



National Technical University of Athens
School of Electrical & Computer Engineering
Department of Computer Science

KU LEUVEN

Katholieke Universiteit Leuven
Faculty of Engineering Science
Arenberg Doctoral School

Modeling and Mitigation of Parametric Time-Dependent Variability in Digital Systems

Dimitrios Rodopoulos

Supervisor:
Prof. Dimitrios Soudris (NTUA)

Co-Supervisor:
Prof. Francky Catthoor (KUL)

Dissertation presented in partial
fulfillment of the requirements for the
degree of Doctor in Engineering
Science: Electrical Engineering

May 2016

Modeling and Mitigation of Parametric Time-Dependent Variability in Digital Systems

Dimitrios RODOPOULOS

Examination committee:

Prof. Dimitrios Soudris (NTUA), supervisor
Prof. Francky Catthoor (KUL), co-supervisor
Prof. Kiamal Pekmestzi (NTUA)
Prof. Guido Groeseneken (KUL)
Prof. Theodora Varvarigou (NTUA)
Prof. Yiannakis Sazeides (UCY)
Prof. Dimitrios Gizopoulos (UOA)
Dr. Ben Kaczer (imec)

Dissertation presented in partial
fulfillment of the requirements for
the degree of Doctor in Engineering
Science: Electrical Engineering

May 2016

This work is partially supported by European Commission FP7 projects 248789-TRAMS and 612069-HARPA.

The Intel® Singel Chip Cloud Computer (SCC) and its Intel® Xeon® Management Concole Personal Computer (MCPC) are available under a material trasnfer agreement between Intel Coropration and ICCS/NTUA.

Content that is reused from publications that the author has (co-)authored (excerpts, figures, tables, etc.) is under copyright with the respective paper publishers (IEEE, ACM, etc) and is cited accordingly in the current text. Content that reused from third-party publications appears with the appropriate copyright note. Reuse of such content by any interested party requires the publishers' prior consent, according to the applicable copyright policies. Content that has not been published before is copyrighted jointly as follows:

© 2016 NTUA – School of Electrical and Computer Engineering

© 2016 KU Leuven – Faculty of Engineering Science

Dimitrios Rodopoulos, 23 Karvela Street, Aghia Paraskevi, 15342 Athens, Greece
drodo@microlab.ntua.gr



ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Modeling and Mitigation of Parametric Time-Dependent Variability in Digital Systems

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Δημήτριος, Α. Ροδόπουλος

Συμβουλευτική Επιτροπή : Δημήτριος Σούντρης

Francky Catthoor

Κιαμάλ Πεκμεστζή

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 18^η Μαΐου 2016.

.....
Δημήτριος Σούντρης
Αν. Καθηγητής ΕΜΠ

.....
Francky Catthoor
Professor KUL

.....
Κιαμάλ Πεκμεστζή
Καθηγητής ΕΜΠ

.....
Prof. Guido Groeseneken
Professor KUL

.....
Θεοδώρα Βαρβαρίγου
Καθηγήτρια ΕΜΠ

.....
Γιαννάκης Σαζείδης
Professor UCY

.....
Δημήτριος Γκιζόπουλος
Καθηγητής ΕΚΠΑ

Αθήνα, Μάιος 2016

.....

Δημήτριος, Α. Ροδόπουλος

Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

*In memoriam,
Aikaterini Kapelari*

Acknowledgments

The current text is the result of an academic process that has been split between two universities: NTUA in Greece and KUL in Belgium. In itself, this endeavor has been both interesting and challenging. During my engagement towards the current thesis, certain people contributed in creating an ecosystem that enabled this work's fortunate completion. I would like to hereby acknowledge their names and sincerely thank them.

Prof. Dimitrios Soudris, supervisor from the NTUA side, is gratefully acknowledged for his continuous support of this work. It is with great wonder that I have observed him operate within the great volatility of the Greek education system, while still carrying his team forward. His trust during the course of this work is greatly appreciated. Prof. Francky Catthoor, supervisor from the KUL side, has been an excellent educator and huge enabler for this work. His way of thinking and his views on research strategy and collaboration principles have been highly efficient and the way they have been communicated has been an excellent academic experience for me. It has been an honor and a privilege to work with him in the course of the current thesis.

I would like to thank Prof. Yannakis Sazeides (UCY, Cyprus) for participating in this work's jury and especially for hosting me at his group in 2015. Prof. Kiamal Pekmestzi (NTUA, Greece), Prof. Guido Groeseneken (KUL, Belgium), Prof. Theodora Varvargiou (NTUA, Greece), and Prof. Dimitrios Gizopoulos (UoA, Greece) are especially acknowledged for their participation in this work's jury. Dr. Ben Kaczer (imec, Belgium) has been a constant point of reference for the BTI/RTN work, since my Master Thesis way back to 2010, and I thank him for always sharing his expertise and for joining my jury. I feel very honored to have this amount of talent review my work and would like to thank them all for their insight on my text.

