{L}{L_{ref}}}} \tag{4.25}$$

Fig. 4.18 shows the defect sets of five different transistors. Note that each transistor has a different number of defects, each defect with different defect parameters (τ_c, τ_e and η), despite being manufactured identically (Dopant concentrations, W, L, V_{th}, I_D, g_m, etc.). After manufacturing, all these defects are unoccupied, while during operation they can capture a carrier and thus degrade their transistor's V_{th} with their $\eta(d)$. The η of each defect varies widely as shown in Fig 4.17. Fig. 4.19 shows the occupancy probability P_{occ} of these defects after four stress (recovery) phases are consecutively applied to transistor 5. Note how most defects have almost certain state ($P_{occ} = 1$ (red) or $P_{occ} = 0$ (blue)), which corresponds to modeling BTI. Only defects, which happen to have τ_c (τ_e) near to the current stress (recovery) duration are uncertain $0 < P_{occ} < 1$. In these defects intra-transistor variability occurs, as these defects could already have captured, but also could have not yet captured a defect.

Intra-Transistor Variability: Most defects are certain as capture τ_c and emission times τ_e are *latest* times, which means that capturing a carrier can occur before this time has passed, but must happen latest at their time value. According to [31] the capture and emission probabilities can be defined for a time t_{stress} (Stress: transistor stimuli (V_{gs}, T) increase) , $t_{recovery}$ (Recovery: transistor stimuli decrease) as follows:

Stress:

$$P_{occ}(t) = P_{occ}(t_i) + \left(\frac{\tau_e}{\tau_e + \tau_c} - P_{occ}(t_i)\right) \cdot \left(1 - e^{\frac{t-t_i}{\tau_{sr}}}\right) \tag{4.26}$$

Recovery:

$$P_{occ}(t) = \frac{\tau_e}{\tau_e + \tau_c} + \left(P_{occ}(t_i) - \frac{\tau_e}{\tau_e + \tau_c}\right) \cdot \left(e^{\frac{t-t_i}{\tau_{sr}}}\right) \tag{4.27}$$

$$\text{with } \tau_{sr} = \frac{1}{\frac{1}{\tau_e} + \frac{1}{\tau_c}} \quad \tau_c = \tau_c(T, V) \quad \tau_e = \tau_e(T, V)$$

With simplified equations for short ($t_{stress} < \tau$) digital signals according to [31]:

$$P_{capture} = \frac{t_{stress}}{\tau_c} \quad P_{emission} = \frac{t_{recovery}}{\tau_e} \tag{4.28}$$

For longer t_{stress} ($t_{recovery}$), then $P_{capture}$ ($P_{emission}$) tend towards 1. Therefore, if more time is spent under stress (recovery), the corresponding equations ensure that capture (emission) of a carrier is more and more likely. Intra-variability occurs in defects with $P_{capture} = 0.5$ and $P_{emission} = 0.5$. Note, that even though just a limited number of defects are the source of intra-variability at a time, it still matters as just a few defects exist in any transistor and a single defect can have a high η (see Fig. 4.17).

4.2.4.2 Transistor level

After the defects and their individual defect parameters are randomly assigned, then the transistor can be characterized. After characterization, a reliability specification is used to determine the upper bound for $\Delta V_{th}(tran)$.

Reliability Specification: The simplified capture probability equation from [31] can be re-formulated as a reliability specification, when t_{stress} is replaced with $t_{rel.spec}$ (e.g., EOL time t_{EOL}) and the capture probability $P_{capture}$ with the desired probability of failure of the circuit P_{fail}:

$$P_{capture} = \frac{t_{stress}}{\tau_c} \quad \Rightarrow \quad \tau_{spec} = \frac{t_{rel.spec}}{P_{fail}} \tag{4.29}$$

$$\Delta V_{th}(tran) = \sum_{o=0}^{n} \eta(d) \text{ with } \tau_c(d) \leq \tau_{spec} \tag{4.30}$$

For each transistor, the upper bound of degradation from manufacturing to EOL is defined as the sum of degradation of all defects with capture times $\tau_c(d)$ *smaller* than the time constant found in the reliability specification τ_{spec}. All defects with longer capture times τ_c have capture probabilities $P_{capture}$ smaller than P_{fail} and thus are not considered to contribute their degradation. These defects are unlikely to capture during operation within the lifetime. Fig. 4.20 shows a graphical explanation of two different reliability specifications on the defect distribution of a transistor.

This reliability specification assumes the transistor to be under constant stress for its entire lifetime t_{EOL}. While this abstraction is pessimistic, it ensures that arbitrary activities can occur within the circuit and that our estimated upper bound is never exceeded. For complex circuits (e.g., our target full microprocessors), where activity extraction is

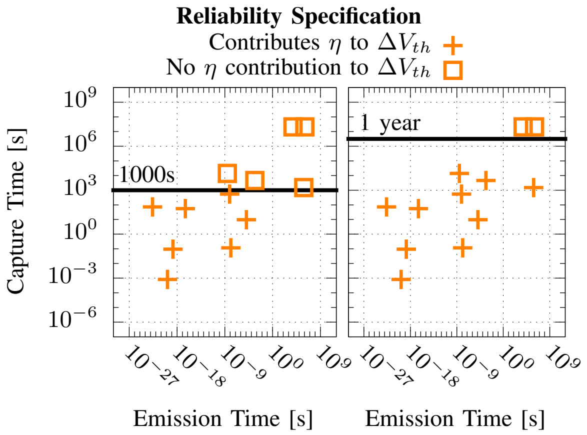


Figure 4.20: Two different reliability specifications applied to transistor 5 (see Fig. 4.18). Reliability specification on the left is $\tau_{spec} = 1000s$ and the resulting $\Delta V_{th}(tran) = 21.3mV$, while on the right $\tau_{spec} = 1year$ with $\Delta V_{th}(tran) = 34.7mV$. The induced ΔV_{th} is directly calculated from the randomly generated parameters (τ_c, τ_e, η) of the defects within transistor 5.

input data dependent and time consuming (for activity extraction options see [73,135]), this assumption is necessary to ensure that reliability is sustained under any arbitrary operation scenario.

To reduce the pessimism, the Longest Continuous Stress time (LCS) can be used to set $t_{rel.spec}$ instead of t_{EOL}. See Section 5.1.4.2 for more details. If it is given, that a given transistor is never stressed longer than a day (e.g., due to shutdown of the system at the end of a workday or an employed aging mitigation/power management technique), than $t_{rel.spec}$ can be set to a day $(86,400s)$. Note that stress times can be arbitrarily chosen. Results are shown in Fig. 4.21. After the day of stress, recovery occurs and carriers are emitted again[3].

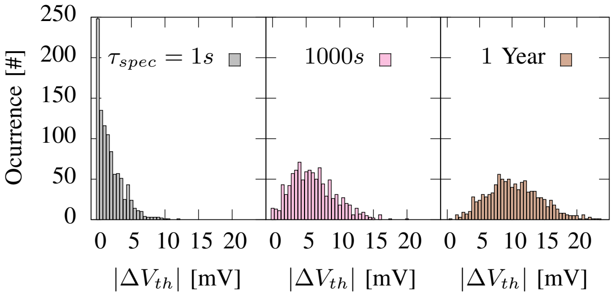


Figure 4.21: ΔV_{th} distributions for 1000 unique pMOS transistors (different set of defects), each with $L = 45nm$ $W = 90nm$ for three different reliability specifications (identical set of 1000 transistors used) $\tau_{spec} = 1s$, $1000s$ and $1year$. A higher reliability specification leads to higher ΔV_{th} and thus higher guardbands. Additionally, the variance increases at higher reliability specifications, i.e. transistors deviate further from each other. This results in a stronger variability across a circuit and thus potentially more variability at the higher abstraction levels (cells, circuits, etc.).

Note, that $t_{rel.spec}$, t_{EOL} and τ_{spec} are different concepts. For example, t_{EOL} is 10 Years, $t_{rel.spec} = 1s$ if LCS$= 1s$ (e.g. periodic microcontroller sleep). A $P_{fail} = 0.01$ results in $\tau_{spec} = 100s$ following eq. 4.29.

The reliability specification calculates $\Delta V_{th}(tran)$ for actual defect sets in transistors instead of worst-case defects (both in number and in individual degradation $\eta(d)$). This is feasible in terms of computational effort (shown later in Section 4.2.4.3) and severely reduces the pessimism involved in our upper bound. In complex circuits with millions to billions of transistors it is a significant overestimation to assume that all these transistors are unlucky during manufacturing and obtain the worst-case defects. In actual circuits, transistors with high $\Delta V_{th}(tran)$ occur next to transistors with low $\Delta V_{th}(tran)$, i.e. they average out in terms of circuit degradation. In Fig. 4.33 in Section 4.2.4.4 we show how this detail on the transistor level, ensures a correct guardband on the circuit level.

[3] Note, that our BTI model assumes no irrecoverable degradation to occur, which is in line with [31,37,47].

Term	Explanation
m	Number of Defects
d	Individual Defect
$\eta(d)$	Impact of Defect d in terms of ΔV_{th}
$occ(d)$	Occupancy State of Defect d (1 = Occupied with Carrier 0 = Without Carrier)
$\Delta V_{th}(d)$	Actual contribution of defect d to total ΔV_{th} of transistor
τ_{spec}	Maximum Capture Time used in Reliability Specification
τ_c, τ_e	Capture Time and Emission Time of a Defect
$D(\tau_e, \tau_c)$	Defect Distribution as a Function of Capture/Emission Time

Table 4.1: Definition and Explanation of Terms of this Section

Note that τ_c and τ_e are functions of voltage and temperature (V_{gs}, T). Therefore, the induced degradation $\Delta V_{th}(tran)$ determined by the reliability specification changes for different V_{gs} and T. In order to provide an upper bound, we assume worst-case temperature T and worst-case V_{gs} in the reliability specification (e.g. $80°C, 1.2V$). In modern micro-processors, the thermal- and power management techniques aim to optimize performance within the thermal/power budget and thus processors operate close their critical temperature and critical voltage continuously [136]. Therefore, only minor pessimism is introduced when considering worst-case V_{gs}, T.

Algorithm 3 Calculate $\Delta V_{th}(tran)$ for the transistors based on their reliability specification τ_{spec}.

```
for every transistor in circuit do
Require: $W, L, T, V_{gs}, \tau_{spec}$
    Generate Random $m$                                              ▷ See Eq. 4.21 and Eq. 4.22
    Generate Random $\tau_e, \tau_c$ from $D(\tau_e, \tau_c)$        ▷ See Eq. 4.23
    Place $m$ defects $d$ in $D(\tau_e, \tau_c)$
    Generate Random $\eta(d)$ for each $d \in D(\tau_e, \tau_c)$     ▷ Eq. 4.24
    for each $o_n$ in $D(\tau_e, \tau_c)$ do
        Check $\tau_c(T, V_{gs}) \leq \tau_{spec}$ for each $o \in D(\tau_e, \tau_c)$
        if $\tau_c(d) \leq \tau_{spec}$ then                         ▷ Defect below Rel.Spec.?
            Set $occ(d) = 1$                                         ▷ Occupied defect
            Set $\Delta V_{th}(d) = \eta(d)$                         ▷ Contribute to $\Delta V_{th}$
        else
            Set $occ(d) = 0$                                         ▷ Remain neutral
            Set $\Delta V_{th}(d) = 0$                               ▷ Do not contribute to $\Delta V_{th}$
        end if
        Set $\Delta V_{th}(tran) = \sum_{i=1}^{m} \Delta V_{th}(i)$
    end for
end for
```

Algorithm 3 details the process of calculating $\Delta V_{th}(tran)$ based on the reliability specification. First the number of defects m is randomly generated, based on the geometry W, L of the transistor following Eq. 4.21. Then capture τ_c and emission times τ_e are generated for each defect following the $D(\tau_e, \tau_c)$.

Then $\eta(d)$ for each defect d is randomly generated based on an exponential distribution and scaled with transistor geometry. After the initialization for the transistor is complete, each defect can be compared against given reliability specification τ_{spec} obtained from Eq. 4.29. If the capture time $\tau_c(d)$ (itself a function of T, V_{gs}) of the defect d is below the specification, then we consider it occupied $occ(d) = 1$ and its $\eta(d)$ contributing to $\Delta V_{th}(tran)$. If $\tau_c(d)$ is larger than τ_{spec}, then the defect is unlikely to be occupied within the lifetime of the transistor and thus it can be discarded $occ(d) = 0$, i.e. it does not contribute to $\Delta V_{th}(tran)$. Finally, the impact of all contributing defects is summed up to obtain the overall transistor degradation $\Delta V_{th}(tran)$.

Note, that both inter- and intra-transistor variability are included in the reliability specification. Different transistors have different set of defects, which corresponds to inter-transistor variability. Defects with a capture probability larger than the failure probability ($P_{capture} > P_{fail}$) are considered in the specification. For example, defects with $P_{capture} = 5\% > P_{fail} = 1\%$ at $t_{rel.spec}$ have $\tau_c \leq \tau_{spec}$ and thus their η added to $\Delta V_{th}(tran)$ accounting for probabilistic capturing, i.e. intra-transistor variability.

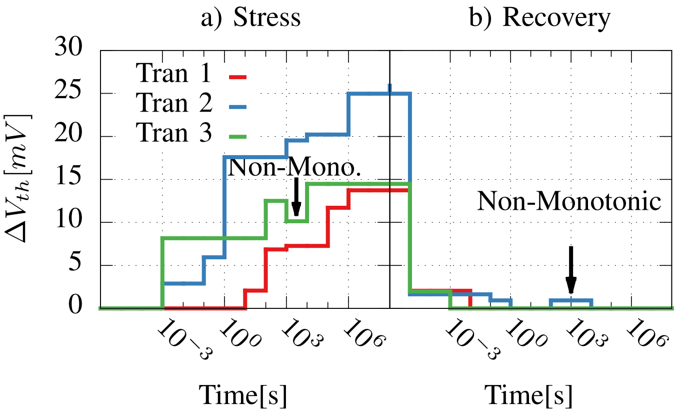


Figure 4.22: Stress and recovery waveforms for transistors with the same geometry (W, L). Due to a different number of defects with different (τ_c, τ_e, η), the observed ΔV_{th} is different despite identical stress and recovery stimuli. This illustrates how transistors, which are manufactured identically, behave differently under stress and recovery. Non-monotonic behavior is due to intra-transistor variability (see Fig. 4.23).

Inter-Transistor Variability: Different transistors exhibit different defects sets and thus their $\Delta V_{th}(tran)$ differs. As the number of defects n changes as well as the defect parameters (τ_c, τ_e, η), a stress curve (Fig. 4.22a) and recovery curve (Fig. 4.22b) features different ΔV_{th} jumps ($\eta(d)$) at different points in time (τ_c for stress and τ_e for recovery). So at a given point in time (e.g., after $10s$), all transistors look different, both from a microscopic perspective (defect occupancies) as well as from a macroscopic perspective (ΔV_{th}).

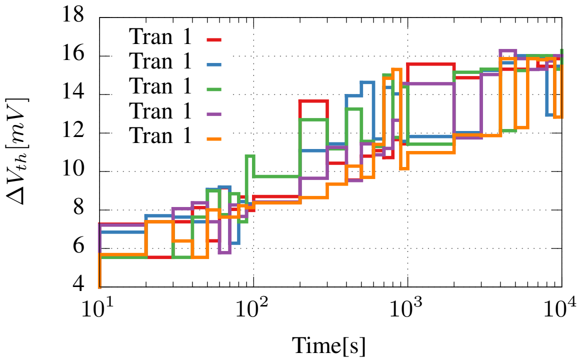


Figure 4.23: A single transistor exhibits different ΔV_{th} under the same stimuli (T, V_{gs}). As the defects are identical for all 5 shown waveforms, the ΔV_{th} fluctuations are of the same height η around τ_c, τ_e of a defect. Each defect can randomly capture or emit a carrier, changing if it contributes to $\Delta V_{th}(tran)$ or not. These random changes can result in non-monotonic ΔV_{th} waveforms.

Intra-Transistor Variability: The same transistor can also exhibit different degradations for the same stimuli. Capture and emission are probabilistic physical processes, i.e. they occur randomly and vary even under identical conditions. Fig. 4.23 shows the same transistor under identical stress with different ΔV_{th} curves.

Optimization Strategy: In order to reduce $\Delta V_{th}(tran)$ from the defect level perspective, it is better to provide an upper bound for continuous stress times. The first option is to check if such a bound intrinsically exists (e.g., transistors in clock tree switch continuously). Alternatively, one could employ a aging/power management technique to force a small LCS (e.g., periodic power gating). This guarantees that the reliability specification results in a low $\Delta V_{th}(tran)$.

4.2.4.3 Cell level

Standard cells consist of multiple transistors with various widths W and lengths L to provide the desired logical function. We use the 45nm NanGate open cell libraries [137], which stores the topology and transistor sizing in netlists. The first step is to annotate the netlist, so that each transistor has a unique name. Then our reliability specification is used to determine $\Delta V_{th}(tran)$ for each transistor in each cell (see Algorithm 3). Based on $\Delta V_{th}(tran)$ we modify the V_{th0} entry to change V_{th} of the transistor in $45\,\mathrm{nm}$ high-performance predictive technology[4] models [3]. Then we use Synopsys SiliconSmart [138] to characterize the entire cell library. See Fig. 4.16 for an overview of our cell level methodology.

Transistor Interactions: Transistors frequently oppose each other. For example, in an inverter during the logic switch from $0 \rightarrow 1$, the pMOS and nMOS are both conducting and pulling the output to V_{DD} and V_{SS}, respectively. Therefore, the actual worst case with respect to an output switching from $0 \rightarrow 1$ is a degraded (high ΔV_{th}) pMOS and a non-degraded nMOS (further details available in Section 6.1).

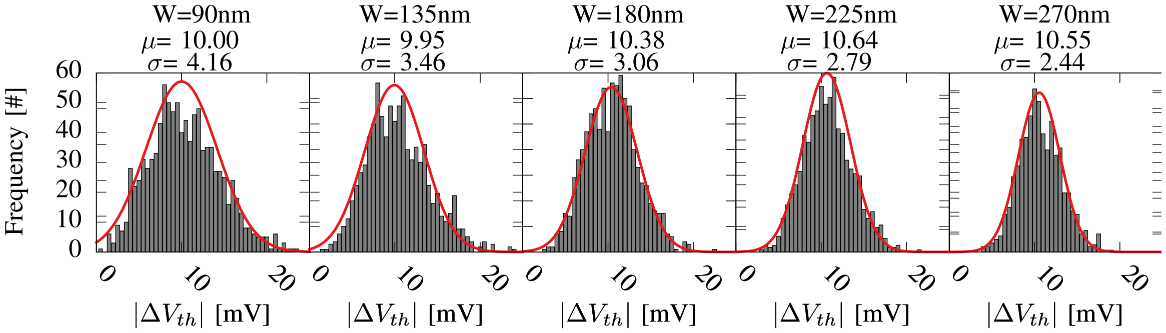


Figure 4.24: $\Delta V_{th}(tran)$ histograms for reliability specification of $t_{rel.spec} = 1year$ for transistors with $L = 45nm$ and shown W. Larger area results in more defects with lower impact $\eta(d)$ per defect, which in turn results in slight increase in the mean μ and a decrease in standard deviation σ of $\Delta V_{th}(tran)$ for higher W.

Transistor Sizing: Transistors have different defect distributions depending on their geometry. Larger transistors feature more defects, but a smaller impact $\eta(d)$ per defect d [113]. Fig. 4.24 shows $\Delta V_{th}(tran)$ for $t_{rel.spec} = 1year$ for $L = 45nm$ and increasing W. This figure, highlights how transistor sizing and thus cell design can affect defect variability directly. Therefore, it is important to take W, L of the transistors into account, when obtaining cell timing.

In order to obtain W, L of the transistors, the netlists of the standard cells in the cell library are parsed/read. Each netlist contains the number of transistors, their connections and the geometry of the transistors in each standard cell (compare 4.26a).

Cell Environment: Depending on the direct environment of the cell (neighboring cells, number of connected cells, etc.), the input signal slew and load capacitance of the cells change. Our previous work [82] showed that impact of these parameters cannot be ignored when estimating cell delays. Fig. 4.25 shows the relative difference in delay of two XOR2 cells ($k = 2$). As illustrated, the input signal slew and output load capacitance changes the impact of TDV, which highlights that cell environment cannot be neglected in TDV estimations.

[4] Our approach can analogously be applied to different technologies. Transistor modelcard and cell netlists are provided by the foundry and automatically considered by the cell characterization tool. Then the steps from Sections 4.2.4.3 and 4.2.4.4 are performed.

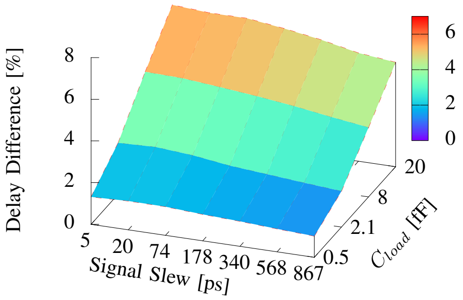


Figure 4.25: Dependence of variability of XOR2 delay on cell environment. The difference between two cell iterations in XOR for pin A and fall delay is shown. The difference between the two cells is not uniform across the cell environment, which highlights that, variability cell timing and thus guardbands must be estimated by taking the cell environment into account.

Cell Iterations: An overview of cell iterations is given in Fig. 4.16 and Fig. 4.26. In order to estimate TDV, we create k unique instances per cell called *cell iterations*. SiliconSmart parses a single copy of each cell, characterizes it in HSPICE (delay and power) and combines it into a liberty file. For k cell iterations, k cell libraries are obtained, which are all different. These k cell libraries are combined to a single *variability cell library*. The variability cell library is a combination of k unique cell libraries each containing a single unique instance per cell, which in turn consists of unique transistors (see Fig. 4.26).

The number of cell iterations k governs computational complexity, but also accuracy (as outliers/corner cases occur solely for high enough k), so its value is a trade-off, which must be carefully made. In order to verify if the chosen sample size k is large enough, we fitted normal distributions against the delay distributions of the cells. The process is visualized in Fig. 4.27 in which we fit distributions for the pin A $\rightarrow$ pin ZN, fall delay in XOR2 for various k. As shown, the mean value μ stabilizes ($\Delta\mu(\Delta k)$ small) quickly after just $k = 100$, but the standard deviation σ is not stable until we go from $k = 500$ to $k = 1\,000$. Therefore, we conclude that $k = 1\,000$ is a representative sample size for the XOR2 cell, i.e. $\Delta\mu(\Delta k)$ and $\Delta\sigma(\Delta k)$ are small enough and no additional benefits are to be expected going to larger sample sizes.

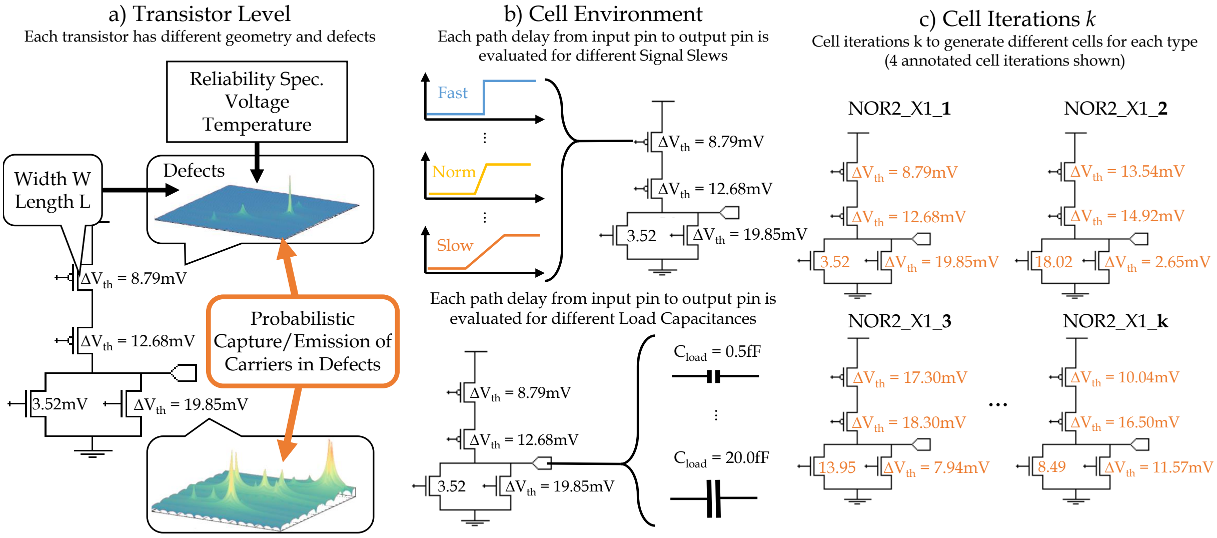

a) Transistor Level
Each transistor has different geometry and defects
Reliability Spec.
Voltage
Temperature
Width W
Length L
Defects
Probabilistic Capture/Emission of Carriers in Defects
ΔV_{th} = 8.79mV
ΔV_{th} = 12.68mV
3.52mV
ΔV_{th} = 19.85mV
b) Cell Environment
Each path delay from input pin to output pin is evaluated for different Signal Slews
Fast
Norm
Slow
ΔV_{th} = 8.79mV
ΔV_{th} = 12.68mV
3.52
ΔV_{th} = 19.85mV
Each path delay from input pin to output pin is evaluated for different Load Capacitances
C_{load} = 0.5fF
C_{load} = 20.0fF
ΔV_{th} = 8.79mV
ΔV_{th} = 12.68mV
3.52
ΔV_{th} = 19.85mV
c) Cell Iterations k
Cell iterations k to generate different cells for each type (4 annotated cell iterations shown)
NOR2_X1_**1**
ΔV_{th} = 8.79mV
ΔV_{th} = 12.68mV
3.52
ΔV_{th} = 19.85mV
NOR2_X1_**2**
ΔV_{th} = 13.54mV
ΔV_{th} = 14.92mV
18.02
ΔV_{th} = 2.65mV
NOR2_X1_**3**
ΔV_{th} = 17.30mV
ΔV_{th} = 18.30mV
13.95
ΔV_{th} = 7.94mV
...
NOR2_X1_**k**
ΔV_{th} = 10.04mV
ΔV_{th} = 16.50mV
8.49
ΔV_{th} = 11.57mV

Figure 4.26: Overview of our proposed methodology from the defect to the cell level. In a) the number n and properties of defects (τ_c, τ_e, η) are randomly generated based on W, L. Then the defect distribution $D(\tau_e, \tau_c)$ is used with the reliability specification $\tau_{rel.spec}$, voltage V_{gs} and temperature T to determine $\Delta V_{th}(tran)$. Every transistor has their unique defect set and therefore their unique $\Delta V_{th}(tran)$ given by the reliability specifications. Middle figure b) illustrates how each cell is evaluated - with respect to cell delay (rise/fall) from input to output pins - under different signal slews and load capacitances. Figure c) shows different cell iterations with unique transistors in each NOR cell. The generation of cells (shown in a) and the evaluation of each cell (shown in b) is repeated for each cell iteration j (shown in c) and each cell type (AND, NOR, etc.). This large one-time effort is necessary to capture the complex impact of variability, as for example 1000 cell iterations are used over 68 cells (NanGate library) under 49 different signal slew/load capacitance combinations for up to 12 input/output pin paths.

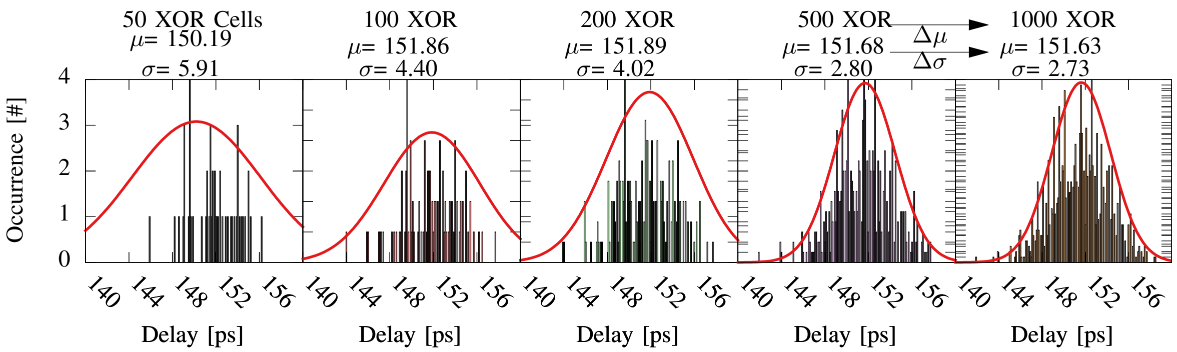


Figure 4.27: Evaluating the sample size for XOR gate. Below 500 samples the distributions are very rough and only at 1000 samples the distribution statistical representation is reached, as $\Delta\mu(\Delta k)$ and $\Delta\sigma(\Delta k)$ are small when switching $k = 500 \rightarrow 1000$, i.e. $\Delta k = 500$.

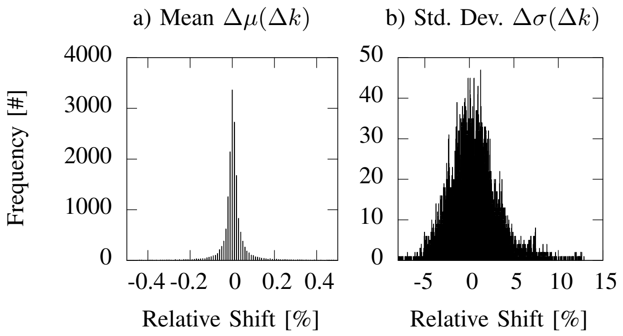


Figure 4.28: Estimation of all path delays (input to output paths, each for rise and fall delay) through all 68 cells for a total of 16 906 HSPICE simulations to evaluate if $k = 1\,000$ is representative. Histograms of the stability $\Delta\mu$ of the mean and the stability $\Delta\sigma$ of standard deviation, when switching $k = 500 \rightarrow 1\,000$. a) shows how the mean μ has fully stabilized and barely shift, while b) shows the majority of path through the cells are within $\Delta\sigma < \pm 3\%$. Therefore, $k = 1000$ is sufficient for our variability analysis.

To verify whether k=1000 is enough to get representative distributions in all cases (all paths in all cells for both rise and fall delay), we investigate the shift in the mean $\Delta\mu(\Delta k)$ and standard deviation $\Delta\sigma(\Delta k)$ of each delay when increasing $k = 500 \rightarrow 1\,000$. Fig. 4.28 shows the relative shift in $\Delta\mu(\Delta k)$ and $\Delta\sigma(\Delta k)$ stays for all 16906 path delays in the NanGate library. The mean $\Delta\mu(\Delta k)$ is stable with less than 0.2% (absolute) shift across all delays. Standard deviation $\Delta\sigma(\Delta k)$ is acceptable with most path delays below 3% shift and rare maximum at 12.8%.

Characterization (Intel Core i7-6700T, 32GB RAM, SSD) of the variability cell library for $k = 1000$ required approx. 35 hours with 10 parallel SiliconSmart instances. Note that this is a one-time offline effort, as the variability library can be re-used for many different circuits and all circuit iterations j. Furthermore, we provide these cell libraries upon publication so that others can start immediately without having to replicate our effort.

Optimization Strategy: Hardening cells in terms of defect variability can be performed by widening (increasing W) the transistors. To identify the most critical transistors in each cell type, the worst cell delay samples (right of histogram in Fig. 4.27) can be checked to find out which transistors caused this severe shift. Then these transistors can be widened to reduce variability (see Fig. 4.24) and increase output current I_D to tolerate degradations, following similar ideas as in our previous work [113]:

Assuming the transistor should deliver a given current I_D to switch fast enough, despite suffering from $\Delta V_{th}(tran)$, then W can be found based upon a first-order approximation for I_D:

$$I_D \approx \frac{\mu}{2} \cdot C_{ox} \cdot \frac{W}{L} \cdot (V_{gs} - V_{th})^2[1 + \lambda(V_{DS} - V_{DSsat})] \tag{4.31}$$

$$\Rightarrow \Delta W = C_1 \cdot \frac{1}{(C_2 - \Delta V_{th})^2} \tag{4.32}$$

with $C_1 = I_D \cdot \frac{2}{\mu \cdot C_{ox}} \cdot L \cdot [1 + \lambda(V_{DS} - V_{DSsat})], C_2 = V_{gs}$ as two constants (as none of the voltages change).

Additionally to our previous work, Fig. 4.24 shows that the standard deviation of $\Delta V_{th}(tran)$ is smaller for wider transistors. Therefore, increasing W should be performed iteratively. First W_{orig} is replaced with up-scaled $W_1 = W_{orig} + \Delta W$ according to Eq. 4.32. Then $\Delta V_{th}(tran(W_1))$ is determined with algorithm 3 and down-scaled to $W_2 = W_1 - \Delta W$ with Eq. 4.32, as $\Delta V_{th}(tran(W_1)) < \Delta V_{th}(tran(W_{orig}))$ and thus requires less width to compensate the degradation. Repeat until this has converged (e.g., $\Delta W < 1nm$) to obtain suitable hardened W to maintain I_D under TDV.

With the modified netlist (wider transistors) k cell iterations can be run and by evaluating the histogram, the designer can verify if he optimized the cell sufficiently to be more resilient against defect variability with less timing guardband but additional area and power overhead due to wider transistors.

Comparison to Traditional Statistical Cell Characterization: Cell characterization tools like SiliconSmart offer support for statistical cell characterization. Variability can be expressed in terms of parameter variability on the transistor level and then the cell characterization tool will characterize the cells under these variations. While this seems very similar to our work, there are several drawbacks to the existing EDA tools. First, transistor parameters can only be varied by Gaussian distributions. While this is useful for most TZV, since most manufacturing imperfections manifest themselves as Gaussian or near-Gaussian distributions of transistor parameters, it does not match TDV. TDV, especially BTI, is not Gaussian distributed [34] [113].

The second limitation is that the variations are uniform for all transistors. Each transistor exhibits different TDV-induced degradation, due to their geometries (W and L) as well as their stress conditions (V and T). As mentioned in previous Section 4.2.4.2, transistors can exhibit significantly different BTI due to differences in their geometry. In existing cell characterization tools, it is not possible to alter the variability based on the geometry or stress conditions of the transistor. This leads to severe under- or over-estimation of BTI on the transistor-level.

The last limitation is that not all cell metrics are expressed with respect to variability. SiliconSmart does only provide the resulting distributions for propagation delay, but other metrics like the power consumption of the standard cells remain at nominal value. It is currently impossible to study the impact of variability on the power consumption in standard cells with cell characterization tools. Existing works have to implement their own circuit simulations (e.g., SPICE) to evaluate power under variability.

Therefore, current cell characterization tools are not suited to study the impact of TDV on standard cells.

Necessity for Brute-Force Monte Carlo Approach: While tools like SiliconSmart offer non-Monte-Carlo-approaches to consider variability during the cell characterization (called sensitivity analysis), this is not possible when considering TDV. As mentioned before, TZV depends on manufacturing and is uniform for all transistors in all cells. Therefore, elegant solutions can be found to avoid simulating every single cell in every possible combination of transistor variability. However, in TDV the transistors have unique distributions based on their geometries and stress conditions. As each transistor is now uniquely affected, we cannot make any assumptions about the induced degradation or the impact of unique degradations on the cell's delay and power. Interpolation across the signal slew and load capacitance table is equally wrong and any clustering/grouping scheme for transistor TDV-induced degradations. Therefore, if each transistor has to be treated as a unique entity, brute-force Monte Carlo approaches are the only option.

4.2.4.4 Circuit Level

Our final goal is to obtain a guardband for complex circuits like processors, in order to tolerate TDV. In this section we discuss several approaches to obtain a guardband with variability simulations at the circuit level and at the end of the section we explain our two novel worst case cell libraries.

Guardband: A guardband is over-designing the circuit, so that degradations can be tolerated without the circuit violating any constraints (e.g., timing or performance constraints). In this work, we define the guardband as a timing guardband as time slack on top of the clock period of the circuit:

$$t_{clk} = t_{delay} + t_{GB} \tag{4.33}$$

The guardband t_{GB} is added to the maximum delay (path delay of critical path) of the circuit t_{delay}, so that as long as $\Delta t_{delay} \leq t_{GB}$ it can be guaranteed, that no timing violations can occur. It is crucial that t_{GB} is carefully selected, as a too large guardband results in large performance loss, while a too small guardband may result in timing violations.

Reference: In order to estimate the guardband, we need to establish a reference, i.e. estimate $t_{delay}(ref)$ (see Fig. 4.32). This reference is the golden sample, the perfect circuit free of any degradations. For TDV this corresponds to $\Delta V_{th}(tran) = 0$ for all transistors. The benchmark circuits were synthesized with Design Compiler with "ultra compile" and the $45\,\mathrm{nm}$ NanGate Library [137] including all 68 cells. In the static timing analysis we obtain $t_{delay}(ref)$, i.e. the fastest clock our circuit can achieve when absolutely no degradation is present. In order to cover a wide range of circuit sizes and cell types, we evaluate RISC microprocessor Rocket with $>$1 000 000 cells [139], then a discrete cosine transformation (DCT) with $>$20 000 complex cells ($>$ 20% DFFR) and b19 $>$10 000 simple cells (AND, OR, etc.) from ISCAS99/ITC99 [140]. For state of the art, we use constant degradation of $\Delta V_{th} = 63\,\mathrm{mV}$ uniformly applied to all transistors. The value $63\,\mathrm{mV}$ correspond to 10 year operation [117]. It results in approximately 10% reduction in the I_{Dsat} (transistor drain current in the saturation mode), which is the industry definition of end of lifetime [127]).

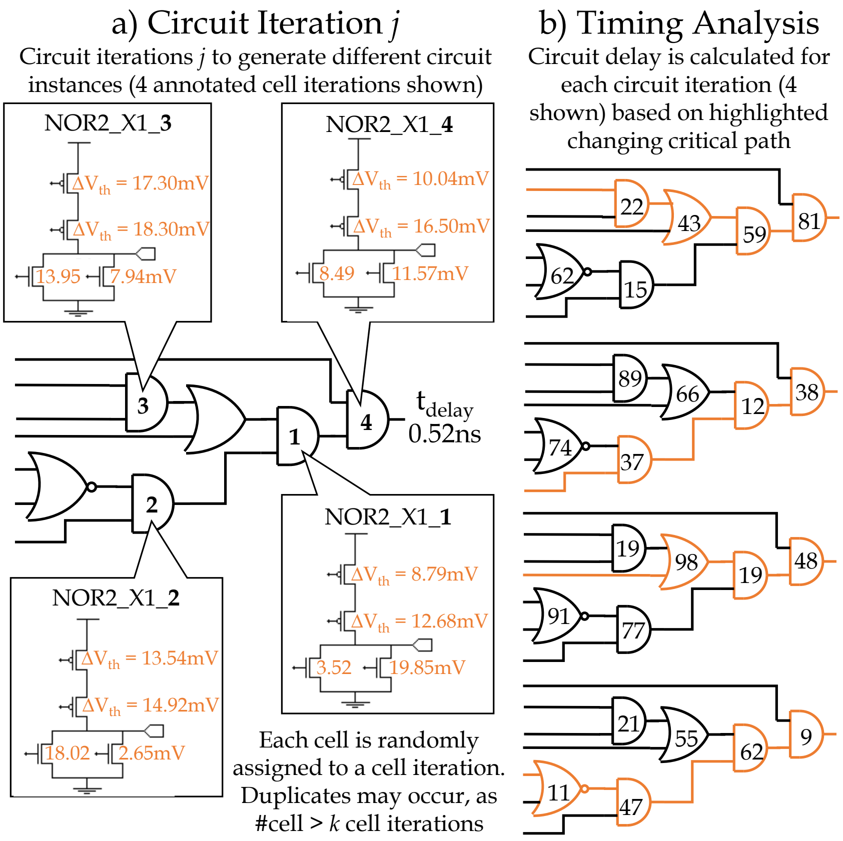

a) Circuit Iteration j
Circuit iterations j to generate different circuit instances (4 annotated cell iterations shown)
NOR2_X1_3
ΔVth = 17.30mV
ΔVth = 18.30mV
13.95
7.94mV
NOR2_X1_4
ΔVth = 10.04mV
ΔVth = 16.50mV
8.49
11.57mV
3
4
1
2
tdelay
0.52ns
NOR2_X1_1
ΔVth = 8.79mV
ΔVth = 12.68mV
3.52
19.85mV
NOR2_X1_2
ΔVth = 13.54mV
ΔVth = 14.92mV
18.02
2.65mV
Each cell is randomly assigned to a cell iteration. Duplicates may occur, as #cell > k cell iterations
b) Timing Analysis
Circuit delay is calculated for each circuit iteration (4 shown) based on highlighted changing critical path
22
43
59
81
62
15
89
66
12
38
74
37
19
98
19
48
91
77
21
55
62
9
11
47

Figure 4.29: Overview of our proposed methodology on the circuit level. For each circuit iteration j shown in a), each cell is randomly assigned to a cell iteration k. In b) the timing analysis is shown, which estimates the delay of the circuit based on the critical path of the circuit. Note, that the critical path (drawn in orange) of the circuit changes based on assigned cells due to the variability, even if the topology is identical. Each of the k timing analysis of a circuit uses a unique assignment/mapping of cell indices to the cells within the circuit.

Circuit Iterations: The previous steps culminate in a variability cell library with k instances for each cell. This library can then be used in timing analysis for circuits. In order to mimic TDV, we randomly assign each cell with index between $[1 : k]$ for each circuit instance. The timing analysis tool (e.g., Synopsys PrimeTime) picks up the corresponding cell (e.g., AND2_X1_37) instance for each cell it encounters and uses their delays to estimate the timing of the entire circuit. This approach is visualized in Fig. 4.29b, where 4 circuit iterations are shown with indices $j \in [1 : 100]$ inside of the cells. Fig. 4.29a illustrates how each index corresponds to a cell with unique non-uniform ΔV_{th} per transistor and therefore a unique degradation/delay pattern. Due to our compatibility with existing EDA tool flows, the mature timing analysis tool automatically takes care of complex tasks, like determining the critical path and extracting the delay for each cell under its cell environment (signal slew, load capacitance).

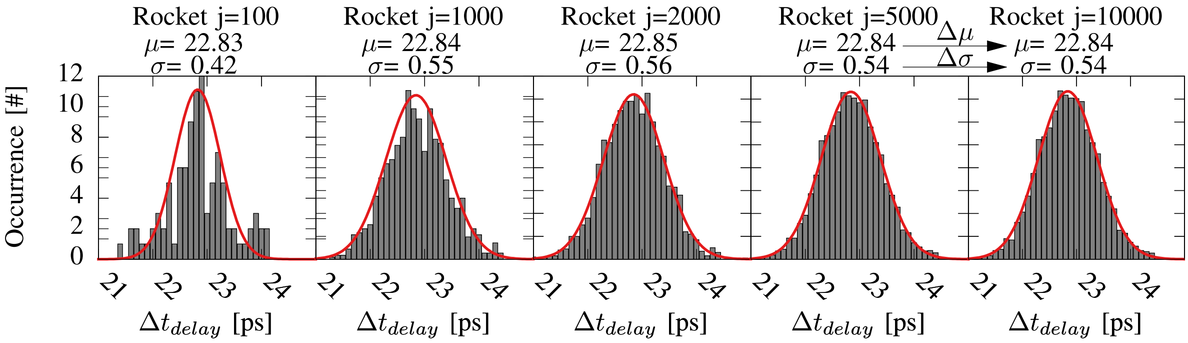


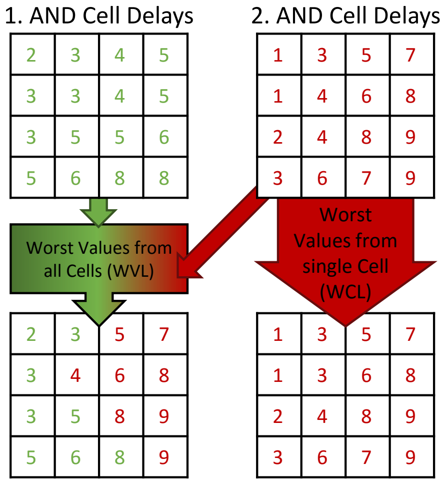
Figure 4.30: Evaluating the sample size j for circuit iterations. 1000 samples is the minimum, while from 5 000 samples onwards the simulations are representative as $\Delta\mu(\Delta j) = 0$ and $\Delta\sigma(\Delta j) = 0$ for $j > 5000$.

Circuit iterations enable us to explore the resilience of the circuit against variations induced by TDV. However, similarly to cell iterations, the number of circuit iterations j should be sufficient to be representative, i.e. j should be chosen large enough that mean μ and standard deviation σ of the circuit delays stabilize. Analogously to cell iterations $\Delta\mu(\Delta j)$ and $\Delta\sigma(\Delta j)$ are minimized by increasing j. In Fig. 4.30 the distributions are shown for different j. For $j > 5\,000$ the distribution stabilizes and both mean μ and standard deviation σ are representative, i.e. $\Delta\mu(\Delta j) = 0$ and $\Delta\sigma(\Delta j) = 0$. Each circuit iteration requires a timing analysis run on a different annotated/indexed circuit netlist. In total, the time consumed is approx. 6 hours for $j = 5\,000$. Therefore, end-to-end time consumption for a new technology including the one-time-effort cell characterization is approx. 41 hours. However, as we already performed $j = 10\,000$ simulations to verify the representativeness of j in all 3 circuits, we present results in Fig. 4.32 and 4.33 for $j = 10\,000$.

Worst-Case Libraries for fast Guardband Estimation: In order to provide a guardband, a circuit designer needs to simulate a representative number of circuit iterations j, extract the delay distribution for the circuit and then extract t_{GB} based on that delay distribution (e.g., with 6σ or the 95-percentile). While this Monte Carlo approach works, we propose a significantly faster approach by extracting worst-case cells from the variability cell library. We create two separate worst-case libraries, each representing a different worst-case scenario (see Fig. 4.31). The first worst-case cell library is the *worst-case value library* (WVL), in which for each cell the worst value (rise/fall path delay entries of that cell) is chosen for every cell environment (i.e. 7x7 delay matrix). Each cell within the WVL is the combination of the worst values found for all k cell iterations of that cell. This cell library can be used within a timing analysis to guarantee in a single timing analysis (i.e. $j = 1$) that an upper bound for delay $t_{delay}(WVL)$ is found, which reduces the computational complexity significantly. The cell library is generic and a one-time effort, as the same library is used in the timing analysis for different circuits (in this work Rocket, b19 and DCT). By estimating $t_{delay}(WVL)$, we can find t_{GB} with:

$$t_{GB}(WVL) = t_{delay}(WVL) - t_{delay}(ref) \tag{4.34}$$

Note that this includes aforementioned (Section 4.2.4.3) transistor interactions. In our inverter example, cell iterations with high ΔV_{th} in the pMOS and low ΔV_{th} for the single nMOS define the rise delay, while for fall


1. AND Cell Delays
2. AND Cell Delays
Worst Values from all Cells (WVL)
Worst Values from single Cell (WCL)

1. AND Cell Delays

2	3	4	5
3	3	4	5
3	5	5	6
5	6	8	8

2. AND Cell Delays

1	3	5	7
1	4	6	8
2	4	8	9
3	6	7	9

Worst Values from all Cells (WVL)

2	3	5	7
3	4	6	8
3	5	8	9
5	6	8	9

Worst Values from single Cell (WCL)

1	3	5	7
1	3	6	8
2	4	8	9
3	6	7	9

Figure 4.31: Difference between the 2 worst-case libraries. The *worst-case value library* (WVL) on the left combined the worst value for each entry within the delay table of a cell, while the *worst-case cell library* (WCL) contains the worst cells as a whole without values from other cells.

delay low ΔV_{th} in pMOS and high ΔV_{th} nMOS cells are taken. These cell iterations feature the worst delays and thus populate the WVL delay tables. For complex cells, complex interactions with elaborate ΔV_{th} mappings are intrinsically considered as the algorithm picks from $k = 1\,000$ cell iterations.