I would also like to thank Dr. Praveen Raghavan, Dr. Antonis Papanikolaou, and Prof. Chrysostomos Nicopoulos for sharing their views on reliability topics.

Their input has always been timely and crucial for the progress of this work. Thanks to the green team members Mohamed Sabry and Georgia Psychou.

I have greatly appreciated the discussions with Christos, Mario and Robert (Erasmus Medical Center, Netherlands) on brain modeling, computational neuroscience and other topics. Great many thanks to my office mates at NTUA and especially JoKo, Harry, Maro, Dimitris, Sotiris, Nikos, and Dionysis. I am also very grateful to have collaborated with many great people during my imec visits: Simone, Pieter, Halil, Prashant, and Matthias. Many thanks to George, Panayota, Lorena, Panayotis, Zacharias, and Kypros for making me feel at home during my three months at UCY. Big ups to John and Vaggelis for joining me in academic wonderment since our first semester as undergrads at NTUA.

During the course of my PhD, I have been fortunate in designing and supervising a number of NTUA thesis projects. I have found this as one of the most rewarding experiences of my time as a PhD candidate. Along these lines, I would like to thank Dimitris, Greg, Giorgos C., Alex, Michalis, Andreas, Christiana, Apostolis, Giorgos P., and Sofia. Each and everyone of them has contributed distinctly in making me a better, researcher, engineer and supervisor. I have been very glad in collaborating with them in pursuit of their studies and I am proud of their progress.

I'd like to thank my friends and especially Christos, Valia, Themistoklis, Stratos, Nick, Anna and Themis for their support. Special thanks to Andriana for being great and for tolerating me at times. I am also glad to have raced with great captains and crew mates, especially Polyanna, Sotiris, Timos, Gary, Katerina, Mr. Tsamis, Michalis, Stelios, and Marilena.

Last, but definitely not least, I would like to thank my father, Anastassios Rodopoulos and my brother Aristeidis Rodopoulos for their support. I have found that even in the face of overwhelming challenges, they have been solid supporters, advisors, and friends; I feel very fortunate to have them in my life.

Extended Abstract

Current and future semiconductor technology nodes, bring about a variety of challenges that pertain to the reliability and dependability of digital integrated systems. Compounds, such as high- κ materials in the transistor gate stack, tend to intensify the time-zero and time-dependent variability of transistors. A case, can certainly be made for the phenomena like Bias Temperature Instability (BTI) and Random Telegraph Noise (RTN). The use of such modern materials is also coupled to an aggressive downscaling trend, which further amplifies matter discretization within the transistor's channel. These are two major manufacturing trends that give rise to time-dependent variability in integrated digital systems. It stands to reason, that as the materials become more variable, the error rates experienced at the circuit- or even system-level are also intensified. Coupled to increasing integrated circuit functionality (namely, the "More than Moore" trend), it is reasonable to expect that future digital chips will be exhibiting reliability profiles that vary across the chip's lifetime.

The goal of the research presented in the current text is to study the above observations both regarding the modeling and the mitigation of time-dependent variability. Target systems, naturally, constitute digital integrated circuits, such as processors and memories thereof. The current research contributes with specific reductions to practice that aim to confirm existing techniques and develop novel insight into the modeling and mitigation of time-dependent variability. As such, the current text is broadly split into two major parts.

The modeling part starts with the reiteration of atomistic modeling concepts, which are very useful in the analysis of phenomena like BTI and RTN, especially for deca-nanometer transistors. As a result, the complexity of atomistic models for integrated circuit aging analysis is made abundantly clear. In order to alleviate the complexity of circuit reliability analysis across the system lifetime, the current research has formulated the concept of the Compact Digital Waveform (CDW). This format targets regions of circuit operation that are similar from a waveform point of view (e.g. similar frequency or duty

cycle) and abstracts them to a single point. This enables striding over circuit lifetime intervals, while retaining key features of atomistic reliability models (e.g. workload dependency).

Still on the modeling front, the current research additionally contributes towards exposing transistor variability information to the architecture level. The metric of choice in this case is the component failure probability (P_{fail}). This is typically required by system designers to appropriately provision their systems with appropriate disabling or correction mechanisms. In order to derive the P_{fail} , this text features the Most Probable Failure Point (MPFP) method, which is applied for the cases of BTI/RTN variability. Two modeling approaches are used for these aging phenomena and observations are drawn regarding the importance of accurate standard deviation capturing (regarding the threshold voltage variability). A statistical reformulation of the MPFP concept is also presented towards handling standard cells, apart from memory components.