The main disadvantage of the WVL is, that it is overly pessimistic, i.e. overestimates t_{GB}. As we combine the worst values from different cell iterations, we create contradictions which could never occur. Again in our inverter example, we have high ΔV_{th} in pMOS for rise delay with low ΔV_{th} in pMOS for fall delay, a direct contradiction. These contradictions result in cell delays, which are impossible to occur in a single real cell. However, they incorporate the transistor interactions to guarantee that actual cells can neither exceed rise nor fall delay, therefore providing a solid upper bound.

Our second worst-case library is the *worst-case cell library* (WCL), in which the worst cells are found based on a fitness function. So from all the k cell instances present in the variability cell library due to k cell iterations, we take the cells according to the following function:

$$F(c) = \sum_{s=0.5ps,l=0.5fF}^{s=867ps,l=20fF} t_{delay}(s,l) \tag{4.35}$$

with c as our cell, $F(c)$ our fitness function, s signal slew and l load capacitance. We sum up all delay values in the tables (rise/fall delay for all paths) to find the cell with the highest sum. Therefore, we find the cell, which has the highest *average delay* without weighting individual cell environments, paths or rise/fall more than others. $F(c)$ determines the worst cell and copies its values to the WCL. The resulting cell library contains the worst cells, i.e. for each cell (the entire matrix as a whole) the worst cell from all k cell iterations is selected. These cells are still realistic (i.e. contains no contradictions as in WVL). This is a weaker upper bound compared to WVL, as individual cells with a heavy skew towards rise delay, could exceed the rise delay of the worst cell, even though the worst cell is worse on average. However, large circuits (like our targeted microprocessors) have long critical paths with many elements, in which cells average out (cell with high rise delay connected to fast rise cells, etc.). Therefore, WCL is frequently sufficient and less pessimistic, but for guarantees, only WVL can be used. The guardband can analogously be found by $t_{GB} = t_{delay}(WCL) - t_{delay}(ref)$.

The resulting guardbands t_{GB} from WVL, WCL and the constant uniform $\Delta V_{th} = 63\,\mathrm{mV}$ can be seen in Fig. 4.33. The guardband for $63\,\mathrm{mV}$ is either under- (for DCT) or over-estimating (for Rocket and b19) TDV. Underestimation leads to timing violations, while the overestimation leads to inefficient systems with high performance loss. Our WCL/WCL libraries are comparable for b19, but further apart for DCT and far apart for Rocket. By design, WVL is always worse than WCL. This highlights how important it is to consider transistor interactions for absolute reliability guarantees with respect to TDV. However, by comparing Fig. 4.33 with Fig. 4.30 we can see that even for $j = 10\,000$ WCL (based on $k = 1\,000$) would be sufficient for reliable operation. WCL employs $t_{GB}(Rocket) = 37$ ps, which is more than sufficient for the 10 000 circuit iterations shown in Fig. 4.30, which at most reaches 25 ps degradation. Compared to the $t_{GB}(Rocket) > 130$ ps based on the constant $63mV$ our WCL approach reduces the guardband by 72%, allowing 100 ps smaller clock periods and thus 11% higher performance.

Optimization Strategy: The circuit designer can manually optimize his circuit with respect to defect variability. But, manual optimization requires searching through $j = 1000$ circuit iterations and manually optimizing synthesis by adding constraints. This manual process is error-prone and not effective, as synthesis is heuristic and may not react to the constraints as desired. Instead, we propose an automatic approach to make the synthesis aware of the degradations introduced by variability (i.e. *variability-aware synthesis* (VAS)). The synthesis algorithm performs the circuit optimization/hardening by itself when synthesizing with the WCL or WVL (similar to our work on aging in [82]). The synthesis tool synthesizes with degraded delay tables from WCL/WVL and thus optimizes cell selection and circuit design for performance in the presence of variability. If a cell was heavily degraded (under a given cell environment) it becomes unfavorable for the synthesis tool (due to prolonged delays) and other options are evaluated by the synthesis algorithm. Therefore, the algorithm intrinsically optimizes the design and selects cells, which maintain small delays in the presence of TDV.

By performing variability-aware synthesis (VAS) with b19 against our WVL library, the synthesis tool intrinsically optimizes the circuit design against defect variability, resulting in additional 48% reduction in the required timing guardbands (see b19-VAS in Fig. 4.33). Synthesizing with WCL results in a 57% reduction in the required timing guardbands.

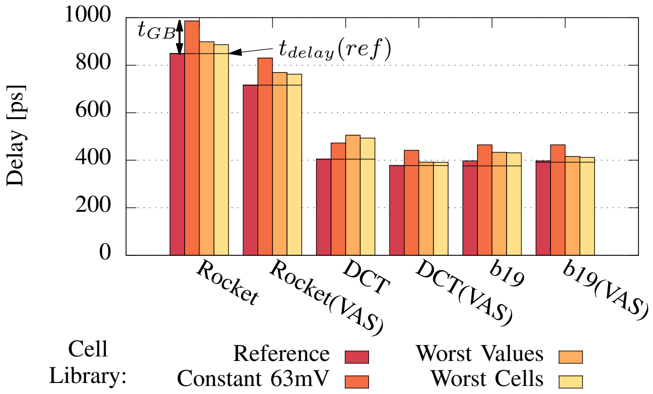


Figure 4.32: Delays for different cell libraries in 3 different circuits. First the RISC microprocessor Rocket [139], then a discrete cosine transformation (DCT) and b19 from ITC99/ISCAS99 [140]. A constant degradation of $63mV$ in each transistor, fails to provide a guarantee in terms of reliability, as it underestimates TDV in DCT. At the same time, it is also inefficient, as both our worst-case approaches provided a smaller (yet safe) guardband for Rocket and b19. In addition, we show the delays of b19 synthesized with variability-aware synthesis (VAS) (see optimization strategy in Section 4.2.4.4).

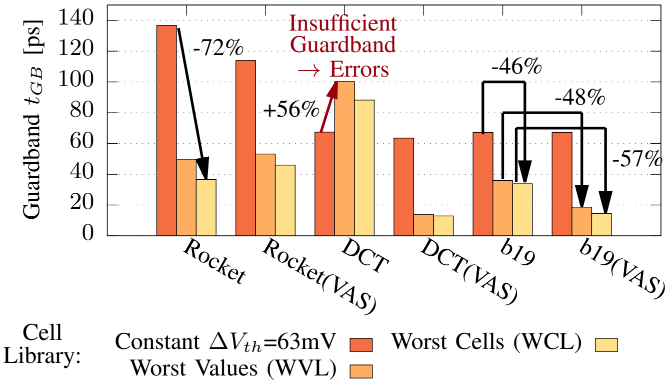


Figure 4.33: Guardbands for different cell libraries in three different circuits. Employed our WCL over state of the art, results in a 72% reduction of the guardband. At the same time, in the DCT a constant degradation to all transistors fails to capture transistor interactions, which underestimates TDV by 56%, potentially resulting in errors. Following our variability-aware synthesis (VAS) optimization strategy, we can additionally reduce the guardband by 48%, respectively 57%.

Comparison to Traditional Static Timing Analysis (STA): Traditional STA offers support for variability studies. STA tools like PrimeTime or PrimeTimeVX use the distributed timing information from cell libraries to estimate timing under variability. Variability-aware cell libraries can currently only support Gaussian distributions in delay. Neither a variation in power nor other distributions are supported. The impact of TDV can be Guassian distributed (see Fig. 4.27), but does not have to be, as this depends on the circuit topology and transistor-level distributions of TDV. Should TDV result in non-Gaussian distributions, TDV cannot be represented correctly in traditional static timing analysis tools.

Additionally, even if TDV is perfectly distributed, the different worst-cases are not correctly encapsulated with PrimeTimeVX. These tools are meant to provide an overview of the impact of TZV variability on the circuits timing. Our approach provides two separate worst-case estimations with our WVL and WCL cell libraries. STA based on traditional statistical cell libraries inaccurately represents TDV, as only mean and standard deviation of a Gaussian distribution are saved and not the absolute worst-case (i.e. missing an accurate representation of the tails of these Gaussian distributions). Cell library corners can unfortunately also not be used to study TDV. The slow-slow (SS) corner of a cell library contains the absolute worst-case with respect to many effects. SS includes TDV, but also TZV, voltage-droops, temperature-effects and many more. Furthermore, it provides a single worst-case point instead of a distribution. Therefore, circuit optimization with sensitivity analysis (e.g., finding circuit elements exhibiting smaller variations due to insensitivity to variability) is not possible. Lastly, variability in power cannot be represented in LVF. Thus, the impact of TDV on power cannot be studied in current EDA STA tools.

Necessity for Brute-Force Monte Carlo Approach: Existing EDA tools offer solutions to avoid costly Monte Carlo Simulations. However, these solutions have strong limitations forcing us towards a Monte Carlo approach. For example, distributions in power are not supported in the LVF format (the statistical extension of the standard cell liberty file format), the tools cannot import the variability information when it comes to power. Therefore, we have to perform simulations on a instance by instance basis to get thousands of power reports to then gather the power information and fit a distribution.

4.3 Integrating Self-Heating as a new Phenomena

Self-Heating is a relatively recent phenomenon, as it relied on the introduction of FinFET transistors (see Section 2.3.4). This contribution highlights how self-heating can be modeled in a $7\,\mathrm{nm}$ FinFET technology. As solely modeling SHE in transistors and small circuits (e.g., ring oscillators) has been performed by the reliability physics community, we used larger circuits to highlight our modeling capabilities. Additionally, this section highlights a frequent misconception between the clock frequency of a circuit and switching frequency of its transistors, which is critical in SHE modeling.

This work employed the circuit framework (see Section 8.1.3.3) to explore SHE in standard cells and large-scale circuits under different workloads. This section only illustrates its use, more details on how SHE is integrated in the circuit framework is given in Section 6.2. This section is based on my publication [141].

4.3.1 Introduction

The Self-Heating Effect (SHE) is a primary reliability concern in the current and future semiconductor technologies [56, 142]. FinFETs embed the channel within an insulating material (gate dielectric) to improve the electrostatics compared to planar MOSFET, yet considerably limit the dissipation of generated heat (see Section 2.3.4 for more details). This results in increased channel temperature T_C, which, in turn, alters the key electrical characteristics of FinFETs such as threshold voltage V_{th}, carrier mobility μ (see Section 2.3.3). a strong frequency dependence in which it is diminished above a certain high frequency [67], i.e. the resulting temperature increase in the channel ΔT_C becomes insignificant. Therefore, state of the art considers SHE negligible in digital circuits [142] because such circuits typically operate in a GHz range.

This work is the first work to demonstrate that the majority of switching frequencies of transistors within a processor are significantly lower than the clock frequency due to the important role of workload-induced activities. We show how transistors in a processor exhibit a wide variety of SHE-induced ΔT_C, demonstrating the necessity to consider the workload-induced activities when it comes to SHE modeling.

As mentioned before, this section focuses on the impact of workload on SHE. For more details on SHE in circuits (especially standard cells) and the SHE simulation framework, see Section 6.2.

4.3.1.1 Technology Dependence of SHE

SHE-induced ΔT_C depends on two primary aspects: a) generated heat and b) heat dissipation. The generated heat corresponds to the power loss across the channel $P_{loss} = I_D \cdot V_{DS}$ [142]. Whereas, heat dissipation is governed by the thermal resistance (R_{th}) and thermal capacitance (C_{th}) of the FinFET device [53]. Both R_{th} and C_{th} are technology dependent as they are determined by the dielectric material, fin geometry, contact resistance, etc. [142] and as such they change *solely* with technology scaling [59] and workloads have no impact on them.

4.3.1.2 Workload Dependence of SHE

Transient heat dissipation depends on the thermal time constant $\tau_{th} = R_{th} \cdot C_{th}$ [58]. For different switching frequencies f_{sw} and duty cycles (on-/off-ratio) λ in FinFETs, different amounts of heat will be generated. For high f_{sw}, the generated heat is rapidly dissipated as C_{th} behaves as a short-circuit and thus lowers the thermal impedance $Z_{th} < R_{th}$ [67]. Conversely, for low f_{sw}, C_{th} behaves as an open connection and thus $Z_{th} \approx R_{th}$, limiting heat dissipation. For more details on the f_{sw} dependence of SHE, see Section 6.2.4.1.

Next to f_{sw} induced by workloads, this work exposes the importance of λ. Typically, SHE in circuits is evaluated based on ring oscillators (e.g., [59, 142]), which they exhibit λ of 0.5. However, for a given f_{sw}, λ governs how

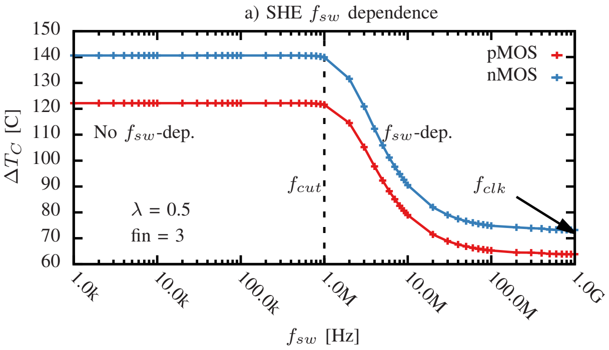
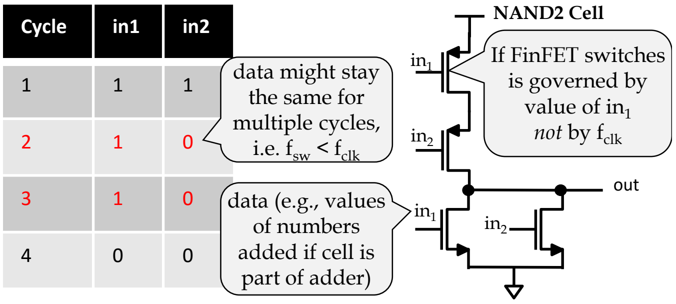


Figure 4.34: a) SHE-induced channel temperature increase ΔT_C as a function of switching frequency f_{sw} of transistors. The temperature drops at the cut-off frequency F_{cut} due to the influence of C_{th}. Below f_{cut} little dependence on the switching frequency is observed, as C_{th} is fully charged (heated) or discharged (cooled down) at those low f_{sw}. b) ΔT_C as a function of λ at different switching frequencies f_{sw}. Lower f_{sw} results in longer ON-times t_{on} charging C_{th}, saturating T_C earlier w.r.t λ.

long C_{th} is charged (heating period, while transistor is ON) or discharged (cooling period, while transistor is OFF). Thus, the impact of SHE on the resulting ΔT_C is directly affected by the workload-induced λ (see Fig. 4.34 and Section 4.3.5). As λ exhibits a wide range in digital circuits (see Section 4.3.5), it cannot be ignored. In summary, both f_{sw} and λ are driven by the induced activities by running workload (see Section 4.3.3), but they are typically uncorrelated.

4.3.1.3 Workload-induced f_{sw} instead of f_{clk}

In digital circuits, f_{clk} are in the GHz range. Therefore, SHE is considered to not be an issue for digital circuits, as Z_{th} approaches low values due the short-circuit via C_{th} at such high frequencies [67] [142]. As a matter of fact, the f_{sw} of a transistor is conceptionally different from f_{clk} (see Fig. 4.35). f_{clk} determines switching in the clock tree, while f_{sw} varies for transistors in the rest of circuit. A large portion of transistors in processors switches significantly below the operating frequency $f_{sw} \ll f_{clk}$ (see Fig. 4.36) because their switching is determined by data that is being generated by workload (e.g., intermediate calculation results) and then later processed by the processor. For example, despite that an SRAM cell might operate in a GHz-range, their transistors forming the cross-coupled inverters will *only* switch if the stored data flips. The latter is completely driven by the running workload. Another example is when an Adder circuit continuously adds small numbers (e.g., $5 + 7$). Hence, its upper bits in the adder will always remain logic 0 even though the Adder circuit itself operates in a GHz range. Thus, the cell which are connected to these upper bits, do not switch at f_{clk} but remain. Therefore, f_{sw} is entirely governed by the workload of the circuit and may reach f_{clk}, yet frequently stays far below f_{clk}.



Cycle	in1	in2
1	1	1
2	1	0
3	1	0
4	0	0

data might stay the same for multiple cycles, i.e. $f_{sw} < f_{clk}$
data (e.g., values of numbers added if cell is part of adder)
NAND2 Cell
If FinFET switches is governed by value of in_1 *not* by f_{clk}
in_1
in_2
out
in_1
in_2

Figure 4.35: Schematic of a NAND2 Cell and its corresponding input data for 4 clock cycles. The switching frequency f_{sw} of the FinFET within the cell depends on the inputs, which is data-dependent (e.g., 5 + 7 = 12 with 5 and 7 as input data for an adder) and does not directly depend on f_{clk}. As data might not change each clock cycle f_{clk} is merely an upper bound for f_{sw}. Therefore, it is important to observe the actual bit switches (changes in in_1 and in_2) according to the running workload.

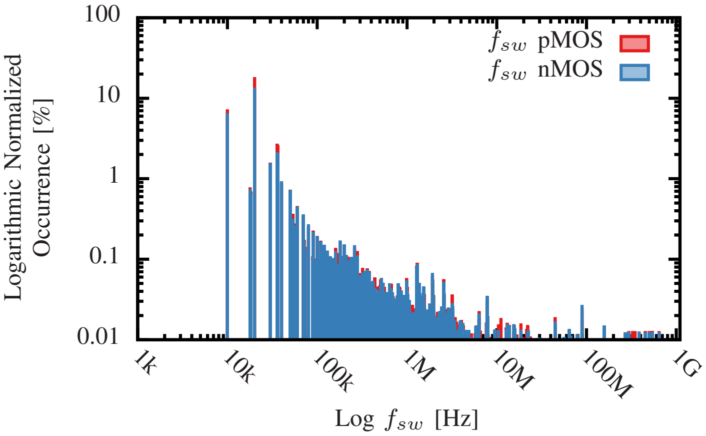


Figure 4.36: f_{sw} histogram for all transistors within the processor (PULP) for 10 different benchmarks (application/workloads). Only a very small subsection (<0.1%) of transistors reaches $f_{sw} = f_{clk} = 1.1$GHz. Therefore, typically $f_{sw} \ll f_{clk}$. At the same time, SHE depends heavily on f_{sw} (see Fig. 4.34a). Therefore, demonstrating $f_{sw} \ll f_{clk}$ validates the necessity to consider the workload in SHE-analysis. For details in experimental setup see Section 4.3.3.

4.3.2 Related Work

For other works see Section 3. To distinguish this work from other works, here is a short overview:

- First work to present the impact of duty cycle λ and actual workload-driven switching frequency f_{sw} on SHE.
- This is the initial report for the impact of SHE on large circuits like microprocessors.

Note, that Section 6.2 covers SHE in transistors and standard cells in more detail as this Section focuses on the impact of the workload on SHE.

4.3.3 Workload Extraction

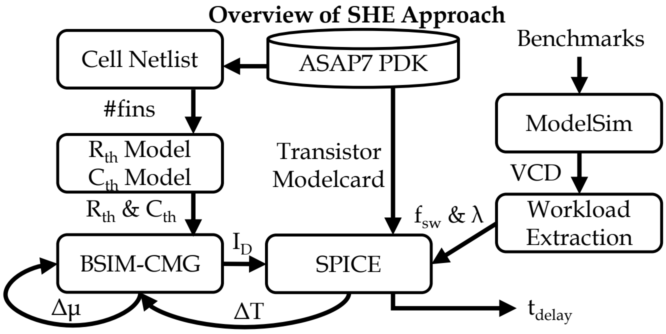


Figure 4.37: We use HSPICE v2017-03.2 which incorporates BSIM-CMG v110 [53] at $V_{DD} = 0.7V$ and $T_{ambient} = 27°C$ (both nominal for [143]). Workload data is extracted from gate-level simulator ModelSim [144], which simulates the processor's netlist (featuring 62 different standard cells and over 250k transistors) at the gate level (boolean representations of signals). f_{sw} is determined by extracting toggle rates (#switches) and dividing it by the total number of cycles. The PULP processor [145] is synthesized with the ASAP7 PDK [143]. The SHE model is the RC-model from BSIM-CMG v110 [53], which is calibrated with 7nm SHE data (R_{th}, C_{th} models) from [59]. u

An overview of our approach is shown in Fig. 4.37. We synthesize the RTL of a processor for maximum performance with the highest optimazation effort. Then, we simulate the generated netlist in a gate-level simulator [144] while a workload is being executed. This provides the signal activity waveform per input/output for every standard cell within the processor's netlist. Afterwards, we extract the signal activity waveform per transistor based on the SPICE netlist of every cell, which are provided inside the PDK. Then, f_{sw} and λ are calculated for every transistor to be used in SHE analysis.

We use Synopsys Design Compiler set to a target delay of 0 (i.e. minimum delay and hence maximum performance) at the "Ultra Compile" (highest effort) optimization setting. Our circuit simulator is HSPICE v2017-03.2 along with the BSIM-CMG v110 transistor compact model. The transistor modelcards are obtained from the 7nm FinFET PDK ASAP7 [143].

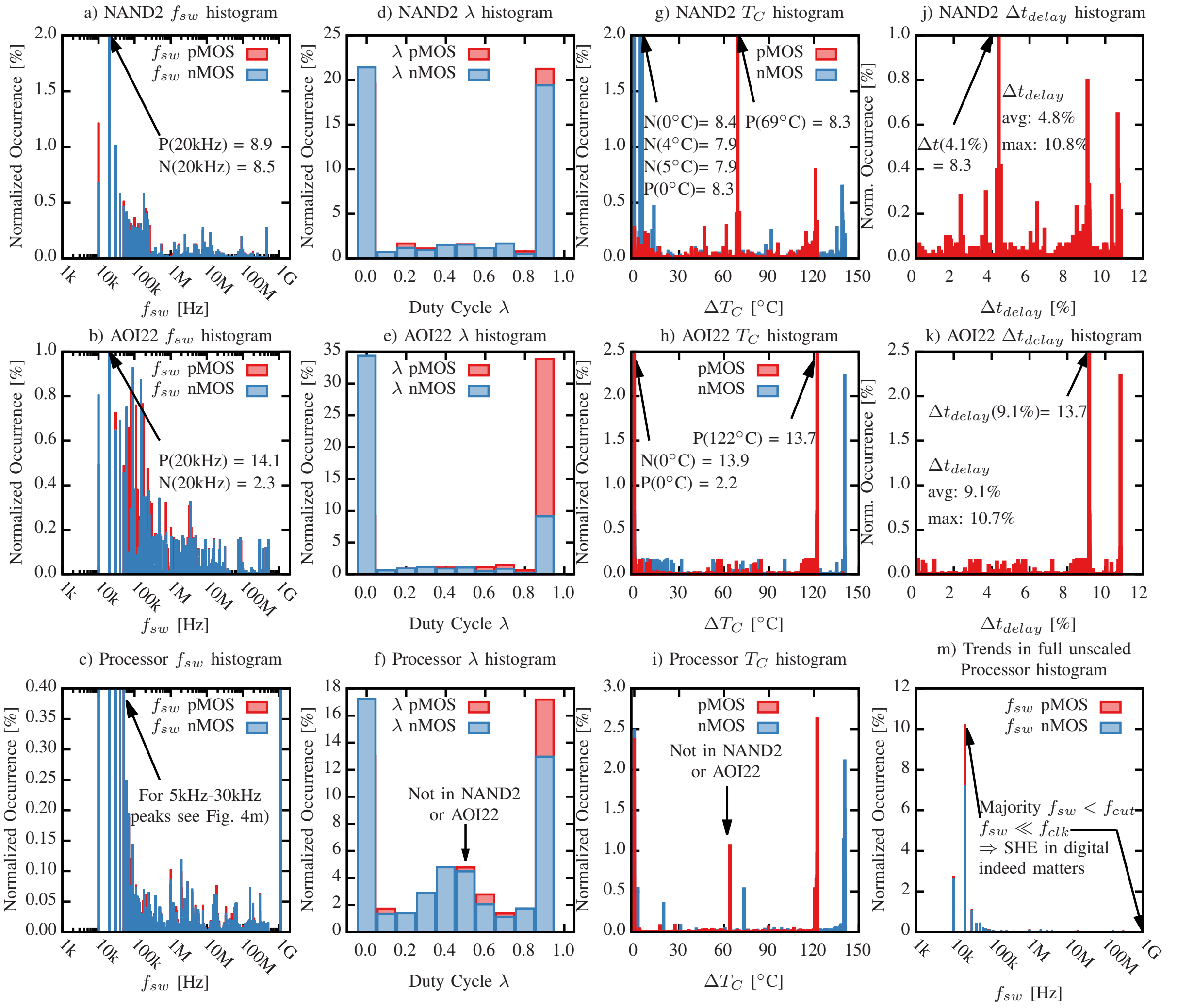


Figure 4.38: Figures (a-c), (g-i) and (j-k) are scaled below the highest peaks for clarity. Note that some peak values (marked with arrows) are 6-7x higher than the figure itself. (d-f) and (m) are unscaled. (a-c) Switching frequency f_{sw} histogram for transistors in standard cells and all transistors in PULP processor. (d-f) Duty cycle λ histogram for transistors in standard cells or processor. (g-i) Channel temperature T_C histogram for transistors in standard cells or processor. (j,k) Propagation delay increase Δt_{delay} histogram, highlighting impact of SHE on digital circuits even if they operate in the GHz range. The wide distribution from 0% to 10.8% highlights the necessity of taking the workload dependence of SHE into account. (m) Full f_{sw} histogram for entire PULP processor running "MatrixMul" to highlight how for the majority of transistors f_{sw} is significantly below f_{clk} and even below f_{cut}. Therefore, SHE matters in digital circuits. However, as seen in i) some transistors are directly connected to f_{clk} (e.g., clock buffers) and thus the workload dependence of SHE must be considered.

4.3.3.1 PULP 32-bit RISC-V Processor

In this work, we study the PULP 32-bit RISC-V processor [145] operating at $f_{clk} = 1.1$GHz in the employed 7nm technology [143]. For reference, a Bubblesort execution results in power values of 4.6mW on average over time and peak power consumption of 57.8mW in an area of 3316μm^2.[5] Fig. 4.36 shows the overall f_{sw} for 10 varied applications [145] being executed on top of the processor. The majority of the transistors (note the logscale)

[5] The power and area numbers are obtained by the synthesis tool and change during ongoing design phases like place and route.

exhibit $f_{sw} < 50$kHz, which is significantly below $f_{clk} = 1.1$GHz. Approximately 0.018% of transistors reach full $f_{sw} = f_{clk} = 1.1$GHz. Therefore, $f_{sw} \ll f_{clk}$ when considering workloads in processors, highlighting the importance of taking the workload into account.

4.3.3.2 Standard Cells NAND2 and AOI22

For a detailed look within a standard cell, see Section 6.2.4.2, this section focuses on workload impact on SHE. Fig. 4.38(a-b) show the extracted f_{sw} for the two most frequent cells within the processor (NAND2 and AOI22 - a combined AND, OR, INV) executing the "bitdescriptor" workload. Fig. 4.38c shows resulting impact of "MatrixMul" workload on on the all processor transistors. Fig. 3(a-c) demonstrate the similarities to Fig. 4.36d, i.e. they are representative. Fig. 4.38(d-f) show the corresponding λ. The peak at low $f_{sw} = 20$kHz manifests themselves as two peaks of relative static behavior with $\lambda < 0.1$ and $\lambda \approx 0.9$.

4.3.3.3 All Standard Cells from Cell Library

All standard cells combined exhibit a wide variety of λ and f_{sw}. Fig. 4.38f shows how when studying an entire cell library, all the cells together exhibit more activity (higher f_{sw} compared to NAND2 and AOI22). This is because some cells in the cell library (e.g., clock buffers) are directly connected to the clock. Fig. 4.38f features a hump around $0.2 < \lambda < 0.6$ not present in Fig. 4.38d and 4.38e, which highlights how some cells feature transistors with more balanced λ due to this higher activity.

4.3.4 Modeling Self-Heating

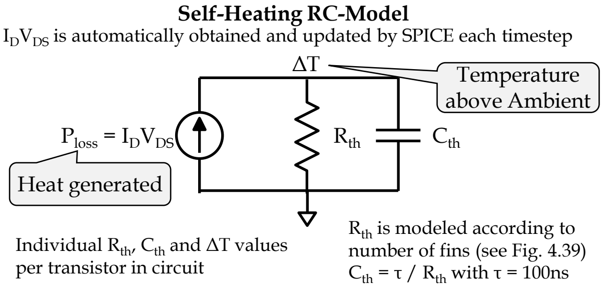

Self-Heating RC-Model
I_DV_{DS} is automatically obtained and updated by SPICE each timestep
ΔT
Temperature above Ambient
$P_{loss} = I_DV_{DS}$
R_{th}
C_{th}
Heat generated
Individual R_{th}, C_{th} and ΔT values per transistor in circuit
R_{th} is modeled according to number of fins (see Fig. 4.39)
$C_{th} = \tau / R_{th}$ with τ = 100ns

Figure 4.39: RC-thermal network is shown, which acts as a low-pass filter at cut-off frequency f_{cut}. Therefore, digital circuits are frequently not considered critical w.r.t. SHE, as they operate in the GHz range.

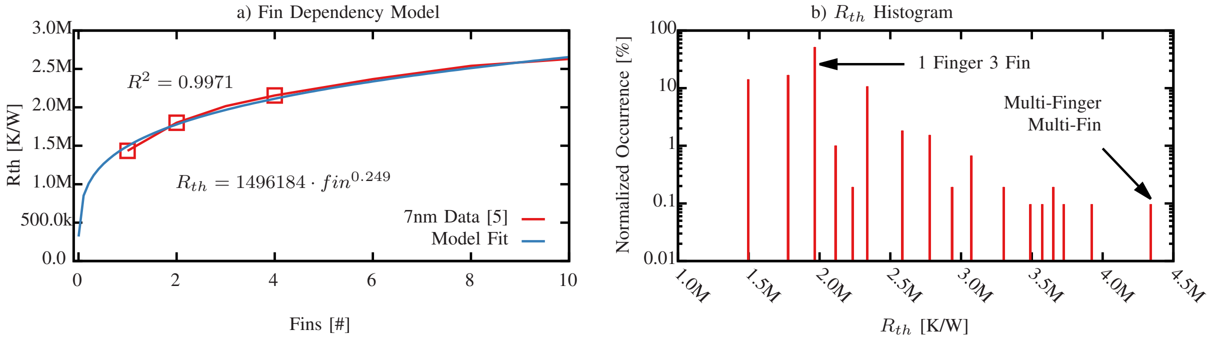


Figure 4.40: a) Fitting of the fin dependence to 7nm FinFET data [59] with a exponential function. The function fits both the data (red squares) and the function plotted in [59] well. b) R_{th} histogram for all transistors within the processor. The transistors stem from all the cells in a cell library and are from single-fin single-finger transistors all the way to multi-fin multi-fin FinFET. With higher number of fins, the substrate is heated by the neighboring fins, increasing R_{th}. C_{th}-scaling is performed according to the total number of Fins, i.e. each fin has identical C_{th} and accumulates to $C_{th.total}$.

The employed SHE model is an RC-thermal network from the industry standard BSIM-CMG [53] shown in Fig. 4.39. The BSIM-CMG parameters R_{th0} and C_{th0} are determined based on R_{th}, C_{th} data from [59]. We modeled R_{th} as a function of #fins according to [59]. As BSIM-CMG implements exponentially (ASHEXP and BSHEXP):

$$R_{th}(fins) = \frac{R_{th0}}{W_{th0} \cdot fingers^{BSHEXP} + fpitch \cdot fins^{ASHEXP}} \tag{4.36}$$

an exponential fit seemed fitting. This fit matched out data well, as shown in Fig. 4.40a). τ_{th} is considered to be $100ns$ according to [58, 59].

$$C_{th} = \frac{\tau_{th}}{R_{th}(fins)} \tag{4.37}$$

The calibrated R_{th}, C_{th} model in BSIM-CMG uses parameters presented in Table 4.2.

Table 4.2: SHE Parameters for 7nm FinFET based on data from [59]. R_{th0} and C_{th0} are BSIM-CMG parameters and do not correspond to actual R_{th} or C_{th}.

Parameter	1 Fin	3 Fins	7 Fins
R_{th0}	5.44×10^{-2}	1.63×10^{-1}	2.95×10^{-1}
C_{th0}	1.83×10^{-6}	6.12×10^{-7}	3.37×10^{-7}
W_{th0}	0	0	0
$fpitch$	2.7×10^{-8}	2.7×10^{-8}	2.7×10^{-8}
ASHEXP	0.249	0.249	0.249
BSHEXP	1	1	1

The SHE-induced rise in channel temperature ΔT_C is shown in Fig. 4.38(g-i). As T_C depends on activity (f_{sw},λ) featuring peaks, the figures equally exhibit large peaks. Note, that nFinFET and pFinFET exhibit different peaks based on the topology of the cells. The NAND2 cell shown in Fig. 4.35 has sequential pFinFET but parallel nFinFET resulting in different f_{sw}, λ distributions and thus cooler nFinFET and hotter pFinFET transistors. Transistors within the AOI22 cells (shown in Fig. 4.38h) are non-switching (constant ON or OFF resulting in $\lambda \approx 0$ or 1). Hence, either low $\Delta T_C = 0°C$ or very high $\Delta T_C = 140°C$ is caused as a result. This demonstrates the importance of studying the workload dependence, as the induced activities determine the severity of SHE.

4.3.5 Impact of SHE on FinFET and Standard Cells

To study the impact of SHE on circuits (e.g., on propagation delay), we used a commercial SPICE [92]. As SPICE solves BSIM-CMG, it models both SHE itself (estimating ΔT_C) as well as the consequence implications of SHE on the FinFET parameters and thus the circuit's delay. The impact of SHE on a standard cell is a rise in its propagation delay t_{delay} due to ΔT_C-induced degradations in the FinFET (e.g., mobility degradation). To evaluate t_{delay}, we simulated rise and fall delay in SPICE for the output pins of the cells at workload-induced λ and f_{sw} at the input pins. Within this SPICE simulation, these activities at the input pins result in SHE-induced ΔT_C in the FinFET, which prolongs the rise/fall delays of the cells by Δt_{delay}.

A wide distribution across transistors can be observed in Figs. 4.38(j-k). NAND2 exhibits Δt_{delay} from 0% up to 10.8% with a clear peak at 4.1% in Fig. 4.38j. The AOI22 cell has the peak at 9.1% in in Fig. 4.38k and is thus affected twice as much on average. Both cells have their peak at approximately 11%. This demonstrates that SHE can induce a 10% shift in delay in digital circuits under actual workloads due to immediate impact of elevated temperatures up to $\Delta T_C = 140°C$ even though the processor is being operated at 1.1GHz. When considering other effects like the acceleration of other degradation effects (BTI, HCI) due to higher channel's temperature [142] [56], the severity of SHE in digital circuits becomes even more serious.

In summary, we demonstrated how running workloads play a major role in SHE because they determine the switching frequency as well as duty cycle of transistors. We presented that the switching frequency of the majority of the transistors in a full processor is in a kHz-range even though the clock frequency of processor is in a GHz-range.

5 Step Two - Accelerating Degradation Models

After the degradation models are now unified into a single model and updated to include the newest phenomena (variability and SHE) from step one (see Section 4), the models must be accelerated. For their models, reliability physicists care about accuracy and matching models to experimental data. Performance is not an issue as they typically model a single transistor or simple circuits like ring oscillators as proof of concept. Hence, the models tend to be accurate but also incredibly complex and thus computational intense. The computational effort to calculate even a single transistor can be immense and on regular desktop PCs exceeds 5 minutes. Therefore, large-scale circuit simulations featuring thousands or even millions of transistors are entirely unfeasible.

This chapter features two contributions to accelerate degradation modeling without losing accuracy. The first contribution is to simplify degradation modeling by solely considering what is necessary for guardband estimations. Physics-based degradation models can be used to provide detailed insights into the response of degradation phenomena to external stimuli, but for the purposes of guardband estimations in custom reliability estimations it is sufficient to calculate the peak degradation. Thus, calculations can be significantly simplified without any loss in accuracy, as intermediate results are not necessary for guardband estimations. This significantly speeds up the computation of degradation models.

The second contribution is to utilize the power or massively parallel compute hardware in the form of graphic cards. Since thousands or millions of transistors need to be estimated in large-scale circuit simulations, the massive parallelism of thousands of compute cores in graphic cards can be fully utilized. This accelerates the estimation further, again without any loss in the accuracy of the estimation of degradation phenomena and their impact on transistors.

5.1 Simplifying Aging Models with Peak Calculations based on Longest Continuous Stress

This section is based on my publication [118].

5.1.1 Bias Temperature Instability is a Key Reliability Issue

Modeling and mitigating aging effects are key challenges of this decade since reliability must not be compromised, while the current nano-CMOS is highly susceptible to aging. Bias Temperature Instability (BTI) is recognized as one of the major aging phenomena due to its considerable ability to degrade the electrical characteristics of MOSFETs. To sustain reliability, aging degradations need to be estimated at design time in order to design the required guardband (i.e. designing the system above specification to tolerate degradation) that protects circuits against aging effects.

The major challenge is that BTI-induced degradations are estimated solely regarding its well-known *long-term* impact. The implications of *short-term* BTI on circuits are unexplored. In fact, reliability physics report that BTI consists of instantaneous (i.e. sub μs) frequency dependent processes, which were uncovered due to advances in measurement techniques [50] (explained in detail in section 5.1.2). The impact of instantaneous BTI is considerable and such instantaneous shifts may suddenly violate the employed guardbands manifesting itself as BTI-induced errors.

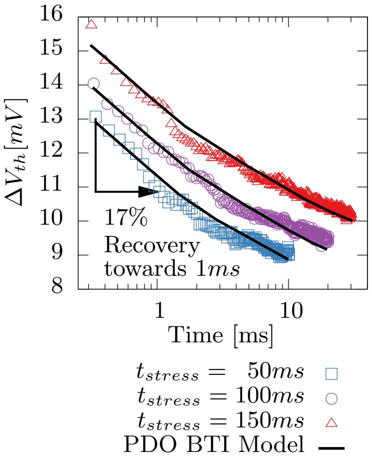


Figure 5.1: BTI recovery, measured on our ultra-fast measurement equipment, validating the short-term behavior of the base BTI model [31] we rely upon in our implementation. Additionally, our measurement highlights the importance of UF measurements as BTI recovers 17% degradation from $0.34ms \rightarrow 1ms$ after stress. Indicating how slow measurement missed the actual impact of BTI.

In order to prevent guardband violations due to instantaneous BTI, a new aging estimation approach is required that considers instantaneous and long-term BTI jointly to design guardbands protecting against both. There are two major challenges: First, guardbands cannot be further increased to incorporate new BTI effects. Actually, with each new technology generation the available design space for guardbands shrinks as different sources of reliability degradation phenomena (aging, noise, process variation, etc.) increase, while the resiliency against them decreases with the decrease in supply voltage [64]. Containing guardbands within available design space (i.e. designing *narrow guardbands*), requires accurate models along with a design methodology that replaces worst-case assumptions with actual occurring aging to minimize overestimation. The second challenge is that BTI-induced degradations must be estimated for instantaneous and long-term BTI at the micro-architecture level, which requires a fast, i.e. computational lightweight model to model such complex circuitry in feasible simulation times.

Our novel contributions:

1. We present a *physical BTI model* incorporating both instantaneous and long-term effects of BTI. It is computationally lightweight to be feasible estimating complex circuitry, while it retains the accuracy of physical models.

2. Adapting existing aging mitigation techniques to reduce the guardbands further by jointly reducing the stimuli of long-term and instantaneous aging.

5.1.2 Instantaneous BTI

BTI is stimulated by transistor activity, i.e. the transistor degrades when it is on (i.e. under stress for time t_{stress}) and recovers when it is in off-state (i.e. in recovery for t_{rec}). Activity waveforms, i.e. series of t_{stress} & t_{rec}, can be summarized with the on-/off-ratio λ and the frequency of the state changes.

5.1.2.1 Exposing Instantaneous BTI

Traditionally, BTI is measured with the measure-stress-measure (MSM) pattern, i.e. stressing the device for t_{stress} and then removing the stress for $t_{measure}$ (from $1ms$ to $1s$) from the device to characterize the device parameters (e.g. ΔV_{th}) [50]. Then ultra-fast (UF) measurement techniques were introduced [146] in which $t_{measure} < 1ms$. UF measurements uncovered that BTI does not solely accumulate over time, degrading reliability, but additionally reacts *instantaneously* (i.e. sub-microsecond) to stimuli with degradation or recovery. In fact, after the stress was removed in MSM, BTI partially recovered from high levels of degradations altering the perception of BTI from its

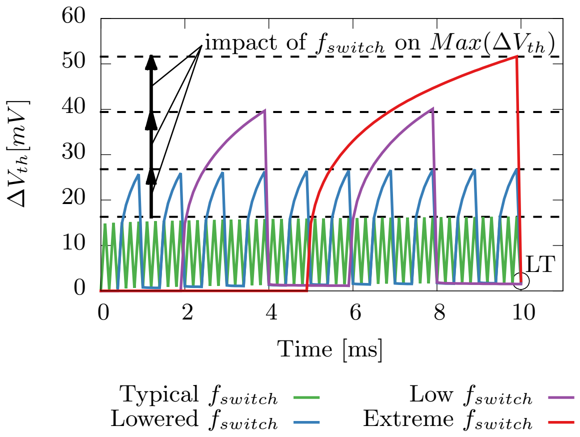


Figure 5.2: Frequency Dependence of BTI ($\lambda = 0.5, T = 80°C, V_{dd} = 1.0V$) with rising $Max(\Delta V_{th})$ for lower f_{sw}. Frequency independent long-term point marked with "LT". f_{sw} based upon Fig 5.5.

actual instantaneous nature. Fig 5.1 shows recovery below $1ms$ measured by UF measurements highlighting how 17% degradation may be missed when $t_{measure} = 1ms$.

5.1.2.2 Frequency Dependence of Instantaneous BTI

We differentiate two frequencies, the operating frequency $f_{operation}$ of the circuit (i.e. the clock frequency) and the switching frequency f_{sw} of individual transistors (i.e. the frequency of transistor switches from on to off states).[1] Note, f_{sw} is only loosely coupled with $f_{operation}$. For example, a memory cell storing the same data or clock gated logic, does not switch ($f_{sw} = 0Hz$) regardless of $f_{operation}$. Therefore, most significant bits of memory or an ALU, storing same values for prolonged times [73], switch infrequently ($f_{sw} \ll f_{operation}$).

Long-term BTI is frequency independent, i.e. multiple waveforms with identical λ but varying f_{sw} lead to identical results [73]. With identical λ, the stress/recovery ratio is fixed and e.g. longer stress phases are compensated by longer recovery phases, ultimately resulting in the same BTI-induced degradation.

Instantaneous BTI, on the other hand, is frequency dependent [147], [8]. As the physical processes of BTI are in reality instantaneous, individual stress phases itself lead to considerable degradations. Instantaneous recovery cannot compensate, if a single instantaneous stress phase already violates the guardband.

Fig. 5.2 shows the BTI-induced degradation ΔV_{th} for three different f_{sw} but identical λ. Instantaneous BTI exhibits degradation peaks after each stress phase. Intolerable degradation ($Max(\Delta V_{th})$) is reached at low f_{sw}, forcing the consideration of instantaneous BTI. Other observations are, that degradation decreases at higher frequencies (also see Fig. 5.3) like [8], [147] reported. At the same time, long-term BTI degradation (marked with "LT") exhibits the same ΔV_{th} when all frequencies are in phase, i.e. being f_{sw} independent as [73] claimed.

Note, that claims like frequency independence due to high $f_{operation}$ [71] are incorrect, as f_{sw} in memory cells, clock gated logic or the most significant bits switch at significantly lower frequencies than $f_{operation}$, i.e. $f_{sw} \ll f_{operation}$ (see Fig. 5.5).

5.1.3 Our Proposed BTI Model

In order to estimate guardbands for circuit designs, a BTI model must consider long-term *and* instantaneous BTI, be computationally lightweight for the feasible employment in complex circuitry, while having a low uncertainty for *narrow guardbands*. We therefore enhanced the physical model from [31] to directly calculate maximum

[1] Note, that f_{sw} is different from the number of switches (toggling rate) used for hot carrier modeling.

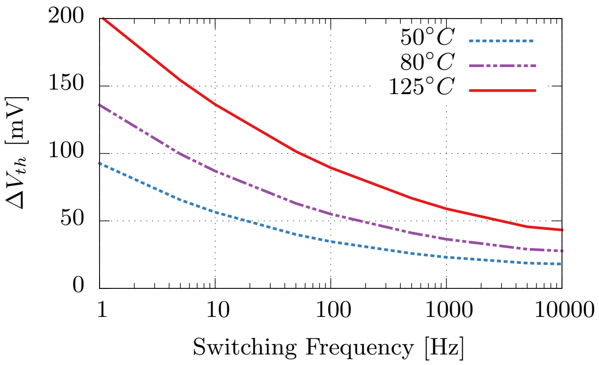


Figure 5.3: ΔV_{th} due $1000s$ with $\lambda = 0.5$ and 1 period continuous stress with $t_{stress} = \frac{1}{f_{sw}}$ for 3 temperatures.

degradation $Max(\Delta V_{th})$ based upon λ and f_{sw} instead of waveforms with stress time t_s / recovery time t_r. Designing guardbands requires solely $Max(\Delta V_{th})$ occurring during the desired lifetime of the circuit (t_{life}), so the tedious calculation of ΔV_{th} over time could be removed. The model is able to predict our UF measurements well (see Fig. 5.1) which validated that it can model instantaneous BTI. Both the original and reshaped model calculate ΔV_{th} by integrating over the defect distribution D and the occupancy probability P_{occ} of the defects. The latter was replaced in our implementation, to calculate only $Max(\Delta V_{th})$.

5.1.4 Original BTI Model

The probabilistic defect occupancy model (PDO) [31] calculates ΔV_{th} for a given temperature T, a voltage V, stress time t_s and recovery time t_r (details in Section 2.2.1.3.2 and 4.1.5.1):

$$\Delta V_{th}(t_s, t_r) = N \cdot \overline{\eta} \int_0^\infty \int_0^\infty D(\tau_e, \tau_c) \cdot P_{occ}(\tau_c, \tau_e; t)\, \mathrm{d}\tau_e \mathrm{d}\tau_c \tag{5.1}$$
$$\text{with } \tau_c = \tau_c(T, V) \text{ and } \tau_e = \tau_e(T, V)$$

BTI is modeled by integrating over two distributions. First $D(\tau_e, \tau_c)$ as the defect distribution, i.e. the distribution of defects characterized with their carrier capture τ_c and emission τ_e times. This characterization of the defect distribution is performed with measurements of the gate dielectric at different T,V as τ_c and τ_e are dependent on the temperature T and voltage V applied to the transistor [73].

The second distribution is the occupancy map P_{occ}, i.e. the occupation probability of a defect given by the current and past activity of the transistor. For a given stress time t_s, all defects with $\tau_c < t_s$ have likely captured a carrier and therefore contribute with their ΔV_{th} towards the overall ΔV_{th}. Then for a given recovery time t_r, all defects currently occupied due previous stress phases and $\tau_e < t_r$ likely release their carrier, i.e. do not contribute to the overall ΔV_{th} any more. According to [31] P_{occ} for digital voltage waveforms can be expressed as:

Stress:

$$P_{occ}(t) = P_{occ}(t_i) + \left(\frac{\tau_e}{\tau_e + \tau_c} - P_{occ}(t_i)\right) \cdot \left(1 - e^{\frac{t_i - t}{\tau_{sr}}}\right) \tag{5.2}$$

Recovery:

$$P_{occ}(t) = \frac{\tau_e}{\tau_e + \tau_c} + \left(P_{occ}(t_i) - \frac{\tau_e}{\tau_e + \tau_c}\right) \cdot \left(e^{\frac{t_i - t}{\tau_{sr}}}\right) \tag{5.3}$$

$$\text{with } \tau_{sr} = \frac{1}{\frac{1}{\tau_e} + \frac{1}{\tau_c}} \quad \tau_c = \tau_c(T, V) \quad \tau_e = \tau_e(T, V)$$

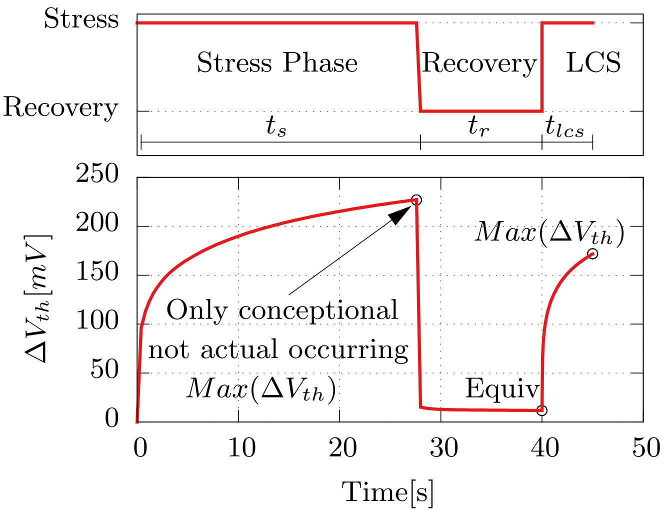


Figure 5.4: Top Plot: Stress and recovery states are illustrated annotated with stress phase t_s, recovery phase t_r and longest continuous stress (LCS) phase t_{lcs}. Bottom plot: Corresponding ΔV_{th} with the equivalent long-term point marked "Equiv" and the final result marked "$Max(\Delta V_{th})$". Note: Unrealistic values used in plot to show principle, as in actual systems $t_{lcs} \ll (t_s, t_f)$. For actual results see Fig. 5.6.

with t_i the time of the i-th switch between on and off state.

Integrating over both distributions from 0 to ∞, captures the impact of all occupied defects (defects capturing a carrier), i.e. contributing their ΔV_{th} to the overall ΔV_{th}.

5.1.4.1 Long-Term Phases

In our estimation of BTI-induced $Max(\Delta V_{th})$, we split the calculation in two parts. First, we estimate the long-term BTI-induced degradation for the desired lifetime t_{life} of the circuit. Then in a second step, we consider the degradation due to instantaneous BTI on top of long-term BTI.

Section 5.1.2.2 explained how long-term BTI is frequency independent if λ remains identical. Therefore, we can model the history of the transistor with overall λ for the entire lifetime and then we can replace the waveform with solely two data points: First a *stress phase* for time t_s, then a *recovery phase* for t_r:

$$t_s = \lambda \cdot t_{life} \tag{5.4}$$

$$t_r = (1 - \lambda) \cdot t_{life} \tag{5.5}$$

5.1.4.2 Longest Continuous Stress Phase

To account for the instantaneous effects of BTI, we introduce the *longest continuous stress (LCS) phase*. After the *stress* and *recovery phase*, we stress the transistor for longest continuous stress occurring in the activity waveform ($max(t_{stress})$):

$$t_{lcs} = max(t_{stress}) \tag{5.6}$$

$$\text{or } t_{lcs} = t_{period}(min((f_{sw})) = \frac{1}{min(f_{sw})} \tag{5.7}$$

Using the longest continuous stress, ensures we catch the worst shift due to instantaneous BTI as longer stress result in higher ΔV_{th} (see Fig. 5.3). By placing the frequency phase at the end, we mimic the occurrence of the longest continuous stress phase at t_{life}, i.e. the worst instantaneous shift occurs on top of the worst long-term degradation ensuring that guardbands can tolerate long-term and instantaneous BTI jointly. As most computing systems are periodical, i.e. execute the same tasks regularly, $max(t_{stress})$ occurs periodically. Therefore $max(t_{stress})$ occurs also at end of lifetime t_{life}, indicating that our worst-case assumption is not an overestimation in most computing systems.

The final activity waveform is shown in Fig. 5.4. On the top, an activity waveform for $\lambda = 0.7$, $t_{life} = 40s$, $f_{sw} = 0.2Hz \rightarrow max(t_{stress}) = 5s$, $T = 80°C$, $V = 1.0V$ is shown, resulting in $t_s = 28s$, $t_r = 12s$ and $t_f = 5s$.

Note that, the end-point in the bottom plot $Max(\Delta V_{th})$ is not the highest plotted ΔV_{th}. Stress phase t_s is solely a concept to reduce computational complexity and not an actual occurring stress phase.

5.1.4.3 Simplified Occupancy Probability

Instead of calculating P_{occ} for an arbitrary waveform, P_{occ} is now calculated for 2 defined transitions between the 3 phases, i.e. stress → recovery → stress at known time-steps ($t_s \rightarrow t_r \rightarrow t_{lcs}$). These defined inputs allow us to simplify the P_{occ} to a 3 step calculation:

1. Stress Phase:

$$P_{occ}(t_s) = \left(\frac{\tau_e}{\tau_e + \tau_c}\right) \cdot \left(1 - e^{\frac{-t_s}{\tau_{sr}}}\right) \tag{5.8}$$

2. Recovery Phase:

$$P_{occ}(t_r) = \frac{\tau_e}{\tau_e + \tau_c} + \left(P_{occ}(t_s) - \frac{\tau_e}{\tau_e + \tau_c}\right) \cdot \left(e^{\frac{t_s - t_r}{\tau_{sr}}}\right) \tag{5.9}$$

3. Longest Continuous Stress Phase:

$$P_{occ}(t_{lcs}) = P_{occ}(t_r) + \left(\frac{\tau_e}{\tau_e + \tau_c} - P_{occ}(t_r)\right) \cdot \left(1 - e^{\frac{t_r - t_{lcs}}{\tau_{sr}}}\right) \tag{5.10}$$

$$\text{with } \tau_{sr} = \frac{1}{\frac{1}{\tau_e} + \frac{1}{\tau_c}} \qquad \tau_c = \tau_c(V, T) \qquad \tau_e = \tau_e(V, T)$$

Originally P_{occ} is calculated recursively for every state switch in the waveform and a recursion depth of #switches (e.g. 1 467 315 for an average transistor while the processor executes “barnes”). In contrast, our new P_{occ} calculation takes exactly 3 steps with a recursion depth of 3:

$$P_{occ} = P_{occ}(t_{lcs}, P_{occ}(t_r, P_{occ}(t_s))) \tag{5.11}$$

Simplifying P_{occ} results in a significant speed-up as P_{occ} is updated every time the voltage or temperature changes to account for the temperature and voltage dependence of $\tau_e(V, T)$, $\tau_c(V, T)$.

As ΔV_{th} is a function of P_{occ} (see eq. 5.1) we obtain:

$$\Delta V_{th}(\lambda, f_{sw}, T, V, t_{life}) = N \cdot \overline{\eta} \int_0^\infty \int_0^\infty D \cdot P_{occ} \, \mathrm{d}\tau_e \mathrm{d}\tau_c \tag{5.12}$$

The model provides the maximum BTI-induced degradation $Max(\Delta V_{th})$ for given operating conditions (λ, f_{sw}, T, V, t_{life}) in milliseconds, as just 3 phases are processed, while still employing detailed modeling of physical processes for the calculation of instantaneous and long-term BTI.

5.1.5 Guardband Estimation

Estimating the guardbands for a circuit requires input parameters for the BTI model to estimate the degradation. Worst-case scenarios are the safe and easy option, i.e. assuming the highest T, worst λ, slowest f_{sw}, etc. However, this leads to guardbands which exceed the available design space.

The alternative is to estimate the aging stimuli based upon workload of the circuit, modeling actual occurring aging [73]. To estimate the activity of the workload, we employ gem5 [148] as cycle accurate system simulator, which simulates the execution of the workload in Linux 2.6 on ALPHA 21264 Out-of-Order processor at $f_{operation} = 2GHz$.

In the following section, we exemplify our approach on the register file of our microprocessor. The implementation is not limited to register files and monitors other microprocessor components (caches, ALUs, etc.) in a analogous manner.

Guardband: In our scenario, the guardband is defined as the BTI-induced degradation in percent of the static noise margin of the SRAM cells in a register [15, 149].

Activity Monitoring: We implemented our own architecture level activity monitor, which estimates the signal probabilities (ratio for 0 or 1) for each bit within the register file. Assuming an SRAM-based register file, we determine the activity based upon the signal probabilities of the storage bits and addressing bits to calculate λ for each transistor within the cells.

Frequency Monitoring: To obtain f_{sw} our activity monitor monitors the longest, shortest and average time for the data stored in the SRAM cells. The longest period defines $min(f_{sw})$ which is of main interest.

Power Estimation: To estimate the power of the processor, we employ McPat [150], which models the static and dynamic power consumption of the ALPHA processor. McPat takes the gem5 activity waveform of each microprocessor component and translate it to power waveforms for each component.

Temperature Estimation: The power waveform for each microprocessor component is passed to the thermal simulator HotSpot. [13]. Together with the floorplan of the microprocessor its temperature can be estimated. Power and temperature estimation is iterative, as the temperature recursively depends on the static power consumption (leakage) of the processor.

These time-consuming steps are performed once, as these aging stimuli do not change as long as the micro-architecture of the processor remains identical.

5.1.5.1 Designing Guardbands

Our workload monitoring provides λ, $min(f_{sw})$ and T for each individual workload. Together with t_{life}, V given by the specification, we can estimate $Max(\Delta V_{th})$ for the given workload with our proposed BTI model.

With $Max(\Delta V_{th})$ known, degraded SPICE simulations of the register file can be performed to estimate if the register file operates within specification. In practice, this means employing Monte Carlo SPICE simulations of SRAM cells to model variability introduced by manufacturing together with aging-induced degradation to verify meeting time constraints (e.g. read access time $<$ clock period) and sufficient resiliency against noise (static noise margin) or radiation (critical charge). A circuit designer can then adapt the guardbands until probability of failure of the circuit P_{fail} is below P_{fail} of the specification. Once the smallest guardband is found, which still satisfies the specification, the circuit designer must employ the guardband in his circuit. For example, he could up-size the transistor widths to make transistors faster and more resilient against noise or reduce the desired operating frequency to increase the timing slack.

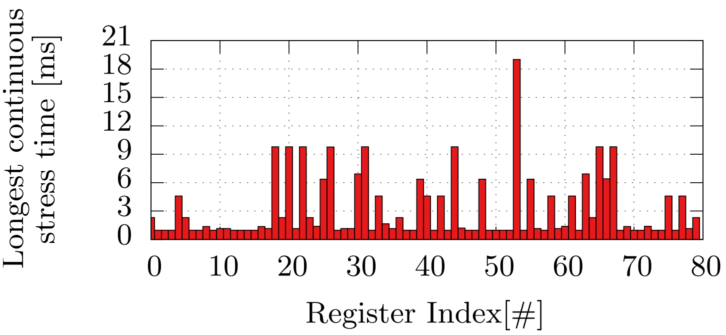


Figure 5.5: Longest time the same value is stored in a register of the register file while executing the "barnes" application at $f_{operation} = 2GHz$. These LCS phases stimulate instantaneous BTI and are the input for the calculation of $Max(\Delta V_{th})$.

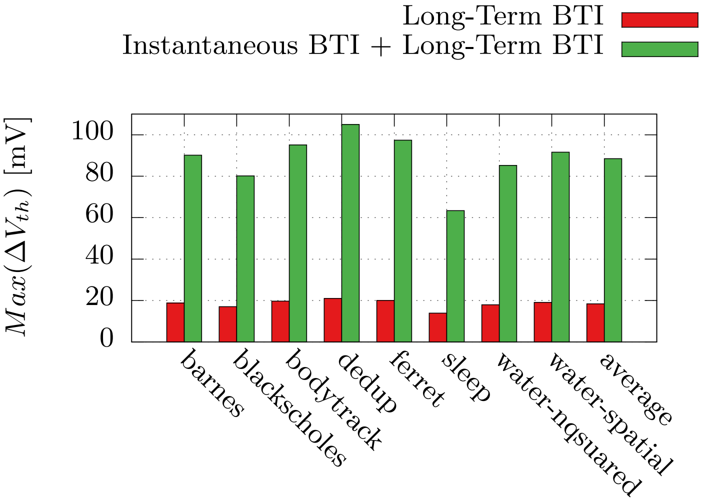


Figure 5.6: $Max(\Delta V_{th})$ for applications at their corresponding operating conditions ($V = 1.0V, T \in [49°C, 85°C]$). Even though $max(t_{stress})$ is similar for all applications different λ, T lead to different $Max(\Delta V_{th})$.

5.1.6 Evaluation

Model Validation: Simplifying the model did introduce only minor inaccuracies, due long-term BTI not being perfectly frequency independent and $max(t_{stress})$ not occurring perfectly at t_{life}. Compared to the carefully validated PDO model [31], which results in $Max(\Delta V_{th}) = 73.12mV$ for low $f_{sw}, T = 125°C, V = 1.5V, \lambda = 0.5$, $t_{life} = 10$ years, our model estimates $Max(\Delta V_{th}) = 72.89mV$, i.e. a deviation of 0.3%. The $Max(\Delta V_{th})$ deviation between PDO and our BTI model for the experiments in Fig. 5.6 was below 0.3%.

Model Performance: To evaluate the performance of the model, we generated an activity waveform with 10^6 data points and compared original PDO [31], PDO with compressed (10^4) waveform [73] and the proposed model in this work. Our model required $0.094s$ to estimate $Max(\Delta V_{th})$, while PDO needed $433.5s$ without and with compression $4.366s$, i.e. speedup of 99x compared against [73] and 4567x against [31]. Note, that the execution times for our model and the compression are almost independent of the waveform length (waveform analysis depends on #points, then computation on fixed #points), while PDO has an execution time proportional to the waveform size. Our model could perform the aging estimation for a DCT circuit featuring 350 015 transistors in 9.2 hours and IDCT circuit in 9.1 hours. DCT-IDCT circuits are often employed in image processing and are 3x larger than a typical RISC processor [82], highlighting how our physical model is feasible for complex circuitry at micro-architecture level. For designs exceeding this level of complexity, our approach in [82] can be used jointly with the BTI model presented in this work.

Intra-Application-Variation: Fig. 5.5 highlights the intra-application variation of the instantaneous BTI stimulant $max(t_{stress})$ across the register file. Most registers change their state regularly during the execution of the "barnes" application, i.e. hold their state not longer than $2ms \rightarrow f_{switching} = 500Hz$. Register 53 exhibits large

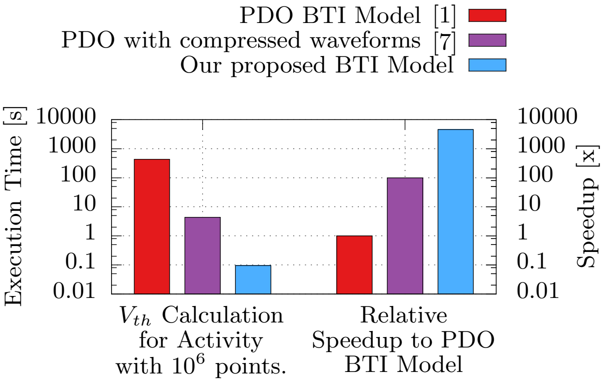


Figure 5.7: Comparison of BTI models, which base themselves on the PDO BTI Model [31]. Our proposed model has 94ms execution time, resulting in a speedup of 4567x compared to PDO.

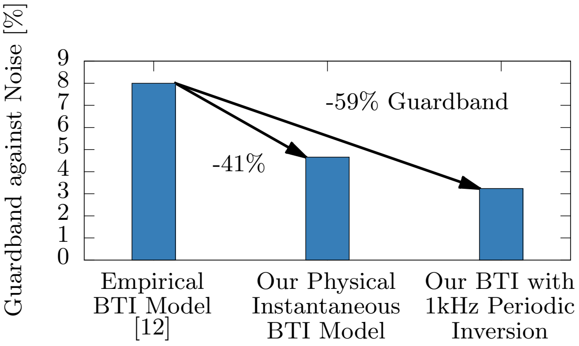


Figure 5.8: Reduction of the required guardband to protect against data corruption due to noise in our register file scenario, due to the estimation of the actual required guardband. Further reduction due to employed aging mitigation technique.

$max(t_{stress})$ as the same state is held for almost $20ms$ which results in $f_{switching} = 50Hz$. Slow switching registers like 53 exhibit $Max(\Delta V_{th}) \approx 100mV$, while the average registers exhibit $63mV$ and the best just $54mV$.

Inter-Application-Variation: While a single application shows different behavior among its registers, the applications themselves are similar to each other in terms of state changes, i.e. the worst and average $max(t_{stress})$ in the entire register file is almost identical for each application. We assume that the register renaming of the out-of-order alpha processor is the main reason for this observation. Even though $max(t_{stress})$ may be similar for the applications, other operating conditions like T,λ are not. Hence the induced $Max(\Delta V_{th})$ shown in 5.6 is different, highlighting that the joint impact between the operating conditions must be considered.

Impact of Instantaneous BTI: In Fig. 5.6 the impact of long-term BTI versus long-term BTI with instantaneous BTI is shown, illustrating that instantaneous BTI contributes $70.1mV$ or 79,2% to the overall average $\Delta V_{th} = 88.4mV$. This motivates mitigating instantaneous BTI as it predominantly governs BTI guardbands.

Impact on Guardbands: In Fig. 5.8 the guardbands of our register file scenario are illustrated for aging stimuli based upon the average of the studied benchmarks. Designing the guardbands with an empirical model leads to 8%, while our approach estimates 4.66% to be sufficient to tolerate the occurring aging-induced degradation. The guardband reduction due to mitigation via periodic inversion is discussed in the next section.

5.1.7 Adapting Existing Aging Mitigation Techniques

As Fig. 5.6 shows, the additional guardband required to tolerate instantaneous BTI is considerable. In order to design *narrow guardbands* mitigation techniques are necessary, to reduce the required guardband. These mitigation techniques should optimize long-term and instantaneous BTI jointly, as both degradations define the required guardband.

Thermal Management/Voltage Scaling: Techniques focusing on reducing a single aging stimulant like thermal management (reduce T) or voltage scaling (reduce V), must take instantaneous aging effects into account. Their policies were designed without f_{sw} in mind as they were evaluated with *empirical models* (i.e. neglecting frequency dependence of BTI) potentially resulting in strong stimulation of instantaneous BTI. For example, thermal management can stall activity (clock gating) to reduce the dynamic power consumption of the circuit, which directly affects f_{sw}. Similarly, when voltage scaling reduces V to save energy or limit generated heat, $f_{operation}$ is lowered to prevent timing errors, which in turn decreases f_{sw} if the same workload is executed. These side-effects must be considered when employing such mitigation techniques, i.e. updating their policies using our proposed BTI model to find the pareto-optimal solution considering all operating conditions jointly including f_{sw} for instantaneous BTI.

Periodic inversion: Originally intended to distribute aging stress as uniformly as possible within circuits, this technique inverts logic signals periodically [151]. Circuits are extended with a flag, indicating if the data is currently *normal* or *inverted*. In *normal* mode everything is regularly processed, while in *inverted* mode data is either processed in an inverted manner (knowing that the result is also inverted) or temporarily returned to its original state during processing. Periodically inverting the entire system ensures that transistors operate close to $\lambda = 0.5$ [151], which reduces aging stimuli for transistors which had $\lambda > 0.5$ while it increases stimuli for $\lambda < 0.5$. This reduces the variability, raising the lower boundary for λ and hence reducing the guardband.

To have a first order approximation for the overhead, we refer to our register file scenario in the Alpha 21264. The 80 registers in the register file would require 1 read and 1 write operation every $1ms$ to invert the data (read, invert, write back) with $f_{inversion} = 1kHz$. With ≈ 25000 reads and ≈ 10000 writes per $1ms$ at $f_{operation} = 2GHz$ across our studied benchmarks the performance and power overhead would be negligible. This rough estimation supports [151] claiming less than 1% performance impact if logic and caches are protected at $t_{period}(inversion) = 1ms$.

Next to its original intent, periodic inversion can be employed to ensure f_{sw} does not fall below a lower boundary. When the logic signals are inverted, the states of all transistors change ensuring $f_{sw} \geq f_{inversion}$. Inverting every $1ms$, i.e. $f_{sw} \geq 1kHz$ would reduce the average $Max(\Delta V_{th})$ across the studied benchmarks by $52mV$, very close to the long-term only result. The employment of periodic inversion together with our BTI model leads to a reduction of 59% of the guardband in our register file scenario.

5.2 Massively Parallel PDO Model

This section is based on my publication [152] and is a brief overview of the GPU-based implementation of the PDO model. As each transistor can be modeled individually, the processing power of GPUs can be harnessed to estimate many transistors in parallel. Instead of speeding up a single transistor BTI estimation like the previous section, this section aims at estimating thousands of transistors simultaneously.

5.2.1 Background PDO Model

For background on the PDO model see Section 2.2.1.3.2. The PDO model is used in the Sections 4.1.2.2 and 4.2.4.1. Therefore, the accelerated implemenation presented in this section, integrates with the contributions presented there and accelerated both the unified model in Sections 4.1.2.2 and the defect variability Sections 4.2.4.1.

5.2.2 Parallel PDO BTI Model

Simulation times for defect-centric BTI models are too large for BTI variability, despite reducing from minutes to seconds with our work in Section 5.1 and [73]. However, current graphic cards offer massive computational resources with massive parallelism (2048 cores in our GTX980). This enables modeling of up to 1024 transistors in parallel to decrease modeling times to milliseconds per transistor (see Fig. 5.9).

In order to implement PDO on the graphic card, it is programmed in CUDA. CUDA is a C-variant targeted towards massive parallel programming of NVIDIA graphic cards. The PDO model is well-suited for the graphic card as little communication is necessary between the computations, as they are uncorrelated. PDO models individual transistors, in which individual defects are randomly occupied or unoccupied according to the occupancy probability P_{occ}. This occupancy probability depends solely on the activity of the transistor. For reliability analysis, either individual activities can be specified (see [73]) or uniform worst-case conditions (e.g., 100% ON-time, 125°C, 1.2V for 2 years) can be assumed. As defect interactions are negligibly small (see Section 4.1.5.1), each defect can be modeled individually. Therefore, each transistor is modeled as a CUDA block containing up to 72 compute threads (72 defects is sufficient for nano-era transistors [31]). Each thread models a single defect. During each time-step of the simulation, each thread receives the activity for the next time step (new voltage, time-step, temperature) and each thread updates capture and emission times according to equation (4.2). Based on the capture/emission times and the current time, each thread updates the occupancy probability given as [31, 113]:

$$P_{occ}(t) = P_{occ}(t_i) + \left(\frac{\tau_e}{\tau_e + \tau_c} - P_{occ}(t_i)\right) \cdot \left(1 - e^{\frac{t-t_i}{\tau_{sr}}}\right) \tag{5.13}$$

$$\text{with } \tau_{sr} = \frac{1}{\frac{1}{\tau_e} + \frac{1}{\tau_c}} \quad \tau_c = \tau_c(T,V) \quad \tau_e = \tau_e(T,V)$$

Then, a dedicated CUDA random number generator cuRAND (part of CUDA default libraries) provides unique uniformly distributed random numbers to each thread. This random number $rand$ is compared against P_{occ}. If P_{occ} is larger or equal than $rand$, then the defect d is occupied and contributes η to the total degradation. If P_{occ} is smaller than $rand$, then defect d is unoccupied and does not contribute η. In programming terms, each thread contributes either η or 0 to a its unique position in a array (its unique thread ID is used as an array index) in shared memory. This ensures that no two threads can write to the same variable, removing the need for access locks and thus improving performance. Now the graphic card uses the "__syncthreads" command to wait for all threads to finish. After all degradation values of each defect are written to the array, all values of the array are summed up to obtain the overall ΔV_{th} of the transistor at this time-step.

These steps are repeated for each time-step, passing activity to each block (modeling a transistor), in which each thread updates its defect parameters (capture/emission times) and determines P_{occ}. Then, cuRAND provides random numbers to each thread and by comparing $rand$ with P_{occ} each thread decides if it writes a 0 or η to its thread index in an array. After all threads are finished, the values are summed and the next time-step is started.

In this process, each transistor has a unique set of defects, which in turn have unique parameters. Therefore, for identical stress, each transistor experiences different ΔV_{th} degradation levels, thus modeling BTI variability.

Our approach is massively parallel as each transistor is modeled in its independent CUDA block, with up to 72 threads for up to 72 defects per transistor. As in current technologies the lowest number of defects is 1-2, up to 1024 transistors can be modeled in parallel as our GTX980 features 2048 CUDA cores. This vastly outperforms multi-core solutions, which can use up to 8 cores and thus 4 transistors in parallel. Recent GTX1080Ti features even 3096 cores, highlighting the growth in CUDA cores in NVIDIA graphics cards and thus the scalability of our BTI modeling towards the future.

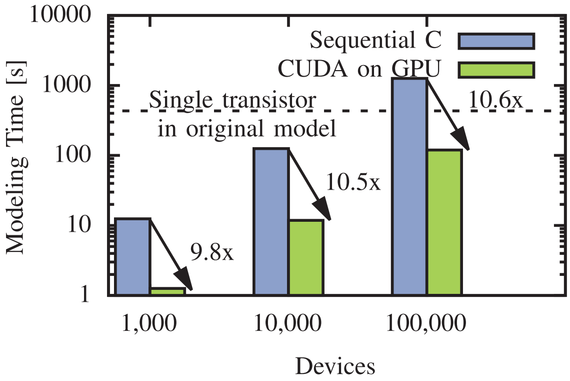


Figure 5.9: Performance of our CUDA implementation of our PDO model for V_{gs} waveforms with 1000 points and different number of transistors. CUDA outperforms sequential C-code by 10x in all cases. Note, that sequential C is further optimized compared to reported performance metrics in Section 5.1.4.2, which are represented by a dashed line for a single MOSFET simulation (3 000 s in MATLAB [153] with code from [31]). Modeling time of 119 s for 100 000 transistors enable circuit level BTI variability modeling (see Section 4.2).

5.2.3 Parallel PDO Model Performance

Fig. 5.9 highlights how our parallel implementation is able to model $100\,000$ transistors in less than $120\,\mathrm{s}$. If normalized to a single transistor, this results in $1.2\,\mathrm{ms}$ modeling time per transistor. The original MATLAB implementation from [31] took about 5 minutes ($3\,000\,\mathrm{s}$) per transistor is shown as a dashed line[2].

Our new sequential C implementation (shown in blue) already outperforms Section 5.1, due to algorithmic optimization (e.g., smaller data structures for smaller memory footprint, only update necessary variables in each time-step, cache intermediate results), but only our massively parallel PDO implementation enables large scale simulations.

To verify this statement, we used our in-house parallel SPICE simulator (see Section 7.1 and [109]) to simulate 32- and 64-bit multipliers with $11\,288$ and $42\,534$ transistors, respectively. Each transistor had a unique set of defects, resulting in a unique occupancy state and thus unique ΔV_{th}. These simulations resulted in $1.42\,\%$ and $1.8\,\%$ BTI-induced propagation delay increase for the 32- and 64-bit multiplier.

[2] Settings in Section 5.1 were different, line shown here is identical conditions (e.g., number of data points and different input data).

6 Step Three - Incorporating Degradation into Standard Tools

After step two, which accelerated the degradation models, the models are now ready to be integrated and incorporated into the standard tools. Standard tools in this context are circuit simulators like SPICE for analog circuits and timing estimation tools like static timing analysis (STA) tools for digital circuits. The timing estimation tools rely on SPICE to generate cell library information (timing and power information for standard cells) for the estimation of power and delay for digital circuits [82]. Hence, this works focuses on the SPICE circuit simulator for both analog and digital circuits, as its the back-end for digital STA tools [82] and directly employed for analog circuit simulations [112].

The first contribution shows that the current worst-case estimations (SS process corner) are not actual worst-case estimations with respect activity/workload-induced degradation through phenomena like BTI. This is critical, as many circuits are general purpose circuits, in which the workload is not pre-determined (e.g., CPUs) and governed by the end-user. In these cases, the activity/workload must not induce degradations beyond the guardband determined at design-time to ensure that functionality and reliability can be maintained. This contribution to find the actual workload-induced worst case, helps determining these necessary guardbands for general purpose circuits. As the pessimism from SS process corners is removed, custom reliability estimations must ensure that they capture the entire detrimental impact of the degradation phenomena.

The second contribution is the integration of new phenomena like self-heating into standard tools. Naturally, newly uncovered degradation phenomena are not immediately supported by the standard tool vendors, so research needs to step in and provide prototype solutions on how these new phenomena can be considered in the standard tools.

6.1 Worst-Case Aging in Standard Cells

This section is based on my publication [85].

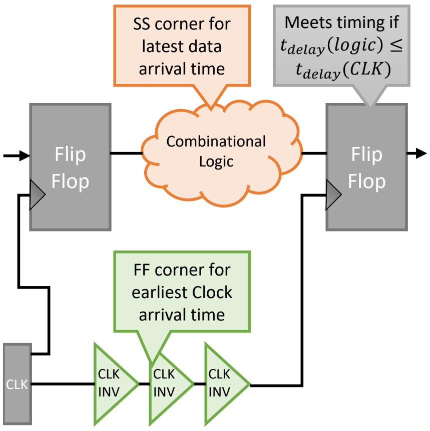

SS corner for latest data arrival time
Meets timing if $t_{delay}(logic) \leq t_{delay}(CLK)$
Flip Flop
Combinational Logic
Flip Flop
FF corner for earliest Clock arrival time
CLK
CLK INV
CLK INV
CLK INV

Figure 6.1: Timing Analysis tools use the theoretical boundaries in the form of SS and FF process corners during a setup and hold timing check. This highlights the necessity for correct theoretical boundaries for standard cell delays with the actual best and actual worst case.

Designing for reliability with respect to aging currently relies on theoretical bounds. Theoretical bounds decouple design from actual occurring conditions, hence simplifying the design process as well as providing guarantees in which any occurring condition (e.g., elevated temperature, bias conditions) can be tolerated. Typical bounds are process corners of standard cell libraries. In the following, we explain how process corners act as theoretical bounds. Additionally, we explain how worst-case timing of circuits is traditionally obtained using the slow-slow (SS) process corner.

Theoretical Bounds (SS and FF Corners): Theoretical bounds are used throughout circuit design, for example during setup and hold timing checks within signoff in timing analysis tools (e.g., Cadence Tempus, Synopsys PrimeTime) shown in Fig. 6.1. Circuit operation is hampered at elevated temperature, lowered voltage, in a badly manufactured sample, i.e. circuit timing is prolonged. Yet, even under such worst-case conditions, the circuit design should exhibit no timing violations. For this purpose, every semiconductor manufacturer provides process corners, typically slow (SS), fast (FF) and typical (TT). These three process corners encompass the worst, best and nominal (typical) state of the standard cells in the circuit. Manufacturing variability, temperature effects, voltage drops and many more effects are incorporated in these process corners to provide absolute theoretical bounds.

These process corners are the worst/best case delay/power information for the standard cells offered by the semiconductor vendor. In the SS corner the lowest supply voltage is combined with the highest temperature for the worst transistor with respect to manufacturing. While this condition might never occur, it is crucial to provide guarantees about the circuit timing. Even under these harsh conditions, the circuit must still meet timing, i.e. it meets timing for every condition the end user could experience. To reduce pessimism, de-rating is applied. It interpolates timing between SS and TT corners for the combinational logic (see Fig. 6.1) and thus results in more realistic timing checks. *However, even when de-rating is applied, it relies on the fact that SS corner is the absolute theoretical upper bound for standard cell delay.*

Aging within SS corner: Aging (e.g., Bias Temperature Instability (BTI)[1]) is considered within process corners. Its induced degradation (e.g., ΔV_{th}) is experimentally obtained, at peak stimuli (high temperature, high voltage) to accelerate degradation. However, these experiments obtain peak degradation occurring *in transistors*. In order to account for aging *in standard cells*, the impact of transistor degradation on cells is simulated, as manufacturing a cell solely out of worst-case transistors (e.g., with highest process variation) is impossible. Thus, during standard cell characterization (e.g., in SPICE simulations) *uniform peak degradation per transistor* (identical worst ΔV_{th} for all transistors) is applied, as state of the art assumes that worst-case transistor degradation results in worst-case cell delay. However, in this work, we demonstrate that uniform peak transistor degradation does not lead to worst-case standard cell delay.

6.1.1 Worst-Case Standard Cell Delay

To determine worst-case standard cell delays, state of the art [74] [79] [81] applies worst-case transistor degradation uniformly to each transistor. In this section, we demonstrate how such a uniform distribution does not necessarily lead to the *actual* worst-case timing. Additionally, we explain how to obtain the actual non-uniform worst-case degradation with respect to propagation delay of standard cells.

Uniform transistor degradation is not worst-case delay: Figure 6.2 shows a NAND logic gate and compares the traditional worst-case (the uniform application of peak transistor degradation ΔV_{th}) with the actual worst-case. The rise propagation delay of a NAND gate does not reach maximum delay when all transistors are uniformly at peak degradation. The rise propagation delay of a standard cell is defined as the time from the switch of a cell input until the cell output switches from logic high (e.g., V is above 90% V_{dd}) to logic low (e.g., V is below 10% V_{dd}). Switching the output from high V to low V is performed by discharging the load capacitance at the output

[1] If PBTI is negligible (e.g., as reported in [154]), then degradation is still present in NMOS transistors in the form of Hot Carrier Injection. HCI can analogously be considered in this work, as it shifts the same transistor parameters (e.g., ΔV_{th}) as BTI (see "Degraded transistors" in Section 6.1.3).

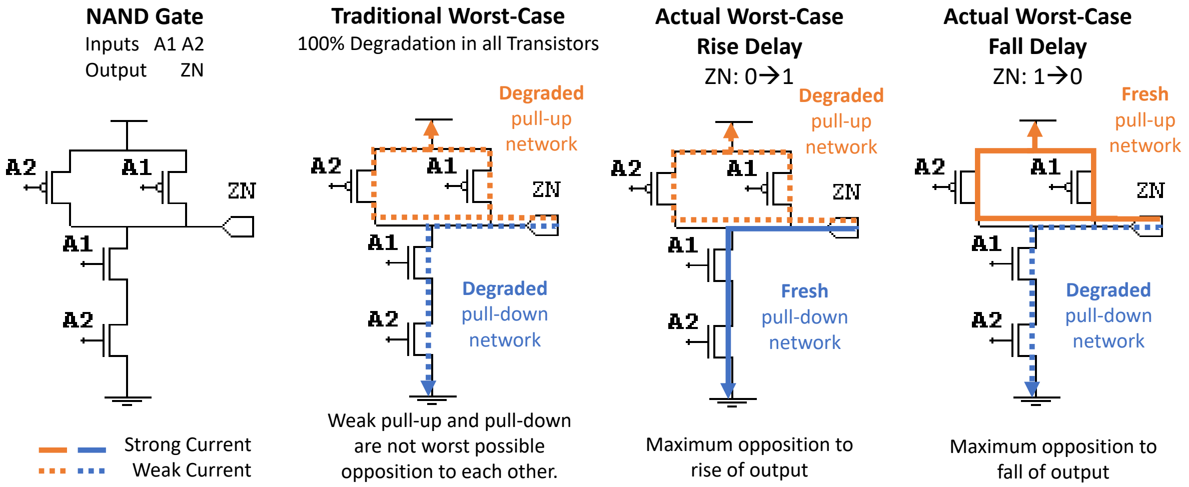

NAND Gate
Inputs A1 A2
Output ZN
A2
A1
ZN
A1
A2
Strong Current
Weak Current
Traditional Worst-Case
100% Degradation in all Transistors
Degraded pull-up network
Degraded pull-down network
Weak pull-up and pull-down are not worst possible opposition to each other.
Actual Worst-Case Rise Delay
ZN: 0→1
Degraded pull-up network
Fresh pull-down network
Maximum opposition to rise of output
Actual Worst-Case Fall Delay
ZN: 1→0
Fresh pull-up network
Degraded pull-down network
Maximum opposition to fall of output

Figure 6.2: Traditional worst-case for the propagation delay in standard cells is a uniform 100% degradation across all transistors. During the switch in the output, both the pull-up and pull-down network are conducting. When the output is rising ($ZN : 0 \rightarrow 1$) the pull-up network increases the voltage at ZN, while the pull-down network opposes that and aims to keep the voltage at logic low ($ZN = 0V$). Therefore, the actual worst case for the rise propagation delay is 100% degradation in the pull-up network (weakest possible rising current) and 0% degradation in the pull-down network (strongest possible opposing current). Fall delay is vice versa. The nMOS and pMOS transistors in this work are regular planar 45nm MOSFETs [3]. However, our approach is not limited to a specific CMOS technology, but analogously can be applied to other technologies exhibiting BTI effects.

over the pull-down network of the cell. Importantly, during the switch, both pull-up and pull-down are conducting, the pull-up network (orange) of the cell opposes the pull-down network (blue). Therefore, the worst-case rise propagation delay occurs when the pull-down network suffers maximum degradation, but the pull-up network is free of degradations (fresh). This results in the lowest driving current from the pull-down network with the maximum opposing current from the pull-up network, i.e. the lowest total charging current to charge the load capacitance and hence the slowest switch from logic high to low.

It is imperative to obtain the actual worst case for each standard cell in terms of timing for SS corners. If the cell delays in the SS corner are not the worst possible values, then no guarantees can be given about the reliability of the designed circuit. A combination of transistor degradations (ΔV_{th} mapped to transistors) could be found, which would result in longer cell delay values than the SS corner. This means, that timing analysis fails to provide guarantees, i.e. timing violations could occur. Especially if de-rating is applied to reduce pessimism, timing violations are not just theoretical, but start to occur under typical operating conditions as safety margins are reduced. To continue to provide guarantees, we propose to characterize the SS corner with the corrected stronger impact of aging at its worst. Existing SS corners should be re-characterized as exemplified in this work.

Obtaining the worst-case delay: In order to obtain the worst-case rise and fall propagation delays of a standard cell, each cell should be analyzed with each combination of transistor degradations (mapping ΔV_{th} to transistors). For instance, abstracting degradation to 10 steps in a 22 transistor AOI221 standard cell, 10^{22} combinations exists, leading to an unfeasible large search space to explore.

Instead, in this work, we exploit circuit topology to significantly reduce the number of combinations to make the search for worst-case cell delays feasible. Not every combination of transistor degradations can occur, as (with respect to aging) transistor activity governs transistor degradation [8]. Circuit topology creates dependencies between transistor activities and hence transistor degradations. For example, in an inverter both transistors share the same gate node as the input and as such must have opposing degradations (if one transistor is under stress (increasing degradation), the other is under recovery (decreasing degradation)). Thus, degradation in these two transistors is correlated. By considering these correlations, we can reduce the search space to all possible transistor degradation combinations, i.e. a feasible search space (see Fig. 6.3). This allows us to find the worst-case rise/fall propagation delay of standard cells in the scope of aging.

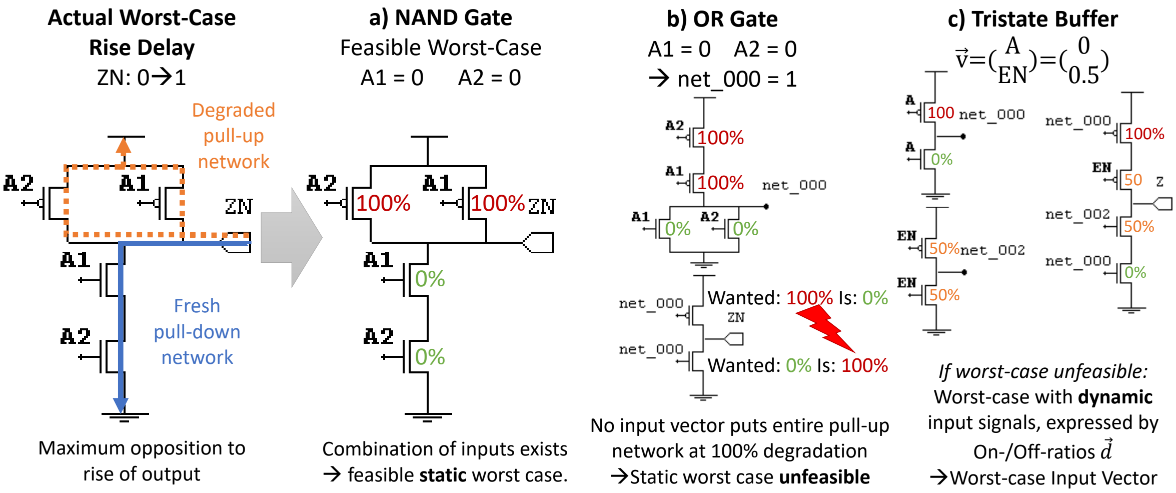


Figure 6.3: Not every duty cycle combination for the transistors can occur, as the topology of the gate connects the gates of the transistors. Three standard cells from the NanGate library [137] are shown. a) the NAND cell can have static inputs which lead to the worst-case rise propagation delay. b) No combination of static inputs can be found for the OR gate to reach its worst case. Hence, the worst-case is based on dynamic inputs ($0 < d < 1$). c) Dynamic worst-case input vector for a tristate buffer. The inputs are not static at logic 0 or logic 1, but instead dynamic input signals. For these signals, the duty cycle d (of-/off-ratio) which results in the worst rise propagation delay is shown each transistor. Note how the values differ from each other, i.e. how a non-uniform degradation results in the worst-case propagation delay

Our contributions within this work are:

1. Demonstrating for the first time, how considering a uniform degradation of worst-case transistors does not necessarily result in the worst-case delay of standard cells. Thus, the created SS corners fail to provide timing guarantees.
2. Creating SS corners with actual worst-case timing through employing worst-case input vectors for standard cells. Our approach efficiently determines the combination of transistor degradation resulting in worst-case standard cell propagation delays.

6.1.2 Related Work

To clearly distinguish our work from state of the art, we provided the following itemized overview (as a reminder on top of the related work Section 3):

1. Estimating the worst-case standard cell rise and fall propagation delay based on worst-case input vectors.
2. Considering the effects of parasitics, input signal slew and load capacitance in our circuit simulations (e.g., integration-based duty cycle abstraction). They change the worst-case input vectors and provide higher accuracy.
3. Explore the impact of worst-case delays on complex circuits like entire microprocessors.

6.1.3 Worst-Case Timing for Standard Cells

The goal of this work is to obtain the worst rise and fall propagation delays of standard cells. For this purpose, our approach employs searches for the worst-case input vectors, i.e. the input signal activities at which these worst delays occur. An overview of our approach is shown as a process flow in Fig. 6.4. We discuss the individual steps in the order of appearance in the figure.

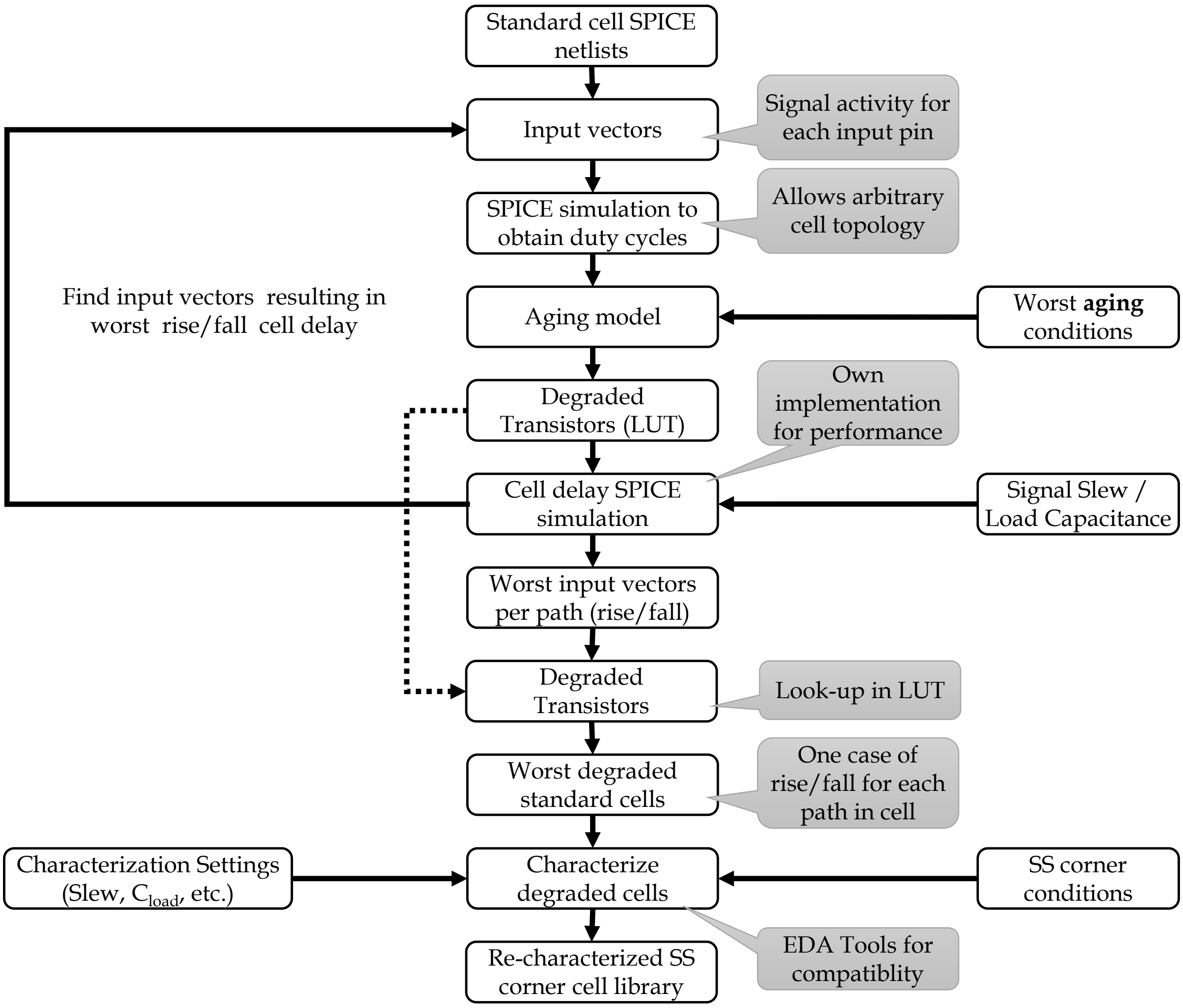

Standard cell SPICE netlists
Input vectors
Signal activity for each input pin
SPICE simulation to obtain duty cycles
Allows arbitrary cell topology
Find input vectors resulting in worst rise/fall cell delay
Aging model
Worst **aging** conditions
Degraded Transistors (LUT)
Own implementation for performance
Cell delay SPICE simulation
Signal Slew / Load Capacitance
Worst input vectors per path (rise/fall)
Degraded Transistors
Look-up in LUT
Worst degraded standard cells
One case of rise/fall for each path in cell
Characterization Settings (Slew, C_{load}, etc.)
Characterize degraded cells
SS corner conditions
Re-characterized SS corner cell library
EDA Tools for compatiblity

Figure 6.4: Process Flow of our approach. The steps 2 until 5 are looped until the worst-case input vector is found. As this loop is repeated many times, we employ our own SPICE-based delay estimation. Steps 6 until 10 are run just once and therefore use standard EDA tools to ensure compatibility with other tools (e.g., timing analysis).

6.1.3.1 Worst-Case Input Vectors

Computational complexity of non-uniform transistor degradation: State of the art [74, 79, 81] assumes that the worst-case standard cell delay occurs under uniform peak transistor degradation. As mentioned before, this weakens the opposition to the signal change (i.e. pull-down network is weakened on a $0 \rightarrow 1$ switch) and therefore does not result in the actual worst-case cell delay. As a result, another combination of (non-uniform) degradation must result into the actual worst-case degradation. However, it is impossible to check every combination of transistor degradation to find that worst-case combination resulting in the worst delay. Cells like AOI221, flip-flops and adders have 15-30 transistors, thus even by discretizing degradation to 10 steps, this results in $10^{15} - 10^{33}$ possible combination to check for worst-case delay.

In order to reduce the search space, this work exploits circuit topology. For example, a full adder has 3 inputs but 10 transistors. Again signal activity is discretized across the inputs in 10% steps from 0% to 100%. For our 3-input adder 10^3 combinations have to be checked, a significant reduction compared to 10^{10} when iterating over transistors. Similarly, AOI221 has 5 inputs, resulting in 10^5 checks. This significantly reduces the search space and makes it computationally feasible. Standard cells feature just a few inputs (2-6 in NanGate cell library [137]), ensuring the feasibility of our approach as a single check is a fast SPICE simulation of a tiny circuit (a standard cell), typically performed in less than a second.