The failure of system components has traditionally been triggering some sort of reaction from the system, at least as far as detectable errors are concerned. Academia and industry have been using the term Reliability, Availability and Serviceability (RAS) in order to refer to such techniques. Their invocation is strictly coupled to the rate of errors that appears at the circuit level and typically comes at a measurable performance cost.

On the mitigation side, the starting point of the current research is a simple, rollback-based RAS technique that aims to recover a system from transient errors. This concept has been implemented on a research-grade many-core platform, in the data plane of which errors are injected at a user-defined rate. The rollbacks make sure the running application is brought to an earlier correct state, so that execution can continue. This fail/stop model, however granular, is creating a measurable drawback in the timely execution of the running application. It is reasonable to explore that a portion of injected errors may not be corrected in order to reduce this performance overhead. In view of this trade-off, the current work explores the trade-off between application correctness, performance and energy budget, in search of the optimal operation points.

The final milestone of the current work is a generally applicable solution for the problem of dependable performance, in view of temporal RAS overheads such as the one illustrated above. Towards that direction, the issue of dependable performance is formulated from scratch and a control-theoretic solution is proposed. More specifically, a PID controller is used in order to manipulate the frequency of a processor, within which RAS schemes are invoked at the price of additional clock cycles. The concept is verified within the current research, both with simple simulations and on a real processing platform.

Εκτεταμένη Περίληψη

Οι σύγχρονες και μελλοντικές τεχνολογίες πυριτίου φέρνουν στο προσκήνιο μία ποικιλία προκλήσεων αναφορικά με την αξιοπιστία των ψηφιακών ολοκληρωμένων κυκλωμάτων. Δομικά στοιχεία, όπως για παράδειγμα τα υλικά με υψηλή διηλεκτρική σταθερά που χρησιμοποιούνται στην πύλη των τρανζίστορ, τείνουν να ενισχύουν την στατική αλλά και χρονικά εξαρτώμενη τους μεταβλητότητα. Αντιπροσωπευτικό παράδειγμα συνιστούν τα φαινόμενα Bias Temperature Instability (BTI) και Random Telegraph Noise (RTN). Η χρήση τέτοιων υλικών υψηλής τεχνολογίας είναι άρρηκτα συνδεδεμένη με την τάση σμήκρυνσης των ολοκληρωμένων κυκλωμάτων, η οποία επιπρόσθετα ενισχύει την διακριτοποίηση της ύλης μέσα στο κανάλι των τρανζίστορ. Αυτές είναι οι δύο κατασκευαστικές τάσεις που προάγουν την στατική και χρονικά εξαρτώμενη τους μεταβλητότητα των ψηφιακών ολοκληρωμένων κυκλωμάτων. Είναι λογικό ότι όσο ενισχύεται η μεταβλητότητα των υλικών, τόσο θα αυξάνονται οι ρυθμοί σφαλμάτων σε επίπεδο κυκλώματος και συστήματος. Σε συνδυασμό με τις συνεχώς διευρυνόμενες λειτουργίες ολοκληρωμένων κυκλωμάτων (την αποκαλούμενη τάση “More than Moore”), είναι λογικό να αναμένουμε ότι μελλοντικά ολοκληρωμένα κυκλώματα θα παρουσιάζουν αξιοπιστία που μεταβάλλεται κατά τη διάρκεια της ‘ζωής’ του ολοκληρωμένου κυκλώματος.

Ο στόχος της παρούσας έρευνας είναι να μελετήσει τις παραπάνω παρατηρήσεις, για την μοντελοποίηση αλλά και την αντιμετώπιση της στατικής και χρονικά εξαρτώμενης μεταβλητότητας. Στοχευόμενα συστήματα είναι, προφανώς, ψηφιακά ολοκληρωμένα συστήματα, όπως επεξεργαστές και μνήμες αυτών. Η παρούσα έρευνα παρουσιάζει πρακτικές εφαρμογές που επιβεβαιώνουν και επεκτείνουν τεχνικές για την μοντελοποίηση και αντιμετώπιση της στατικής και χρονικά εξαρτώμενης μεταβλητότητας ολοκληρωμένων συστημάτων. Συνεπώς, το παρόν κείμενο είναι διαχωρισμένο σε δύο κύρια μέρη.