Cell input vectors: A standard cell has n inputs i_n and m outputs o_m, each of these inputs i_n can feature a signal activity from 0% ($i_n = 0$) to 100% ($i_n = 1$). Based on the activity at the inputs, for each of the k transistors within the cell, a duty cycle d_k (on-/off-ratio of the transistor) occurs based on the topology of the standard cell. These d_k govern aging and therefore are later used to calculate the resulting aging-induced degradation. According to the duty cycle d_k of a transistor, the transistors exhibits the ON-phase at the end, i.e. a duty cycle d_k of 30% is a transistor which is 70% of the time period OFF followed by 30% of the time period ON (high gate voltage). This allows us to ensure that the peak degradation (ΔV_{th}) occurs at the end of the simulation (i.e. no BTI recovery) and in all transistors simultaneously, which is necessary to obtain worst-case propagation delays of cells.

Our approach iterates through all the inputs combinations with each input sweeping from from 0% $\rightarrow$ 100% activity. Step size (e.g., 10%) and step size optimization is discussed in Section 6.1.3.2. An iteration consists of a n-dimensional vector $\vec{i}$ with an activity value in each dimension. This vector $\vec{i}$ is called *input vector* and results in $\vec{d}$ the *duty vector* and *output vector* $\vec{o}$:

$$\vec{i} = \begin{pmatrix} i_1 \\ \vdots \\ i_n \end{pmatrix} \quad \Rightarrow \quad \vec{d} = \begin{pmatrix} d_1 \\ \vdots \\ d_k \end{pmatrix} \text{ and } \vec{o} = \begin{pmatrix} o_1 \\ \vdots \\ o_m \end{pmatrix}$$

Obtaining transistor duty cycles: In order to determine the corresponding duty cycles d_k, each $\vec{i}$ results in a circuit simulation (e.g. in SPICE). n Piece-Wise Linear (PWL) voltage sources are created to generate a pulsed waveform for each i_n which matches the given signal activity. Then based on the netlist for standard cell, SPICE provides the d_k for all k transistors by calculating the on-/off-ratio of the signal at the gate terminal of each transistor. We obtain d_k for transistor k by calculating the integral of the voltage signal V_G at the gate of the transistor and dividing it by the integral of the supply voltage (which is always logic 1, i.e. $d = 1$):

$$d_k = \frac{\int_{t_{start}}^{t_{end}} V_G(k)\ dt}{\int_{t_{start}}^{t_{end}} V_{dd}\ dt} \tag{6.1}$$

Employing SPICE simulations for these calculations, instead of abstracted signal equations (e.g., based on the boolean function) [81] has two main advantages. First, standard cells can have different implementations, for example a NAND cell with a fan-out of four can be implemented in two different ways. Option one is a NAND cell with four wide (e.g., 270nm) transistors for sufficient drive strength to achieve a fan-out of four. Option two is a multi-stage cell with four minimum width transistors (e.g., 90nm) AND cell followed by an inverter with two wide transistors to achieve a fan-out of four. For more complex cells like AOI, flip-flops and adders, up to 10 different implementations exist in commercial cell libraries. Importantly, each implementation has different duty cycles due to the different topology, resulting in a different worst-case input vector. Using SPICE considers circuit topology and extracts the actual duty cycles and thus determines the worst-case input vector for that specific cell topology.

Secondly, the resulting duty cycle can be estimated by considering standard cells under the joint impact of parasitics (capacitive, resistive), input signal slews and load capacitances. Our previous work in [82] and Fig. 6.6 shows how large the impact of these effects is on the cell delay, thus it must be considered.

Degraded transistors: With $\vec{d}$ known, the degradation (e.g., ΔV_{th}) of each transistor is estimated in an aging model. The model should be a physics-based aging model to correctly incorporate the duty cycle dependence of BTI [8] [113]. BTI depends on V, T and d [155] [113] and for a worst case, i.e. a theoretical upper bound, V, T are kept at their worst values at all times. Typical values for a $45nm$ technology are $V_{aging} = 1.2V, T_{aging} = 125°C$ [137]. The duty cycle d_k for each transistor k is extracted for each input vector as explained in the previous section.

In order to incorporate the degradation in SPICE, we propose to change the transistor modelcards. These transistor modelcards contain the parameters for the transistor model (e.g., BSIM4) and are provided by the semiconductor vendors for SS, TT and FF corners. In order to obtain the SS transistor modelcard without aging as a starting point, we estimate the peak degradation per transistor with maximum V, T, d and subtract that ΔV_{th} from the current V_{th} value found in the SS modelcard. In this manner, we maintain the shifts of other effects (e.g., process variability) and only removed aging. To estimate ΔV_{th}, our approach employs the aging model of our previous work [113] (which is based on [31]) as it features accurate duty cycle modeling. As V, T are constantly kept at their worst values (V_{aging}, T_{aging}) and d is the only dynamic value, it is possible to create a degraded transistor Look-Up-Table (LUT). For all duty cycles d ranging von 0 to 1 in 0.01 increments we estimate ΔV_{th} and incorporate that in a modelcard. This results in 101 nMOS and 101 pMOS degraded transistor model cards. This reduces the computational complexity, as the computationally intensive aging models are only used to create a LUT. Thus, 202 entries from as many aging model estimations are replacing calling the aging models for each transistor in thousands of cells (as seen in commercial cell libraries). This degraded transistor LUT is re-used for cell characterization.

Cell delay estimation under different input vectors: After the degraded transistor modelcards are ready and $\vec{d}$ is determined, the propagation delays of the standard cells are determined with SPICE. The standard cell netlists are annotated to use the degraded transistors from the LUT. For example, if for the current input vector it is determined that a transistor has a $d = 0.36$, then the netlist is edited so that this transistor uses the matching degraded transistor modelcard with BTI-induced degradations at $V_{aging} = 1.2V, T_{aging} = 125°C, d = 0.36$. Algorithm 4 describes the general overview of the estimation of the worst-case input vectors for a 2-input standard cell. For standard cells with more inputs, the number of for-loops increases to sweep the duty cycles for each input.

Cell characterization tools like SiliconSmart [138] could be used for the same purpose, but they simulate all conditions (rise/fall delay for each output pin) every time. This is unnecessary, as the worst-case input vectors are different for rise and fall delay (see Fig. 6.2), forcing us to discard at least half the results (e.g, rise delay simulation while the circuit is configured with a worst-case vector for fall delay). Therefore, we implemented our own tool, which obtains the worst-case input vectors by only simulating the necessary cases.

Worst-case input vectors: This entire process is looped over all the possible combinations of inputs vectors. For each input vector $\vec{i}$, the duty cycles d_k are determined, resulting degraded transistor parameters obtained from the LUT and then rise/fall delays estimated in the degraded cell SPICE simulations. If the rise or fall delay in this iteration is longer than the current worst-case rise or fall delay, than the input vector of this iteration is stored as the worst-case input vector for rise/fall delay at that output. After all input vectors have been simulated, the worst-case input vector for each output in both rise/fall delay is found. By iteratively searching for the worst vectors for every path (input to output pin) and case (rise or fall) individually, we can guarantee that we find the input vectors resulting in the worst propagation delay for every path.

Worst-case degraded standard cells: After the worst input vectors are found, we create matching degraded standard cell netlists for each vector. For each cell and each $\vec{i}$, we create an annotated netlist. Each worst vector $\vec{i}$ is the worst case for an output pin, resulting in worst rise/fall delay for that pin. For this vector $\vec{i}$, each transistor in the netlist gets degraded parameters from the LUT, according to its d under worst $\vec{i}$ and thus creates the worst degraded standard cell for that case.

Cell characterization: Our goal is to re-characterize the SS corner cell library in which aging effects are correctly considered. Therefore, we employ the standard EDA characterization tool to maintain full compatibility with the rest of the EDA tools by creating liberty files. A "liberty file" is a standard file format containing delay tables for each output pin in each standard cell under different signal slews and output load capacitances. SiliconSmart characterizes the cells with worst degraded cell SPICE netlists, i.e. annotated netlists in which each transistor has a degradation according to the worst-case input vectors of that cell. The tool creates a liberty file based on these worst standard cell netlists.

The conditions of the characterization include different signal slews and output load capacitances. As discussed previously, considering these effects is mandatory for accurate results [82]. Furthermore, the SS corner mandates worst conditions for delay estimation, i.e. $V_{SS} = 0.8V, T_{SS} = 125°C$ instead of $V_{aging} = 1.2V, T_{aging} = 125°C$. The low voltage with high temperature results in the worst possible cell delay. Note, that this is indeed unrealistic, but as mentioned before, the SS corner is meant as a theoretical upper bound for delay. Therefore, the discrepancy between aging and delay conditions is not an artefact, but instead purposefully chosen to be the individual worst conditions for aging and characterization.

Obtaining re-characterized SS process corner: For each case (rise/fall, each output pin) our approach determined a worst-case input vector and created a corresponding liberty file during cell characterization. Now we extract solely the meaningful delay information from each liberty file and merge it into a single liberty file, which will be our SS process corner cell library. In practice, each liberty file contains a single case, e.g., worst *fall* delay for output pin ZN. Worst *rise* delay for pin ZN is in a different liberty with its own $\vec{i}$. As each delay is the obtained under different worst-case transistor degradations, the merged cell library is our re-characterized SS corner. It contains the actual worst-case propagation delay values of the standard cells and as such can serve as a sound theoretical upper bound for delay in a setup and hold check.

Necessity for Brute-Force Approach: Our approach is a brute-force search through the search space, which is necessary to accurately obtain the worst-case input vectors. It seems like a crude solution, but it is necessary as more elegant abstract approaches introduce severe inaccuracies.

To exemplify, we discuss two solution: a) functional analysis: duty cycle extraction based on Boolean function of standard cell (frequently employed in automated test pattern generation (ATPG)) b) dependency graphs: mapping transistor activity correlation (from circuit topology) to a dependency graph to solve for duty cycles or use graph criteria (e.g., find paths with maximum weights from input to output and relate the weights) to obtain worst-case input vectors from graph directly.

Both solutions result in inaccurate duty cycles. Slowly changing signals (large slews) result in long times in which a transistor is neither at V_{dd} or V_{ss} potential, i.e. neither in full stress/recovery. Similarly, parasitics (especially charging parasitic capacitances) delays or prolongs stress/recovery. Additionally, as mentioned before, cells often feature different implementations with different topologies. All these issues can only be considered in circuit simulation (e.g., in SPICE), for example by using our integration-based duty cycle extraction.

Similarly, cell delay estimations are also significantly affected by both topology of the individual implementation as well as signal slews, load capacitances and parasitics (see Fig. 6.6). Therefore, all commercial cell characterization tools (e.g., Synopsys SiliconSmart, Cadence Liberate) create liberty files with delay tables based on SPICE circuit simulations as other solutions introduce unacceptable inaccuracies.

6.1.3.2 Scalability of Input Vector Estimation

Our approach scales exponentially with the number of inputs with the base as the granularity of the input for loops: $\#simulations = c^n$ with n as the number of inputs and c as the step-width (e.g. 1%) for the signal activities of the inputs. For larger cells with 6 or more inputs, this is still computationally intense. To combat this issue, the step-width c can be coarse-grained for initial simulations and iteratively moved to finer granularity:

$$c = 20 \text{ for } i_1 \in [0, 100] i_2 \in [0, 100] \Rightarrow \vec{i}_{worst}(i_1, fall) = \begin{pmatrix} 0.2 \\ 0.4 \end{pmatrix}$$

$$c = 5 \text{ for } i_1 \in [10, 30] i_2 \in [30, 50] \Rightarrow \vec{i}_{worst}(i_1, fall) = \begin{pmatrix} 0.25 \\ 0.35 \end{pmatrix}$$

$$c = 1 \text{ for } i_1 \in [20, 30] i_2 \in [30, 40] \Rightarrow \vec{i}_{worst}(i_1, fall) = \begin{pmatrix} 0.23 \\ 0.37 \end{pmatrix}$$

Algorithm 4 Obtain Worst-Case Input Vector and Worst-Case Delay for Standard Cell

```
Require: $V, T, f, t_{slew}, C_{load}$ and standard cell netlist
for $i_1, i_2, ..., i_n \in [0, 100]$ do                    ▷ Loop for each input $i_1$ till $i_n$
    Set $\vec{i} = \begin{pmatrix} i_1 \\ \vdots \\ i_n \end{pmatrix}$
    Add PWL voltage sources based on $\vec{i}$ and $t_{slew}$
    Add duty cycle measurement statements to netlist
    Add output capacitance $C_{load}$
    Simulate netlist in SPICE at $V_{dd} = V, T_{sim} = T$
    Get $\vec{d}$ and $\vec{o}$
    Load original netlist
    Annotate each transistor $k$ with $d_k \in \vec{d}$
    Add PWL voltage sources with $i_n = 50$ and $f(i_n) = 2^n$ with input signal slew $t_{slew}$
    Add delay measurement statements to netlist
    Add output capacitance $C_{load}$
    Simulate netlist in SPICE at $V_{dd} = V, T_{sim} = T$
    Get $\forall i_n \in \vec{i} : t_{delay}(i_n, rise)$ and $t_{delay}(i_n, fall)$
    for every $i_n \in \vec{i}$ do
        if $t_{delay}(i_n, rise) > t_{delay}^{worst}(i_n, rise)$ then
            Set $t_{delay}^{worst}(i_n, rise) = t_{delay}(i_n, rise)$
            Set $\vec{i}_{worst}(i_n, rise) = \vec{i}$
        end if
        if $t_{delay}(i_n, fall) > t_{delay}^{worst}(i_n, fall)$ then
            Set $t_{delay}^{worst}(i_n, fall) = t_{delay}(i_n, fall)$
            Set $\vec{i}_{worst}(i_n, fall) = \vec{i}$
        end if
    end for
end for
```

In our studied *NanGate* library [137] standard cells feature up to 6 inputs. On a normal Desktop PC (Core i5, 8GB RAM, SSD) finding the worst-case input vectors for a 2-input standard cell takes ≈ 7 seconds, while the rare complex 6-input standard cells take ≈ 2 hours.

Note, that the characterization under a range of input vectors $\vec{i}$ is trivially parallelizable and thus can be significantly enhanced on multi-core CPU. Our implementation is merely a proof of concept and not optimized in terms of performance, i.e. single-threaded CPU without aforementioned granularity stepping. Our studied NanGate library with just a few gates with 5 or 6 inputs did not require such optimizations, while larger cell libraries might.

6.1.4 Evaluation

Experimental Setup: Our work uses the NanGate FreePDK45 Generic Open Cell Library [137] for the netlists (with and without parasitics) of the standard cells. NanGate provides matching 45nm predictive technology models [3] transistor models. We employed the open-source *ngspice* as our SPICE engine. The BTI aging models are from [113] based on [31]. The aging calculations were performed at worst-case stimuli $V_{aging} = 1.2V, T_{aging} = 125°C$, while the cell characterization followed SS conditions $V_{SS} = 0.8V, T_{SS} = 125°C$.

6.1.4.1 Worst Input Vectors

We evaluate our approach by reporting the worst-case input vectors and the resulting propagation delay in a selection of standard cells. Our approach works for all combinational cells in the NanGate library[2], however as the internal duty cycles can only be shown on space consuming circuit schematics, only a few interesting examples can be shown to prove our concept. However, our introduction with Fig. 6.2 and Fig. 6.3 feature additional cases.

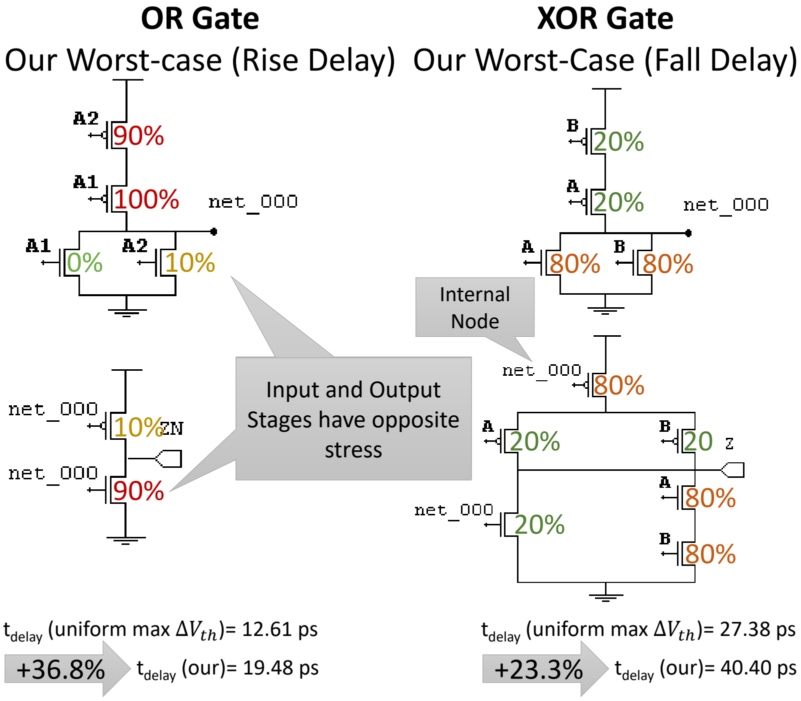

OR Gate
Our Worst-case (Rise Delay)
XOR Gate
Our Worst-Case (Fall Delay)
A2
90%
A1
100%
net_000
A1
0%
A2
10%
net_000
10%
ZN
net_000
90%
Input and Output Stages have opposite stress
B
20%
A
20%
net_000
A
80%
B
80%
Internal Node
net_000
80%
A
20%
B
20
Z
A
80%
net_000
20%
B
80%
t_{delay} (uniform max ΔV_{th})= 12.61 ps
+36.8%
t_{delay} (our)= 19.48 ps
t_{delay} (uniform max ΔV_{th})= 27.38 ps
+23.3%
t_{delay} (our)= 40.40 ps

Figure 6.5: Comparison of the traditional worst-case and our worst-case for the OR2_X1 and XOR2_X1 gate from NanGate library. Our worst-case has a 36.8% higher rise propagation delay in OR2 and 23.3% higher fall delay in XOR2 than the traditional worst-case, despite less aging-induced degradation per transistor.

1) OR2_X1: The OR2_X1 cell is implemented as a 2-input NOR Gate with fan-out 1 (X1) inverter stage. Therefore, one part of pull-up network is the PMOS transistor in the output stage (the inverter) and a part of the pull-down is the nMOS transistor in the output stage. However, our approach placed the main degradation in the earlier input stage, directly connected to the inputs. For this specific cell a signal activity of 0% for input A1 and 10% for input A2 leads to the highest degradation in the rise propagation delay (see Fig. 6.5). This non-uniform transistor degradation leads to 36.8% higher than the traditional worst-case with uniform $d_k = 1$ for all k transistors. This highlights, that is imperative to consider the non-uniform duty cycles created by the worst-case input vectors, as this 36.8% underestimation of propagation delay leads to unreliable systems. Should the OR2_X1 be stimulated by the worst-case input vectors, then timing violations would occur, as the clock frequency would be set to aggressively.

2) XOR2_X1: In Fig. 6.5, the XOR2_X1 cell is shown, as an example of a complex cell, i.e. a cell with internal nodes (transistor not connected directly to the input). While a simple cell NAND2_X1 has an obvious worst-case (see Fig. 6.2), the XOR2_X1 is hard to solve manually, despite exhibiting just 2 inputs. The challenge of the XOR2_X1 cell are 10 transistors, some in parallel, some sequential and some dependent of the output of others (i.e. connected to an internal node). Our brute-force automated approach solves complex cells effortlessly.

[2] Note, that the resulting signal activities and duty cycles are round numbers due to the clean 2:1 pmos:nmos sizing of the transistors within the NanGate library. For more optimized cells with different size ratios, like in commercial libraries, both signal activities and duty cycle are more interesting.

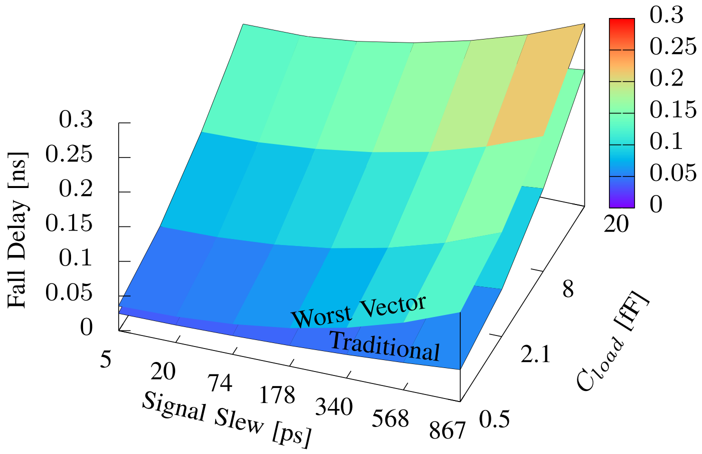


Figure 6.6: Impact of input signal slew and load capacitance on both the traditional worst-case (all transistors 100% duty cycle) and our worst-case (based on worst-case input vector) for the `XOR2_X1` gate from NanGate library. Due to the considerable differences, considering both signal slew as well as load capacitance is important to obtain accurate delays.

Impact of Signal Slew and Load Capacitance: Fig. 6.6 shows the impact of the signal slew and load capacitance on the circuit. As the impact of both signal slew and load capacitance is significant, it must be considered (in line with [28] [79] [82]). As signal slews and load capacitances can only be accurate considered in a SPICE circuit simulation, the necessity for a brute-force approach is clear.

Asymmetry for Inputs: Most standard cells are not symmetrical with respect to their topology. In `NAND2_X1`, the nMOS transistor connected to A1 ages less if the nMOS transistor connected to A2 is off. The lower transistor (nMOS:A2) acts as a resistor, increasing the potential at the source terminal of the upper transistor (nMOS:A1). BTI is driven by the gate-source potential difference [8] and thus the upper transistor ages less. This is called the stacking effect [83] and, as we employ circuit simulations, is automatically considered by our duty cycle extraction. The impact of the stacking effect, is that for the same input signal slew and load capacitance, $t_{delay}^{worst}(A2, fall) = 13.1ps$, while $t_{delay}^{worst}(A1, fall) = 10.3ps$. Details like these, which have a 30% impact ($10.3ps$ compared to $13.1ps$) on the result can only be considered by crude brute-force circuit simulation approaches.

6.1.4.2 Cell Library and Timing Analysis

Our approach can estimate worst-case timing for small circuits with limited number of inputs like standard cells, but also sense-amplifiers, ring-oscillators, etc. However, any commercial design vastly exceeds the capabilities of both our approach and circuit simulators, as they feature millions of cells, i.e. billions of transistors (e.g., current microprocessors). In order to determine the worst-case timing of such circuits, to determine the correct (maximal yet safe) clock frequency, we must employ timing analysis tools like *Synopsys PrimeTime*. Timing analysis tools can estimate the timing behavior in complex circuits. Our approach is fully compatible to these existing EDA tool flows, i.e. created compatible SS corners. Thus, we can leverage their timing analysis to evaluate the worst-case timing of complex circuits.

In order to re-characterize the SS corner, we employ our approach as described in the previous section. In practice, in Fig. 6.2 we obtain the duty cycles for the worst-case rise delay on the path A1:ZN. We then use the cell library characterization tool SiliconSmart to characterize the standard cell under that netlist and thus obtain a liberty file with the worst timing for the path A1:ZN(rise). We repeat this process for all paths A1:ZN(fall), A2:ZN(rise) and A2:ZN(fall), i.e. 3 more characterizations. We discard all the mis-matched timing information (e.g. A1:ZN(fall) information during the characterization of the cell with worst-case duty cycles based on A1:ZN(rise)). Then the 4 remaining cell timing tables are concatenated to form a full characterization of `NAND2_X1` in which each path was characterized with its corresponding worst-case input vector, i.e. the actual worst-case delay. This is then repeated for all other cells.

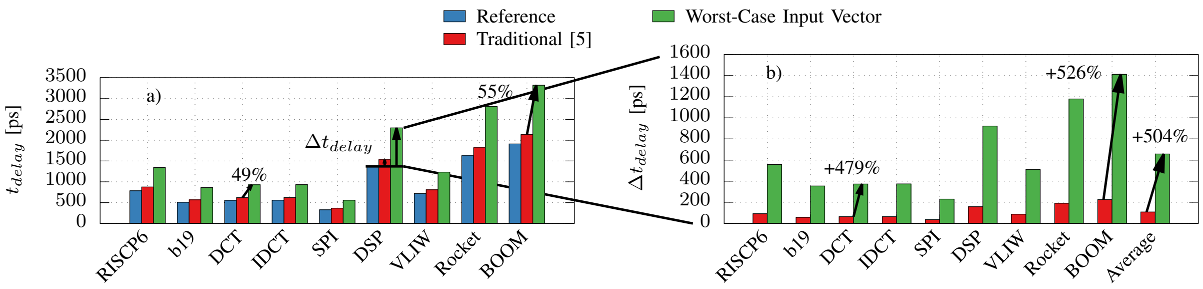


Figure 6.7: a) Comparison of the guardband (increase in clock period to tolerate aging-induced degradation) between our worst-case input vector and the traditional uniform degradation approach. b) Comparison of the guardband (increase in clock period to tolerate aging-induced degradation) between our worst-case input vector and the traditional uniform degradation approach

Table 6.1: Cell count of each circuit

Circuit	Cells	Source
B19	4 587	ITC'99 [140]
SPI	1 734	OpenCores [156]
DCT	26 694	UT Austin [82]
IDCT	26 754	UT Austin [82]
RISC6P	11 188	ASIP Designer [157]
VLIW	9 071	ASIP Designer [157]
FFT	2 235	ASIP Designer [157]
DSP	5 314	ASIP Designer [157]
Rocket	21 797	Berkeley [158]
BOOM	93 113	Berkeley [159]

With this re-characterized SS corner, we can perform a worst-case timing analysis of any complex circuit, as this cell library is a regular liberty file. Fig. 6.7a shows the delays of the various circuits for the traditional worst-case (uniform peak degradation for all transistors [81]) and our worst-case (worst-case input vectors). On average, each circuit delay is 52% longer with our worst-case compared to the traditional worst-case. The DCT (discrete cosine transformations) circuit exhihibits the smallest difference with 49.65%, while the Berkeley Out-or-Order Machine (BOOM) processor [159] shows the largest difference with 55.59%.

The shift in the critical path delay Δt_{delay} is shown in Fig. 6.7b. This isolates the impact of aging on the circuit delay, as the reference delay is subtracted. Fig. 6.7b illustrates the significance of our worst-case approach versus the traditional worst-case, as the impact of BTI on the circuit delay is on average 504%. It is lowest for DCT with 479% and highest for BOOM with 526%. Therefore, considering the traditional worst-case based on uniform peak transistor degradation underestimates aging-induced degradation by up to 526%. This results in timing violations if the circuit frequency is chosen based on these delays.

6.2 Self-Heating in Digital Circuits

This section aims to incorporate SHE into a SPICE circuit framework. To illustrate an application of this SHE integration in standard tools, we explore the impact of SHE on standard cells, the building blocks of large-scale digital circuits. This section is based on my publication [160].

6.2.1 Introduction

SHE is a recently uncovered degradation phenomena, which leads to elevated channel temperatures of transistors during their operation (see Section 2.3.4 for details). This work explores SHE in standard cells. Standard cells are logic gates (e.g., AND, OR, INV) and more complex blocks (e.g., AOI, buffer, adder, MAC) and their performance in characterized (simulated/measured) and stored in a cell library.

The contributions of this work are as follows:
(1) Estimating the SHE in single transistors and full standard cells with calibrated 14nm FinFET model-cards as well as evaluating the corresponding impact on the delay.
(2) Demonstrating how extracting the stimuli for SHE, such as switching frequency and duty cycle, can be performed in an automatic manner for every transistor within any circuit.

6.2.2 Background

We explored self-heating in general in Section 2.3.4 as well as SHE in SRAM arrays (SRAM cells and their periphery) in [161]. For the impact of workload (executed software) on self-heating in a full processor (large digital circuit), we refer to Section 4.3.

6.2.2.1 Calibrated Intel 14nm FinFET for Self-Heating

In our previous work we used data from a Intel 14nm FinFET transistor [162] to obtain a calibrated set of transistor model parameters, which is then used in a transistor model (e.g., the industry standard BSIM-CMG [53]). This set of parameters is called a “modelcard” and describes the electrical behavior of a FinFET based on many electrical and physical properties of the transistor (e.g., gate length, doping parameters, threshold voltage). It is typically used in circuit simulations like SPICE (e.g., HSPICE, Spectre, Virtuoso).

Our calibration uses Technology Computer Aided Design (TCAD), which is software, which simulates the properties of a transistor on a atomic scale. Materials and its manufacturing (deposition, annealing, etc.) is simulated in a process called S-process. When the TCAD model matches the measured data from a transistor (e.g., its C-V and I-V responses) then TCAD can be used to export a transistor model card.

We followed the electrical calibration with a SHE calibration (obtaining C_{th} and R_{th}) in [57]. Since in S-process the materials and their properties are modeled, extracting the thermal resistance R_{th} and capacitance C_{th} can be done by enabling and performing thermal simulations. One such thermal simulation for a DC scenario (constant voltage applied to the terminals) is shown in Fig. 2.24. For more details about the calibrated transistor model please refer to our previous work of electrical [162] and thermal [57] calibration.

6.2.2.2 Circuit Reliability Framework

Our circuit reliability framework is presented in Section 8.1. This framework can estimate the impact of aging (BTI and HCD) on circuits. For this purpose, it simulated the circuit with its activity (e.g., an addition of several numbers in an adder) and measured the voltages at each terminal (gate, source, drain, bulk) of each transistor. For the sake

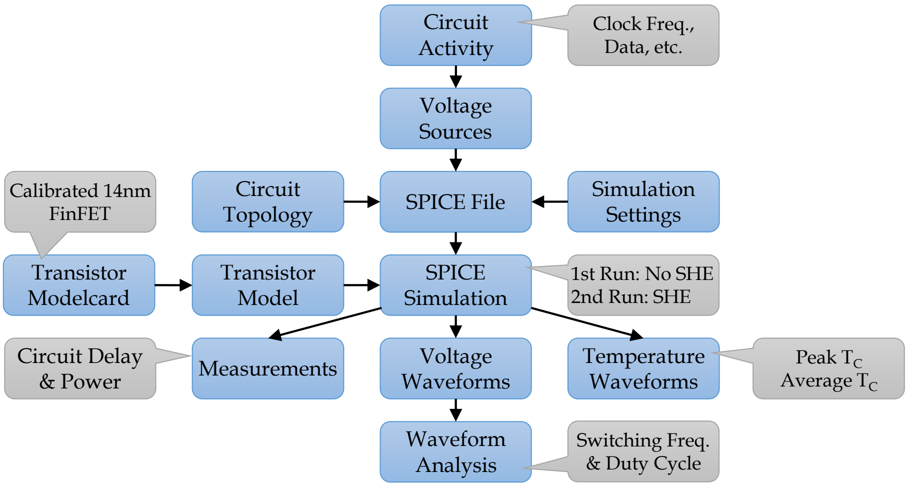


Figure 6.8: Flowchart of the proposed framework to automatically extract SHE-induced temperatures (per transistor) and its impact (on the circuit).

of simplicity, this work assumes a SPICE input file with voltage sources connected to the circuit topology, i.e. already existing activity. However, do note, that our tool can generate voltage sources for external inputs (for details see Section 8.1). In this section, we extend the framework to include self-heating in transistors as an additional degradation effect, which illustrates how SHE could be integrated analogously into any standard tool.

6.2.3 Circuit Reliability Framework - Extensions

The circuit reliability framework (see Section 8.1) had to be extended in two aspects: 1) Extract SHE-induced temperatures for each transistor. 2) Measure the properties of each transistor (e.g., switching frequency, duty cycle). The flowchart of our approach (highlighting just the relevant parts of the overall framework) is shown in Fig. 6.8.

6.2.3.1 Self-Heating

Self-heating is considered by the transistor model BSIM-CMG [53], which is the industry standard for SPICE simulations. In order to leverage BSIM-CMG, it needs R_{th0} and C_{th0} parameters (input values for the actual R_{th} and C_{th} calculation) as well as exponents for fin and other dependencies.

Since HSPICE is the employed SPICE simulator in the circuit reliability framework, we can use two different methods to access the temperature of each transistor. We can access the 5th terminal (next to gate, drain, source, gate) at which the temperature is modeled via the RC-thermal model of BSIM-CMG (see Section 2.3.4 and 4.3.4 for more details). Alternatively, we can use a transistor template (`lx594` for SHE) of HSPICE [92]. The difference is that the 5th terminal is an option offered by BSIM-CMG and works both in the internal implementation within HSPICE, as well as the external Verilog-A implementation. The template is HSPICE-function to access transistor properties (e.g., threshold voltage is another template). The two methods are generally numerically close, but do not match at all times (e.g., one rises before the other in a transition). To leave the choice up to end-user, our framework employs both methods to extract the channel temperature of the transistors. The evaluation in this work uses the 5th terminal.

The temperature is recorded at each point in time of the simulation for each transistor. So if 28 transistors are used (see Table 6.2) with 1ns time steps for 100ns, then 2800 temperatures are reported. Our next step, then analyses this data for the ease of use for the end-user.

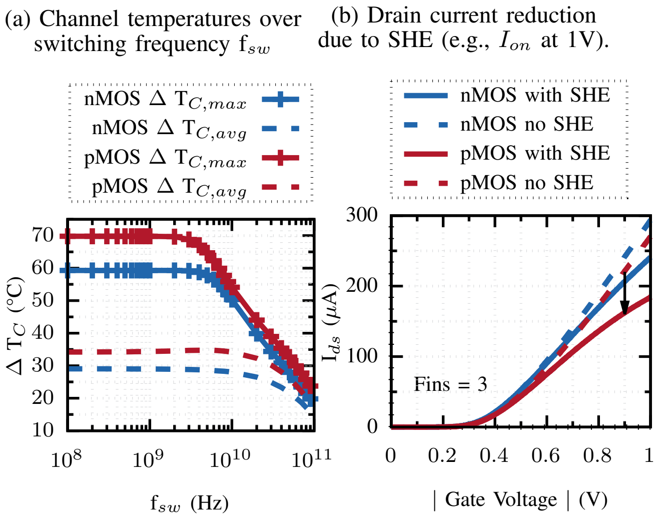


Figure 6.9: Transistor SHE elevates channel temperature and reduces ON-current.

6.2.3.2 Waveform Analysis

The circuit reliability framework reports voltage waveforms (gate, drain, source and bulk voltage for each point in time) and temperature waveforms for each transistor. We process these waveforms to extract the key parameters. For the voltage waveforms, we extract key activity parameters. If the voltage drops below 20% V_{dd} or above 80% V_{dd}, then we consider this as a transition. By measuring the number of transitions per transistor and dividing it by the simulation time, we obtain the switching frequency per transistor. Similarly, the ON-time of the transistor (voltage above 80% V_{dd}) divided by the total simulation time is the duty cycle (on-/off-ratio of the transistor). For self-heating we calculate peak and average (arithmetic mean) temperature of each transistor. All this information is then merged in a single transistor report to guide the designer to transistor hotspots and why these transistors are particularly hot.

6.2.4 Evaluation

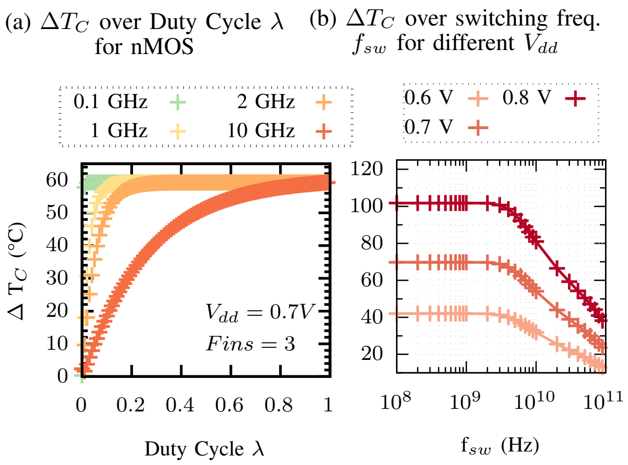
The experimental setup is our circuit reliability framework (see Fig. 6.8 and Section 8.1) which employs HSPICE v2019-20.SP3 as its circuit simulator. An overview of the used standard cells is given in Table 6.2 and we used the transistor model-cards from [162] and [57]. The supply voltage and number of fins is 0.7V and 3 fins unless otherwise noted. Red lines are pMOS transistors and blue lines are nMOS transistors.

6.2.4.1 Self-Heating in Transistors

In our employed 14nm FinFET technology, SHE manifests itself with a maximum of 60-70 $^{\circ}C$ increase in channel temperature for pMOS and nMOS when a duty cycle of 50% is applied. This is shown in Fig. 6.9a. The average temperature (dashed lines) is approximately half that with 30-35 $^{\circ}C$. Fig. 6.10b shows that increasing V_{dd} significantly increases SHE and Fig. 6.10a shows the impact of the duty cycle. For lower frequencies, the impact of the duty cycles becomes less and less, which is important as this work showed that for actual software the switching frequency (f_{sw}) is frequently in the kHz-range (see Section 4.3)..

6.2.4.2 Self-Heating in Standard Cells

We selected different standard cells, to highlight how differently SHE behaves in each cell. NAND2 and AOI21 are among the most used cells in standard cell circuit designs. INV was selected as its well-known and AND3 and the


(a) ΔT_C over Duty Cycle λ for nMOS

(b) ΔT_C over switching freq. f_{sw} for different V_{dd}

Figure 6.10: Transistor SHE-induced ΔT_C dependencies.

Name	Description	Bool/Function	# Transistors
INV_X1	Inverter	$\overline{A}$	2
NAND2_X1	Not-AND	$\overline{A1} \wedge A2$	4
AOI21_X1	AND-OR-Inverter	$B \wedge (A1 \vee A2)$	6
AND3_X1	3-input AND gate	$A1 \wedge A2 \wedge A3$	8
FA_X1	Full-Adder	In: A + B + CIN Out: S + COUT	28

Table 6.2: Description of standard cells.

FA (full adder) were selected since their behavior was interesting. The "_X1" means a fan-out of 1. The majority of the evaluation focuses on AND3 cell, as it exhibited the most distinct SHE behavior. In Fig. 6.11 we show the impact of different number of fins and supply voltage in both peak and maximum temperature. Note in particular, that for higher voltages (see Fig. 6.11e) not only the peak but also the shape of the curve changes and that for low number of fins (Fig. 6.11b) the nMOS is hotter than pMOS, while for more fins, this reverses (pMOS hotter than nMOS in Fig. 6.11d and 6.11f).

Fig. 6.12 shows maximum SHE-induced channel temperature increase (ΔT_C) in a comparison of the other cells (other than AND3). Note, how for INV in Fig. 6.12a the nMOS and pMOS are roughly equal, where as the AOI21 cell in Fig. 6.12c shows a wide temperature distribution. The frequency dependency across the cells also widely varies, in Fig. 6.12d two nMOS transistors exhibit a different frequency dependency and two pMOS differ as well. In general, each cell has 2-4 transistors, which reach peak ΔT_C, while most other transistors remain cold. This means, that selective hardening of these transistors (e.g., using more fins in these transistors to counteract the I_{ds} loss) should be considered to harden circuits against SHE.

As SHE hampers the performance of the transistors (see Fig. 6.9), the propagation delay of the standard cell also prolongs. Interestingly, not each cell is equally susceptible to SHE. For example, the NAND2 cell in Fig. 6.13b shows for different load capacitances C_{load} and slew rates (rate of change of the input voltages; steep or shallow transitions) an almost uniform delay increase of approximately 3%. On the other hand, the AND3 cell shows a strong dependence on C_{load} and weak dependence on the slew rate in Fig. 6.13a. Overall the AND3 gate in Fig. 6.13a features up to 15% delay degradation, which follows the trend of the more complex gates (up to 12% for FA).

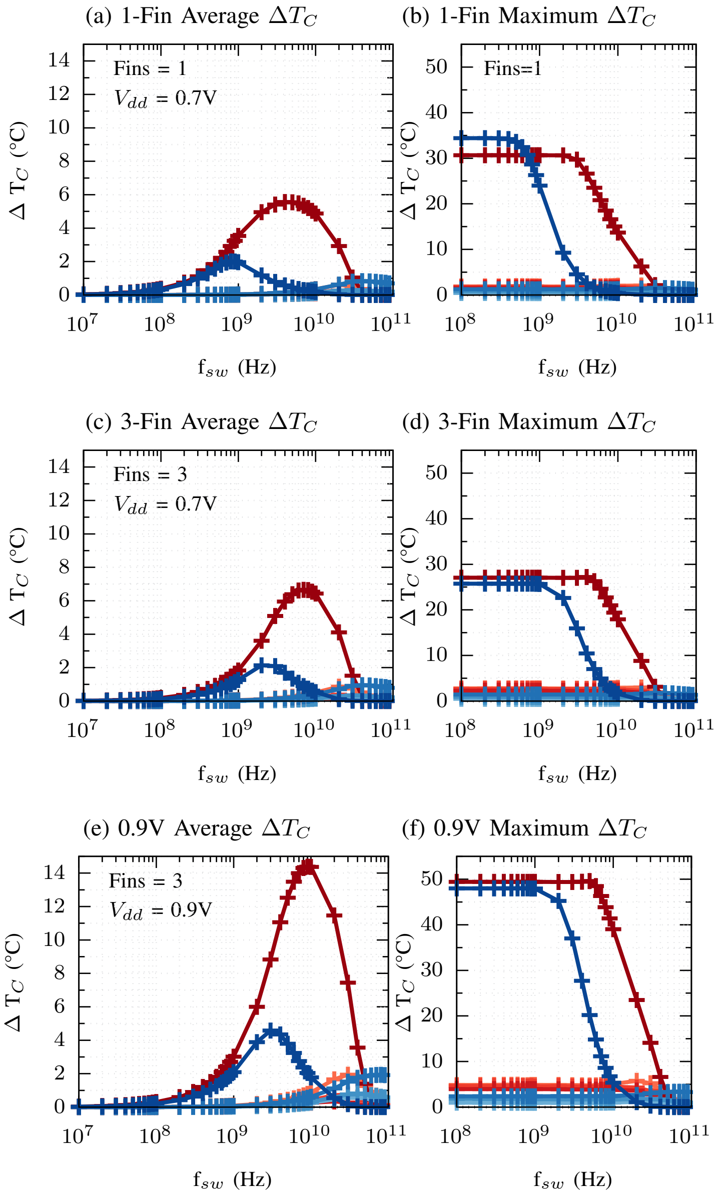


Figure 6.11: AND3_X1 Gate under various conditions.

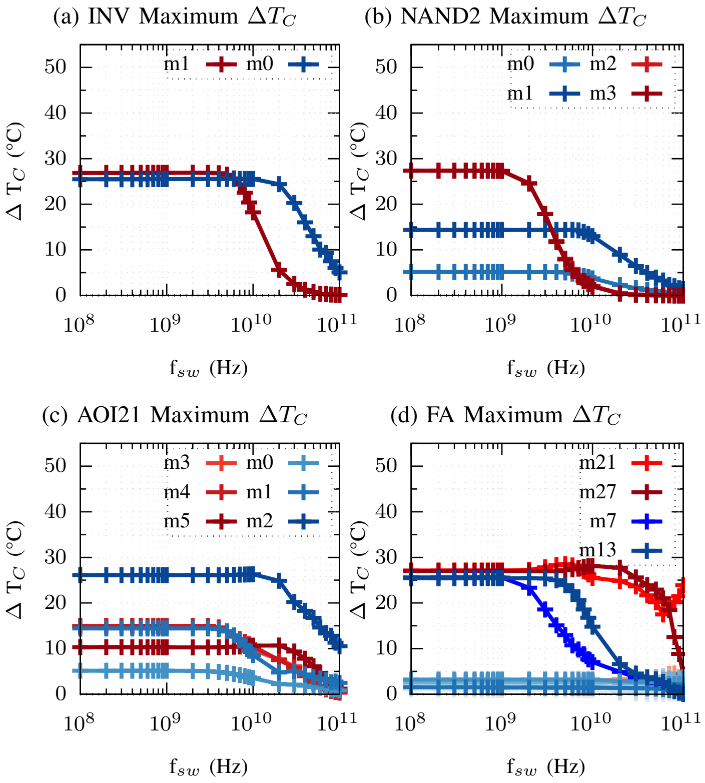


Figure 6.12: Comparison of maximum ΔT_C across different cells at 0.7 V, Fins=3, C_{load}=20fF

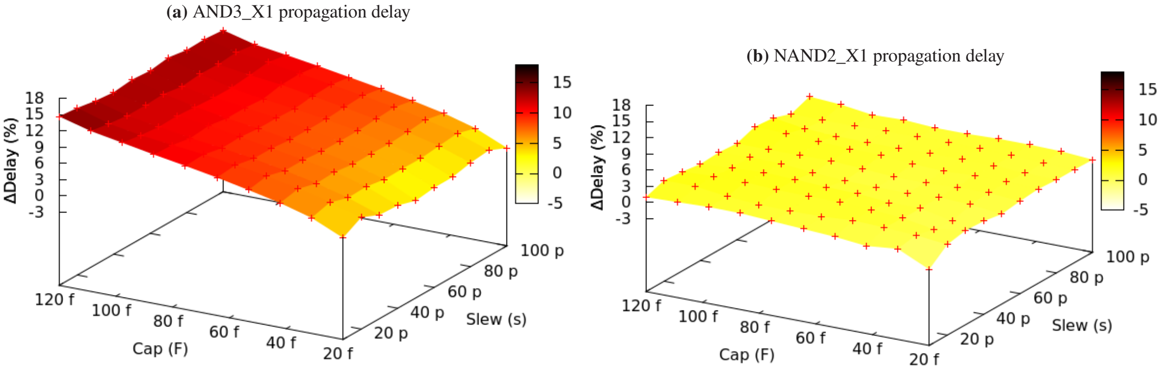


Figure 6.13: Delay shift in cells at 0.7 V, Fins=3. [163]

7 Step Four - Accelerating Standard Tools

Step three incorporated degradation models into the standard tools (SPICE circuit frameworks) and made them ready for custom reliability estimations. Now, the standard tools need to be accelerated as circuits are ever-growing with the ever-continuing scaling of technology allowing for ever-increasing number of transistors within a single circuit.

Step four is therefore the acceleration of SPICE via the massively parallel compute power found in graphic cards. This allows circuit simulations of large-scale analog and digital circuits with over a hundred thousand transistors without any loss in accuracy. This section is based on my publication [164].

7.1 GPU-SPICE: GPU-based Analogue/Mixed-Signal SPICE Circuit Simulator

Circuit simulations are an indispensable tool during circuit development to predict and validate circuit behavior. Simulation Program with Integrated Circuit Emphasis (SPICE) simulations [165] with transistor models like Berkeley Short-channel IGFET Model (BSIM) [52] is the industry standard for more than a decade. It enables circuit designers to estimate key circuit characteristics like delay, signal-to-noise ratio, voltage drops, power consumption, etc.

However, accurate results from SPICE simulations come at the cost of high computational complexity. SPICE simulations are computationally intensive as thousands of transistors need to be modeled, each with dozens of equations within the transistor model [53]. This is in direct conflict with the ever-growing demand to simulate larger circuits in SPICE. As circuits grow in number of transistors and thus complexity, EDA vendors offered heuristics like Static Timing Analysis (STA) to evaluate the circuit. The use of heuristics is necessary to make the signoff phase (verification of circuit functionality before fabrication) feasible with respect to time, as SPICE simulations were unfeasible for large circuits. However, these heuristics cannot solve the issue for every use case. For example, analogue designers still must use SPICE to verify timing, noise and other metrics during the signoff phase of circuit design. More importantly, recent automotive safety specification ISO 26262 demands full SPICE accuracy for stringent Tool Confidence Level 1 (TCL1) [94]. Hence, previous solutions to simulate large circuits in SPICE, like FastSPICE [91] cannot be used in these applications. FastSPICE sacrifices accuracy for faster simulations, to enable large circuits to be simulated (50-500K transistors [90]). Such FastSPICE variants are fast (3-30x [90]) but at the cost of inaccuracies up to 15% [91]. However, sacrificing accuracy is unacceptable for safety critical applications like ISO 26262. Yet, both analogue designs as well as automotive controllers can be large, i.e. reach or exceed millions of transistors (e.g., Infineon Aurix TC297TA with three 32-bit CPU cores and >8MB in-chip memory targeted for ISO26262). Therefore, there is a new demand for high-performance SPICE variants, i.e. high performance circuit simulations *at full accuracy*.