Το πρώτο μέρος αφορά τη μοντελοποίηση και ξεκινά με μία επανάληψη βασικών εννοιών ατομιστικής προσομοίωσης, που χρησιμοποιείται εκτεταμένα για φαινόμενα όπως BTI και RTN, ειδικά για τρανζίστορ σε διαστάσεις δεκάδων

νανομέτρων. Κατά συνέπεια, εκτίθεται η πολυπλοκότητα των ατομιστικών μοντέλων για ανάλυση γήρανσης ολοκληρωμένων κυκλωμάτων. Η παρούσα έρευνα αντιμετωπίζει την υψηλή πολυπλοκότητα της μοντελοποίησης αξιοπιστίας προτείνοντας την έννοια της «συμπαγούς ψηφιακής κυματομορφής» (Compact Digital Waveform – CDW). Αυτό το πρότυπο απεικόνισης κυματομορφών απομονώνει διαδοχικές περιοχές ενός σήματος που έχουν παρόμοια μορφή (π.χ. βάσει συχνότητας) και τις συνοψίζει σε ένα σημείο. Αυτό επιτρέπει στην προσομοίωση να υπερπηδά διαστήματα ζωής του κυκλώματος, ενώ διατηρούνται τα χαρακτηριστικά του σήματος τα οποία είναι αναγκαία για την ορθότητα του ατομιστικού μοντέλου (εξάρτηση από την λειτουργία του κυκλώματος).

Παραμένοντας στο μέτωπο της μοντελοποίησης, η παρούσα έρευνα συνεισφέρει επιπρόσθετα στην έκθεση της μεταβλητότητας των τρανζιστορ στο επίπεδο αρχιτεκτονικής. Η μετρική που επιλέγεται είναι η πιθανότητα αποτυχίας του εκάστοτε υποσυστήματος (P_{fail}). Τέτοιες μετρικές είναι απαραίτητες στους αρχιτέκτονες υπολογιστών, έτσι ώστε να μπορούν να λάβουν τα κατάλληλα σχεδιαστικά μέτρα απενεργοποίησης ή διόρθωσης σφαλμάτων εντός της αρχιτεκτονικής. Για τον υπολογισμό του P_{fail} χρησιμοποιείται η τεχνική Most Probable Failure Point (MPFP), η οποία εφαρμόζεται στο παρόν κείμενο για την περίπτωση μεταβλητότητας λόγω BTI/RTN. Με βάση δύο μοντέλα μεταβλητότητας, αποδεικνύεται η σημαντικότητα του σωστού υπολογισμού τυπικής απόκλισης της μεταβολής τάσης κατωφλίου για τον υπολογισμό του P_{fail} .

Η αποτυχία ορισμένων υποσυστημάτων σε ένα ολοκληρωμένο κύκλωμα συνήθως προκαλεί κάποια αντίδραση από εσωτερικούς μηχανισμούς, τουλάχιστον όσον αφορά ανιχνεύσιμα σφάλματα στο υλικό. Ο όρος «Αξιοπιστία, Διαθεσιμότητα και Επισκευασιμότητα» (Reliability, Availability and Serviceability – RAS) χρησιμοποιείται στη βιομηχανία και ακαδημία για να περιγράψει τέτοιες τεχνικές. Η χρήση τέτοιων τεχνικών αντιμετώπισης συνδέεται άρριχτα με τον ρυθμό εμφάνισης σφαλμάτων σε κυκλωματικό επίπεδο.

Το δεύτερο μέρος της παρούσας έρευνας αφορά την αντιμετώπιση της χρονικά εξαρτώμενης μεταβλητότητας. Σημείο έναρξης είναι μία απλή RAS τεχνική που χρησιμοποιείται για την αντιμετώπιση παροδικών σφαλμάτων στο υλικό και βασίζεται σε επαναφορά εκτέλεσης. Η τεχνική αυτή υλοποιείται σε μία ερευνητική πλατφόρμα πολλών πυρήνων, στο χώρο δεδομένων της οποίας εισάγονται σφάλματα με έναν προκαθορισμένο ρυθμό. Οι επαναφορές εκτέλεσης εγγυόνται ότι η εκτέλεση της εφαρμογής επανέρχεται σε σωστή αρχιτεκτονική κατάσταση έτσι ώστε να μπορεί να συνεχιστεί σωστά. Ωστόσο, αυτό το μοντέλο διακοπής/εκτέλεσης, όσο λεπτομερείς και να είναι οι επαναφορές, επιβαρύνει την επίδοση της εκτέλεσης προσθέτοντας ένα χρονικό κόστος στην εκτέλεση. Είναι ιδιαίτερα ενδιαφέρον να διερευνήσουμε κατά πόσο ένα υποσύνολο των σφαλμάτων μπορεί να αγνοηθεί, με σκοπό να μειώσουμε την χρονική επίδραση των επαναφορών εκτέλεσης. Στην παρούσα έρευνα, παρουσιάζουμε

αναλυτικά τον συμβιβασμό μεταξύ της ορθότητας εκτέλεσης, της επίδοσης και της καταναλυσόμενης ενέργειας με σκοπό την απομόνωση του βέλτιστου συνδυασμού.