To achieve that, EDA vendors employ multi-core parallelism in SPICE with support for cluster computing in Spectre [94] and HSPICE [92]. However, in recent years, research took a different direction and instead started to exploit the massive parallelism in graphic cards (GPU) [100–103, 106]. GPUs provide thousands of computing cores (e.g. 3584 CUDA cores in a NVIDIA GTX 1080 Ti), enabling levels of parallelism similar to compute clusters. Additionally, GPUs are well-suited for the computational demands of SPICE as it is essentially a numerical solver

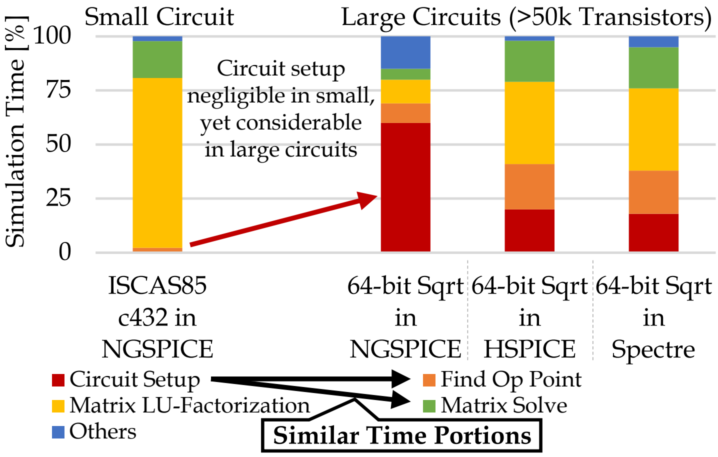


Figure 7.1: The circuit setup is only revealed as an equally important SPICE phase with respect to simulation times for large circuits (>50K transistors). Circuit setup scales worse than other SPICE phases (see Section 7.1.3). Therefore, for larger circuits (sqrt from [166] with 85K transistors) the circuit setup starts to consume considerable amount of time. This observation is universal across all EPFL benchmark circuits (sqrt, square, multiplier and divisor) in both open-source NGSPICE and commercial SPICE variants HSPICE and Spectre. If the other phases utilize their GPU implementations (e.g. [101–103, 105, 105, 106]), they shrink further, thus highlighting the importance of a GPU implementation of circuit setup.

for large equation systems [165]. As such, it is well-suited for the GPU computing architecture, which is designed for large-scale numerical matrix/vector computations.

7.1.1 Circuit Setup: The emerging Bottleneck in large Circuit SPICE Simulations

When evaluating SPICE (regardless which variant), researchers consistently observed that matrix factorization was the performance bottleneck [102, 103]. The circuit matrix (A) has to be factorized in two triangular matrices L (lower triangular matrix) and U (upper triangular matrix), so that $A = L \cdot U$. Factorizing large sparse matrices (dimension $> 1M$ with $> 20M$ entries) and then solving the matrix was extensively researched over the course of a decade both by industry (e.g., NVIDIA [107]) as well as academia [101–103]. Evaluating the transistor model (i.e. linearizing and solving transistor equations) was the other performance bottleneck in SPICE, which was overcome on the GPU by industry (e.g., Agilent [100], but no EDA vendors) and academia [106].

These advances in LU factorization and transistor model evaluation made the simulation of large (>1M transistors) circuits feasible. Yet, none of the previously mentioned SPICE implementations on the GPU, were evaluated with really large circuits. GPU SPICE researchers used standard benchmarks (e.g, ISCAS85 [167]) with less than <100K transistors [100, 107] [105] and the LU factorization research evaluated their algorithms (not entire SPICE) directly with matrix benchmarks [101–103]. The focus on LU factorization and device linearization is clear, as for smaller circuits, both phases are the major contributor to total execution time. However, when observing execution times of SPICE phases in larger circuits (which is done for the first time in this work) the circuit setup (parsing and representing circuit topology) may take up to 58% of the total execution time (see Fig. 7.1). While for the smaller ISCAS85 benchmark circuit [167], the circuit setup phase is completely negligible, in the large circuit, the circuit setup becomes as important as the other phases.

This can be explained by the lack of scalability of the circuit setup. Parsing the circuit netlist and creating an internal representation of the circuit topology is a process, which scales quadratically with circuit size in their current implementations. Creating the internal representation of circuit topology requires tracking which circuit components (e.g., transistor, resistor) are connected to each other. Therefore, every time a path from one node to another node is found, all previous circuit nodes are checked to examine if either the start or end node already exists. This process is used as it is simple to implement and is quite fast for the small numbers of nodes (e.g. below 25K nodes) found in small circuits. Yet, it scales badly to large number of nodes (100K transistors with each a drain, gate, source and bulk node equals to 400K nodes) as the number of comparisons scales quadratically ($\mathcal{O}(n^2)$) with the number of nodes. Note that compared to other parsing/setup problems (e.g., graph theory), circuit setup does more than just creating a internal representation (i.e., a graph) from a text description. For instance, deduplication of nodes, linking devices to their models (FinFET, FDSOI, MOSFET, resistor, capacitor, voltage sources, etc.), checking the corresponding files (transistor modelcards (their electrical parameters) exist and are valid, etc. This overhead is why it scales so much worse than comparable problems in computer science theory.

Fig. 7.1 demonstrates how the circuit setup consumes a considerable part of the execution time in both open-source (NGSPICE [95]) and commercial (HSPICE [92] and Spectre [94]) SPICE variants. Empirically, the commercial variants scale better than the open-source NGSPICE, but even there the circuit setup execution times become comparable to the other intense phases (similar portion to operation point and matrix solve) at approximately 250K circuit nodes (80K transistors). This is despite the fact, that the circuit setup is executed once, while the transient operation uses the solvers in a loop. Both commercially and in open-source SPICE, circuit setup (netlist parsing, matrix construction) scales just so much worse than the highly optimized solvers. Transient simulation steps severely affect overall simulation runtime and thus the contribution of the circuit setup to overall simulation runtime. For this reason, the number of steps is typically minimized, and we used sufficient steps to estimate a single cycle for propagation delay test (similar to library characterization tools). *We are the first to reveal the circuit setup as a performance bottleneck.* To overcome this bottleneck, this work presents a novel high-performance circuit setup in two implementations. The first is our database-based single-threaded CPU implementation, targeted towards the simplest possible solution (e.g., if no GPU is available). Our second implementation exploits the GPU to achieve a massively parallel circuit setup, which analyzes circuits up to millions of transistors. These contributions therefore provide sufficiently fast nodal analyses and enable SPICE users to simulate large circuits.

Additionally, our circuit setup is implemented in an open-source SPICE variant to enable other researchers to employ, adapt and extend our proof of concept or to reproduce our results. Our work results in identical internal representations of circuit topology in SPICE and as such is fully compatible with existing SPICE implementations. Therefore, it can be directly employed with the existing GPU-based LU factorization and device evaluation. Therefore, these phases are outside the scope of this work.

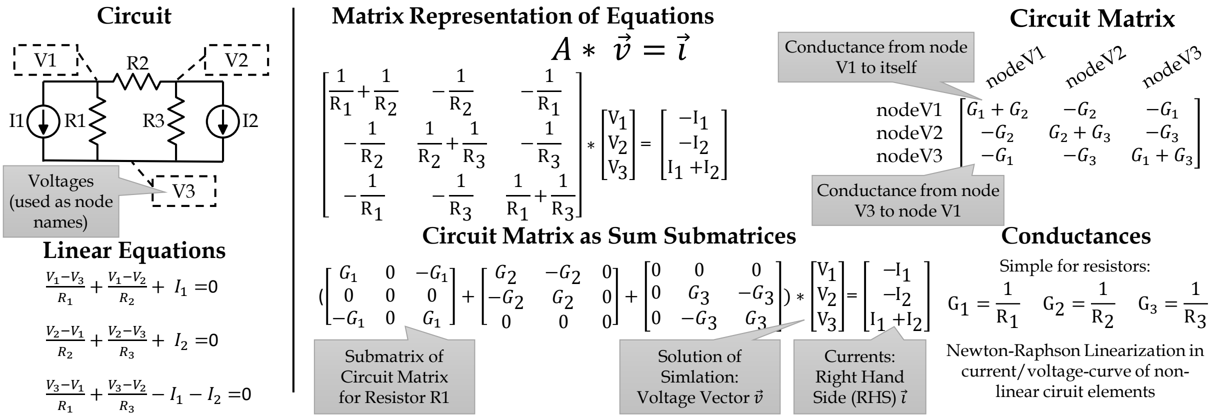
The contributions of this work are:

1. Revealing the circuit setup as a performance bottleneck in SPICE simulations for large circuits.
2. We present the first GPU-based circuit setup. Our GPU implementation of circuit setup overcomes this SPICE performance bottleneck in large circuits.
3. First application of a sorting algorithms for circuit setup, enabling existing algorithms to be employed for circuit simulations.

7.1.2 Related Work

To clearly distinguish our work from state of the art, we provided the following itemized overview (as a reminder on top of the related work Section 3):

1. We integrate our work in an already parallel SPICE simulation, resulting in the first SPICE implementation, which features parallel device evaluation, parallel device linearization and parallel circuit setup on the GPU. Other works focused on single phases and did not provide the code for a full SPICE version.
2. Our work outperforms multi-core SPICE implementations, even highly optimized current commercial SPICE flavors.


Circuit
V1
V2
V3
R1
R2
R3
I1
I2
Voltages (used as node names)
Linear Equations
$\frac{V_1-V_3}{R_1}+\frac{V_1-V_2}{R_2}+I_1=0$
$\frac{V_2-V_1}{R_2}+\frac{V_2-V_3}{R_3}+I_2=0$
$\frac{V_3-V_1}{R_1}+\frac{V_3-V_2}{R_3}-I_1-I_2=0$
Matrix Representation of Equations
$A * \vec{v} = \vec{\imath}$
Circuit Matrix
Conductance from node V1 to itself
nodeV1
nodeV2
nodeV3
Conductance from node V3 to node V1
Circuit Matrix as Sum Submatrices
Conductances
Simple for resistors:
$G_1=\frac{1}{R_1}$ $G_2=\frac{1}{R_2}$ $G_3=\frac{1}{R_3}$
Newton-Raphson Linearization in current/voltage-curve of non-linear ciruit elements
Submatrix of Circuit Matrix for Resistor R1
Solution of Simlation: Voltage Vector $\vec{v}$
Currents: Right Hand Side (RHS) $\vec{\imath}$

$$\begin{bmatrix} \frac{1}{R_1}+\frac{1}{R_2} & -\frac{1}{R_2} & -\frac{1}{R_1} \\ -\frac{1}{R_2} & \frac{1}{R_2}+\frac{1}{R_3} & -\frac{1}{R_3} \\ -\frac{1}{R_1} & -\frac{1}{R_3} & \frac{1}{R_1}+\frac{1}{R_3} \end{bmatrix} * \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} -I_1 \\ -I_2 \\ I_1+I_2 \end{bmatrix}$$

$$\begin{array}{c|ccc} & nodeV1 & nodeV2 & nodeV3 \\ \hline nodeV1 & G_1+G_2 & -G_2 & -G_1 \\ nodeV2 & -G_2 & G_2+G_3 & -G_3 \\ nodeV3 & -G_1 & -G_3 & G_1+G_3 \end{array}$$

$$\left(\begin{bmatrix} G_1 & 0 & -G_1 \\ 0 & 0 & 0 \\ -G_1 & 0 & G_1 \end{bmatrix} + \begin{bmatrix} G_2 & -G_2 & 0 \\ -G_2 & G_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & G_3 & -G_3 \\ 0 & -G_3 & G_3 \end{bmatrix}\right) * \begin{bmatrix} V_1 \\ V_2 \\ V_3 \end{bmatrix} = \begin{bmatrix} -I_1 \\ -I_2 \\ I_1+I_2 \end{bmatrix}$$

Figure 7.2: Overview of definitions for SPICE. Shown are circuit nodes (denoted by the voltages V1-V3), 3 resistors (R1-R3) and 2 current sources (denoted by the current I1 and I2). First the elements are linearized (see Fig. 7.3 for non-linear elements), then equations are formed for each node and finally circuit matrix is constructed to represent the circuit behavior.

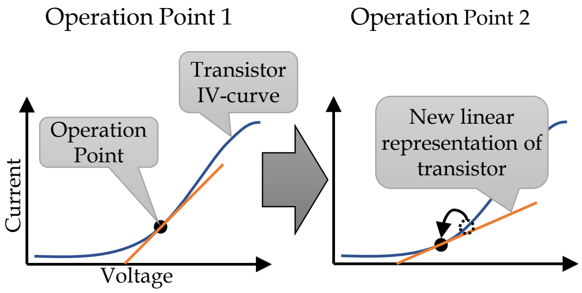


Figure 7.3: Two iterations of the search for the operation point with the corresponding linearization of a MOSFET transistor.

7.1.3 SPICE Background

7.1.3.1 Definitions

Circuit Node: A circuit node is a point within the circuit topology. For example, one terminal of a transistor (gate, drain, source, bulk) or the positive/negative terminal of a capacitor (see Fig. 7.2). Some nodes like supply voltage (VDD) or ground (GND) are connected to many circuit elements.

Circuit Element: A circuit element is a component within the circuit. This can be a physical component like a transistor, resistor or capacitor as well as a virtual element like an ideal voltage or current source.

Request: A command to obtain a conductance for a path through a circuit element (e.g. from the gate to the drain of a MOSFET). Requests are identified by a pair of circuit node IDs, representing starting and end node (see top of Fig. 7.7). Requests are directional as circuit elements (e.g., diodes) are not symmetric with respect to resistance/conductivity.

Operating Point: The operating point of a circuit element is the bias voltage of a non-linear element at which the element is linearized. It is the electrical representation of the voltage value in a linear approximation according to the Newton-Raphson method (see Fig. 7.3).

Circuit Matrix: The circuit matrix A stores $G(i, j)$ with conductance G for a path from the node with circuit node index i to the node with circuit node index j (path defined by position (i, j)). These node indices are unique integers assigned to each unique node (a string in the netlist) during netlist parsing (see Fig. 7.2). The conductance describes the combined unidirectional behavior of the circuit elements connecting from node i to node j. Note that the sparse matrix is stored in Compressed Sparse Column (CSC) format:

$$
A = \begin{pmatrix}
\underset{(0,0)}{9} & 0 & 0 & \underset{(0,3)}{12} & 0 \\
0 & 0 & \underset{(1,2)}{7} & 0 & \underset{(1,4)}{22} \\
0 & \underset{(2,1)}{16} & 0 & 0 & 0 \\
0 & 0 & \underset{(3,2)}{7} & 0 & \underset{(3,4)}{13}
\end{pmatrix} \tag{7.1}
$$

$$
\begin{aligned}
val &= \begin{pmatrix} \underset{(0,0)}{9} & \underset{(2,1)}{16} & \underset{(1,2)}{7} & \underset{(3,2)}{7} & \underset{(0,3)}{12} & \underset{(1,4)}{22} & \underset{(3,4)}{13} \end{pmatrix} \\
row &= \begin{pmatrix} \underset{(0)}{0} & \underset{(1)}{2} & \underset{(2}{1} & \underset{2)}{3} & \underset{(3)}{0} & \underset{(4}{1} & \underset{4)}{3} \end{pmatrix} \\
col &= \begin{pmatrix} 0 & \underset{(0)}{1} & \underset{(1)}{2} & \underset{(2)}{4} & \underset{(3)}{5} & \underset{(4)}{7} \end{pmatrix}
\end{aligned}
$$

The CSC format stores all matrix values top-to-bottom then left-to-right in the val array. Then all row indices in the row array. Lastly, the number of elements in column i as $col[i+1] - col[i]$, thus the first element is always 0 and the last element always to total Number of Non-Zero elements (NNZ).

7.1.3.2 SPICE Simulation

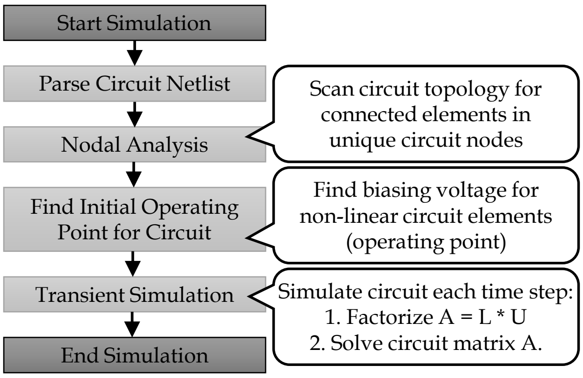

Start Simulation
Parse Circuit Netlist
Nodal Analysis
Find Initial Operating Point for Circuit
Transient Simulation
End Simulation
Scan circuit topology for connected elements in unique circuit nodes
Find biasing voltage for non-linear circuit elements (operating point)
Simulate circuit each time step:
1. Factorize A = L * U
2. Solve circuit matrix A.

Figure 7.4: Brief overview of the major steps in a SPICE simulation. For details of parsing and circuit setup see Fig. 7.6

An overview of the SPICE simulation flow is given in Fig. 7.4. Now, we explore the individual steps:

Parse Netlist: Circuits in SPICE are represented in a circuit topology file, called a circuit netlist. This file is parsed to extract circuit topology, number of circuit nodes and circuit elements, solve all includes of other files, simulation settings, etc. During parsing each entity (element, node, etc.) with a name (string) gets a unique integer assigned to it.

Flatten Netlist: After the netlist is parsed, all circuit hierarchy (subcircuits, etc.) is removed and the whole circuit topology is flattened to solely circuit elements (transistors, resistors, etc.).

Circuit Setup: circuit setup is called “cktSetup” in NGSPICE, “setup” in HSPICE and “parsing” in Spectre. It creates an internal representation of circuit topology by constructing the circuit matrix. The circuit topology is represented by a two-dimensional linked list in SPICE. For each path through a circuit element, pairs (start node, end node) called *requests* are created according to the circuit netlist (see top of Fig. 7.6). For example, a MOSFET $m13$ has its gate connected to node $n12$, drain to node $n8$, source to $n37$ and bulk to $n37$. This results in sixteen pairs (e.g. $(n8, n12)$, $(n37, n8)$, $(n12, n37)$, $(n37, n37)$) to be inserted in the linked list. For each node, the linked list is entered based on its node parse ID (see Parse Netlist step). If no node exists at that position, then the linked list is extended to the ID of the node. Should the node already exist, it is inserted at correct position (end node IDs ascend monotonically). Then, the pointer to that (start node, end node) pair is linked to the currently processed path. To insert the end node, all previous end nodes (thousands for VDD, GND and other interconnected nodes (bus, clock, etc.)) are traversed, which consumes a lot of time. Even if the end node already exists, then the list for the start node has to be traversed anyway to know the pointer to that end node and to link that node to the path of the circuit element currently processed. This linked list is reasonably efficient process for small circuits with limited circuit elements and thus limited paths, but the linked list approach is unsuitable for large circuits as it does not scale well. Note, that other data structures are used in other SPICE simulators (e.g., in Xyce [96]) and that this is solely an explanatory example.

Circuit Matrix: After the circuit setup is complete (i.e., the entire netlist is parsed), then a circuit matrix is constructed. This matrix has n number of rows and columns, with n being the number of unique nodes in the circuit. The values in the matrix are the conductances G of the node in row n_{row} to the node in the column n_{col}.

Then SPICE continues with LU-factorization, device evaluation and other steps of the transient simulation. Related work [106] [100] [101] [103] [102] already focused on these phases, so they are outside of the scope of this work.

Parity with original circuit setup: For this work, it is important to note, that our circuit setup results in identical data structures and values compared to existing circuit setup. Our internal representation of the circuit topology is 100% identical to the original NGSPICE implementation and thus the later SPICE phases (e.g., matrix LU factorization, matrix solving) are not affected in any way (neither runtime, nor accuracy, nor convergence).

7.1.4 Our Parallel Circuit Setup

The circuit setup in NGSPICE relies on linked lists as their data structure [104] during the circuit setup. This algorithm scales with $\mathcal{O}(n^2)$ and as such is not suitable for large circuits with lots of circuit elements. Note, that our approach does not rely on the existence of the linked list or a specific format for the circuit matrix. Hence, it is applicable to other simulators of the SPICE family (PSPICE, LTSPICE, ELDO, etc.) or SPICE-like simulators like FastHenry (inductance extraction tool) [168].

7.1.4.1 SQL Circuit Setup

Our first attempt to provide a faster circuit setup, was implementing the circuit setup with a SQLite database. SQLite is an open-source self-contained implementation of the Standard Query Language (SQL) [169]. It builds a B-Tree to store information and allows insertion, search and deletion of an element in $\mathcal{O}(log(n))$ [169]. SQLite is used to support the creation of the linked list. Every element inserted in the linked list is also inserted in the SQL database with a direct link to its index (address of element) inside the linked list. This allows us to search for already existing nodes in SQL (hence in $\mathcal{O}(log(n))$) and if such a node is found, SQL returns the index of the last circuit element attached to that node. Then SQL requires an update to our last element. This action again is in $\mathcal{O}(log(n))$. If the node pair is already part of the list, then SQL returns the index again in $\mathcal{O}(log(n))$. After all paths requests are served our supporting SQL database can be deleted as the circuit matrix is now ready for creation and all paths are linked to their node pair.

In summary, our non-invasive SQLite implementation supports the creation of the linked list, by improving the look-up for pre-existing nodes and finding the correct insertion location of a node pair faster. As we show in our evaluation (Section 7.1.5), this results in a negligible overhead for the circuit setup of small circuits (i.e. we have no high initialization cost, but the circuit is not large enough to make SQL faster than the linked list), but a decent speedup for large circuits. The commercial HSPICE and Spectre are comparable in performance to our SQL solution. However, as we show in Table 7.2 SQL still requires an hour for circuit setup of large circuits. In fact, of all simulation phases, Spectre consumes typically the 2nd most time in circuit setup, see Table 7.1. Thus, while our SQL implementation needs no additional hardware in the form of a GPU, it is not fast enough for really large circuits.

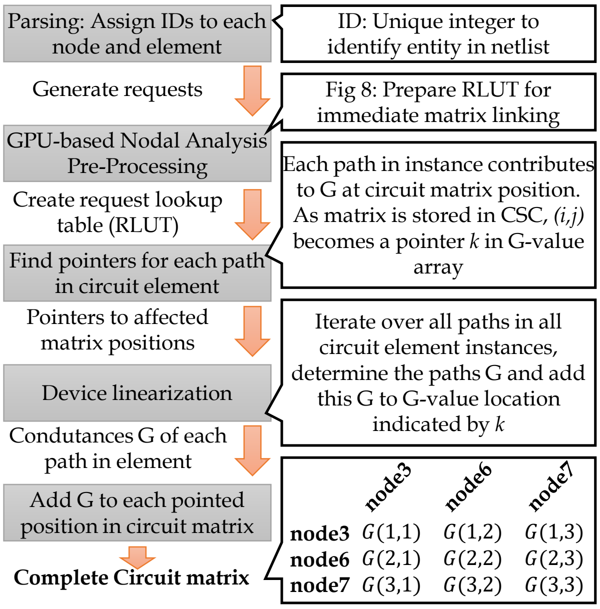

Parsing: Assign IDs to each node and element
ID: Unique integer to identify entity in netlist
Generate requests
Fig 8: Prepare RLUT for immediate matrix linking
GPU-based Nodal Analysis Pre-Processing
Each path in instance contributes to G at circuit matrix position. As matrix is stored in CSC, (i,j) becomes a pointer k in G-value array
Create request lookup table (RLUT)
Find pointers for each path in circuit element
Pointers to affected matrix positions
Iterate over all paths in all circuit element instances, determine the paths G and add this G to G-value location indicated by k
Device linearization
Condutances G of each path in element
Add G to each pointed position in circuit matrix
Complete Circuit matrix
node3
node6
node7
node3 G(1,1) G(1,2) G(1,3)
node6 G(2,1) G(2,2) G(2,3)
node7 G(3,1) G(3,2) G(3,3)

Figure 7.5: Process flow of Radix circuit setup

7.1.4.2 GPU-based Radix Circuit Setup

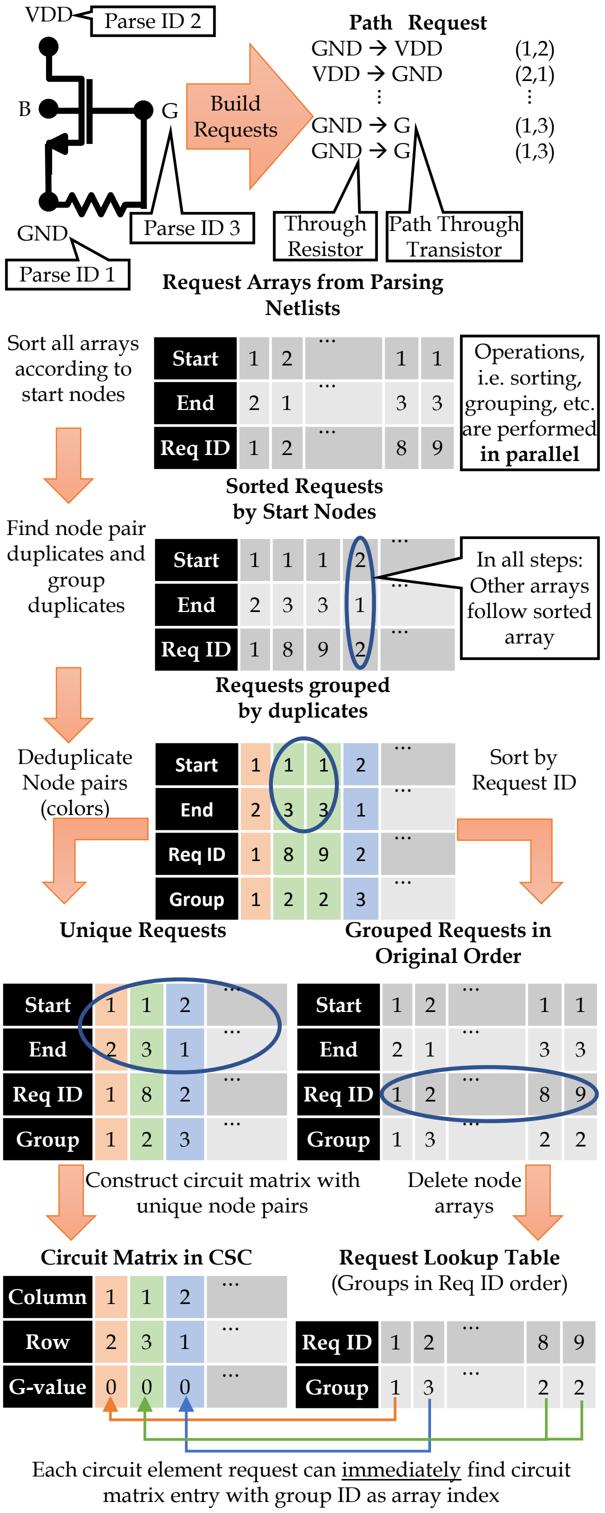
Our second GPU implementation uses a sorting algorithm (Radix sort) to construct the internal representation of the circuit topology. This is a novel and elegant use-case for a sorting algorithm to be employed in circuit setup. Importantly, using Radix sorting for circuit setup leads to the following advantages:

- Massive parallelism to utilize the massive parallel hardware of the GPU
- Independence of the runtime on the order (ascending, descending, random) of the input data
- Accepts any number of data points (for example not solely 2^n data points)
- Low initialization overhead

While our SQLite-supported circuit setup is fast, it is still not fast enough for circuits beyond 100K circuit elements. While its big-O-notation with $\mathcal{O}(log(n))$ is certainly desired, it is single-threaded and requires significant main memory, as circuit elements are stored in both the SQL database and linked list. In order to process the circuit topology in parallel and be computationally lightweight, we restructure the circuit setup and circuit matrix creation. It is based on an implementation of the massively-parallel Radix sorting algorithm on the GPU. Applying a sorting algorithm to circuit setup is not straightforward, so we explain our process in the following paragraphs and Fig. 7.5.

General Overview: First, we pre-process the requests to obtain the Request Look Up Table (RLUT). After pre-processing is done, we build the circuit matrix immediately in a single step. Finally, using the RLUT we can link each request immediately with its corresponding circuit matrix location. This completes the circuit setup and allows device linearization and matrix solving in NGSPICE to occur.

Pre-Processing: Pre-processing is shown in Fig. 7.6. First, all requests are stored in three request arrays. The first array is the start node array (start node of path), followed by an end node array (end node of path) and finally request indices (unique ascending integers to identify a request from a path). After the parsing, all requests are stored in their arrays. Then the arrays are sorted with the Radix algorithm (explained in the next section) according to the start nodes (i.e. (1,1) before (2,1)) and secondary by end nodes (i.e. (1,2) is before (1,3); see sorting section). In this section, all arrays follow the order of the sorted array. Therefore, after sorting the nodes, the request ids followed this order (the second request (2,1) with ID 2, is still (2,1,2) just now stored in each array at the 4th position).


VDD
Parse ID 2
B
G
GND
Parse ID 1
Parse ID 3
Build Requests
Path Request
GND → VDD (1,2)
VDD → GND (2,1)
⋮ ⋮
GND → G (1,3)
GND → G (1,3)
Through Resistor
Path Through Transistor
Request Arrays from Parsing Netlists
Sort all arrays according to start nodes
Operations, i.e. sorting, grouping, etc. are performed **in parallel**
Sorted Requests by Start Nodes
Find node pair duplicates and group duplicates
In all steps: Other arrays follow sorted array
Requests grouped by duplicates
Deduplicate Node pairs (colors)
Sort by Request ID
Unique Requests
Grouped Requests in Original Order
Construct circuit matrix with unique node pairs
Delete node arrays
Circuit Matrix in CSC
Request Lookup Table (Groups in Req ID order)
Each circuit element request can immediately find circuit matrix entry with group ID as array index

Sorted Requests by Start Nodes

Start	1	2	…	1	1
End	2	1	…	3	3
Req ID	1	2	…	8	9

Requests grouped by duplicates

Start	1	1	1	2	…
End	2	3	3	1	…
Req ID	1	8	9	2	…

Start	1	1	1	2	…
End	2	3	3	1	…
Req ID	1	8	9	2	…
Group	1	2	2	3	…

Unique Requests

Start	1	1	2	…
End	2	3	1	…
Req ID	1	8	2	…
Group	1	2	3	…

Grouped Requests in Original Order

Start	1	2	…	1	1
End	2	1	…	3	3
Req ID	1	2	…	8	9
Group	1	3	…	2	2

Circuit Matrix in CSC

Column	1	1	2	…
Row	2	3	1	…
G-value	0	0	0	…

Request Lookup Table
(Groups in Req ID order)

Req ID	1	2	…	8	9
Group	1	3	…	2	2

Figure 7.6: Process flow of our nodal pre-processing to obtain the Request Look Up Table (RLUT) and circuit matrix in CSC format in parallel. With RLUT each request can immediately find its matching entry in the circuit matrix, as requests are in order of appearance and have the index to access the conductance value G in the val array.

After the requests are sorted, an array is added in which the node arrays are grouped. Each duplicate request[1] (e.g. (1,3) and (1,3)) has the same group ID. As the requests are sorted, grouping can be performed in parallel. The entire array is divided in thousands of sections (overlapping by a single request for corner cases), with each section processed by a single thread. Each thread compares a request with its left neighbor. If both requests are identical, the current request gets assigned a 0, if they are different it is a 1. Thus, each 1 marks a boundary between groups of duplicates. Then the parallel prefix algorithm (moving sum algorithm, a part of radix, see [170]) assigns each group a unique ascending group ID (shown as colors in Fig. 7.6).

After each request has its group ID, our pre-processing splits in two parts. The left part of Fig. 7.6 is used to construct the circuit matrix (faster than via linked list), while the right part creates the RLUT, i.e. linking each request to a position in the matrix.

In the left part of Fig. 7.6, we copy the left-most request in each group to a new set of arrays. To ensure high performance, finding the left-most request in a group and copying it is performed in parallel. By using the same overlapping sections as before, each thread looks for different group IDs and checks if they are different. If the group ID is different, the request is copied to the new array, if not, nothing is copied. Thus, the new arrays feature all unique requests, i.e. we deduplicated the original request arrays. These unique requests indicate where in the circuit matrix a conductance will be placed later, as each request stems from a path. Therefore, based on the start node and end node array, an empty matrix in CSC format can be created. As we know each location within the circuit matrix, row and col values can be determined (see eq. 7.1). The value array val can be allocated and initialized with zeros, but their G values will be determined during the device linearization later.

In the right part of Fig. 7.6, we sort according to request ID on the original arrays. These arrays still feature all requests, including the duplicates. By sorting according to request ID, we restore the original order of the requests. However, now the group ID of each request is known. The important detail is that the group ID integer is exactly the index of the val array to which the request has to be linked. By storing the pointer to the memory location of $val[groupID]$, we can link each request (and thus each path) to the corresponding circuit matrix entry. This is performed by directly linking paths to the CSC format and no intermediate linked list is necessary. Therefore, the two arrays with request ID and group ID form a table called the Request Look Up Table (RLUT). In order to link each path in each circuit element, each request is processed iteratively. For each request we add the pointer to $val[groupID]$ to the corresponding path. As the RLUT is in the original order of the requests, this iteration and lookup in RLUT is in constant time $\mathcal{O}(1)$. During device linearization, which traverses over each path in circuit instance, we now know immediately where to add our conductance G value.

Sorting: Our aforementioned methodology is extremely fast, as everything is in parallel and scales well. However, it relies on fast parallel sorting, as we sort by start nodes, end nodes and later by request ID in the right side on millions of array elements (requests outnumber number of transistors vastly, see Table 7.2). For this purpose, we selected the Radix sorting algorithm for which high performance GPU implementations exist. We used the four-way parallel Radix [170]. This implementation builds on top of parallel prefix algorithm, which was also used in our pre-processing during grouping. Details of four-way parallel Radix [170] are outside of the scope of this publication, but an explanation of general parallel Radix is given in Fig. 7.7 and its caption. Radix is a high performance massively parallel sorting algorithm, which is perfectly suited for our purposes. One important aspect is that Radix is a stable sorting algorithm, so by sorting according to a primary aspect followed by secondary aspect is possible. So (3,1); (1,4); (2,1); (1,1) is first sorted according to the *secondary* aspect (2,1); (1,1); (3,2); (1,4) and then according to the *primary* aspect (1,1); (1,4); (2,1); (3,2). In the second sort equal values (1,1) and (1,4) did not change order with respect to each other, which represents the stable property.

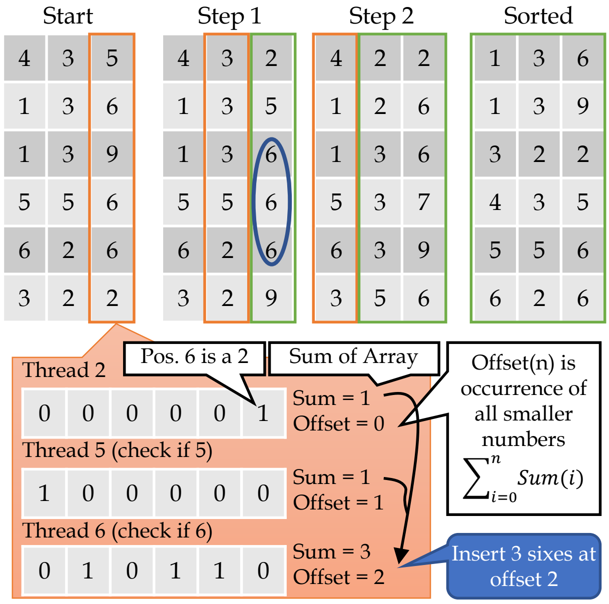


Figure 7.7: Parallel Radix sorting algorithm. Data is broken into blocks (digits in this example, 3 bits in real implementation), then each block is sorted individually. A number of threads equal to the block range are spawned. Each thread writes to an array if its specific value (e.g. a 6) is found at the current position. The array is filled with 0 if at the current position (e.g. no 6 at position 1) the value was not found or filled with a 1 if the value was found (e.g. found 6 at position 2). Then the sum of the array is calculated in each thread to know how frequently that value occurred and the offset is determined as the sum of all occurrences of all smaller numbers. Then the thread can write its values for the number of occurrences at the offset (e.g. write three 6 after offset of 2). After the first block is sorted, the second block (higher value bits) can be sorted.

7.1.4.3 Parallel Implementation of Prefix Sum Calculation and Radix Sorting

Parallel Prefix Sums: A parallel prefix algorithm calculates a *implicit* or *explicit* prefix of an array with elements e_i:

$$\text{explicit prefix: } \sum_{i=0}^{n} e_i \qquad \text{implicit prefix: } \sum_{i=0}^{n-1} e_i$$

Input: [3 0 0 4 1 5 6 7]

explicit: [3 3 3 7 8 13 19 26] implicit: [0 3 3 3 7 8 13 19]

The two prefixes have different use-case scenarios: explicit prefixes are used for summation, while implicit prefixes help in creating indexes. See the values “offset” and “sum” in Fig. 7.7 for examples. In this work, we employ the “Work-Efficient Parallel Scan” prefix sum algorithm presented in [171], which bases itself on [172]. This prefix sum algorithm consists of 2 phases: the up-sweep (see Fig. 7.8) and down-sweep (see Fig. 7.9). In the up-sweep, the algorithm calculates the sum of the values in the array by combining the sums logarithmically. While this is sufficient to calculate only the sum, it is not a complete prefix. A complete prefix has the correct sum of all previous values (including the current value) at each position within the array. To achieve that, the down-sweep is necessary, which fills in all the positions. The advantages of this implementation are that it uses solely $log(n)$ steps (n is array length) in both up- and down-sweep and is heavily parallelizable.

The individual steps (arrows in Fig. 7.8 and 7.9) can be parallelized. However, to further increase performance in larger arrays, an additional division of the array into blocks can be employed. This additional division of the data

[1] Duplicates occur, if multiple circuit elements are parallel to each other, i.e. connecting the same nodes.

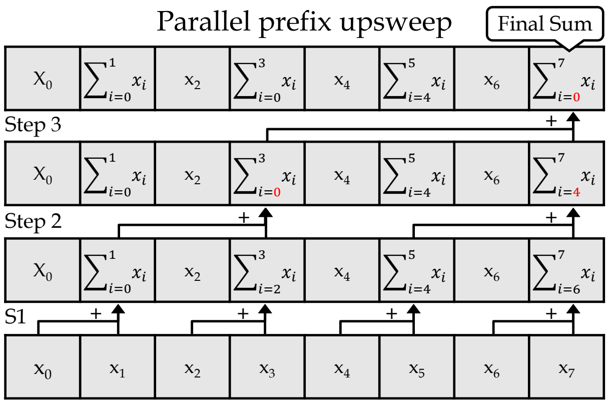


Figure 7.8: Up-sweep of employed prefix algorithm from [171]. The up-sweep calculates in $log(n)$ steps the overall sum of all the values in the array. This itself is sufficient for summation ("sum" in Fig. 7.7, but insufficient for indexing "offset" in Fig. 7.7).

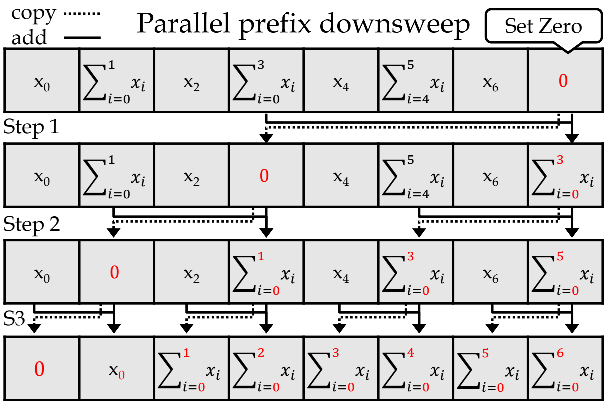


Figure 7.9: Down-sweep of employed prefix algorithm from [171]. The down-sweep fills the entire array with the prefix values (sum until that position) instead of just calculating the overall sum. This is frequently necessary for indexing (e.g., finding the position of an item in a list).

ensures allow additional parallelism to utilize the massive parallel hardware architecture of a GPU and to reduce the memory footprint of each thread (to overcome bandwidth and memory hardware limitations). Detailed explanation provided in Fig. 7.10.

Parallel Radix: Radix is a sorting algorithm which can perform high-performance sorting and is well-suited for employment on the GPU [173] [170]. Our work employs 4 way parallel Radix sort based on [170]. This Radix sort algorithm breaks integers down to the bit-level and sorts the integers according to n-bit patterns. The length of the patterns is arbitrary for the algorithm and should be chosen according to the hardware architecture. It determines both memory footprint of each thread as well as the maximum parallelism employed by Radix. Longer bit patterns result in larger pattern tables (i.e. higher memory consumption). However, each pattern can individually be checked by an individual thread, i.e. longer patterns result in a more parallelism. Our implementation uses 4-bit patterns, resulting in 16 patterns total. These 16 patterns each can be checked by a single thread, which fits nicely in a CUDA warp (up to 32 CUDA cores which execute a single SIMD instruction lockstepped). 5-bit patterns resulting in full utilization (32 threads) of the warp would leads to fragmented writing to local memory, as each thread writes not in a multitude of 2^n. Therefore, we chose 4-bit patterns as the best trade-off between warp utilization and preventing cache/memory fragmentation.

Another important setting is the block size of the parallel prefix algorithm. We used a block size of 1024, as this leads to using the maximum number of threads (1024) per CUDA kernel (the programming CUDA entity). Note how only on step 1 up-sweep in Fig. 7.8 and step 3 on down-sweep in Fig. 7.9 the full number of threads is used. Therefore, the initial number of threads corresponds to the maximum of the hardware architecture, as memory/bandwidth considerations are secondary. In each following step (2,...,n in the up-sweep or 1,..,n-1 in the down-sweep) the algorithm uses exponentially less threads, significantly reducing memory and bandwidth footprints. Therefore even

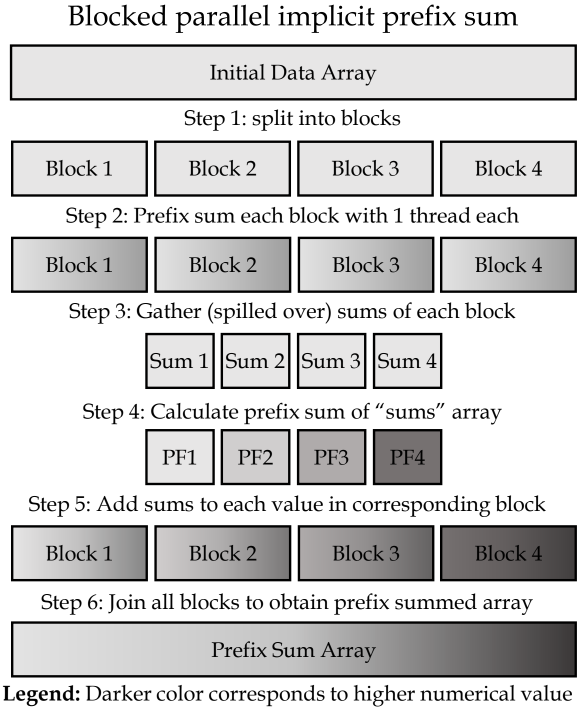

Blocked parallel implicit prefix sum
Initial Data Array
Step 1: split into blocks
Block 1
Block 2
Block 3
Block 4
Step 2: Prefix sum each block with 1 thread each
Block 1
Block 2
Block 3
Block 4
Step 3: Gather (spilled over) sums of each block
Sum 1
Sum 2
Sum 3
Sum 4
Step 4: Calculate prefix sum of "sums" array
PF1
PF2
PF3
PF4
Step 5: Add sums to each value in corresponding block
Block 1
Block 2
Block 3
Block 4
Step 6: Join all blocks to obtain prefix summed array
Prefix Sum Array
Legend: Darker color corresponds to higher numerical value

Figure 7.10: Divide prefix summation into blocks. Each block uses multiple threads to calculate the implicit prefix sum with up-sweep (see Fig. 7.8) and down-sweep (see Fig. 7.9). The final sum of each block is stored in an additional array called "sums". The "sums" array is then also prefix summed in parallel, which updates all array entries to the prefix sum (3rd row in the figure). This prefix sum is then added to each block by the threads which were assigned to each block. This finishes the prefix calculation. Each block utilizes multiple threads while the array is divided into many blocks. This results in massive parallelism, i.e. good utilization of the GPU with each thread occupying just a small memory footprint (as it operates on a small data block).

if in these two specific steps, the full number of threads might exceed the limits hardware architecture (resulting in GPU memory offloading to main memory), overall performance is still maximized as the parallelism in the other steps (which exponentially decreases) is as high as possible.

circuit setup is user input dependent. Therefore, it is impossible to make assumptions about the input data. The end-user can use SPICE to perform any circuit setup on any circuit with correct syntax. This results in three key issues: 1) The input circuit could be very small, leading to diminishing returns on our circuit setup optimized for parallelism and large circuits. Our solution has to create up to 1024 threads, initializing the prefix and radix algorithm by allocating local and global memory structures, etc., resulting in potentially large overhead. However, our evaluation shows that less than $1s$ overhead is present for small circuits compared to original NGSPICE, which is tolerable.

2) Our input data is not aligned to exact multitudes of either our block size or thread count, resulting in asymmetric load for our threads. Some blocks might have just a handful of values, which might lead to aliasing or corner cases in other algorithms. In practice, this is not an issue for our implementation, as blocks with less data simply finish earlier. Our approach does not require handling of specific edge cases for non-aligned data. The prefix algorithm simply calculates the sum for a smaller array and finishes earlier and Radix has slightly fewer values to sort. None of the employed algorithms require alignment to specific values (e.g., determined by the hardware architecture or the algorithmic structure). Additionally, algorithm runtimes are not affected, as this is governed by the execution time of the slowest threads. These threads remain the ones operating on regular filled blocks, leading the maximum operations per thread, which equals to the regular runtime. No filling with dummy data or corrections at the edges of data are necessary.

3) Input data can arrive in any order, including any given worst-case order. Sorting is frequently prolonged if an ascending list needs to be sorted descending or vice versa. However, the end-user can provide circuit topology in any order, i.e. any permutation of the lines in the input file results in a valid SPICE netlist, i.e. a valid input file.

However, Radix sorting does not exhibit a dependency on input data [170] [173]. Sorting an ascending data, a uniform distributed data or descending data to descending, requires the same amount of operations and thus results in the same runtime. Therefore, all permutations of the circuit netlist the end-user can input into SPICE results in the same runtime for the circuit setup. This is especially important, as SPICE is frequently used as a back-end tool, e.g. in library characterization tools like Synopsys SiliconSmart. Using SPICE as a back-end tool might result in valid but corner case netlists. For example, a flattened netlist devoid of any hierarchy with millions of similarly named nodes (e.g., node1, node2, etc.). These tools might also re-order these nodes in any way the other tool saw fit, resulting in particular orders (e.g., ascending or descending a logic path). Therefore, pre-sorted lists might be encountered frequently. Thus, being impartial to the order of input data is important for the circuit setup of SPICE.

Summary: Our Radix-based implementation is a novel use-case scenario for a sorting algorithm used for circuit topology analysis (circuit setup) in a circuit simulator. It enables us to pre-process requests, construct the circuit matrix and link requests to the circuit matrix with massive parallelism. This vastly outperforms the original NGSPICE circuit setup (see Section 7.1.5). Our Radix implementation also outperforms our first SQL implementation of the previous section, as sorting and matrix construction is parallelized utilizing thousands of cores in the GPU compared to a single CPU thread.

7.1.5 Evaluation

Experimental Setup: We run the experiments on a Desktop PC with a Core i5-4570 with 3.2GHZ and 8GB RAM. The graphic card is a NVIDIA GTX980 with 4GB GDDR3 RAM. The software NGSPICE-26 [95] is the baseline for our evaluation. The experimental CUSPICE [107] (a derivative of NGSPICE) is the basis of our work, but still in experimental development stage, thus currently not fully functional (as of Apr. 2018) and as such cannot be used to evaluate or compare circuits of this size (due to convergence issues). To represent modern circuit simulators (NGSPICE is quite dated), we use the Xyce 7.0.0 simulator [96] (parallel build using OpenMPI). Xyce features modern data structures and multi-processing to use the available cores (4 in our Core i5-4570), resulting in much higher performance than NGSPICE.

Circuits: To explore the performance of our circuit setup, we cannot use the standard ISCAS85 benchmark circuits [167]. These ISCAS circuits are too small and emphasize static overheads over scalability. Therefore, we used the EPFL circuits [166] and our own circuits[2] The EPFL circuits are meant to evaluate and challenge synthesis tools. For this purpose, EPFL circuits feature challenging topology and are significant in size. This makes them perfect as a stress test for our circuit setup. As the EPFL circuits feature irregular topology (to challenge synthesis), we additionally evaluate large standard circuits like multipliers, AES encryption and discrete cosine transform (DCT). All these circuits were synthesized from VHDL/Verilog hardware language using the Synopsys Design Compiler tool (using ultra compile setting for highest performance, i.e. shortest propagation delay). Synthesis used the Nangate 45nm standard cell library, which provides spice cell netlists.

Importance of Circuit Setup: The circuit setup in small circuits contributes only a minor part to the overall simulation time. However, in large circuits, this changes. Table 7.1 evaluates current commercial SPICE with the large SQRT circuit. The circuit setup phases are not the dominant phase, which is the LU factorization (which explains the heavy focus of research on that phase), but circuit setup is indeed comparable to other important phases like the matrix solve or operating point phase. In general, it is approximately 20% of overall simulation time and as such contributes considerably to the overall execution time of the circuit simulation. *Circuit setup is now non-negligible in large circuits and should be optimized to reduce overall simulation time.*

[2] The EPFL multiplier has an entirely different implementation and topology (e.g., approximately 3x transistors) than our regularly synthesized multiplier, as it was designed as a synthesis challenge [166]. Hence, we report values for both 64-bit Multipliers.

Table 7.1: Relative Execution Time of SPICE Phases in Spectre

	EPFL Circuits [166]			
Phase	Divisor	Multiplier	Sqrt	Square
Circuit Setup	24%	19%	20%	17%
Operating Point	18%	17%	21%	20%
Matrix Lu Factorization	33%	35%	38%	38%
Matrix Solve	18%	16%	19%	16%
Others	7%	13%	2%	9%

Generic GPU-SPICE Implementation: Our implementation in SPICE makes no assumptions about circuit topology or circuit elements. All results (final or intermediate, e.g. the resulting circuit matrix) are a perfect match to original NGSPICE. Therefore, every circuit that can be simulated in NGSPICE can be solved in GPU-SPICE. Both sequential and combinatorial (large or small) circuits synthesized in different technologies are possible. Our resulting voltages are almost indistinguishable ($< 0.1\%$) from the commercial HSPICE results with minor differences resulting from the differences in the numerical settings/algorithms (e.g. the matrix solver or gmin values).

Table 7.2: Evaluation of Circuit Setup Time in Different Simulators

Circuit Name	Circuit Size					Circuit Performance			
	General			Non-Zero Matrix Elements		Circuit Setup Time [s]			
Our Circuits	Transistors	Requests	Matrix Cols	Total	Max in Column	NGSPICE [95]	Xyce [96]	SQL	Radix
MAC 64-bit	87,096	5,574,920	392,320	3,340,543	185,364	1,484.74	9.67	22.72	3.24
AES	87,235	5,584,084	393,616	3,293,525	189,412	1,569.00	10.79	23.83	3.32
DCT	353,184	22,603,924	1,605,066	13,485,834	793,179	47,211.88	38.53	95.60	13.12
Multiplier 64-bit[3]	42,534	2,722,696	191,663	1,630,990	90,006	244.44	4.71	11.03	1.64
Multiplier 128-bit	236,639	15,145,928	1,064,524	9,021,114	523,191	16,876.14	26.79	62.44	8.81
Multiplier 256-bit	1,051,970	58,911,350	4,734,895	39,954,126	2,240,289	394,093.62	123.80	3,634.23	89.63
EPFL Benchmarks [166]	Transistors	Requests	Matrix Cols	Total	Max in Column	NGSPICE [95]	Xyce [96]	SQL	Radix
Divisor 64-bit	371,631	20,811,597	1,670,657	12,421,637	1,203,341	132,032.71	43.00	183.10	21.10
Multiplier 64-bit[3]	141,616	7,930,757	636,977	4,794,077	269,461	1,828.47	16.31	60.91	7.37
Sqrt 64-bit	255,361	15,458,954	1,082,543	9,195,146	554,498	20,173.21	25.15	81.37	10.22
Square 64-bit	85,106	4,766,125	382,603	2,866,549	162,937	1,003.89	9.78	48.19	5.37

Circuit Properties: Table 7.2 summarizes our evaluated circuits. The circuits range from 42 thousand (k) to >1 million (M) transistors with approximately 70 times the number of requests. This highlights the importance of high-performance circuit setup, as the circuit setup scales with the number of requests and not number of transistors. Matrix Cols describes the number of columns of the circuit matrix, which represents dimension of the matrix (i.e. n for the nxn matrices). As the circuit matrices are sparse, we report the number of non-zero elements and the maximum number of non-zero elements in a single column (typically the column representing either GND or VDD). These circuit properties affect runtime. For example, the 15M requests of the 128-bit multiplier result in 9M non-zero elements, hence 6M requests were removed as duplicates. This highlights how even simple operations like deduplication demands parallel execution, as millions of elements are processed.

Original NGSPICE Circuit Setup: For the original circuit setup we evaluate all circuits in Fig. 7.11. Note the logarithmic scale on the y-axis. For the smallest ISCAS benchmarks setup is below a single second and even for the largest benchmarks it does not exceed 100s (1.5 minutes). This highlights how for such small circuits, previously the circuit setup phase in SPICE could not be identified as a performance bottleneck. The circuit setup phase is negligible for circuits that small. However, in EPFL benchmarks and our circuits we range from minutes (e.g., 26 minutes for AES) to hours (e.g., 13 hours for DCT) to days (e.g., 4.5 days for 256 multiplier). This illustrates, how for larger circuits, the circuit setup can consume considerable amount of time and should be accelerated.

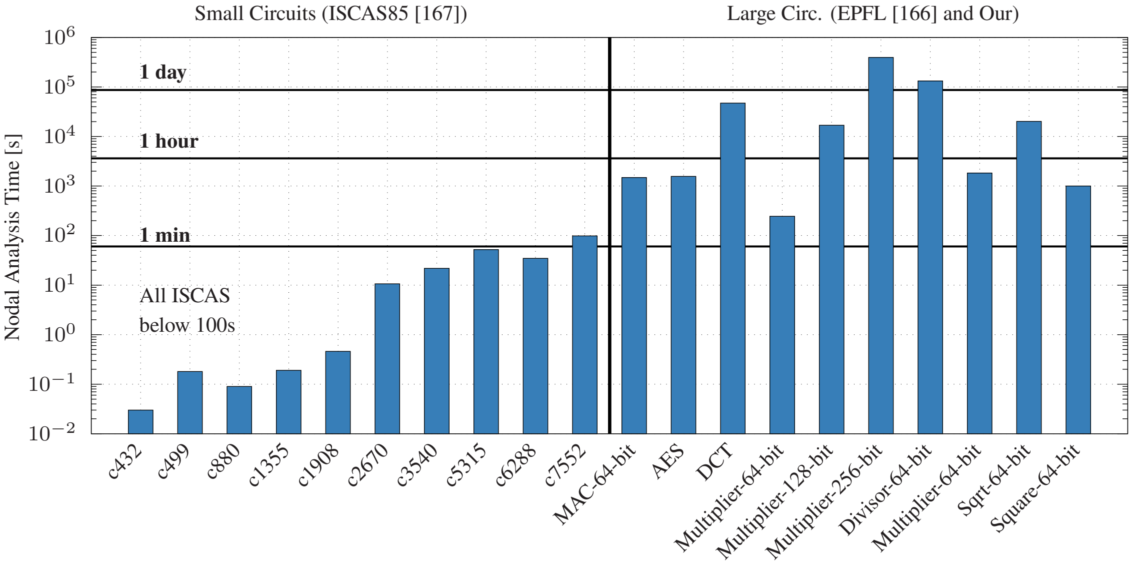


Figure 7.11: Setup time for the ISCAS85 [167], EPFL [166] and our circuits in NGSPICE [95]. Smaller circuits exhibit setup times below $100s$. Yet, larger circuits show how circuit setup exceeds hours (128-bit multiplier, DCT) to days (256-bit multiplier) and thus consumes considerable time.

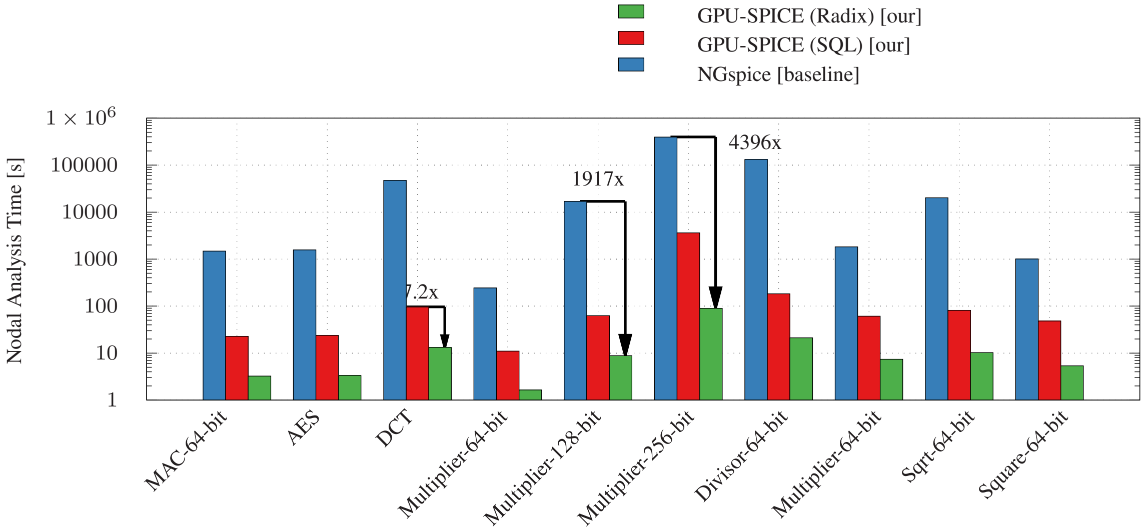


Figure 7.12: Evaluation of circuit setup time for our synthesized circuits. Radix is 7x to 40x faster than SQL, which in turn is significantly faster than our baseline NGSPICE. For the 128-bit multiplier, circuit setup dropped from 5.5 hours to $62s$ for our SQL implementation and $8.8s$ for our Radix implementation. For larger circuits like 256 multiplier and divisor, the speedup is even higher with 4396x compared to NGspice and 40x to our SQL.

Our Circuit Setup: Our two circuit setup implementations (Radix and SQL) are evaluated with all circuits. The ISCAS benchmarks are not shown as the Radix circuit setup is faster than $1s$. For example, the largest ISCAS circuit c7552 with 14942 transistors executes circuit setup for $9.8s$ in NGSPICE and drops to $3.3s$ in SQL and $0.7s$ in Radix. The large circuit results are shown in Fig. 7.12. Our implementations reduce the time necessary for the circuit setup for the entire range of large circuits. With encryption (AES), image processing (DCT) and arithmetic (MAC) circuits we covered a wide range of combinatorial circuits in which Radix vastly outperforms NGSPICE and SQL. To observe the scaling, we use the multipliers, as they feature similar circuit topology and only differ in size. For the smaller 64-bit multiplier the analysis drops from 4 minutes in NGSPICE to $11s$ SQL and below $2s$ Radix. In the middle, the 128-bit multiplier sees a drop in circuit setup time from $19878s$ NGSPICE (5.5 hours) to $62s$ SQL (1 minute) and $8.8s$ Radix, providing 318x respectively 2257x faster circuit setup. For the huge 256-bit multiplier, analysis drops from $394093s$ in NGSPICE (4.5days) to $3634s$ in SQL (1 hour) and $89s$ in Radix (1.5 minutes). This illustrates why the SQL implementation was insufficient, as its speedup dropped to 108x as the database could not fit in main memory anymore. At the same time our Radix implementation is 40x faster than SQL for this huge circuit, resulting in an overall speedup of 4396x. The Xyce simulator scales much better than NGSPICE, since it uses multiple cores and modern data structures. For the 128-bit multiplier, Xyce is faster than commercial simulators (see below) with just 1.38x the runtime of Radix. For smaller circuits like the AES, DCT, MAC and EPFL circuits it is on average 3x, respectively 2x (EPFL), slower than Radix. In summary, Xyce outperforms our SQL implementation as well as the commercial simulators in the circuit setup phase. However, our GPU-based Radix approach outperforms all examined simulators including the Xyce simulator.

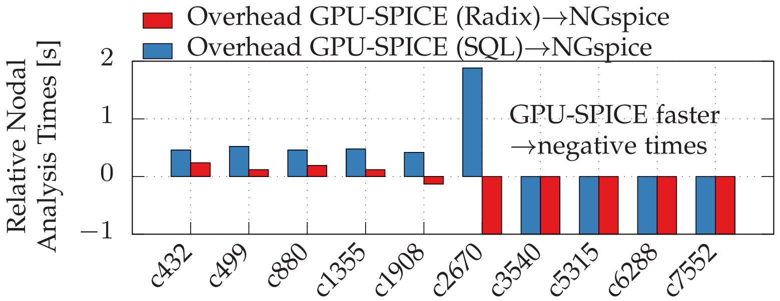


Figure 7.13: Comparing circuit setup times in small circuits to illustrate the overhead $t_{overhead} = t_{analysis}(our) - t_{analysis}(NGSPICE)$ of our approach (e.g., by copying data to and from the GPU). As shown, both SQL and Radix are below $< 2s$ slower for even the smallest circuits and thus feature negligible overhead.

Overhead: To evaluate the overhead, we return to the ISCAS circuits in Fig. 7.13. Their smaller size makes them ideal to highlight static overhead in the circuit setup. Fig. 7.13 illustrates that our overhead ($t_{overhead} = t_{analysis}(our) - t_{analysis}(NGSPICE)$ for both implementations is below $1s$ and thus negligible.

Commercial SPICE: Synopsys HSPICE and Cadence Spectre are the leading commercial SPICE flavors. We evaluated circuit setup (called setup in HSPICE and Spectre) for the 64-bit multiplier in HSPICE 2017.3 and Spectre APS 15.4 both configured to 8 threads (i.e. multi-core circuit setup), which resulted in the fastest setup times on our hardware. Note that Fig. 7.1 already establishes that circuit setup consumes a significant portion of the total execution time in commercial SPICE flavors. We could not solely run circuit setup, but had to evaluate DC or TRAN simulations to comply with the software. Consequently, we could not evaluate larger circuits than the 64-bit multiplier in HSPICE and Spectre, since the entire simulation consumes unfeasible amounts of time. In the examined 64-bit multiplier, our Radix circuit setup is 4.49x faster than Spectre and 2.32x faster than HSPICE, confirming that we also outperform commercial software. Note, that this evaluates circuit setup and not overall simulation time.

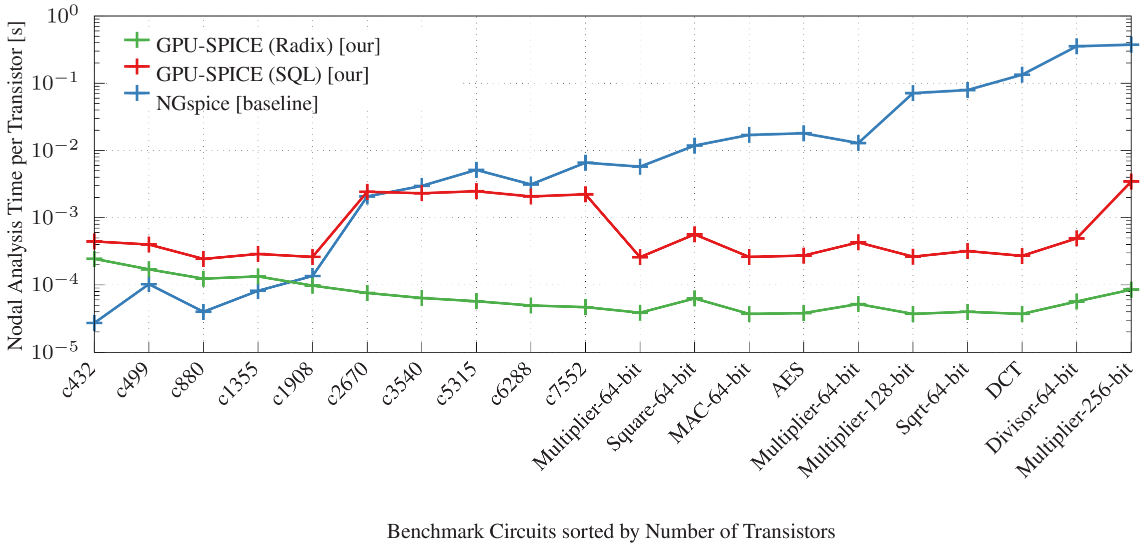


Figure 7.14: Normalizing circuit setup times shows how the algorithms scale. If the algorithm scales well, than the amount per transistor should stay constant (approx. linear $\mathcal{O}(n)$) instead of growing linearly ($\mathcal{O}(n^2)$). NGSPICE has less initialization overhead, but scales badly. Our Radix circuit setup implementation scales excellent with near constant circuit setup time per transistor.

Circuit Setup Time per Transistor: In order to evaluate the scaling behavior of our two approaches and state of the art, we evaluate the circuit setup time, normalized per transistor in Fig. 7.14. The total circuit setup time (shown in Fig. 7.11 is divided by the number of transistors in the simulated circuits (see Table 7.2 for large circuits). Fig. 7.14 shows how the original implementation (NGSPICE) increases 4 orders of magnitude in circuit setup time per transistor towards larger circuits. This clearly highlights, why for larger circuits a new approach was necessary.

Our SQL circuit setup implementation in GPU-SPICE shows better scaling. Beyond c2670 with 5132 transistors, SQL utilizes less computational time per transistor as NGSPICE. Before that point, both Radix and SQL are worse due to the initialization of the GPU and SQL database. The hump from c2670 to c7552 in SQL is mainly due to the specific topology of the larger ISCAS benchmarks (multiple copies of smaller ISCAS benchmarks), which results in slightly longer lookups in the SQL database. While the SQL solution features near-linear scaling $\mathcal{O}(n)$, it has its limits towards large circuits. The SQL database for the Divisor-64-bit and Multiplier-256-bit did not fit into the main memory of the employed desktop PC and therefore the operating system had to page (offload memory to the hard disk). This results in much more computational time per transistor, as visible by the incline at these two large circuits.

The Radix implementation in GPU-SPICE uses less memory (4GB on the GPU vs. 8GB main memory) and therefore scales better towards the larger circuits. It features a slight incline towards Divisor-64-bit and Multiplier-256-bit as the bandwidth from GPU to GPU-memory becomes a bottleneck. However, Radix is at least one order of magnitude better than our SQL implementation and scales excellently. For small circuits, the initialization, i.e. moving data from main memory to the GPU, thread management, etc.) consumes time. Beyond c1908 with 3374 transistors, Radix consumes less time per transistor (and overall, see Fig. 7.13) than state of the art NGSPICE. Our experimental results verify the algorithmic complexity of our SQL and Radix approaches as $\mathcal{O}(n)$.

Total Simulation Times: We did not compare total simulation times, as this is outside of the scope of this work and heavily depends on the settings used. A transient simulation for 10k cycles of a digital circuit takes significantly longer than a transient simulation with a single cycle used for propagation delay estimation. Circuit setup is independent of settings and thus fair to compare. All simulations finished, i.e. we could verify that circuit setup worked fine.

8 Custom Reliability in Circuits

With the previous four steps complete, the degradation models are improved, accelerated and incorporated into standard tools. Now the final goal can be achieved, which is to estimate the reliability of circuits under custom conditions. For a user specified workload, temperature, voltage and lifetime, the tool can accurately determine the required timing guardband to maintain reliability. Compared to traditional worst-case estimations, this improves the performance of the circuit without any loss in reliability.

8.1 Circuit Reliability Analysis for Custom Use-Case Scenarios

To provide an interesting use-case for our custom reliability estimation, we study an entire SRAM memory array which is used as a register file in a microprocessor. For this purpose, this section does not just discuss the custom reliability framework by itself (see Section 8.1.3.3), but also the surrounding infrastructure (see Section 8.1.3.4) to obtain all the data necessary for a use-case (e.g., activity in the circuit). This section is based on my publication [174].

8.1.1 SRAM Array - The Typical Test Circuit

SRAM memory is the fastest memory type, used in the most demanding memory applications such as buffers, register files and L1 cache, operating at full f_{clk}. Thus, evaluating its reliability (and therefore guardband, affecting effective performance) is important and frequently used to benchmark a technology [86, 175]. However, studying SRAM memory cells in isolation misrepresents their importance for SRAM array performance.

In this work, we show how SRAM memory cells (C) represent just a small fraction of the overall performance (e.g., read or write delay) of an SRAM memory array. Instead, we highlight how periphery like the Write Driver (WD) and the Sense Amplifier (SA) govern performance. Other periphery, like address decoders, are latched and thus do not affect array delay. Hence, they are not considered in this work, even though they could be included.

In this work, we present a fully automated SRAM framework, which is built on top of our own Circuit Aging Reliability Analysis Tool (CARAT). This SRAM framework can create an SRAM array with various settings (size of memory array, etc.), extract activity for the array from a processor simulator and then estimate aging-induced degradation in the array to identify weaknesses in the circuit design with respect to reliability. Our high level flow is shown in Fig. 8.1.

8.1.1.1 Related Work

Studying reliability can be performed at different abstraction levels and with different accuracies. At the higher abstraction level, reliability is studied of large circuits (exceeding 100k transistors), which is solely possible by breaking the circuits down to standard cells, but not down to the transistor level. Works like [82, 84, 85] have shown how large circuits like entire microprocessors can be studied by characterizing standard cell libraries under the effects of aging.

This works aims at lower abstraction levels (up to 100k transistors), as circuits are broken down to the transistor level. This allows us to study analogue and mixed-signal circuits, as these cannot be broken down to standard cells.

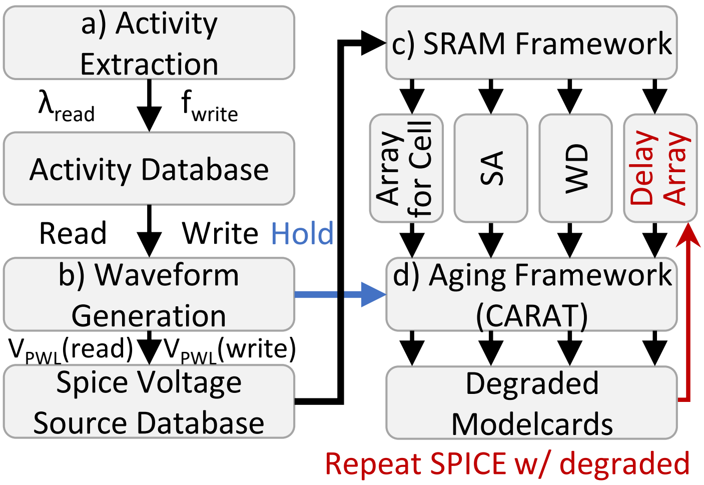


Figure 8.1: Overview of entire work. Delay array is simulated with fresh and degraded modelcards to obtain delay difference. Indices a) to d) correspond to Fig. 8.3.

Additionally, circuit simulation on the transistor level feature higher accuracy, as more of the transistor interactions (for example, pull-up versus pull-down networks as shown in [85]) are captured and circuits are not evaluated with abstracted delay and power tables for standard cells. Therefore, instead of taking the workload purely as signal probabilities of standard cell input pins [82, 85], we take individual voltage waveforms per transistor into account. So instead of duty cycle and switching frequency, we have full V_G, V_D, V_S, V_B waveforms which allow the aging models to more accurately consider recovery and thus provides a more accurate degradation value per transistor.

As a representative circuit, we study an SRAM array as SRAM are frequently studied in reliability [86]. However, the majority studies the SRAM memory cells in isolation missing the periphery with SA or WD [73, 86]. Reliability in periphery is mainly reported with isolated SA studies in [87, 88] and a single BTI report in WD with cells in [89]. For the SA, the work in [88] claims workloads from a processor simulator, yet solely records read frequency ("read activation") (f_{read}) and then simulates simple read 0 and read 1 patterns in SPICE to translate read operations to transistor duty cycle (λ_{tran}) and switching frequency f_{sw} *for transistors in the SA alone*.

For a detailed comparison to other circuit reliability frameworks, see Section 3.5.

8.1.1.2 Novel Contributions

The novel contributions of this work are as follows:

1. We simulate an entire SRAM 32 x 64 bit array including SA, WD, pre-chargers under the impact of realistic workloads from a processor, bringing SA, WD and C together for the first time. This reveals the vulnerable transistors across the entire SRAM array.
2. Our workload and activity stretches far beyond solely f_{read}. We monitor a large set of activities (see Fig. 8.3a) for each circuit. This work monitors an entire processor register file (2048 SRAM bits) including 64 columns for SA, WD data. This allows for the first time to uncover the interaction between C, SA and WD during all three read, write and hold operations.
3. We map the full activity on each transistor in the array and simulate everything in SPICE. Then, we integrate this in our own aging framework (CARAT) to study each transistor with its full voltage waveform both for detailed BTI and HCD analysis. The automated tool is implemented with C-code for performance-critical sections for a simulation time of under 2h, illustrating feasibility of large circuit studies.

8.1.2 Background

To explain our reliability framework, we first need to provide some background in SRAM and aging.

8.1.2.1 SRAM Periphery

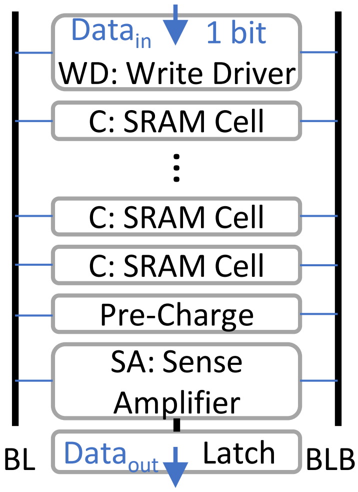


Figure 8.2: Block schematic of an array column with periphery

The periphery of an SRAM array is important with respect to the overall performance of the array. During read, an SRAM cell barely pulls its bit line (BL) down by 50-100mV, before the SA supports the read and the latch stabilizes the SA output. Similarly, strong WDs are necessary to flip a cell to its opposite value for reliable writes, as the SRAM cell itself opposes the write operation (if the opposite value (to the stored value) is written). Actually, the periphery determines the majority of t_{delay}, compared to the contributions of the cells alone, as we illustrate in Fig. 8.6.

8.1.2.2 Transistor Aging

Bias Temperature Instability (BTI) is stimulated by temperature and voltage, specifically V_{gs} and V_{ds} [9]. These voltage waveforms are frequently abstracted to by On-/Off-ratio of transistors λ_{tran} and switching frequencies f_{switch} [118]. For more details about BTI see Section 2.2.1. With respect to our SRAM array, its transistors in SA and WD, λ_{tran} depend on the balance ratios (more operations with either 0's or 1's) of read (λ_{read}) and write (λ_{write}) operations.

For Hot Carrier Degradation (HCD) stimulation is also by temperature and voltage, with V_{gs} and even stronger dependency (compared to BTI) on V_{ds}. For more details on HCD see Section 2.2.2. HCD activation is given by $f_{read.SA}$ and $f_{write.WD}$ (different from (lower) cell frequencies $f_{read.C}$ and $f_{write.C}$). The WD and C see high current when bit lines discharge and C has to be flipped (writing opposite values), hence $f_{flip.C}$ contributes to the total flip frequency of the WD $f_{flip.WD}$, which are both important for HCD in C and WD, respectively. This is not considered in state of the art, as the additional load of periphery on the SRAM C can only be considered if periphery like SA, WD, etc. are considered.

8.1.3 Circuit Reliability Framework

Our circuit reliability framework consists of four parts, which are explained in their corresponding subsections. A high level overview is given in Fig. 8.1.

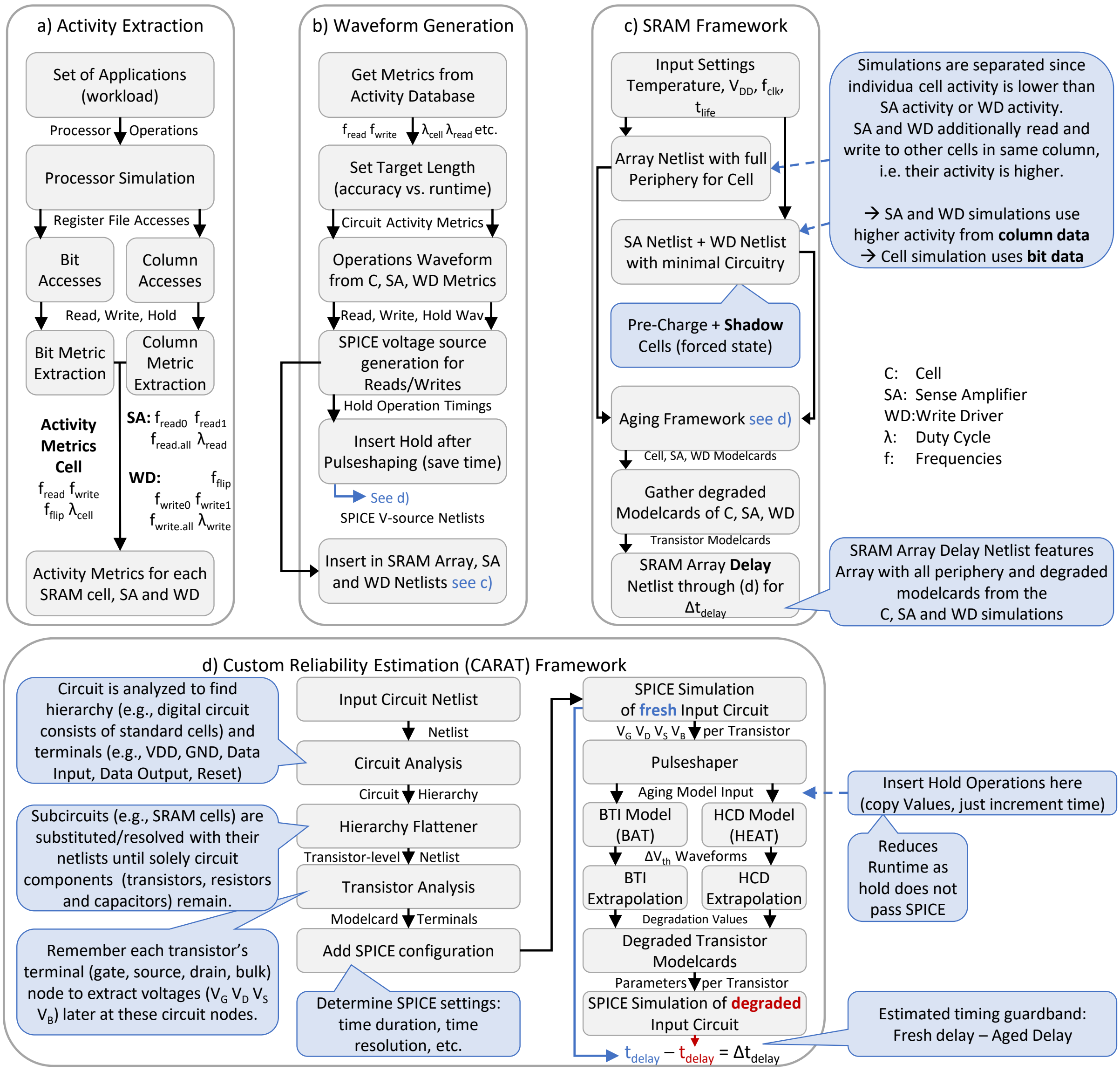

a) Activity Extraction
Set of Applications (workload)
Processor Operations
Processor Simulation
Register File Accesses
Bit Accesses
Column Accesses
Read, Write, Hold
Bit Metric Extraction
Column Metric Extraction
Activity Metrics Cell
f_{read} f_{write} f_{flip} λ_{cell}
SA: f_{read0} f_{read1} $f_{read.all}$ λ_{read}
WD: f_{flip} f_{write0} f_{write1} $f_{write.all}$ λ_{write}
Activity Metrics for each SRAM cell, SA and WD
b) Waveform Generation
Get Metrics from Activity Database
f_{read} f_{write} λ_{cell} λ_{read} etc.
Set Target Length (accuracy vs. runtime)
Circuit Activity Metrics
Operations Waveform from C, SA, WD Metrics
Read, Write, Hold Wav
SPICE voltage source generation for Reads/Writes
Hold Operation Timings
Insert Hold after Pulseshaping (save time)
See d)
SPICE V-source Netlists
Insert in SRAM Array, SA and WD Netlists see c)
c) SRAM Framework
Input Settings Temperature, V_{DD}, f_{clk}, t_{life}
Array Netlist with full Periphery for Cell
SA Netlist + WD Netlist with minimal Circuitry
Pre-Charge + Shadow Cells (forced state)
Aging Framework see d)
Cell, SA, WD Modelcards
Gather degraded Modelcards of C, SA, WD
Transistor Modelcards
SRAM Array Delay Netlist through (d) for Δt_{delay}
Simulations are separated since individua cell activity is lower than SA activity or WD activity. SA and WD additionally read and write to other cells in same column, i.e. their activity is higher.
→ SA and WD simulations use higher activity from column data
→ Cell simulation uses bit data
C: Cell
SA: Sense Amplifier
WD:Write Driver
λ: Duty Cycle
f: Frequencies
SRAM Array Delay Netlist features Array with all periphery and degraded modelcards from the C, SA and WD simulations
d) Custom Reliability Estimation (CARAT) Framework
Circuit is analyzed to find hierarchy (e.g., digital circuit consists of standard cells) and terminals (e.g., VDD, GND, Data Input, Data Output, Reset)
Subcircuits (e.g., SRAM cells) are substituted/resolved with their netlists until solely circuit components (transistors, resistors and capacitors) remain.
Remember each transistor's terminal (gate, source, drain, bulk) node to extract voltages (V_G V_D V_S V_B) later at these circuit nodes.
Input Circuit Netlist
Netlist
Circuit Analysis
Circuit Hierarchy
Hierarchy Flattener
Transistor-level Netlist
Transistor Analysis
Modelcard Terminals
Add SPICE configuration
Determine SPICE settings: time duration, time resolution, etc.
SPICE Simulation of fresh Input Circuit
V_G V_D V_S V_B per Transistor
Pulseshaper
Aging Model Input
BTI Model (BAT)
HCD Model (HEAT)
ΔV_{th} Waveforms
BTI Extrapolation
HCD Extrapolation
Degradation Values
Degraded Transistor Modelcards
Parameters per Transistor
SPICE Simulation of degraded Input Circuit
$t_{delay} - t_{delay} = \Delta t_{delay}$
Insert Hold Operations here (copy Values, just increment time)
Reduces Runtime as hold does not pass SPICE
Estimated timing guardband: Fresh delay – Aged Delay

Figure 8.3: a) Processor activity extraction, which is stored as activity metrics for each circuit.
b) Metrics translated to operation waveforms (Read, Write, Hold over time), then into SPICE voltage sources.
c) C, SA & WD simulations to extract values for degraded modelcards (see Fig. 8.1) with their activity (bit for C, col. for SA/WD) then delay simulation (fresh vs. degraded)
d) Circuit Reliability Framework (CARAT)

8.1.3.1 Activity Extraction

The simulator gem5 [148] simulates a complete processing system, including main memory and the ARM CPU, clocked at 1 GHz. This allows us to extract a representative activity trace of even the innermost memory of the CPU, the register file. Gem5 was extended to report the current clock cycle and values of all read and write accesses to the register file. This way, all activities in the 64 bits (SRAM cells) of the 32 registers can be analyzed. The access traces of various workloads (benchmark applications) are extracted and we deduce data for each bit and column (see Fig. 8.3a for metrics). Importantly, the activity of SA or WD is different from C, since single SA or WD support an entire column (see Fig. 8.2). Each read (write) to any C in that column passes the same SA (WD) and thus degrades that SA or WD, while activity distributes across Cs: $f_{read.SA} \gg f_{read.cell}$ and $f_{write.WD} \gg f_{write.cell}$.

8.1.3.2 Waveform Generation

SPICE requires voltage sources for the pins of our netlists (SA enable, WL, CLK, etc.). For this purpose, we generate read and write waveforms matching λ_{read}, λ_{write}, $f_{flip.WD}$, etc. (see Fig. 8.3b). Hold operations are introduced to bring f_{read} and f_{write} from f_{clk} down to requested $f_{read.SA}$, $f_{write.WD}$ by altering the input to the BTI/HCD models. Thus, instead of simulating $> 1e8$ operations doing nothing, we insert hold operations without loss of accuracy (since nothing changes during hold) to reduce SPICE runtime.

8.1.3.3 CARAT Aging Framework

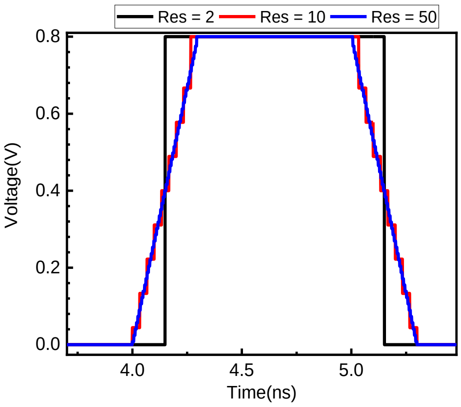


Figure 8.4: Shaping a rising and falling voltage waveform with the three different resolution settings. At resolution = 2, the shape is only roughly approximated, while at resolution = 50 the shape is approximated at high accuracy. The resolution is thus a trade-off between simulation accuracy and simulation time.

The aging framework starts with a SPICE circuit simulation of the circuit (see next Section for details about the netlists). The circuit consists of 14nm FinFET transistors with modelcards from [162]. The activity of the circuit is given by the description of the voltage sources at the input pins of the circuit (see previous section). In our case, this is done by the waveform generator (Fig. 8.3b). It creates all data and control signals (e.g., WL, CLK, SA enable, WD enable) for our SRAM array.

From SPICE, V_D, V_G, V_S, V_B and temperature T over time for each transistor are extracted, which is later used by our aging models to estimate aging-induced degradation. These five entities (four voltages + temperature) are each saved into a single file per transistor as waveforms over time.

Then, a pulseshaper creates valid inputs for the BTI model (BAT [9]) and the HCD model (HEAT [42]). As the aging models are computationally intensive, the pulseshaper removes glitches (i.e. smooths the signal slightly to remove over- and undershooting of the voltages) and discretizes the waveforms with a given resolution (see Fig. 8.4). This resolution factor is between 1 and 100, with 100 being the full SPICE waveform and lower values representing abstracted versions of the waveforms to further enhance the simulation speed. The actual resolution value is up to the end-user to balance accuracy against simulation time. In this work, resolution is set to 100. If full accuracy is required for large circuits (e.g., in security critical applications) and thus resolution is forced to 100, then GPU-accelerated aging models can be employed to feasibly simulate 100k transistors despite using defect-centric aging models [152].

As the SPICE simulation can only cover a limited amount of operations and not the entire lifetime of a circuit (10 years of operation at billions of operations per second is infeasible), we have to extrapolate the aging degradation. For this purpose, both BTI and HCD output from the corresponding models [9] [42] are extrapolated by matching constant stress (DC) curves with the actual calculated waveform-induced degradation (AC stress). For BTI, we shift the DC degradation curve down (towards lower ΔV_{th}) to match the currently computed AC stress curve (see Fig. 8.5a) and to then continue the AC stress curve. This exploits the universal power law of BTI [9] [34]. For HCD we

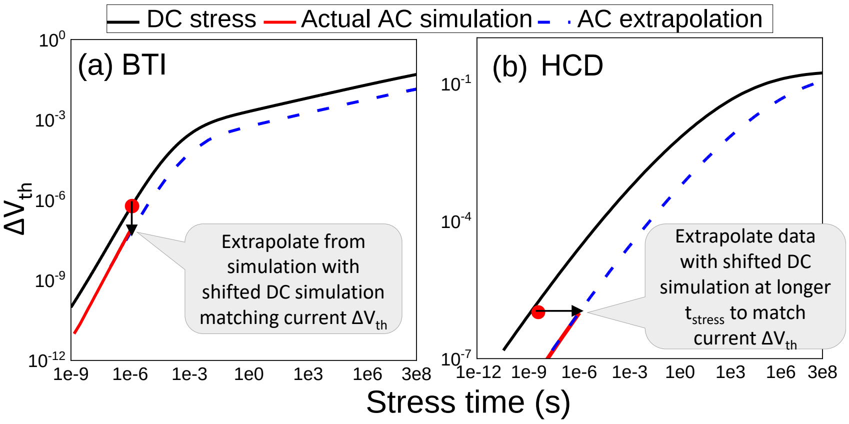


Figure 8.5: Extrapolation of BTI and HCD aging waveforms. a) BTI waveforms from DC (i.e. constant stress) simulation data are vertically shifted down (lower degradation values, but same slope) to match the actual simulation data (red curve) and continue it (blue dashed line). This exploits the universal power law of BTI [9]. b) HCD waveforms are horizontally shifted (to longer stress times) as HCD features no recovery and as such its degradation under AC stress can always be recreated with slower/shifted DC stress curves.

shift the DC degradation curve towards longer t_{stress} (summation of ON-time for a transistor). Since HCD features no recovery [42], we can model AC degradation as a slower DC degradation (e.g. 50% ON-time resolution in half degradation speed, i.e. twice the stress time) as shown in Fig. 8.5b.

Finally, the framework creates the degraded modelcards for each transistor (shifting V_{th} and other parameters based on the output of the aging models). Then another SPICE simulation (with degraded transistors) is performed to report the degraded delay. The degraded delay is then compared to the initial delay from the waveform extraction simulation (i.e. undegraded/fresh delay) and Δt_{delay} is determined.

8.1.3.4 SRAM Framework

The SRAM framework (see Fig.8.3c) creates four netlists, to minimize the amount of simulated transistors as much as possible without loss of any accuracy (as each simulation still contains all components affecting the circuit under test):

1. SRAM array with single C, SA and WD. Simulated with activity of single SRAM cell (1 bit).
2. SA with Pre-Charge reading from a shadow SRAM cell (cell with nodes Q and QB set via high impedance voltage source) to read the desired values. Netlist is simulated with the read activity of an entire column of the SRAM array (representative of all activity for a single SA).
3. WD with Pre-Charge writing to a shadow SRAM cell set to the correct values to write while flipping the cell (additional load on the WD) or without flip. Netlist is simulated with the writes of an entire SRAM array column (representative of all activity for a single WD).
4. Full SRAM array with multiple cells (for parasitics) to obtain delay at various points in the circuit with and without degradations.

Each netlist is then passed through the aging framework to obtain the degraded modelcards for each transistor under the correct activities (see Fig. 8.1)

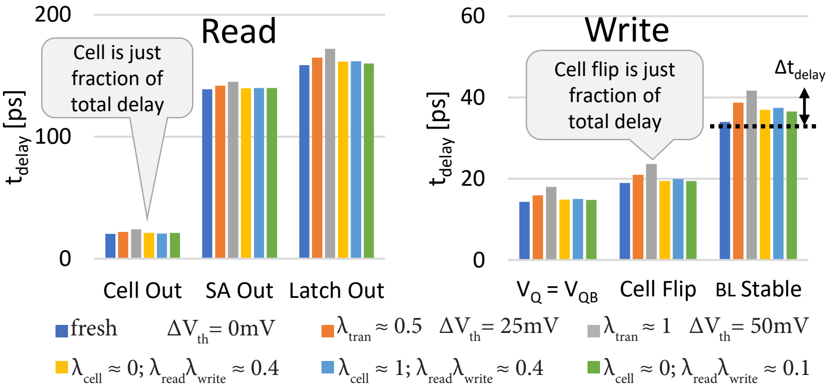


Figure 8.6: a) Read delay measured from SRAM cell out (BL, BLB), sense amplifier output and latch output. b) Write delay until internal nodes (Q, QB) cross their voltages, until the cell flips and until bit line (BL) stabilizes.

8.1.4 Evaluation

For the evaluation, we compare static degradation (0, 25 and 50 mV ΔV_{th} applied uniformly to all transistors in the SRAM array) to three actually workload-driven degradation scenarios in which we applied individual ΔV_{th} to each transistor based on its voltage waveforms.

8.1.4.1 Static Degradation

Fig. 8.6 show $t_{delay.read}$ and $t_{delay.write}$ for different points in the SRAM array. The dark blue is the fresh circuit without degradation. Orange and grey apply static uniform degradations of $\Delta V_{th} = 25mV$ and $50mV$ (corresponding to $\lambda_{tran} \approx 0.5$ and ≈ 1) to each transistor within the entire array. These three scenarios are our baseline scenarios without taking workload into account. Note, that static 50 mV ΔV_{th} is comparable to a worst-case estimation (SS process corner) for the SRAM array.

8.1.4.2 Workload-Induced Degradation

For the workload-induced degradations, we selected three bits (memory cells) from the SRAM array for the cell activity and the corresponding column activity for the SA and WD activity. We selected two comparisons (red and blue arrows in legend of Fig. 8.7). First, the impact of different λ_{cell} for identical λ_{read}, λ_{write} (two bits in the same column, have identical periphery, but different λ_{cell}). The second comparison is the impact of different λ_{read}, λ_{write} for similar cells (comparable λ_{cell}) to evaluate the role of the aging in periphery on the overall delay. Fig. 8.7 shows $\Delta t_{delay.read}$ and $\Delta t_{delay.write}$ due to BTI and HCD in the array, this highlights the differences already seen in Fig. 8.6 clearer as it isolates the impact of aging.

All cells are heavily unbalanced ($\lambda_{cell} \approx 0$ for yellow and green, with ≈ 1 for blue, which results in $\lambda_{tran} \approx 1$ for one nMOS and pMOS in C and $\lambda_{tran} \approx 0$ (i.e. fresh) for the other two. In Fig. 8.7a, we can observe that reading a stored 1 and then measuring the delay at the cell (“cell 1”), at the SA (“sense 1”) and finally, at the output of the latch (“latch 1”) does not show the same increase. For “cell 1”, yellow is higher, while for “sense 1” and “latch 1” blue is higher. This indicates that when studying the cell without periphery, one must conclude that λ_{cell} plays a major roll, while in reality, aging in SA and WD matters. Fig. 8.7a furthermore shows that for “cell 0” and “cell 1”, the cells behave as expected, performing slower when the degraded side (due to unbalanced aging $\lambda_{cell} \approx 0$ (yellow), ≈ 1 (blue)) has to charge the bit lines. Yet, for “sense 0” and “latch 0” compared with “sense 1” and “latch 1”, this disappears and the similarity in SA and WD degradation dominates overall t_{delay} (λ_{read}, λ_{write} is identical for yellow and blue). This highlights clearly, the importance of periphery when designing for reliability.