Η τελευταία συνεισφορά της παρούσας έρευνας είναι η γενίκευση του προβλήματος αξιόπιστης επίδοσης ψηφιακών συστημάτων, ειδικότερα υπό την επίδραση τεχνικών RAS. Σε αυτή τη κατεύθυνση, παρουσιάζεται η μαθηματική θεμελίωση του προβλήματος της αξιόπιστης επίδοσης και προτείνεται μία τεχνική αυτομάτου ελέγχου για την αντιμετώπισης της πρώτης. Ποιο συγκεκριμένα, ένας ελεγκτής PID χρησιμοποιείται για να διαμορφώσει την συχνότητα ενός επεξεργαστή κατά τη διάρκεια της εκτέλεσης. Η αποτελεσματικότητα αυτής της τεχνικής παρουσιάζεται τόσο μέσα από προσομοιώσεις όσο και μέσα από υλοποίηση λογισμικού που εκτελείται σε πραγματική πλατφόρμα.

Uitgebreide Samenvatting

De huidige en toekomstige evolutie van de procestechnologie, leidt tot een verscheidenheid van uitdagingen die betrekking hebben op de betrouwbaarheid en de veilige toepassing van digitale geïntegreerde systemen. Het gebruik van nieuwe materialen zoals hoge- κ materialen tussen de poort en het kanaal van de transistor versterkt de initiele en tijdsafhankelijke variabiliteit van de transistoren. Dit verhoogt de reeds aanwezige Bias Temperatuur Instabiliteit (BTI) en Random Telegraph Noise (RTN). Het gebruik van dergelijke moderne materialen wordt eveneens gekoppeld aan een trend naar agressieve schaalverkleining die de discretisatie in het transistorkanaal verder versterkt. Dit zijn twee belangrijke trends die leiden tot een verhoogde tijdsafhankelijke variabiliteit in geïntegreerde digitale systemen. Het spreekt voor zich, dat wanneer de materialen meer variabiliteit vertonen, ook de foutmarges op het circuit- of zelfs systeem-niveau groter worden. Gekoppeld aan de toenemende functionaliteit van geïntegreerde schakelingen (de “ More than Moore ” trend), is het redelijk om te verwachten dat de betrouwbaarheid in toekomstige digitale chips onder druk komt and dat deze zal fluctueren over de levensduur van de chip.

Het doel van het huidige werk is om de bovenstaande observaties te bestuderen zowel ten aanzien van de modellering als naar de mitigatie van de tijdsafhankelijke variabiliteit. De voornaamste doelsystemen vormen digitale geïntegreerde schakelingen, zoals processoren en geheugencomponenten. De huidige tekst voorziet ook praktische realisaties die bijdragen tot dat doel, zowel door bestaande technieken te bevestigen als door de ontwikkeling van nieuwe inzichten in het modelleren en de mitigatie van tijdsafhankelijke variabiliteit.

Het eerste deel omvat het modelleren en dit begint met een overzicht van atomistische modelleringconcepten, die zeer bruikbaar zijn bij de analyse van verschijnselen als BTI en RTN. Dit geldt vooral voor deca-nanometer transistoren, maar deze technieken zijn ook zeer complex. Om de complexiteit van de betrouwbaarheidsanalyse te verminderen over de levensduur van het

systeem, bevat het huidige werk een voorstel voor een compact digitale golfvorm (CDW) concept. Deze formulering exploiteert tijdsintervallen in de circuitwerking die vergelijkbaar zijn qua golfvorm (bijvoorbeeld soortgelijke frequentie of aan-af periode) en abstraheert deze naar een enkele vergelijking. Dit maakt het mogelijk om over grote tijdsintervallen heen te springen, met behoud van de belangrijkste kenmerken van atomistische betrouwbaarheidsmodellen (bijvoorbeeld afhankelijkheid van de werklast).

De huidige tekst draagt ook bij tot het introduceren van de transistorvariabiliteit tot op het architectuurniveau. Hiervoor wordt de component-defectwaarschijnlijkheid (P_{fail}) geselecteerd. Die wordt meestal opgelegd door de systeemontwerpers om de juiste uitschakel- of correctiemechanismen te bepalen. Om deze P_{fail} af te leiden, wordt een Meest Waarschijnlijke Defect Punt (Most Probable Failure Point of MPFP) methode voorgesteld, en die wordt toegepast voor de gevallen van BTI en RTN variabiliteit. Twee modelingsbenaderingen worden gebruikt voor deze verouderingsverschijnselen en observaties worden afgeleid met betrekking tot het belang van een juiste standaardafwijking (ten aanzien van de variatie van de drempelspanning). Daarnaast wordt een herformulering van het MPFP concept gepresenteerd, specifiek bedoeld voor het ontwerp met standaardcellen.

Het falen van systeemcomponenten is van oudsher bestudeerd in systeemontwerp, althans wat detecteerbare fouten betreft. Zowel de academische wereld als de industrie hebben de term Betrouwbaarheid, Beschikbaarheid en Onderhoudsgemak (Reliability, Availability and Serviceability of RAS) geïntroduceerd om te verwijzen naar dergelijke technieken. Hun toepassing is strikt gekoppeld aan de snelheid van de fouten die op het circuitniveau verschijnen.

Aan de mitigatiezijde, is het uitgangspunt een eenvoudige, terugrol-gebaseerde RAS techniek die gericht is op het herstellen van voorbijgaande fouten in het systeem. Dit concept is geïmplementeerd op een platform met meerdere processoren. In het datapad hiervan worden fouten geïnjecteerd met een ingestelde snelheid. Het terugrollen zorgt ervoor dat de actieve toestand in de toepassing op een eerdere opgeslagen toestand gebracht wordt, zodat de uitvoering daarna heropgestart en voortgezet kan worden. Dit faal/stop-model, ook als het zeer fijnkorrelig toegepast wordt, creëert een meetbare vertraging in de uitvoering van de lopende applicatie. Als gevolg daarvan wordt een gedeelte van de geïnjecteerde fouten niet gecorrigeerd, om de performantiereductie te beperken. Gezien deze wisselwerking, stellen we in het huidige werk voor om de afweging te analyseren tussen de applicatiecorrectheid, de performantie en het energiebudget, op zoek naar de optimale werkingspunten.

De laatste mijlpaal van het huidige werk is een algemeen toepasbare formulering voor het probleem van betrouwbare systemen, gezien de RAS meerkosten zoals

hierboven aangegeven. Om een betrouwbare performantie te herstellen wordt een controle-theoretische oplossing voorgesteld. Meer specifiek wordt een PID controlemethode gebruikt om de frequentie van een processor bij te sturen, met de granulariteit van meerdere klokcycli. Dit concept is ook praktisch geverifieerd zowel met eenvoudige simulaties als op een reeel processorplatform.

Contents

Contents	xvii
List of Figures	xxi
List of Tables	xxvii
1 Introduction	1
1.1 Context	1
1.2 Reliability Background	3
1.3 Functional vs. Parametric Reliability Violations	7
1.4 Contributions and Text Structure	9
2 Prior Art	15
2.1 Reliability Analysis	16
2.1.1 Injection	16
2.1.2 Detection	18
2.2 Reliability Modeling	18
2.2.1 Environment-Inherent Reliability Violations	19
2.2.2 Platform-Inherent Reliability Violations	20
2.3 Parametric Mitigation Within Package	23

2.3.1	Single Die	23
2.3.2	Multiple Dies	23
2.4	Parametric Mitigation Across Packages	24
2.4.1	Hardware Solutions	24
2.4.2	Software Solutions	25
2.5	Overview of Representative Groups	28
2.6	Current and Future Reliability Needs	29
2.7	Critical Appreciation and Contributions	31
3	Compact Digital Waveform for Circuit Aging Modeling	35
3.1	Background	35
3.2	The Atomistic BTI Model	38
3.3	Proposed Tool Support	40
3.4	CDW Verification for BTI Modeling	42
3.5	Thermal & Bias Applicability	48
3.6	Design Flow Compatibility	54
4	Time-Dependent Variability in the P_{fail}	59
4.1	Background	59
4.2	The MPFP Methodology	61
4.3	BTI Variability Models	63
4.3.1	Time-Zero Variability	64
4.3.2	Time-Dependent Variability and Aging	64
4.4	Simulation Flow and Results	66
4.5	Discussion and Extensions to SSTA	72
4.5.1	Reformulating the MPFP – Inverter Case Study	73
4.5.2	Generalizing to Complex Gates and Standard Cell Paths	76