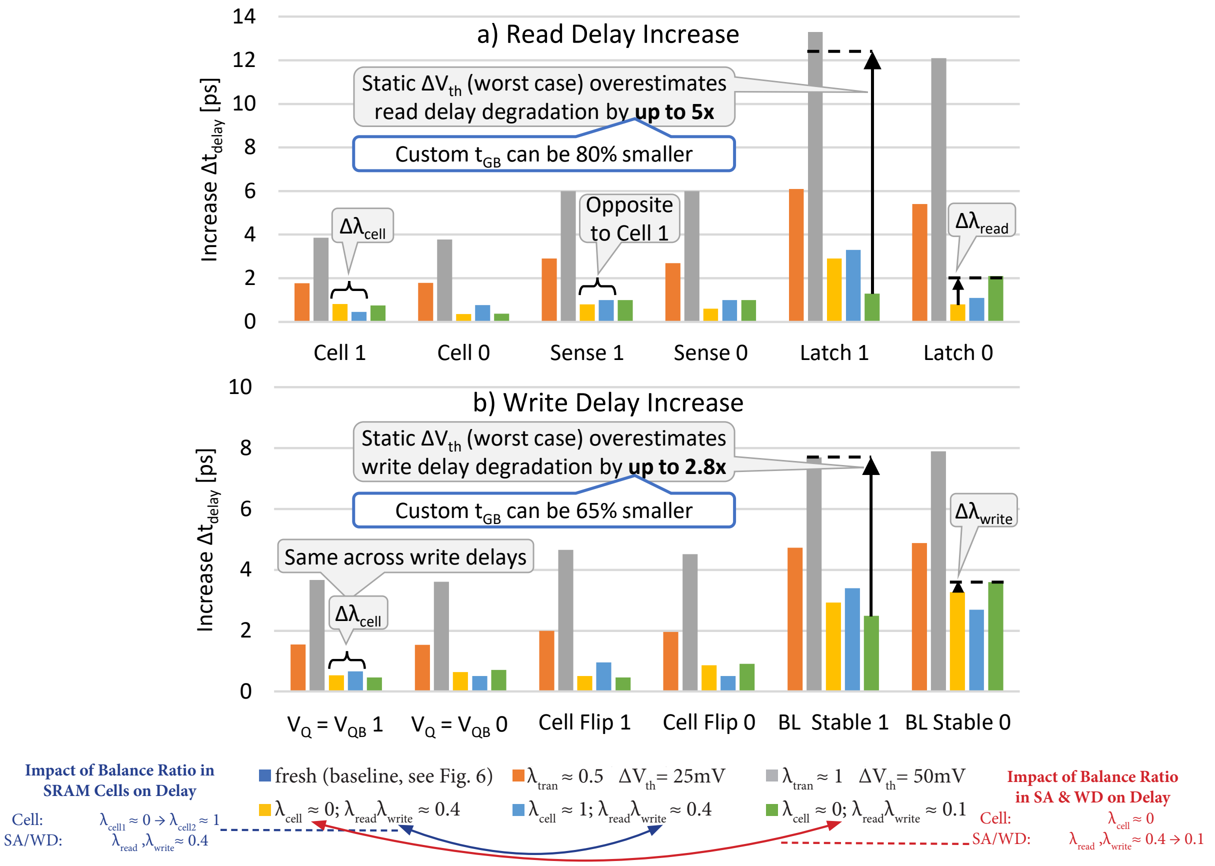


Figure 8.7: a) Read delay increase for static degradation (25 and 50mV in each transistor), different cell balance ratios and periphery balance ratios. Higher λ_{cell} (blue) increases t_{delay} over (yellow), while higher λ_{read} and λ_{write} (yellow) results in higher t_{delay} than (green) for read 1 but not read 0
b) Write delay follows the trends observed in read delay. Yet, overestimation due to static V_{th} is much smaller compared to read delay.

When taking periphery into account, it is important, that we do not use static degradations, but actual workload-driven degradations. Considering homogeneous ΔV_{th} for all transistors significantly overestimates both $\Delta t_{delay.read}$ (see "latch 1" in Fig. 8.7a) and $\Delta t_{delay.write}$ (see "BL Stable 1" in Fig. 8.7b), highlighting the importance of taking actual processor activity into account. Simple estimations are severely pessimistic leading to over-designed timing guardbands and thus ultimately a loss in performance (lower f_{clk}).

To study how much the difference in workload matters, we can compare yellow vs. green. This second comparison shows the impact of different λ_{read}, λ_{write} for identical cells. As mentioned before, with the SA connected to C, it reverses the side with high $\Delta t_{delay.read}$ and high λ_{cell}, since the SA is much stronger and thus its λ_{read} governs overall $\Delta t_{delay.read}$ for 0 and 1, regardless of which side of the cell is degraded. For "sense 0" and "sense 1" the actual λ_{read} does not have a large impact on $\Delta t_{delay.read}$. Yet for "latch 0" and "latch 1" the impact of λ_{read} can clearly be observed as yellow and green swap who has higher $\Delta t_{delay.read}$.

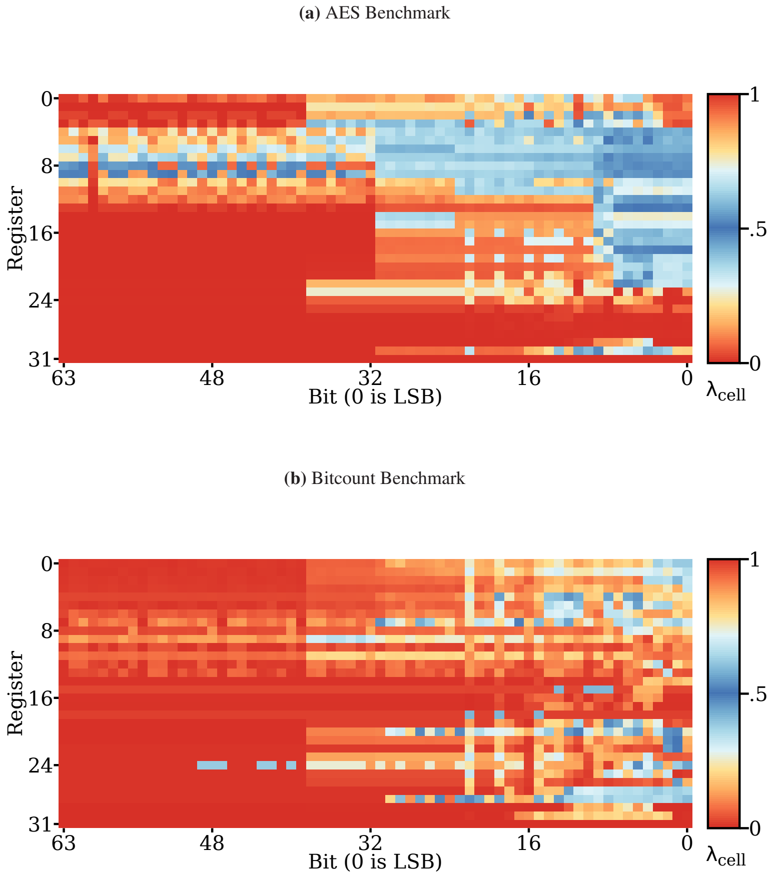
The write operation (seen in Fig. 8.7b) "$V_Q = V_{QB}$ 1" describes when the two voltages at the internal nodes of the SRAM cell are equal (approximately $\frac{V_{DD}}{2}$), "Cell Flip 1" when the memory cell reached a stable storage of a 1 ($V_Q \approx V_{DD}$ for storing a 1) and "BL Stable 1" until the bit line reached the stable value ($V_{BL} \approx V_{DD}$ for storing a 1). The latter happens last, because the cross-coupled inverters in the memory cell support the cell flip after they are over their tipping point ($V_Q = V_{QB}$). In terms of $\Delta t_{delay.write}$, the SRAM cell resist the write if its value is flipped. Hence, it is easier to write a value, in which the SRAM cell resists with degraded transistors (yellow has shorter $\Delta t_{delay.write}$ than blue). This is apparent across "$V_Q = V_{QB}$", "Cell Flip" and "BL Stable 1", which all match λ_{cell} (yellow ($\lambda_{cell} \approx 0$) higher $\Delta t_{delay.write}$ for storing 0's and blue ($\lambda_{cell} \approx 1$) higher $\Delta t_{delay.write}$ for storing 1's). Therefore, for the write operation, the actual cell degradation is clearly visible and must be taken into account. This is in contrast to the read operation, where periphery overshadows the cells.

Periphery still plays a role, as comparing yellow with green in Fig. 8.7b also shows a difference in $\Delta t_{delay.write}$. For the cell internal delays "$V_Q = V_{QB}$" and "Cell Flip" the degradation of the WD does not matter much ($\Delta t_{delay.write}$ of yellow and green are comparable). Yet, for "BL Stable 1" the asymmetric/unbalanced degradation (stemming from different λ_{write}) in the WD matters as it takes longer for the degraded side in the driver to charge the rather large bit line capacitance.

In summary, for the read operation, aging in the SA and its output latch are crucial to take into account as the majority of $\Delta t_{delay.read}$ is governed by them. For the write operation, degradation in the cell matters the most, while the WD plays a secondary but non-negligible role. Evaluating reliability with static degradation severely overestimated Δt_{delay} for both read and write. Thus, when hardening an entire SRAM array against aging, both periphery and workloads must be taken into account.

Comparing worst-case estimations (equivalent to SS process corners) in grey, compared to the the workload cases results in $80\,\%$ smaller timing guardband for the read operation and a $65\,\%$ smaller timing guardband for the write operation. This clearly highlights the need for custom reliability estimations and the considerable pessimism involved in worst-case estimations.

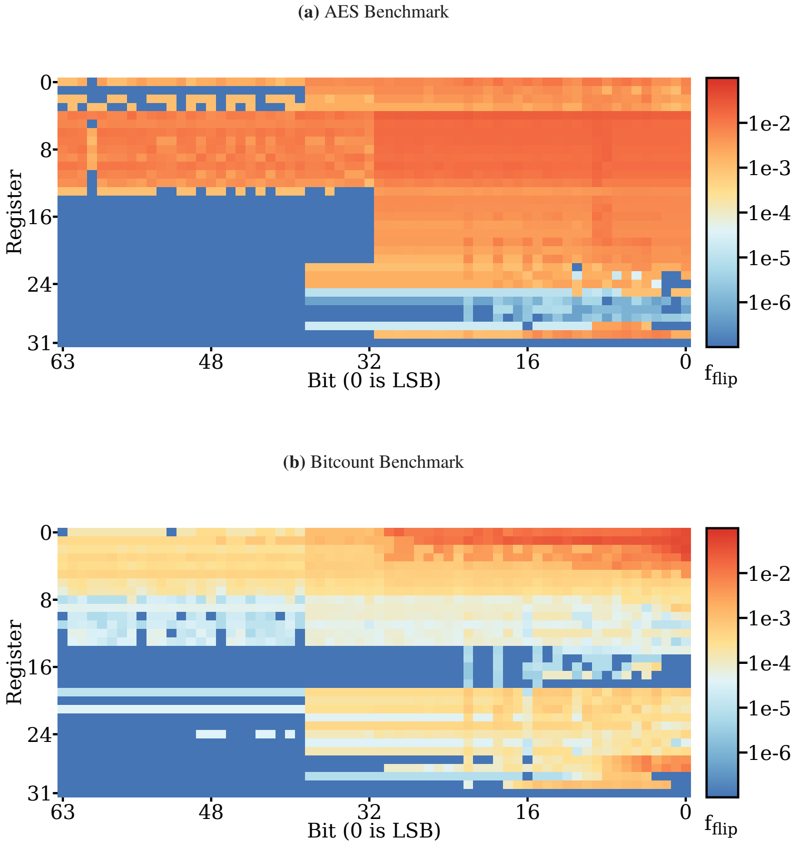
8.1.4.3 Register File Workload in a Processor


(a) AES Benchmark

(b) Bitcount Benchmark

Figure 8.8: a) SRAM balance ratio (duty cycle) in "AES" application b) SRAM balance ratio (duty cycle) in "bitcount" application. Bits above 40 are barely accessed, since the computation uses mostly 32-bit values and calculation of virtual addresses are done in 40 bit. "AES" shows much more balanced λ_{tran} as encryption algorithms try to mimic white noise.

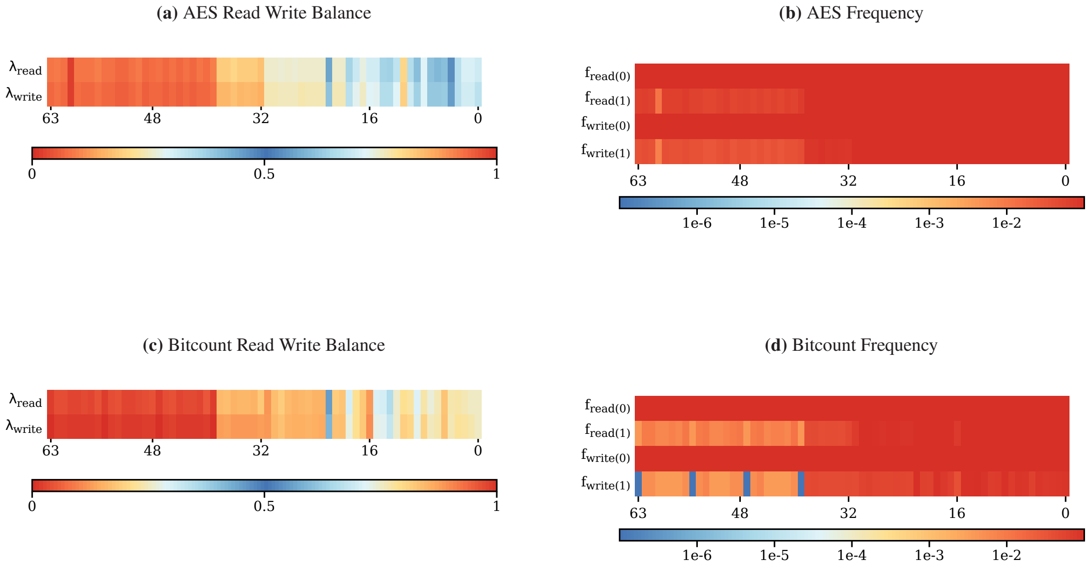
Fig. 8.8 show the duty cycles λ_{cell} (ratio of SRAM cell storing 1's or 0's) for two benchmark applications "AES" and "bitcount", illustrating the diversity in workloads (applications). Both applications feature essentially zero writes in the upper bits. These upper bits are barely accessed, as applications rarely require 64-bit values (e.g. int variables are typically signed 32-bit variables). In this case, the ARM CPU uses 40-bit virtual addresses to store 32-bit variables, which is clearly visible in the Fig. 8.8 as the cut-off. The cells above bit 40 are thus essentially static, resulting a very unbalanced reads λ_{read} (all read 0's) and unbalanced λ_{cell} (storing 0's). For the lower bits, "AES" shows very balanced (blue) λ_{cell} and thus balanced λ_{tran}. This makes sense, since encryption algorithms try to behave as randomly as possible (i.e. as close to white noise as possible) to protect against side-channel attacks. Other applications are not designed in this manner. Thus, for "bitcount" only a few lower bits are balanced (blue) and most are deep red (extremely unbalanced), indicating λ_{tran} close to 0 or 1 (depending if primarily 0's or 1's are stored). This highlights that different workloads might share general trends (higher bits: governed by CPU architecture, memory architecture, etc.), while at the same time being clearly distinct from each other (lower bits: governed by application data and instructions), indicating that separate hardening against aging for these areas might be useful, with more attention focusing on the static (and thus higher degraded) architecture-governed areas.


(a) AES Benchmark

(b) Bitcount Benchmark

Figure 8.9: a) Bit flip frequency in "AES" a) Bit flip frequency in "bitcount" application. "AES" flips the bits much more and consequently has significantly higher load on the WD. This highlights why taking the workload into account is important for both SRAM cells and periphery.

The flipping frequency f_{flip} shown in Fig. 8.9 follows largely the trends of the duty cycle λ_{cell}. This makes sense, as frequent flipping (high f_{flip}) should on average result in a more balanced λ_{cell}. However, for "AES" the registers 4-12 and bits 32-0 show a very uniform f_{flip}, despite some irregularities in the λ_{cell} plot in the same area. Similarly, for "bitcount" in registers 0-3 and bits 32-0 there is a clear f_{flip} hotspot in Fig. 8.9, while in Fig. 8.8, we see unbalanced λ_{cell} above bit 4. As λ_{cell} is important for SRAM cell design, but f_{flip} indicates the load for the corresponding WD, these discrepancies highlight how unfortunately λ_{cell} data alone cannot hint at the workload of SRAM cell periphery.



(a) AES Read Write Balance

(b) AES Frequency

(c) Bitcount Read Write Balance

(d) Bitcount Frequency

Figure 8.10: a) and c) Read λ_{read} and write λ_{write} balance. "AES" shows much more balanced reads and writes than "bitcount" b) and d) Read and write frequencies ($f_{read}(0)$, $f_{read}(1)$, $f_{write}(0)$,$f_{write}(1)$) for "AES" and Bitcount. Both workloads are comparable and write/read in the low kHz-range.

Instead, Fig. 8.10 shows the column activity for the 2 benchmarks "AES" and "bitcount", which is the activity seen by the periphery, with λ_{read} governing the balance of SA and λ_{write} governing the balance of the WD. As the majority of cells store a logic 0, the $f_{read}(0) \gg f_{read}(1)$ and $f_{write}(0) \gg f_{write}(1)$ resulting in an overall $\lambda_{read} \approx 0$ and $\lambda_{write} \approx 0$ for most bits (especially the upper bits). Additionally, since most reads and writes operate on 0's, the periphery ages asymmetrically, which affects overall circuit delay. This indicates that SA and WD should receive hardened transistors which provide currents for reading and writing 0's. For lower bits, things balance out more and "bitcount" reaches $\lambda_{read} \approx 0.2$, while "AES" reaches $\lambda_{read} \approx 0.5$. This shows, that not just for the cells, but also for the periphery, there are clear general trends governed by circuit architecture (CPU-architecture, memory-architecture, etc.) as well as unique behavior by the workload running on the circuit. Thus, as for the cells, both aspects must be considered when designing reliable SRAM array periphery.

8.2 Aging-Aware Voltage Scaling

To explore custom reliability estimations further, we present another use-case scenario in order to highlight their strength. In this section, we explore the impact of voltage scaling on aging. The aging framework from Section 8.1.3.3 is employed to estimate the impact of BTI-induced degradations on the circuit. As power scaling is considered, this estimation occurs under dynamic supply voltages. To fully highlight the modeling capabilities, guardband estimation and checking if timing can still be met occurs under ultra-fast voltage scaling (switches within 1 µs). This section is based on my publication [121].

8.2.1 Ultra-Fast Voltage Scaling is a Reliability Challenge

Increasing the supply voltage (V_{dd}) allows to boost the CPU performance [136] due to the higher operating frequency, but decreasing V_{dd} helps avoiding critical temperatures.

Ultra-fast voltage scaling: The joint fulfilling of both performance and thermal constraints necessitates to switch the voltage very frequently. However, each V_{dd} switch invokes a performance penalty due to the inoperative phases. This is unavoidable since the power supply would be unstable during switching due to charging/discharging the chip's capacitances [176]. To increase the efficiency, manufactures started implementing *ultra-fast* voltage regulators where V_{dd} switching moved into the sub-micron regime like the Intel Haswell CPU which switches between voltage levels within less than 1μs [176, 177], reducing the performance penalty of voltage scaling.

Aging effects: In the nano-scale era, aging effects are at the forefront of reliability concerns due to their momentous ability to cause hardware failures. During the operation of transistors (i.e. applying/ceasing electric fields) the Bias Temperature Instability (BTI) aging mechanism[1], leads to continuously breaking/annealing Si-H bonds at the Si-SiO_2 interface as well as capturing/emitting charges in the oxide vacancies inside the transistor's SiO_2/high-κ dielectric [8]. Overtime, generated defects manifest as a gradual shift in the threshold voltage of a transistor (V_{th}). Aged (i.e. slower) transistors degrade the reliability of on-chip systems as they become less resilient to timing violations manifesting in errors.

Guardband: To sustain reliability during the entire lifetime of an on-chip system, designers employ a guardband, i.e. a slack time ($t_{guardband}$) that is added to the nominal delay of chip ($t_{nominal}$), to tolerate the slower operation due to aging.

$$\begin{aligned} f_{clock} &= \frac{1}{t_{clock}} \text{ ; } t_{clock} = t_{nominal} + t_{guardband} \\ t_{operation} &> t_{clock} \Rightarrow \text{Timing violations} \end{aligned} \tag{8.1}$$

Aging in the scope of voltage scaling: In fact, aging is accelerated/decelerated based on the strength of electric fields and thus based on V_{dd} [8]. Hence, ΔV_{th} indeed *follows* the tendencies of V_{dd} controlled by the employed voltage scaling technique, i.e. higher V_{dd} $\rightarrow$ higher aging-induced ΔV_{th} and vice versa. Importantly, switching V_{dd} in an ultra-fast manner opens the door for emerging ***transient errors***, as the V_{dd} will be dropped much faster than the speed of aging recovery, as it will be demonstrated in Section 8.2.2. In practice, such transient errors may appear immediately after switching from high to low V_{dd} level due to the temporary violation of the guardband. In Fig. 8.11, we show how $t_{operation}$ temporarily grows larger than t_{clock} after switching to a lower V_{dd} level. This is because of the high ΔV_{th}, originating from the previous high V_{dd} level along with the negligible recovery within a transition time of $<1\mu$s. Recent measurements in [177] through an on-chip hardware monitor validated

[1] We focus solely on BTI as it is responsible for the highest degradation compared to other aging mechanisms [15]. However, our work is applicable to any mechanism featuring recovery, like Hot Carrier Injection.

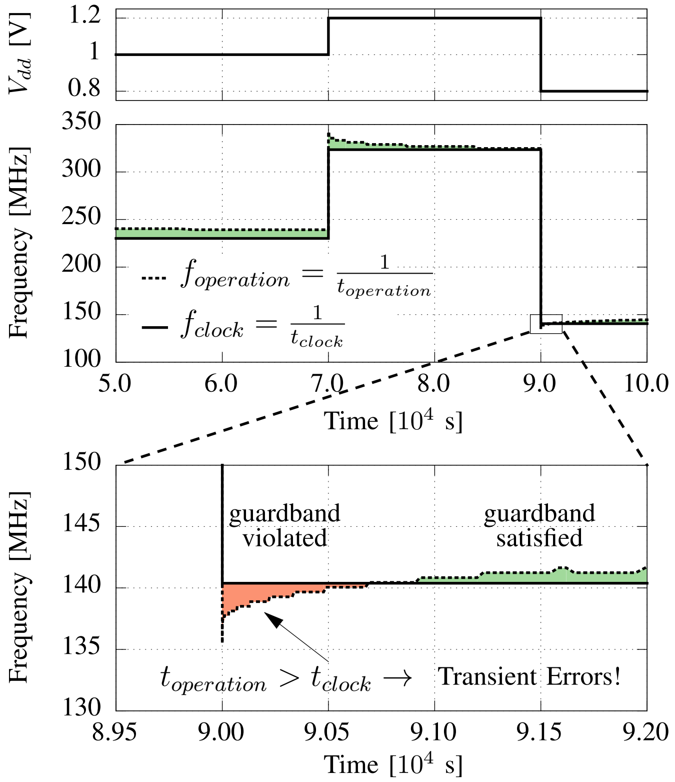


Figure 8.11: Aging in conjunction with ultra-fast voltage scaling may lead to transient errors due to the temporary violation of guardband

the theoretical prediction [178] of a sudden drop in the frequency (see Fig. 8.11) after the switch from high to low voltage level.

Therefore, aging effects should better be investigated jointly with voltage scaling. Otherwise, reliability may be unsustainable due to the hidden short-term effects of aging.

Our novel contributions within this paper are as follows:
(1) We explore for the first time the *short-time* effects that aging in conjunction with voltage scaling has on reliability. This is unlike state-of-the-art which treats aging only as a long-term deleterious effect [15, 25].
(2) To proactively avoid aging-induced transient errors, we propose a technique that adaptively tunes the guardband at runtime towards employing a small, yet sufficient one. Thereby, our technique still maintains the benefits of ultra-fast voltage switching and avoids the high performance loss that incurs from employing non-efficient guardbands.

8.2.2 Aging-Induced Transient Errors

As soon as a pMOS is turned on, the BTI mechanism occurs and generates defects that shift the V_{th}. The induced ΔV_{th} is determined by the strength of V_{dd} as Fig. 8.12(a) shows, where ΔV_{th} due to different V_{dd} levels is presented. However, when V_{dd} is switched to a lower level, a partial recovery of the generated defects starts to take place as Fig. 8.12(b) demonstrates. *State-of-the-art (e.g., [15, 179]) considers that recovery solely occurs when the pMOS is turned off (i.e. $V_{gs} = 0V$).*

However, recent measurements [177] as well as state-of-the-art physics-based BTI modeling [34] demonstrated that an intrinsic recovery occurs as soon as V_{dd} is switched to a lower level proving that recovering aging effects do not necessitate turning the pMOS off. To evaluate that, we employ the state-of-the-art Transient Trap Occupancy Model (TTOM) of BTI [34]. As seen in Fig 8.12(b), switching V_{dd} from 1.0V down to 0.9V and 0.8V reduces ΔV_{th} by 43% and 59%, respectively. This is in contrast to [180] which shows that voltage scaling has no impact on aging.

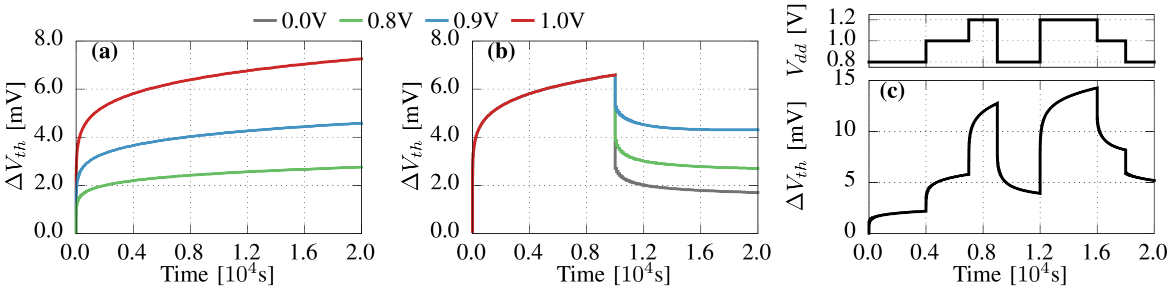


Figure 8.12: Overview of the degradation and recovery of the BTI aging mechanism and its relations with voltage scaling.
(a) Aging degradation is determined by the strength of V_{dd}, i.e. higher V_{dd} leads to higher ΔV_{th}.
(b) Although the transistor is still on, switching the voltage to a lower level allows an intrinsic recovery to occur contrary to state-of-the-art that assumes recovery only occurs when at 0V.
(c) Aging degradation *follows* the tendencies of voltage scaling. This demonstrates the necessity to *jointly* investigate aging and voltage scaling (as we propose) and not *separately* (as state-of-the-art does)

This is due to employing models that are not capable to capture aging under voltage scaling. Note [180] like others also assumes only long-term effects of aging. Additionally, Fig. 8.12(c), illustrates how aging degradation *follows* the tendencies of voltage scaling. All in all, V_{dd} govern aging effects and therefore it is indispensable to investigate them *jointly* with voltage scaling.

In fact, increasing V_{th} results in decreasing the transistor drain current (I_D) which elongates its delay [14]. As a result, aging increases the delay of the chip's critical path ($t_{operation}$) due to the delay increase of its individual transistor (t_{delay})[2].

$$t_{operation} = \sum_{i=1}^{n} t_{delay}(i) : i \in \text{critical path transistors} \tag{8.2}$$

$$t_{delay} \propto \frac{1}{I_D} \text{ with } I_D \propto (V_{dd} - V_{th} - \Delta V_{th})^2 \tag{8.3}$$

Susceptibility to aging degradation: Besides its role in *governing* aging, V_{dd} also determines the susceptibility to the induced degradation, i.e. the impact that ΔV_{th} has on increasing $t_{operation}$. In Fig. 8.13, we present how the same of aging degradation ($\Delta V_{th} = 10\text{mV}$) leads to a stronger shift in $t_{operation}$ at lower V_{dd} levels. This is consistent with what it can be derived from Eq. 8.3 where the impact of ΔV_{th} on the t_{delay} magnifies when V_{dd} becomes smaller. This hints to our key idea of revealing the transient errors induced by aging in conjunction with voltage scaling.

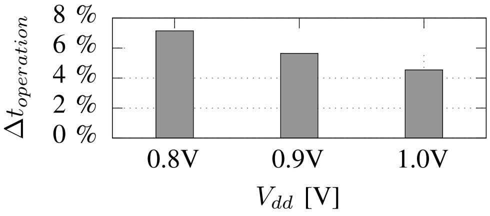


Figure 8.13: SPICE simulations of a ring oscillator with aging modeling from [34] demonstrate that the susceptibility to aging increases as V_{dd} scales down

[2] As aging may change which path is critical, works like [181] can be employed to determine the set of potentially critical paths after aging. For simplicity, our method is presented with respect to a single critical path

Transient Errors: In state-of-the-art, aging is treated as a long-term problem where degrading the reliability of on-chip systems is in the order of months or even years. This is because aging gradually shifts V_{th}. However, employing ultra-fast voltage scaling changes the situation.

While degradation/recovery of aging still occurs gradually, the impact of aging on reliability becomes sudden in the presence of ultra-fast voltage scaling due to the negligible recovery that is feasible within such tiny transition times (i.e. $<1\mu$s). Therefore, the high ΔV_{th}, that was induced at the previous high V_{dd} level, will be carried to next low V_{dd} level where a higher susceptibility to aging degradation exists. Such a conjunction between the high aging degradation and the high aging susceptibility may lead to a temporary violation of the employed guardband (i.e. $t_{operation} > t_{clock}$) and thus to executing operations at that time results in transient errors (see Fig. 8.11). This explains the relevance of aging short-term effects. Despite some works (e.g., [25]) study aging under different V_{dd} levels, such a conjunction between the high aging degradation and the high aging susceptibility was neglected.

8.2.3 Guardbands to Sustain Reliability

Designing the required guardband that sustains reliability (i.e. protects on-chip systems from errors induced by the slower operation of aged transistors) may be either *static* at design-time or *dynamic* at runtime. In both cases, the guardband may be either *optimistically* or *pessimistically* designed.

Optimistic Static Guardband: The designer estimates the aging-induced ΔV_{th} under the worst-case scenario which comes from constantly applying the highest V_{dd} during the entire lifetime (e.g., 10 years). Then, the guardband is designed through calculating the increase in $t_{operation}$ due to the estimated ΔV_{th}, i.e. $t_{guardband} = \Delta t_{operation}$ at the highest ΔV_{th} and the highest V_{dd}. Importantly, such a guardband will be *optimistic* because it does not take into account that V_{dd} may be switched to a lower level causing a conjunction between the high aging degradation (induced at the previous high V_{dd} level) and the high susceptibility (exists at the next low V_{dd} level). In the past, recovery had sufficient time to compensate the higher susceptibility at lower V_{dd} by reducing ΔV_{th} during the voltage switch, but with the introduction of ultra-fast voltage scaling, the OSG approach became unreliable. It may lead to transient errors because such an *optimistic* guardband may temporarily be violated at runtime (see Fig. 8.11).

Pessimistic Static Guardband: To overcome transient errors, the designer may consider the worst-case scenario in both aging degradation and aging susceptibility. In such a case, the guardband is designed through calculating the $\Delta t_{operation}$ based on the worst-case ΔV_{th}, which is caused by constantly applying the highest V_{dd} along with the worst-case aging susceptibility, which comes from switching to the lowest V_{dd}, i.e. $t_{guardband} = \Delta t_{operation}$ at the highest ΔV_{th} and the lowest V_{dd}. Indeed, the designed guardband is able to overcome all transient errors unlike the previous case. However, such a guardband is *pessimistic* (i.e. larger than what actually be needed at runtime) as it considers the worst-case conjunction, where V_{dd} is always scaled from the highest to the lowest V_{dd} level. Therefore, a considerable performance loss may incur due to the unnecessarily low operating frequency.

Dynamic Guardbands: To avoid the high performance loss inherent to pessimistic static guardbands, the guardband may *dynamically* be adapted at runtime based on a hardware monitor that provides delay measurements (e.g., [182]). In practice, the on-chip system periodically (at every t_{update}) checks the monitor and adapts the $t_{guardband}$ according to the current delay increase. It is noteworthy that enabling the monitor to get the measurement imposes aging stress on its transistors and hence frequent access leads to rapidly aging the monitor. Therefore, dynamically adapting the guardband based on periodically reading the monitor is a double-edged sword. On the one hand, infrequent reading through employing an ***optimistic*** t_{update} (e.g., in the order of seconds) avoids aging the monitor. However, it leads to overcoming only the long-term effects of aging since short-term effects, originating from the ultra-fast voltage scaling, occur in a significantly shorter period of time (i.e. $<1\mu$s). On the other hand, frequently reading the monitor through employing a ***pessimistic*** t_{update}, which must be smaller than the switching time of V_{dd} (i.e. $<1\mu$s), overcomes both short and long-term effects of aging but it concurrently imposes a severe aging stress on the monitor itself and thus it rapidly ages resulting in a high degree of uncertainty with respect to monitor readings.

8.2.4 Our Proposed A-GEAR Technique

To counteract short *and* long-term effects of aging with minimum performance loss, we propose a novel technique that employs an Adaptive Guardband for short- and long-term aging Effects AwaReness (A-GEAR). It is based on an offline (i.e. design-time) analysis, where we investigate the impact that different aging degradations at different V_{dd} levels – which are available within the chip [183] – have on the critical path delay. The analysis results are then used to build an *interpretation table* which interprets the current state of aging degradation to the corresponding guardband that the on-chip system actually needs. This table is employed at runtime to allow an efficient *adaptation* of the guardband (i.e. selecting small, yet sufficient guardbands) based on the existing operation conditions, i.e. the current degradation (ΔV_{th}), the previous and next voltage levels (V_{dd} and V'_{dd}).

Aging Effects Investigation

To obtain the current state of aging degradation, we assume the availability of a hardware delay monitor that measures the delay increase at runtime. A wide range of implementations of such monitors has been proposed which, in practice, measure the delay through a ring oscillator and then compare the result with the original/reference delay to capture the delay increase (i.e. $\Delta t_{monitor}$) at any point of time. For instance, state-of-the-art monitor [177] is able to provide its measurements within $<1\mu$s and for different voltages ($V_{dd} \in [0.8-1.4]$V). Authors showed, that their monitor can be implemented through a very simple circuit and hence adds just minor costs/overheads [177].

Once the delay increase ($\Delta t_{monitor}$) is known, the current aging degradation state ΔV_{th} can be estimated[3] To achieve that, the hardware monitor circuit (i.e. the ring oscillator) is modeled through a SPICE netlist along with the BSIM4 transistor model [14] on 22nm PTM technology [3]. Then, we employ our aging framework from Section 8.1 that models the impact of BTI on V_{th} and, more importantly, is able to take the voltage dynamics into account. This enables us to accurately consider the joint effect of voltage scaling and aging degradation on reliability. Table 8.1 shows an example of such an analysis when an 11-stage ring oscillator is examined.

To the best of our knowledge, the employed aging modeling within this work is the exclusive one that is able to consider the intrinsic aging recovery due to scaling the voltage down (see Fig. 8.12(b)) from a physical perspective. While the *empirical* aging model [178] is able to consider voltage fluctuations, the aging modeling [34] that we employ is based on modeling the underlying *physical processes* behind aging and hence we can model a wide range of voltages, temperatures, etc. with a high degree of certainty. This is indispensable to achieve our goal of exploring the short and long-term effects of aging where we need to accurately investigate aging degradation within very fine-grained time steps (i.e. microseconds) and under highly-fluctuating voltage conditions (i.e. ultra-fast voltage scaling).

Table 8.1: Example of the resulting ΔV_{th} due to different ΔT and V_{dd}

	V_{dd}		
$\Delta t_{monitor}$	0.8V	0.9V	1.0V
5%	7mV	9mV	12mV
10%	14mV	17mV	21mV
15%	19mV	23mV	29mV
20%	24mV	30mV	38mV

Guardband Estimation

Once, the ΔV_{th} is known, we then estimate the required guardband at the each V_{dd}. This can be achieved through simulating the impact of that particular ΔV_{th} on the chip's critical path delay in SPICE according to different V_{dd}.

[3] To consider the intrinsic variability of BTI [184], the distribution $\Delta V_{th}(\mu, \sigma)$ could be calculated [184] and worst case of ΔV_{th} (e.g. 6σ) selected as the upper bound for degradation. However, our recent model [34] only models the mean $\Delta V_{th}(\mu)$.

Table 8.2 shows an example of resulting guardband for different conjunctions between aging degradation (ΔV_{th}) and next voltage level (V'_{dd}). It is noteworthy that, the same ΔV_{th} results in different delay increases in the monitor itself and in the critical path (i.e. $\Delta t_{monitor} \neq \Delta t_{operation}$ at the same ΔV_{th}). This is because different circuits may have varied transistors sizes (i.e. different I_D) and therefore the same ΔV_{th} results in different delay increases. For instance, the guardband, at $\Delta V_{th} = 20$mV and $V'_{dd} = 0.8$V, results in 5.3 difference corresponding to a 27% underestimation if we solely rely on the monitor measurement without interpreting it (see Table 8.2).
This illustrates why we cannot directly rely on the monitor to select our guardband unless we interpret its measurement to the corresponding delay increase in the critical path of chip.

As explained in Section 8.2.1 and motivated in Figs. (8.11, 8.13), circuits become more susceptible as V_{dd} scales down. Therefore, guardbands increase if V_{dd} is switched to a lower level and/or ΔV_{th} increases, as it can also be observed in Table 8.2.

Table 8.2: The same aging-induced ΔV_{th} results in a different delay increase in the critical path compared to the monitor itself

Aging	Supply Voltage V'_{dd}					
	0.8V			1.0V		
ΔV_{th}	$\Delta t_{monitor}$		$\Delta t_{operation}$	$\Delta t_{monitor}$		$\Delta t_{operation}$
5mV	3.5%	→	4.5%	2.6%	→	3.2%
10mV	7.2%	→	9.1%	4.6%	→	6.4%
15mV	10.8%	→	14.0%	6.9%	→	9.6%
20mV	14.2%	→	19.5%	9.2%	→	12.7%

Runtime Adaptation to select Guardbands

Based on the aforementioned design-time investigations presented in Tables (8.1, 8.2), an interpretation table can be extracted to be employed at runtime. Such a table contains the required guardbands that are actually needed to tolerate the delay increase in the critical path according to different operating conditions. Such a table is a two-dimensional (nxm) array, where n is the total number of $\Delta t_{monitor}$ steps and m is the total number of V_{dd} levels. In practice, for each $\Delta t_{monitor}$ step, we calculate the corresponding ΔV_{th} within the hardware monitor transistors. Then, we apply that calculated ΔV_{th} to the critical path of our on-chip system to estimate the delay increase ($\Delta t_{operation}$). Table 8.3 shows an example of the resulting table that will be implemented within the chip to be employed by our runtime algorithm (see Algorithm 5 and further details are in the next section) that present the hardware implementation of our proposed A-GEAR technique. A hardware monitor may have fine-grained $\Delta t_{monitor}$ steps, leading to a large n. To reduce n, we store only the entries, which lead to a different guardbands. As guardbands correspond to the small set of frequency levels within a CPU and hence are coarse-grained in comparison. This allows the feasibility of implementing the table in hardware.

Overcoming Short-term Effects of Aging: Whenever a voltage switch is triggered the responsible control circuit reads, from the hardware monitor, the current delay increase ($\Delta t_{monitor}$) at the requested new voltage level (V'_{dd}). Then, it obtains from our implemented look-up table the required guardband that sustains a reliable operation based on ($\Delta t_{monitor}$, V'_{dd}). To further optimize our technique, we additionally exploit the intrinsic recovery that is inherent to switching to a lower voltage level (see Fig. 8.2.2(b) and Section 8.2.2). As recovery is a exponential function [8], it is worthwhile to adapt the guardband again after a short period of time to exploit the recovery ($t_{recovery}$) which, in turn, enables us to avoid applying a non-efficient guardband (i.e. larger than what the system actually needs).

Overcoming Long-term Effects of Aging: On the other hand, voltage scaling may not be triggered for a prolonged interval of time and hence the employed narrow guardband may become insufficient due to the gradual degradation of aging (i.e. the well-known long-term effect of aging). Therefore, to also counteract long-term effects of aging

Table 8.3: Example of the hardware table, interpreting the hardware monitor delay to a guardband of chip at different voltages.

Monitor	Supply Voltage V_{dd}		
$\Delta t_{monitor}$	0.8V	0.9V	1.0V
5%	6.3%	6.4%	7.3%
10%	12.2%	12.5%	13.6%
15%	18.4%	18.5%	19.0%
20%	24.1%	24.1%	26.3%

while employing narrow guardbands we regularly update the guardband, at every t_{update} similar to [182], based on $(\Delta t_{monitor}, V_{dd})$ after rechecking the hardware monitor measurement.

Distinction from existing techniques: Various adaptive guardband techniques have been proposed (e.g., [182]). However, our A-GEAR technique distinguishes itself from others through the following novelties:

- It considers the short *and* long-term effects of aging, instead of solely long-term effects, which prevents transient errors.
- It interprets the aging monitor measurement to the corresponding guardband that the chip's critical path actually needs, instead of directly applying the measurement itself as a guardband, which prevents wrong guardbands.
- It considers the intrinsic recovery of aging in the on-state of the transistor, recently demonstrated [34, 177], which provides efficient guardbands.
- It considers, while adapting the guardband, the impact of voltage scaling on the susceptibility to aging, which allows a correct estimation of guardbands.

Algorithm 5 Algorithm of our hardware A-GEAR technique

```
Require: Current, new voltages ($V_{dd}$, $V'_{dd}$), Timer, Look-up Table
for every trigger $\in$ (voltage switch, timer expired) do
    if voltage switch then
        Read $\Delta t_{monitor}$ at $V'_{dd}$                                  ▷ monitor
        Get $t_{guardband}$ at $(\Delta t_{monitor}, V'_{dd})$                  ▷ look-up
        Set frequency $f_{clock}$
        Switch to $V'_{dd}$
        if $V'_{dd} < V_{dd}$ then                                              ▷ intrinsic recovery
            Set timer to $t_{recovery}$
            Wait until timer expired
            Read $\Delta t_{monitor}$                                           ▷ monitor
            Get $t_{guardband}$ at$(\Delta t_{monitor}, V_{dd})$                ▷ look-up
            Set frequency $(f_{clock})$
        end if
    else if timer expired then
        Read $\Delta t_{monitor}$                                               ▷ monitor
        Get $t_{guardband}$ at $(\Delta t_{monitor}, V_{dd})$                   ▷ look-up
        Set frequency $(f_{clock})$
    end if
    Set timer to $t_{update}$
end for
```

8.2.5 Experimental Setup

To evaluate our A-GEAR technique and to quantify the short-term effects of aging, we implemented the following:

Thermal Estimation: First, the gem5 simulator [148] extracts the activities of the running application on top of the single-core Alpha CPU[4]. Then, the McPAT simulator [150] provides for a 22nm technology, the V_{dd} levels of the Alpha CPU along with the corresponding maximum frequency and static/dynamic power consumption of the CPU's components at each V_{dd}. Afterwards, the Hotspot thermal simulator [13] estimates the temperature of the CPU's components based on the extracted activity and power. In our experiments, we employed diverse applications from the PARSEC [185] and SPEC2006 [186] benchmark suites exhibiting diverse activities/powers and hence thermal behaviors. In addition, we executed them on top of the Linux OS to consider a more realistic scenario than bare-metal execution.

Dynamic Thermal Management (DTM): We implemented the state-of-the-art DTM technique, namely "Intel Turbo Boost 2.0", from the Intel Haswell CPU [136]. It works as follows [183]: it checks every 1ms whether the critical temperature (e.g., $T_{crit} = 80°$C) is reached or not. If yes, it decreases frequency by one step (e.g., 133 or 200MHz) and it scales the V_{dd} down to the corresponding V_{dd} level of new frequency. If T_{crit} is not yet reached, the frequency is, instead, increased by one step and the V_{dd} is scaled up to the corresponding V_{dd} level.

Aging Estimation: As explained in Section 8.2.4, we estimate aging effects with state-of-the-art BTI aging modeling [34]. Based on the voltage trace, which is resulted from the thermal behavior of the running application and the employed DTM technique, we estimate the corresponding aging degradation trace. The latter enables us to quantify the short and long-term effects of aging jointly with voltage scaling towards capturing when the guardband is violated.

Evaluated Scenarios: For a fair comparison and a more general evaluation, we consider the following four scenarios:
(1) *Base*: The unmodified (i.e. nominal) CPU which is not protected against aging (i.e. no guardband is employed).
(2) *Optimistic Static Guardband (OSG)*: The CPU is protected against only the long-term effects of aging (see Section 8.2.3).
(3) *Pessimistic Static Guardband (PSG)*: The CPU is protected against the short and long-term aging effects (see Section 8.2.3).
(4) *A-GEAR*: The CPU is protected against short and long-term effects of aging through adapting the guardband at runtime based on our proposed technique described in Algorithm 5.

It is noteworthy that Base and OSG are unreliable designs as errors due to aging may occur. Whereas, PSG and A-GEAR are reliable designs as they prevent errors due to aging.

8.2.6 Evaluation, Comparison And Advantages

Since transient errors due to the short-term effects of aging occur only when V_{dd} is switched to a lower level, we show in Fig. 8.14 the total number of falling edges after analyzing the resulting voltage trace of each application. The reason behind the variety in voltage traces is that the applications have different thermal behaviors and thus they differently trigger the DTM technique. As a result, different applications exhibit different rates of transient errors that are induced by the short-term effects of aging.

[4] In many-core system, our A-GEAR needs to be implemented in each core individually to consider different V_{dd} levels per core

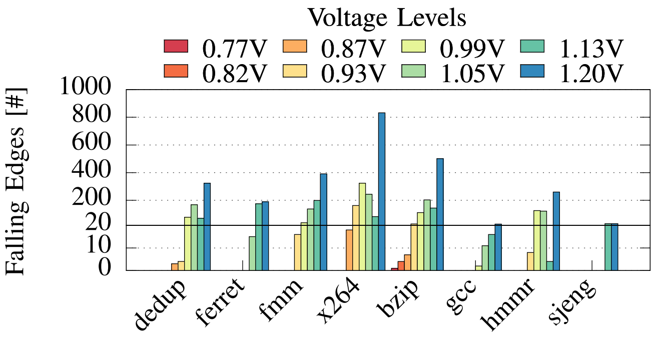


Figure 8.14: The number of falling edges due to reducing V_{dd} one step (e.g. $0.99V \rightarrow 0.93V$) within the resulting voltage traces as on-chip system is susceptible to transient errors only there

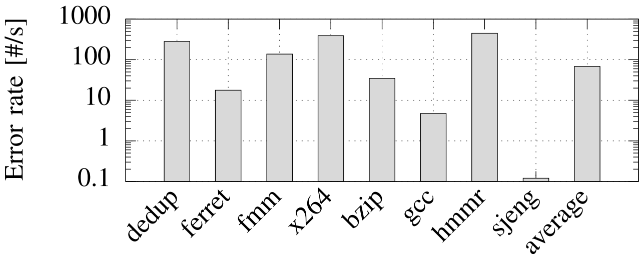


Figure 8.15: Error rate after 1 year operating at $V_{dd} = 1.2$V. Note, employing A-GEAR prevents all errors due to the employment of sufficient guardbands.

To quantify the latter, we demonstrate in Fig. 8.15 the total the number of occurring transient errors per second in OSG (i.e., not counteracting short-term effects of aging). In such a case, the designed guardband is 17% which is the resulting aging degradation at the end of a 10 years lifetime when the highest V_{dd} (1.2V) is constantly applied. As shown in Fig. 8.15, designing a guardband – that is unaware of the short-term effects of aging – leads to unreliable behavior due to the high rate of transient errors (on average 94 errors/s). In practice, the $\Delta V_{th} = 44m$V that a static guardband of 17% is able to tolerate becomes lower when V_{dd} is switched down and therefore the guardband may temporarily be violated, at the falling edges, resulting in transient errors.

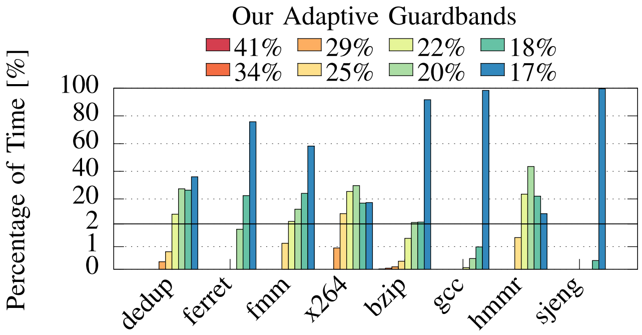


Figure 8.16: Percentage of time spent at each guardband that is *adaptively* selected at runtime through our A-GEAR technique

Our A-GEAR technique *adaptively* selects the sufficient guardband that sustains a reliable operation. The distribution of the selected guardbands at runtime for different applications is presented in Fig. 8.16. As observed, the minority of time is spent within the large guardbands. This is because of the efficient selection of our guardbands due to the exploitation of intrinsic recovery. To evaluate the latter, we demonstrate in Fig. 8.17 the normalized execution time of each application. Compared to applying the OSG technique that protects the on-chip system against only long-term aging effects, our A-GEAR technique overcomes both short and long-term aging effects and it comes with merely 1% overhead on average.