5	Dependable Performance under Fine-Grain Rollbacks	81
5.1	Background	82
5.2	Rollback-Based Transient Error Mitigation	84
5.3	Experimental Setup	88
5.3.1	Target Platform	90
5.3.2	Error Injection & Application Correctness	90
5.4	Rollback Mitigation per Execution Node	94
5.5	Experimental Results	97
5.5.1	Building the Pareto Space	97
5.5.2	Experiments with a Hard Constraint	101
6	DVFS-Assisted Performance Dependability	105
6.1	Background	106
6.2	Problem Statement	108
6.3	Simulation-Based Verification	115
6.4	Experimental Verification	119
7	Conclusion	127
7.1	General Remarks	127
7.2	Future Work	129
A	Guidance Framework Methodology	131
	Bibliography	135
	Curriculum Vitae	163
	List of Publications	165

List of Figures

1.1	Transistor downscaling trends [258] © The Economist Newspaper Ltd, London	2
1.2	The chain of events from a physical mechanism to a system- or user-perceived failure [231]	4
1.3	Horizontal constraint propagation from analysis to modeling; constraints also propagate vertically across abstraction levels [231]	6
1.4	Variable circuit timing starts as parametric at the combinational logic, it appears functional when clocked and then as performance variability with timing speculation (all above diagrams are qualitative).	10
1.5	Schematic overview of the contributions of the current text . .	13
2.1	Classification framework for analysis and modeling of reliability violations on digital systems	22
2.2	Classification framework for parametric reliability techniques .	27
2.3	Crude timeline of indicative European projects in the vicinity of performance dependability and reliability [228]	31
3.1	Artist’s illustration of the differences between the average RD model and the sought-after atomistic modeling schemes for BTI/RTN	37

3.2	Artist's impression of a CDW compared to the SPICE and VCD waveforms: Signal regions with similar f and α are grouped in a single CDW point. This concept alleviates the complexity of atomistic BTI simulations over large time intervals, since it reduces the number of model iterations [234].	41
3.3	Flowchart of wavapprox tool: Each V_{gs} signal is handled according to its initial format, finally providing a CDW approximation [234].	42
3.4	Accuracy assessment (signal integrity) of the wavapprox tool that produces CDW representations, using a 1 μ s bit stream. It should be noted that SPICE requires 17089 points to represent that signal and a VCD representation uses 521 points [234]. . .	43
3.5	Accuracy assessment of the atomistic BTI model of Equation 3.7 for various degrees of CDW-compression [234].	45
3.6	Errors from a single iteration of Equation 3.7 on bit streams of different f_{avg} . The error of the proposed model is restricted to small defect subsets, as f_{avg} is increased [234].	47
3.7	Equations 3.9 and 3.10 enable full support of bias and temperature variations in the CDW-based BTI simulation framework [234].	48
3.8	Before resolving occupancy of defects with the CDW, time constants need to be scaled [234]	49
3.9	Distinction between "fast" and "slow" defects [234]	51
3.10	Applying the concept of pseudo-transient BTI/RTN simulation on a single device [234].	52
3.11	Applying the concept of pseudo-transient BTI/RTN simulation on a single device: the short- and long-term BTI/RTN impact for the workloads considered in Figure 3.10 [234].	53
3.12	Use of pseudo-transient BTI analysis for the estimation of time-dependent functional yield of an SRAM circuit	57
4.1	Schematic of the target circuit (pass transistors ignored)	62
4.2	Failure criterion analysis based on the SNM hold	62
4.3	Probability of a single failure point coordinate. The MPFP maximizes the coordinate-wide product of probabilities.	64

4.4	MPFP estimation setup used in the current work	67
4.5	The MPFP is more sensitive to the standard deviation of threshold voltage shifts, rather than the mean.	68
4.6	The inspected V_{dd} bump is not effective for SRAM cell reliability in case of aggressive variability. This claim remains valid only for the SNM hold criterion used in the current work.	69
4.7	The defect-centric model introduces a time-dependent increase of the V_{th} shift standard deviation. This is an additional and desirable feature on top of average aging models. We note that the color bars of the above Figures are presented in arbitrary units, whereas the actual proper unit of measurement for this probability density is Volts per second.	70
4.8	The average BTI modeling approach fails to properly capture the impact of time-dependent variability on the P_{fail} . Apparently, this course of action deviates by more than three orders of magnitude if stochastic ΔV_{th} behavior is unaccounted for.	71
4.9	Navigating the $\mathbf{x}$ space for accurate approximation of the inverter delay distribution. The χ^2 distribution is useful in bounding failure region F	74
4.10	Inverter delay analysis and approximation of set F for P_{fail} extraction	77
4.11	Results of the coordinate descent implementation of Algorithm 1. One of the minimum delay points is identified within ~ 160 NanoTime iterations.	78
4.12	Iterative convolution of PDF_y from Figure 4.10c provides the delay PDF, CDF and, eventually, the P_{fail} for a chain of four inverters.	80
5.1	Various applications, abstracted using a combination of Computation Kernels (CKs) and Accumulation Kernels (AKs) [230] .	86
5.2	Hybrid (HW-SW) mitigation of errors in application data, for the kernel types of Definitions 5.4 and 5.5. Error injection stages are depicted, in lieu of real transient errors [230].	87
5.3	Steps of the mitigation methodology for a hypothetical application, being ported on a many-core platform. The concept is illustrated for two execution nodes and their kernels [230]. . . .	88