Finally, compared to applying the PSG technique that, similar to ours, is able to overcome short and long-term aging effects our A-GEAR reduces the overhead by 10% on average and up to 21%.

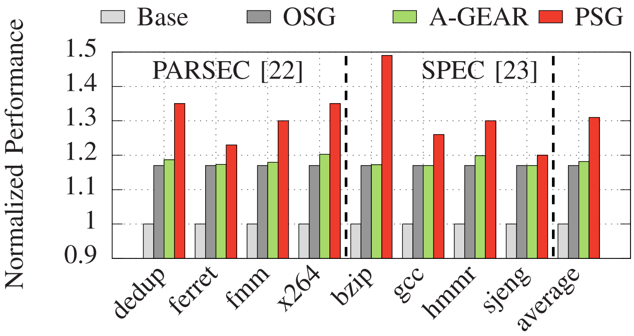


Figure 8.17: Performance loss evaluation

Monitor Degradation: As explained in Section 8.2.3, each access to the hardware monitor imposes an aging stress on it. In fact, A-GEAR accesses the monitor only when the V_{dd} scaling is triggered in addition to the regular update at 1s. Our competitor here is dynamic guardband-based techniques (see Section 8.2.3) when they aim to overcome short and long-term effects of aging. In such a technique, the monitor should very frequently be accessed (i.e. $t_{update} = 1\mu$s) to sustain reliability. Compared to such a case, we mitigate the monitor aging by 4.1x (i.e. we reduce the aging-induced ΔV_{th} in the transistors of the monitor's reference, after a lifetime of 10 years, from 14.03mV to 3.39mV).

8.2.7 Custom Reliability Estimations to Estimate Guardband for Voltage Scaling Use-Case

Regular SS process corners do not consider these short-term aging effects. Hence, a guardband estimated with the traditional FF to SS comparison (see Section 6.1) might not be sufficient to maintain reliability. However, in practice, the guardband is so overly pessimistic (as effects such as temperature and PV are also at their worst), that details such as these can be ignored. The SS corner is just that more pessimistic than the static PSG and OSG presented in this section, as not just aging is considered. This is the key advantage to such pessimistic assumptions: 1) They simplify the reliability estimation. 2) Their pessimism guarantees reliability. Of course, at the cost of considerable performance losses.

Our custom reliability estimation framework (in this section mainly the transistor modeling part was discussed) allowed us to determine OSG, PSG and ultimately A-GEAR. This allows us to design guardbands, while considering not just voltage-scaling but additionally the short-term aging effects induced by voltage scaling. For this purpose, accurate modeling with respect to dynamic voltage waveforms is crucial.

Therefore Section 8.1 showed the importance of activity and how guardbands can be reduced if they are considered, while this section showed the importance of considering dynamic voltage waveforms. This was the key motivation for choosing these two use-cases to highlight the application of our custom reliability estimations. First, reducing the required guardband considerably in Section 8.1 by considering workload. Secondly, to ensure that these tight custom guardbands are not violated, even by recently uncovered short-term aging effects, as the modeling and simulation can handle challenging use-case scenarios with dynamic voltages in Section 8.2.

9 Conclusion

9.1 Limitations

This work features several limitations, mainly with respect to CMOS technologies. First of all, the models and tools are developed for regular MOSFET and FinFET transistors. As seen in Section 1.4.2, new phenomena might emerge (Section 1.4.2.5) and materials change (Section 1.4.2.4) in newer generations (e.g., gate-all-around transistors, nanosheets) and then the underlying assumptions of this work might not be valid anymore. Therefore (as with almost all reliability research), each new technology generation these assumptions should be checked (e.g., the assumption "elevated temperatures make transistors perform worse").

Additionally, even if all assumptions hold, on new generations the parameters have to change. Since the assumptions held, the presented models still hold (i.e., the equations and physics in the model do not have to change) but their calibration does not. For new generations, the models have to be re-calibrated, i.e. their responses have to be fitted to measurement of that new technology. Once all model parameters are updated, the same tools can then be used to create custom reliability estimations in new generations.

Except new generations, this work also may not apply to other transistor technologies. Tunnel Field Effect Transistors (TunnelFET) or Negative Capacitance Field Effect Transistors (NCFET) are different ways to use a transistor with various advantages over traditional CMOS technologies (e.g., better ON-/OFF-ratio). With respect to this work, their entire foreign structure (new materials, new geometries, etc.) means that again assumptions have to be checked and the models have to be updated. In fact, for NCFET transistors we have already performed these steps [187, 188]. This clearly shows how robust the presented methodology is.

9.2 Future Work

Future work would be to continuously update the reliability models to each new generation and transistor technology. As mentioned in the above Section 9.1, the models need to be continuously updated and re-calibrated.

Additionally, it is imperative, that the most important degradation phenomena are covered. For example, this work does not include Electromigration (EM) in the wires and connections of the chip, as the standard tools already consider EM in a custom manner. Instead of rough generalizations, EM is already considered for the individual layout of the power delivery (supply voltage, ground, etc.) and data network (logic connections). Should additional phenomena appear, which severely alter the reliability of a circuit/chip, then it must be included in a custom reliability estimation to ensure the guardband is still sufficiently large at all times (to maintain reliability).

9.3 Summary

This work provided a four-step process to integrate custom reliability estimations for analogue and digital circuits. Step one improved and unified degradation models and made them ready for circuit reliability estimations (instead of pure transistor reliability estimations). Step two accelerated these models, so that large-scale estimations (featuring thousands to millions of transistors) are feasible. Step three integrated these degradation models into the standard tools, to make them aware of custom reliability estimations (instead of just worst-case estimations). Step four accelerated these standard tools, to make cope with the demand of the large-scale analogue simulations of today.

With these contributions, a custom circuit reliability can be estimated for a given temperature, voltage, lifetime and workload. Each of these inputs can be a single peak value (e.g., $125\,^{\circ}\mathrm{C}$), an average value (e.g., $86\,^{\circ}\mathrm{C}$ for application “MatrixMul”) or values over time (e.g., $66\,^{\circ}\mathrm{C}$ at $1\,\mathrm{s}$, $69\,^{\circ}\mathrm{C}$ at $2\,\mathrm{s}$, etc.). Similarly, these values can be provided for each transistor individually, for subcircuits or for the entire circuit. These trade-off between detailed information over time/space and simply using peak values/full-circuit information is simulation accuracy versus simulation time. The tools themselves do not determine this, the end-user can decide his tolerance for inaccuracies in his results versus the effort to obtain these values (temperature simulations, voltage measurements, etc.).

With these custom reliability estimations, the circuit designer can determine the accurate guardband. Sufficiently large to tolerate the accumulated degradations within the circuit during the specific operation (use-case/application) of the circuit (during its lifetime) and thus maintaining reliability. The guardband ensures no loss of data and maintains functionality. At the same time, the guardband is as small as possible to not impose unnecessary performance losses. In fact, considering custom reliability in our SRAM framework reduced the necessary timing guardband by $80\,\%$ for the read delay and $65\,\%$ for write delay compared to traditional worst-case estimations.

In times of ever-increasing degradations (and thus ever-increasing guardbands) , these guardband reductions are utterly necessary to keep degradations at bay while further gaining the improvements (less power, more performance) from future CMOS technologies. Instead of utilizing all performance gains by a new technology to tolerate the high degradation levels (thus nullifying the advantage compared to the previous generation), now with custom reliability estimations, technology scaling can be beneficial for a few more generations.

List of Figures

List of Tables

Bibliography

[1] J. Henkel, L. Bauer, J. Becker, O. Bringmann, U. Brinkschulte, S. Chakraborty, M. Engel, R. Ernst, H. Härtig, L. Hedrich, A. Herkersdorf, R. Kapitza, D. Lohmann, P. Marwedel, M. Platzner, W. Rosenstiel, U. Schlichtmann, O. Spinczyk, M. Tahoori, J. Teich, N. When, and H. J. Wunderlich, "Design and architectures for dependable embedded systems," in *CODES*, 2011.

[2] M. Bohr, "A 30 year retrospective on dennard's mosfet scaling paper," *IEEE Solid-State Circuits Society Newsletter*, vol. 12, no. 1, pp. 11–13, 2007.

[3] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45 nm early design exploration," *IEEE Transactions on Electron Devices*, vol. 53, no. 11, pp. 2816–2823, 2006.

[4] M. H. Lee, S.-T. Fan, C.-H. Tang, P.-G. Chen, Y.-C. Chou, H.-H. Chen, J.-Y. Kuo, M.-J. Xie, S.-N. Liu, M.-H. Liao, C.-A. Jong, K.-S. Li, M.-C. Chen, and C. W. Liu, "Physical thickness 1.x nm ferroelectric HfZrOx negative capacitance FETs," in *IEDM*, 2016.

[5] K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer, "High-performance cmos variability in the 65-nm regime and beyond," *IBM journal of research and development*, 2006.

[6] ISO 13053-1:2011, "Quantitative methods in process improvement — six sigma — part 1: Dmaic methodology."

[7] S. Salamin, V. M. van Santen, H. Amrouch, N. Parihar, S. Mahapatra, and J. Henkel, "Modeling the interdependences between voltage fluctuation and bti aging," *TVLSI*, 2019.

[8] S. Mahapatra, N. Goel, S. Desai, S. Gupta, B. Jose, S. Mukhopadhyay, K. Joshi, A. Jain, A. Islam, and M. Alam, "A Comparative Study of Different Physics-Based NBTI Models," *T-ED*, 2013.

[9] N. Parihar, N. Goel, S. Mukhopadhyay, and S. Mahapatra, "Bti analysis tool—modeling of nbti dc, ac stress and recovery time kinetics, nitrogen impact, and eol estimation," *IEEE Transactions on Electron Devices*, vol. 65, no. 2, pp. 392–403, 2017.

[10] E. Cartier, A. Kerber, T. Ando, M. Frank, K. Choi, S. Krishnan, B. Linder, K. Zhao, F. Monsieur, J. Stathis, and V. Narayanan, "Fundamental aspects of hfo2-based high-k metal gate stack reliability and implications on tinv-scaling," in *Electron Devices Meeting (IEDM), 2011 IEEE International*, pp. 18.4.1–18.4.4, Dec 2011.

[11] B. Tudor, J. Wang, C. Sun, Z. Chen, Z. Liao, R. Tan, W. Liu, and F. Lee, "MOSRA: An efficient and versatile MOS aging modeling and reliability analysis solution for 45nm and below," in *ICSICT*, 2010.

[12] M. Selim, E. Jeandeau, and C. Descleves, "Design-reliability flow and advanced models address ic-reliability issues.," in *ERMAVSS@ DATE*, pp. 1–4, 2016.

[13] M. R. Stan, K. Skadron, M. Barcella, W. Huang, K. Sankaranarayanan, and S. Velusamy, "Hotspot: a dynamic compact thermal model at the processorarchitecture level," *Microelectronics Journal*, 2003.

[14] Y. Chauhan, S. Venugopalan, M. Karim, S. Khandelwal, N. Paydavosi, P. Thakur, A. Niknejad, and C. Hu, "BSIM - Industry standard compact MOSFET models," in *ESSCIRC*, 2012.

[15] H. Amrouch, V. M. van Santen, T. Ebi, V. Wenzel, and J. Henkel, "Towards interdependencies of aging mechanisms," in *ICCAD*, 2014.

[16] D. Pierce and P. Brusius, "Electromigration: A review," *Microelectronics Reliability*, vol. 37, no. 7, pp. 1053–1072, 1997.

[17] J. Lienig and M. Thiele, *Fundamentals of Electromigration-Aware Integrated Circuit Design*. Springer, 2018.

[18] E. Ogawa, J. Kim, G. Haase, H. Mogul, and J. McPherson, "Leakage, breakdown, and tddb characteristics of porous low-k silica-based interconnect dielectrics," in *Reliability Physics Symposium Proceedings, 2003. 41st Annual. 2003 IEEE International*, pp. 166–172, March 2003.

[19] S. Tous, E. Wu, and J. Sune, "A compact analytic model for the breakdown distribution of gate stack dielectrics," in *Reliability Physics Symposium (IRPS), 2010 IEEE International*, pp. 792–798, May 2010.

[20] J. Martin-Martinez, B. Kaczer, R. Degraeve, P. J. Roussel, R. Rodriguez, M. Nafria, X. Aymerich, B. Dierickx, and G. Groeseneken, "Circuit design-oriented stochastic piecewise modeling of the postbreakdown gate current in mosfets: Application to ring oscillators," *Device and Materials Reliability, IEEE Transactions on*, vol. 12, no. 1, pp. 78–85, 2012.

[21] G. Bersuker, D. Heh, C. Young, H. Park, P. Khanal, L. Larcher, A. Padovani, P. Lenahan, J. Ryan, B. H. Lee, H. Tseng, and R. Jammy, "Breakdown in the metal/high-k gate stack: Identifying the weak link in the multilayer dielectric," in *Electron Devices Meeting, 2008. IEDM 2008. IEEE International*, pp. 1–4, Dec 2008.

[22] T. Nigam, A. Kerber, and P. Peumans, "Accurate model for time-dependent dielectric breakdown of high-k metal gate stacks," in *Reliability Physics Symposium, 2009 IEEE International*, pp. 523–530, IEEE, 2009.

[23] J. Henkel, T. Ebi, H. Amrouch, and H. Khdr, "Thermal management for dependable on-chip systems," in *ASP-DAC*, 2013.

[24] W. Wang, V. Reddy, A. T. Krishnan, R. Vattikonda, S. Krishnan, and Y. Cao, "Compact modeling and simulation of circuit reliability for 65-nm cmos technology," *IEEE Transactions on Device and Materials Reliability*, vol. 7, no. 4, pp. 509–517, 2007.

[25] V. B. Kleeberger, M. Barke, C. Werner, D. Schmitt-Landsiedel, and U. Schlichtmann, "A compact model for NBTI degradation and recovery under use-profile variations and its application to aging analysis of digital integrated circuits," *Microelectronics Reliability*, 2014.

[26] M. A. Alam, "Ece 695a reliability physics of nanotransistors," Jan 2013.

[27] B. Kaczer, T. Grasser, P. J. Roussel, J. Franco, R. Degraeve, L.-A. Ragnarsson, E. Simoen, G. Groeseneken, and H. Reisinger, "Origin of nbti variability in deeply scaled pfets," in *IRPS*, 2010.

[28] H. Kükner, S. Khan, P. Weckx, P. Raghavan, S. Hamdioui, B. Kaczer, F. Catthoor, L. V. 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, pp. 182–193, March 2014.

[29] J. Bhaskarr Velamala, K. Sutaria, H. Shimizu, H. Awano, T. Sato, G. Wirth, and Y. Cao, "Compact Modeling of Statistical BTI Under Trapping/Detrapping," *T-ED*, 2013.

[30] G. Rzepa, J. Franco, B. O'Sullivan, A. Subirats, M. Simicic, G. Hellings, P. Weckx, M. Jech, T. Knobloch, M. Waltl, *et al.*, "Comphy—a compact-physics framework for unified modeling of bti," *Microelectronics Reliability*, vol. 85, pp. 49–65, 2018.

[31] J. Martin-Martinez, B. Kaczer, M. Toledano-Luque, R. Rodriguez, M. Nafria, X. Aymerich, and G. Groeseneken, "Probabilistic defect occupancy model for NBTI," in *IRPS*, 2011.

[32] J. H. Stathis, S. Mahapatra, and T. Grasser, "Controversial issues in negative bias temperature instability," *Microelectronics Reliability*, vol. 81, pp. 244–251, 2018.

[33] S. Mahapatra, *Recent Advances in PMOS Negative Bias Temperature Instability: Characterization and Modeling of Device Architecture, Material and Process Impact.* Springer, 2022.

[34] N. Goel, T. Naphade, and S. Mahapatra, "Combined trap generation and transient trap occupancy model for time evolution of NBTI during DC multi-cycle and AC stress," in *IRPS*, 2015.

[35] G. Rzepa, M. Waltl, W. Goes, B. Kaczer, and T. Grasser, "Microscopic oxide defects causing bti, rtn, and silc on high-k finfets," in *2015 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2015.

[36] M. Toledano-Luque and B. Kaczer, "Characterization of individual traps in high-κ oxides," in *Bias Temperature Instability for Devices and Circuits*, pp. 597–614, Springer, 2014.

[37] T. Grasser, H. Reisinger, P. J. Wagner, F. Schanovsky, W. Goes, and B. Kaczer, "The time dependent defect spectroscopy (TDDS) for the characterization of the bias temperature instability," in *IRPS*, 2010.

[38] 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, pp. 704–714, June 2014.

[39] W. Goes, F. Schanovsky, and T. Grasser, "Advanced modeling of oxide defects," in *Bias temperature instability for devices and circuits*, pp. 409–446, Springer, 2014.

[40] V. Velayudhan, F. Gamiz, J. Martín-Martínez, R. Rodríguez, M. Nafria, and X. Aymerich, "Influence of the interface trap location on the performance and variability of ultra-scaled mosfets," *Microelectronics Reliability*, vol. 53, no. 9-11, pp. 1243–1246, 2013.

[41] M. Moras, J. Martin-Martinez, R. Rodriguez, M. Nafria, X. Aymerich, and E. Simoen, "Negative bias temperature instabilities induced in devices with millisecond anneal for ultra-shallow junctions," *Solid-state electronics*, vol. 101, pp. 131–136, 2014.

[42] U. Sharma and S. Mahapatra, "A spice compatible compact model for hot-carrier degradation in mosfets under different experimental conditions," *IEEE Transactions on Electron Devices*, vol. 66, no. 2, pp. 839–846, 2018.

[43] U. Sharma and S. Mahapatra, "Modeling of hcd kinetics under full v g–v d space, different experimental conditions and across different device architectures," *IEEE Journal of the Electron Devices Society*, vol. 8, pp. 1354–1362, 2020.

[44] K. Takeuchi, T. Nagumo, and T. Hase, "Comprehensive SRAM design methodology for RTN reliability," in *VLSIC*, 2011.

[45] T. Grasser, K. Rott, H. Reisinger, M. Waltl, J. Franco, and B. Kaczer, "A unified perspective of RTN and BTI," in *IRPS*, 2014.

[46] N. Ayala, J. Martin-Martinez, R. Rodriguez, M. Nafria, and X. Aymerich, "Unified characterization of RTN and BTI for circuit performance and variability simulation," in *ESSDERC*, 2012.

[47] B. Kaczer, S. Mahato, V. V. de Almeida Camargo, M. Toledano-Luque, P. J. 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 *IRPS*, 2011.

[48] D. Rodopoulos, S. B. Mahato, V. 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 *ICICDT*, 2011.

[49] R. Wang, M. Luo, S. Guo, R. Huang, C. Liu, J. Zou, J. Wang, J. Wu, N. Xu, W. Wong, S. Yu, H. Wu, S. W. Lee, and Y. Wang, "A unified approach for trap-aware device/circuit co-design in nanoscale CMOS technology," in *IEDM*, 2013.

[50] H. Reisinger, O. Blank, W. Heinrigs, A. Muhlhoff, W. Gustin, and C. Schlunder, “Analysis of nbti degradation- and recovery-behavior based on ultra fast vt-measurements,” in *IRPS*, 2006.

[51] L. Gerrer, J. Ding, S. M. Amoroso, F. Adamu-Lema, R. Hussin, D. Reid, C. Millar, and A. Asenov, “Modelling RTN and BTI in nanoscale MOSFETs from device to circuit: A review,” *Micro. Rel.*, 2014.

[52] B. Sheu, D. Scharfetter, and P.-K. Ko et. al., “BSIM: Berkeley short-channel IGFET model for MOS transistors,” *JSSC*, 1987.

[53] M. V. Dunga, C.-H. Lin, A. M. Niknejad, and C. Hu, “Bsim-cmg: A compact model for multi-gate transistors,” in *FinFETs and Other Multi-Gate Transistors*, pp. 113–153, Springer, 2008.

[54] S. Ramey, A. Ashutosh, C. Auth, J. Clifford, M. Hattendorf, J. Hicks, R. James, A. Rahman, V. Sharma, A. S. Amour, and C. Wiegand, “Intrinsic transistor reliability improvements from 22nm tri-gate technology,” in *IRPS*, 2013.

[55] C. Prasad, L. Jiang, D. Singh, M. Agostinelli, C. Auth, P. Bai, T. Eiles, J. Hicks, C. Jan, K. Mistry, *et al.*, “Self-heat reliability considerations on intel’s 22nm tri-gate technology,” in *IRPS*, pp. 5D–1, IEEE, 2013.

[56] S. Liu, J. Wang, Y. Lu, D. Huang, C. Huang, W. Hsieh, J. Lee, Y. Tsai, J. Shih, Y.-H. Lee, *et al.*, “Self-heating effect in finfets and its impact on devices reliability characterization,” in *2014 IEEE International Reliability Physics Symposium*, pp. 4A–4, IEEE, 2014.

[57] O. Prakash, G. Pahwa, Y. S. Chauhan, and H. Amrouch, “Transistor self-heating: The rising challenge for semiconductor testing,” in *VTS*, IEEE, 2021.

[58] S. Makovejev, S. Olsen, and J. Raskin, “Rf extraction of self-heating effects in finfets,” *TED*, 2011.

[59] D. Jang, E. Bury, R. Ritzenthaler, M. G. Bardon, T. Chiarella, K. Miyaguchi, P. Raghavan, A. Mocuta, G. Groeseneken, A. Mercha, D. Verkest, and A. Thean, “Self-heating on bulk finfet from 14nm down to 7nm node,” in *IEDM*, 2015.

[60] V. Camargo, B. Kaczer, T. Grasser, and G. Wirth, “Circuit simulation of workload-dependent RTN and BTI based on trap kinetics,” *Micro. Rel.*, 2014.

[61] H. Amrouch, V. M. van Santen, and J. Henkel, “Interdependencies of Degradation Effects and their Impact on Computing,” *IEEE Des. Test.*, 2016.

[62] A. Gebregiorgis, S. Kiamehr, F. Obori, R. Bishnoi, and M. Tahoori, “A Cross-Layer Analysis of Soft Error, Aging and Process Variation in Near Threshold Computing", in Proceedings of Design,” in *DATE*, 2016.

[63] R. Saeidi, M. Sharifkhani, and K. Hajsadeghi, “Statistical analysis of read static noise margin for near/sub-threshold sram cell,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2014.

[64] K. Aadithya, A. Demir, and S. Venugopalan et. al., “Accurate Prediction of Random Telegraph Noise Effects in SRAMs and DRAMs,” *TCAD*, 2013.

[65] T. B. Tang, A. F. Murray, and S. Roy, “Methodology of statistical rts noise analysis with charge-carrier trapping models,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2010.

[66] S. Makovejev, S. Olsen, and J.-P. Raskin, “Rf extraction of self-heating effects in finfets,” *IEEE Transactions on Electron Devices*, vol. 58, no. 10, pp. 3335–3341, 2011.

[67] H. Jiang, S. Shin, X. Liu, X. Zhang, and M. A. Alam, “The impact of self-heating on hci reliability in high-performance digital circuits,” *LED*, 2017.

[68] K. Jenkins and K. Rim, “Measurement of the effect of self-heating in strained-silicon mosfets,” *IEEE Electron Device Letters*, vol. 23, no. 6, pp. 360–362, 2002.

[69] H. Amrouch, B. Khaleghi, and J. Henkel, “Optimizing temperature guardbands,” in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*, pp. 175–180, IEEE, 2017.

[70] A. Timar and M. Rencz, “Real-time heating and power characterization of cells in standard cell designs,” *Microelectronics Journal*, vol. 44, no. 11, pp. 977 – 985, 2013. Thermal investigations of integrated circuits in systems at THERMINIC’11.

[71] B. Tudor, J. Wang, C. Sun, Z. Chen, Z. Liao, R. Tan, W. Liu, and F. Lee, “MOSRA: An efficient and versatile MOS aging modeling and reliability analysis solution for 45nm and below,” in *ICSICT*, 2010.

[72] J. Chen, S. Wang, and M. Tehranipoor, “Critical-reliability Path Identification and Delay Analysis,” *J. Emerg. Technol. Comput. Syst.*, 2014.

[73] H. Amrouch, J. Martin-Martinez, V. M. van Santen, M. Moras, R. Rodriguez, M. Nafria, and J. Henkel, “Connecting the physical and application level towards grasping aging effects,” in *IRPS*, 2015.

[74] D. Lorenz, M. Barke, and U. Schlichtmann, “Aging analysis at gate and macro cell level,” in *Proceedings of the International Conference on Computer-Aided Design*, 2010.

[75] J. Fang and S. S. Sapatnekar, “The impact of bti variations on timing in digital logic circuits,” *IEEE Transactions on Device and Materials Reliability*, vol. 13, pp. 277–286, March 2013.

[76] N. Koppaetzky, M. Metzdorf, R. Eilers, D. Helms, and W. Nebel, “RT level timing modeling for aging prediction,” in *DATE*, 2016.

[77] Y. Lu, L. Shang, H. Zhou, H. Zhu, F. Yang, and X. Zeng, “Statistical Reliability Analysis Under Process Variation and Aging Effects,” in *DAC*, 2009.

[78] M. Ebrahimi, F. Oboril, S. Kiamehr, and M. B. Tahoori, “Aging-aware Logic Synthesis,” in *ICCAD*, 2013.

[79] J. Chen, S. Wang, and M. Tehranipoor, “Critical-reliability path identification and delay analysis,” *J. Emerg. Technol. Comput. Syst.*, vol. 10, pp. 12:1–12:21, Mar. 2014.

[80] K. C. Wu and D. Marculescu, “Aging-aware timing analysis and optimization considering path sensitization,” in *DATE*, 2011.

[81] W. Wang, Z. Wei, S. Yang, and Y. Cao, “An efficient method to identify critical gates under circuit aging,” in *ICCAD*, 2007.

[82] H. Amrouch, B. Khaleghi, A. Gerstlauer, and J. Henkel, “Reliability-Aware Design to Suppress Aging,” in *DAC*, 2016.

[83] F. Firouzi, S. Kiamehr, and M. B. Tahoori, “A Linear Programming Approach for Minimum NBTI Vector Selection,” in *GLSVLSI*, 2011.

[84] V. M. van Santen, H. Amrouch, and J. Henkel, “Modeling and mitigating time-dependent variability from the physical level to the circuit level,” *TCAS-I*, 2019.

[85] V. M. van Santen, H. Amrouch, and J. Henkel, “New worst-case timing for standard cells under aging effects,” *IEEE Transactions on Device and Materials Reliability (TDMR)*, 2019.

[86] C. Auth, A. Aliyarukunju, M. Asoro, D. Bergstrom, V. Bhagwat, J. Birdsall, N. Bisnik, M. Buehler, V. Chikarmane, G. Ding, *et al.*, “A 10nm high performance and low-power cmos technology featuring 3 rd generation finfet transistors, self-aligned quad patterning, contact over active gate and cobalt local interconnects,” in *2017 IEEE International Electron Devices Meeting (IEDM)*, pp. 29–1, IEEE, 2017.

[87] I. Agbo, M. Taouil, S. Hamdioui, P. Weckx, S. Cosemans, P. Raghavan, and F. Catthoor, “Comparative bti analysis for various sense amplifier designs,” in *2016 IEEE 19th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS)*, pp. 1–6, IEEE, 2016.

[88] D. Kraak, M. Taouil, I. Agbo, S. Hamdioui, P. Weckx, S. Cosemans, and F. Catthoor, “Impact and mitigation of sense amplifier aging degradation using realistic workloads,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 12, pp. 3464–3472, 2017.

[89] I. Agbo, M. Taouil, S. Hamdioui, P. Weckx, S. Cosemans, and F. Catthoor, “Bti analysis of sram write driver,” in *2015 10th International Design & Test Symposium (IDT)*, pp. 100–105, IEEE, 2015.

[90] Synopsys, “FineSim.” `https://www.synopsys.com/verification/ams-verification/circuit-simulation/finesim.html`.

[91] M. Rewienski, “A perspective on fast-spice simulation technology,” *Simulation and Verification of Electronic and Biological Systems*, 2011.

[92] Synopsys, “HSPICE.” `https://www.synopsys.com/verification/ams-verification/circuit-simulation/hspice.html`.

[93] M. Graphics, “ELDO.” `https://www.mentor.com/products/ic_nanometer_design/analog-mixed-signal-verification/eldo-platform`.

[94] Cadence, “Spectre Accelerates Parallel Simulator.” `https://www.cadence.com/content/cadence-www/global/en_US/home/tools/custom-ic-analog-rf-design/library-characterization/spectre-accelerated-parallel-simulator.html`.

[95] H. Vogt, M. Hendrix, and P. Nenzi, “Ngspice users manual version 27,” 2017.

[96] S. Hutchinson, E. Keiter, R. Hoekstra, H. Watts, A. Waters, T. Russo, R. Schells, S. Wix, and C. Bogdan, “The xyce parallel electronic simulator–an overview,” in *Parallel Computing: Advances and Current Issues*, pp. 165–172, World Scientific, 2002.

[97] E. R. Keiter, H. K. Thornquist, R. J. Hoekstra, T. V. Russo, R. L. Schiek, and E. L. Rankin, “Parallel transistor-level circuit simulation,” in *Simulation and Verification of Electronic and Biological Systems*, pp. 1–21, Springer, 2011.

[98] J. Power, A. Basu, J. Gu, S. Puthoor, B. M. Beckmann, M. D. Hill, S. K. Reinhardt, and D. A. Wood, “Heterogeneous system coherence for integrated cpu-gpu systems,” in *2013 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 457–467, IEEE, 2013.

[99] L. Han, X. Zhao, and Z. Feng, “Tinyspice: A parallel spice simulator on gpu for massively repeated small circuit simulations,” in *DAC*, 2013.

[100] R. E. Poore, “Gpu-accelerated time-domain circuit simulation,” in *Custom Integrated Circuits Conference, 2009. CICC’09. IEEE*, pp. 629–632, IEEE, 2009.

[101] L. Ren, X. Chen, Y. Wang, C. Zhang, and H. Yang, “Sparse lu factorization for parallel circuit simulation on gpu,” in *DAC*, 2012.

[102] X. Chen, L. Ren, Y. Wang, and H. Yang, “Gpu-accelerated sparse lu factorization for circuit simulation with performance modeling,” *TPDS*, 2015.

[103] K. He, S. X. D. Tan, H. Wang, and G. Shi, “Gpu-accelerated parallel sparse lu factorization method for fast circuit analysis,” *TVLSI*, 2016.

[104] F. Lannutti, P. Nenzi, and M. Olivieri, “Klu sparse direct linear solver implementation into ngspice,” in *MIXDES*, 2012.

[105] F. Lannutti, “What Does It Take to Accelerate SPICE on the GPU?,” in *GTC*, 2013.

[106] A. M. Bayoumi and Y. Y. Hanafy, “Massive parallelization of spice device model evaluation on gpu-based simd architectures,” in *IFMT*, 2008.

[107] F. Lannutti, “CUSPICE: The revolutionary NGSPICE on CUDA Platforms,” in *MOS-AK ESSDERC/ESSCIRC Workshop*, 2014.

[108] T. A. Davis and E. Palamadai Natarajan, “Algorithm 907: Klu, a direct sparse solver for circuit simulation problems,” *TOMS*, 2010.

[109] V. M. van Santen, H. Amrouch, and J. Henkel, “Reliability estimations of large circuits in massively-parallel gpu-spice,” in *IOLTS*, pp. 143–146, IEEE, 2018.

[110] R. H. Tu, E. Rosenbaum, W. Y. Chan, C. C. Li, E. Minami, K. Quader, P. K. Ko, and C. Hu, “Berkeley reliability tools-bert,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 10, pp. 1524–1534, 1993.

[111] J. B. Bernstein, M. Gurfinkel, X. Li, J. Walters, Y. Shapira, and M. Talmor, “Electronic circuit reliability modeling,” *Microelectronics Reliability*, vol. 46, no. 12, pp. 1957–1979, 2006.

[112] A. Toro-Frias, R. Castro-López, E. Roca, F. Fernández, J. Martin-Martinez, R. Rodriguez, and M. Nafria, “A fast and accurate reliability simulation method for analog circuits,” in *2015 International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, pp. 1–4, IEEE, 2015.

[113] V. M. van Santen, J. Martin-Martinez, H. Amrouch, M. M. Nafria, and J. Henkel, “Reliability in super- and near-threshold computing: A unified model of rtn, bti, and pv,” *TCAS-I*, 2018.

[114] S. Jain, S. Khare, S. Yada, V. Ambili, P. Salihundam, S. Ramani, S. Muthukumar, M. Srinivasan, A. Kumar, S. K. Gb, *et al.*, “A 280mV-to-1.2 V wide-operating-range IA-32 processor in 32nm CMOS,” in *ISSCC*, 2012.

[115] G. Ruhl, S. Dighe, S. Jain, S. Khare, and S. R. Vangal, “IA-32 processor with a wide-voltage-operating range in 32-nm CMOS,” *IEEE Micro*, 2013.

[116] J. Myers, A. Savanth, R. Gaddh, D. Howard, P. Prabhat, and D. Flynn, “A Subthreshold ARM Cortex-M0+ Subsystem in 65 nm CMOS for WSN Applications with 14 Power Domains, 10T SRAM, and Integrated Voltage Regulator,” *JSSC*, 2016.

[117] H. Amrouch, S. Mishra, V. van Santen, S. Mahapatra, and J. Henkel, “Impact of BTI on dynamic and static power: From the physical to circuit level,” in *IRPS*, 2017.

[118] V. M. van Santen, H. Amrouch, J. Martin-Martinez, M. Nafria, and J. Henkel, “Designing Guardbands for Instantaneous Aging Effects,” in *DAC*, 2016.

[119] S. M. Amoroso, C. M. Compagnoni, A. Ghetti, L. Gerrer, A. S. Spinelli, A. L. Lacaita, and A. Asenov, “Investigation of the RTN Distribution of Nanoscale MOS Devices From Subthreshold to On-State,” *IEEE Electron Device Letters*, 2013.

[120] K. Takeuchi, T. Nagumo, S. Yokogawa, K. Imai, and Y. Hayashi, “Single-charge-based modeling of transistor characteristics fluctuations based on statistical measurement of RTN amplitude,” in *VLSIT*, 2009.

[121] V. M. van Santen, H. Amrouch, N. Parihar, S. Mahapatra, and J. Henkel, “Aging-aware Voltage Scaling,” in *DATE*, 2016.

[122] A. Asenov, S. M. Amoroso, and L. Gerrer, “Progress in the simulation of time dependent statistical variability in nano cmos transistors,” in *2014 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, 2014.

[123] S. Realov and K. L. Shepard, “Analysis of Random Telegraph Noise in 45-nm CMOS Using On-Chip Characterization System,” *TED*, 2013.

[124] J. Martin-Martinez, J. Diaz, R. Rodriguez, M. Nafria, and X. Aymerich, “New Weighted Time Lag Method for the Analysis of Random Telegraph Signals,” *EDL*, 2014.

[125] V. Gupta and M. Anis, “Statistical Design of the 6T SRAM Bit Cell,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2010.

[126] X. Wang, A. R. Brown, B. Cheng, and A. Asenov, “Statistical variability and reliability in nanoscale FinFETs,” in *IEDM*, 2011.

[127] C.-Y. Su, M. Armstrong, L. Jiang, S. Kumar, C. Landon, S. Liu, I. Meric, K. Park, L. Paulson, K. Phoa, *et al.*, “Transistor reliability characterization and modeling of the 22ffl finfet technology,” in *Reliability Physics Symposium (IRPS), 2018 IEEE International*, pp. 6F–8, IEEE, 2018.

[128] S. K. Saha, “Modeling process variability in scaled cmos technology,” *IEEE Design Test of Computers*, 2010.

[129] M. Alioto, G. Palumbo, and M. Pennisi, “Understanding the effect of process variations on the delay of static and domino logic,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2010.

[130] D. Sengupta and S. S. Sapatnekar, “Predicting circuit aging using ring oscillators,” in *ASP-DAC*, 2014.

[131] X. Jiao, Y. Jiang, A. Rahimi, and R. K. Gupta, “Slot: A supervised learning model to predict dynamic timing errors of functional units,” in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, pp. 1183–1188, March 2017.

[132] S. E. Rauch, “Review and reexamination of reliability effects related to nbti-induced statistical variations,” *TDMR*, 2007.

[133] J. Fang and S. S. Sapatnekar, “The impact of BTI variations on timing in digital logic circuits,” *IEEE Transactions on Device and Materials Reliability*, 2013.

[134] S. Kiamehr, P. Weckx, M. Tahoori, B. Kaczer, H. Kukner, P. Raghavan, G. Groeseneken, and F. Catthoor, “The impact of process variation and stochastic aging in nanoscale vlsi,” in *IRPS*, 2016.

[135] V. B. Kleeberger, P. R. Maier, and U. Schlichtmann, “Workload- and instruction-aware timing analysis - the missing link between technology and system-level resilience,” in *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, June 2014.

[136] J. Charles, P. Jassi, N. Ananth, A. Sadat, and A. Fedorova, “Evaluation of the Intel Core i7 Turbo Boost feature,” in *IISWC*, 2009.

[137] S. I. Initiative *et al.*, “Nangate open cell library,” *Available: http://www.si2.org/openeda.si2.org/projects/nangatelib*, 2011.

[138] Synopsys, “SiliconSmart.” `https://www.synopsys.com/implementation-and-signoff/signoff/siliconsmart.html`.

[139] K. Asanović, R. Avizienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelevitz, S. Karandikar, B. Keller, D. Kim, J. Koenig, Y. Lee, E. Love, M. Maas, A. Magyar, H. Mao, M. Moreto, A. Ou, D. A. Patterson, B. Richards, C. Schmidt, S. Twigg, H. Vo, and A. Waterman, “The rocket chip generator,” Tech. Rep. UCB/EECS-2016-17, EECS Department, University of California, Berkeley, 2016.

[140] F. Corno, M. S. Reorda, and G. Squillero, “Rt-level itc’99 benchmarks and first atpg results,” *IEEE Design Test of Computers*, 2000.

[141] V. M. van Santen, H. Amrouch, P. Sharma, and J. Henkel, “On the workload dependence of self-heating in finfet circuits,” *TCAS-II*, 2019.

[142] W. Ahn, S. Shin, C. Jiang, H. Jiang, M. Wahab, and M. Alam, “Integrated modeling of Self-heating of confined geometry (FinFET, NWFET, and NSHFET) transistors and its implications for the reliability of sub-20nm modern integrated circuits,” *Microelectronics Reliability*, vol. 81, pp. 262 – 273, 2018.

[143] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, “Asap7: A 7-nm finfet predictive process design kit,” *Microelectronics Journal*, vol. 53, pp. 105 – 115, 2016.

[144] M. Graphics, “Modelsim,” 2007.

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

[146] H. Reisinger, U. Brunner, W. Heinrigs, W. Gustin, and C. Schlunder, “A Comparison of Fast Methods for Measuring NBTI Degradation,” *TDMR*, 2007.

[147] T. Grasser, B. Kaczer, H. Reisinger, P.-J. Wagner, and M. Toledano-Luque, “On the frequency dependence of the bias temperature instability,” in *IRPS*, 2012.

[148] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, “The Gem5 Simulator,” *SIGARCH Comput. Archit. News*, 2011.

[149] S. Kumar, C. Kim, and S. Sapatnekar, “Impact of nbti on sram read stability and design for reliability,” in *ISQED*, 2006.

[150] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, “The McPAT Framework for Multicore and Manycore Architectures: Simultaneously Modeling Power, Area, and Timing,” *ACM Trans. Archit. Code Optim.*, 2013.

[151] E. Gunadi, A. A. Sinkar, N. S. Kim, and M. H. Lipasti, “Combating Aging with the Colt Duty Cycle Equalizer,” in *MICRO*, 2010.

[152] V. M. van Santen, J. Diaz-Fortuny, H. Amrouch, J. Martin-Martinez, R. Rodriguez, R. Castro-Lopez, E. Roca, F. V. Fernandez, J. Henkel, and M. Nafria, “Weighted time lag plot defect parameter extraction and gpu-based bti modeling for bti variability,” in *2018 IEEE International Reliability Physics Symposium (IRPS)*, pp. P–CR, IEEE, 2018.

[153] C.-M.-M. O. Ong, *Dynamic simulation of electric machinery using Matlab/Simulink.* Prentice-Hall PTR,, 1998.

[154] S. Novak, C. Parker, D. Becher, M. Liu, M. Agostinelli, M. Chahal, P. Packan, P. Nayak, S. Ramey, and S. Natarajan, “Transistor aging and reliability in 14nm tri-gate technology,” in *IRPS*, 2015.

[155] D. Rodopoulos, S. 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 *ICICDT*, 2011.

[156] ORSoC AB, “OpenCores.” https://opencores.org/.

[157] Synopsys, “ASIP Designer.” `https://www.synopsys.com/dw/ipdir.php?ds=asip-designer`.

[158] K. Asanović, R. Avizienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelevitz, S. Karandikar, B. Keller, D. Kim, J. Koenig, Y. Lee, E. Love, M. Maas, A. Magyar, H. Mao, M. Moreto, A. Ou, D. A. Patterson, B. Richards, C. Schmidt, S. Twigg, H. Vo, and A. Waterman, “The rocket chip generator,” Tech. Rep. UCB/EECS-2016-17, EECS Department, University of California, Berkeley, 2016.

[159] C. Celio, D. A. Patterson, and K. Asanović, "The berkeley out-of-order machine (boom): An industry-competitive, synthesizable, parameterized risc-v processor," Tech. Rep. UCB/EECS-2015-167, EECS Department, University of California, Berkeley, 2015.

[160] V. M. van Santen, L. Schillinger, and H. Amrouch, "Self-heating effects from transistors to gates," in *2021 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, pp. 1–4, IEEE, 2021.

[161] H. Amrouch, V. M. van Santen, O. Prakash, H. Kattan, S. Salamin, S. Thomann, and J. Henkel, "Reliability challenges with self-heating and aging in finfet technology," in *IOLTS*, pp. 68–71, IEEE, 2019.

[162] S. Mishra, H. Amrouch, J. Joe, C. K. Dabhi, K. Thakor, Y. S. Chauhan, J. Henkel, and S. Mahapatra, "A simulation study of nbti impact on 14-nm node finfet technology for logic applications: Device degradation to circuit-level interaction," *TED*, 2019.

[163] L. Schilliger, "Evaluating the impact of self-heating on neural network accelerators," *Master Thesis, KIT, Germany*, 2020.

[164] V. M. van Santen, F. L. F. Diep, J. Henkel, and H. Amrouch, "Massively parallel circuit setup in gpu-spice," *IEEE Transactions on Computers*, 2020.

[165] A. Vladimirescu, *The SPICE book.* John Wiley & Sons, Inc., 1994.

[166] L. Amarú, P.-E. Gaillardon, and G. De Micheli, "The epfl combinational benchmark suite," in *IWLS*, 2015.

[167] F. Brglez, "A neutral netlist of 10 combinational benchmark design and a special translator in fortran," in *ISCAS*, 1985.

[168] M. Kamon, M. J. Tsuk, and J. K. White, "Fasthenry: A multipole-accelerated 3-d inductance extraction program," *IEEE Transactions on Microwave theory and techniques*, vol. 42, no. 9, pp. 1750–1758, 1994.

[169] M. Owens and G. Allen, *SQLite.* Springer, 2010.

[170] L. Ha, J. Krüger, and C. T. Silva, "Fast four-way parallel radix sorting on gpus," in *Computer Graphics Forum*, vol. 28, Wiley Online Library, 2009.

[171] H. Nguyen, *Gpu Gems 3.* Addison-Wesley Professional, first ed., 2007.

[172] G. E. Blelloch, "Prefix sums and their applications," tech. rep., Technical Report CMU-CS-90-190, School of Computer Science, Carnegie Mellon University, 1990.

[173] N. Satish, M. Harris, and M. Garland, "Designing efficient sorting algorithms for manycore gpus," in *IPDPS*, pp. 1–10, IEEE, 2009.

[174] V. M. van Santen, S. Thomann, C. Pasupuleti, P. R. Genssler, N. Gangwar, U. Sharma, J. Henkel, S. Mahapatra, and H. Amrouch, "Bti and hcd degradation in a complete 32× 64 bit sram array–including sense amplifiers and write drivers–under processor activity," in *IRPS*, pp. 1–7, IEEE, 2020.

[175] S. Y. Wu, C. Y. Lin, M. C. Chiang, J. J. Liaw, J. Y. Cheng, S. H. Yang, S. Z. Chang, M. Liang, T. Miyashita, C. H. Tsai, C. H. Chang, V. S. Chang, Y. K. Wu, J. H. Chen, H. F. Chen, S. Y. Chang, K. H. Pan, R. F. Tsui, C. H. Yao, K. C. Ting, T. Yamamoto, H. T. Huang, T. L. Lee, C. H. Lee, W. Chang, H. M. Lee, C. C. Chen, T. Chang, R. Chen, Y. H. Chiu, M. H. Tsai, S. M. Jang, K. S. Chen, and Y. Ku, "An enhanced 16nm CMOS technology featuring 2nd generation FinFET transistors and advanced Cu/low-k interconnect for low power and high performance applications," in *IEDM*, 2014.

[176] E. Burton, G. Schrom, F. Paillet, J. Douglas, W. J. Lambert, K. Radhakrishnan, M. J. Hill, *et al.*, "FIVR—Fully integrated voltage regulators on 4th generation Intel® Core™ SoCs," in *APEC*, 2014.

[177] S. Satapathy, W. H. Choi, X. Wang, and C. Kim, "A revolving reference odometer circuit for BTI-induced frequency fluctuation measurements under fast DVFS transients," in *IRPS*, 2015.

[178] C. Zhou, X. Wang, W. Xu, Y. Zhu, V. Reddi, and C. Kim, “Estimation of instantaneous frequency fluctuation in a fast DVFS environment using an empirical BTI stress-relaxation model,” in *IRPS*, 2014.

[179] X. Li, J. Qin, and J. Bernstein, “Compact Modeling of MOSFET Wearout Mechanisms for Circuit-Reliability Simulation,” *TDMR*, 2008.

[180] T.-B. Chan, J. Sartori, P. Gupta, and R. Kumar, “On the efficacy of nbti mitigation techniques,” in *DATE*, 2011.

[181] J. Chen, S. Wang, and M. Tehranipoor, “Efficient Selection and Analysis of Critical-reliability Paths and Gates,” in *GLSVLSI*, 2012.

[182] C. R. Lefurgy, A. J. Drake, M. S. Floyd, M. S. Allen-Ware, B. Brock, J. A. Tierno, and J. B. Carter, “Active Management of Timing Guardband to Save Energy in POWER7,” in *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-44, 2011.

[183] J. Henkel, H. Khdr, S. Pagani, and M. Shafique, “New trends in dark silicon,” in *DAC*, 2015.

[184] A. Kerber and T. Nigam, “Challenges in the characterization and modeling of BTI induced variability in metal gate / High-k CMOS technologies,” in *IRPS*, 2013.

[185] C. Bienia, S. Kumar, J. P. Singh, and K. Li, “The PARSEC Benchmark Suite: Characterization and Architectural Implications,” in *PACT*, pp. 72–81, 2008.

[186] J. L. Henning, “SPEC CPU2006 Benchmark Descriptions,” *SIGARCH Comput. Archit. News*, 2006.

[187] H. Amrouch, G. Pahwa, A. D. Gaidhane, J. Henkel, and Y. S. Chauhan, “Negative capacitance transistor to address the fundamental limitations in technology scaling: Processor performance,” *IEEE Access*, 2018.

[188] V. M. van Santen, S. Thomann, Y. S. Chauchan, J. Henkel, and H. Amrouch, “Reliability-driven voltage optimization for ncfet-based sram memory banks,” in *2021 IEEE 39th VLSI Test Symposium (VTS)*, pp. 1–7, IEEE, 2021.