5.4	Initial step of the proposed mitigation methodology, as outlined in Figure 5.3. Application kernels are mapped to execution nodes and their types are identified [230].	89
5.5	Schematic overview of the SCC layout and of our experimental setup [230]	91
5.6	Scope of the error injection per SCC tile and examples of SNR degradation due to errors in the decoder’s data plane [230]. . .	92
5.7	Second step of the methodology: implementing the rollbacks on demand for every encountered kernel type [230].	95
5.8	When DFS is used along with mitigation, the timing overhead is transformed to an average power equivalent [230].	97
5.9	Exploration of the trade-off between the decoder’s delay, energy and output fidelity for various rates of DUE injection [230] . .	99
5.10	As the injected FIT rate increases, a more aggressive DFS is required to keep timely arriving video frames [230].	102
5.11	Performance evaluation of the MJPEG decoder runtime with a hard constraint on output fidelity. DFS successfully reclaims slack, up to a certain upper bound of injected FIT [230].	104
6.1	Synoptic classification of representative prior art samples on RAS techniques, their performance implications and attempts to reclaim the latter in a systematic way [226]	108
6.2	Interplay between cycle noise and budget for a series of TNs [226]	109
6.3	DVF for various cycle noises and budgets, assuming a constant reference frequency multiplier. In this case, PVF follows exactly the same behavior [226].	110
6.4	Block diagram of the proposed performance dependability scheme [226]	113
6.5	Numerical illustration of the average performance dependability criterion of Equation 6.12 [226]	115
6.6	Setup simulating the execution of a TN sequence [227]	116
6.7	Step response of the proposed PID scheme [227]	118
6.8	Static multipliers may miscompensate cycle noise [227]	119

6.9	The r parameter can improve energy efficiency [227]	120
6.10	Increased r values lead to energy savings [227]	120
6.11	The default system scenario is defaced in the presence of cycle noise [226].	121
6.12	Transient view of the PID control functionality on the X3470 [226]	122
6.13	Evaluating the PID controller with the target application on one core of an X3470, for a variety of MTTFs. Average DVF across 10 runs are presented within a 95% confidence interval. Circular markers show DVF at the end of each run [226].	123
6.14	Illustration of the mean performance dependability criterion of Equation 6.12, for the range of μ , λ and $m_{\max}$ values encountered in our experiments [226]	124
6.15	Experimentally derived boundaries for slack reclaiming capabilities of our proposed performance dependability setup. The feasibility region predicted by Equation 6.12 and shown in Figure 6.14 is successfully verified through our measurements [226].	125
A.1	Binary splits leading to the classification framework used to map prior art. In the above example, “Work 3” is a hybrid, given that it is mapped both in “B.1” and “B.2”.	132
A.2	Example of our classification framework methodology [231]	134

List of Tables

3.1	Atomistic model configuration for yield analysis of the target SRAM circuit, according to BTI/RTN measurements [265] and the 90 nm Predictive Technology Model [1], as initially disclosed in [234]	55
4.1	Model parameter explanation and values used for average BTI aging model	65
5.1	Optimal configurations of the slack reclaiming knobs according to the $(k_{\text{DFS}} \%, k_{\text{WI}} \%)$ format [230]	100
5.2	Guarantees on application correctness and timing based on the proposed mitigation and slack reclaiming with DFS [230]	101
6.1	Based on Definition 6.6 and Equation 6.4, a zero target slack should be (intuitively) equivalent to a zero target DVF, namely $s[n] = 0 \Leftrightarrow \text{DVF}[n]$. We prove this statement using perfect induction, assuming that $s[0] = \text{DVF}[0] = 0$, which can be expected at the beginning of execution for a series of TNs. Hence, <i>when performance variability needs to be mitigated (i.e. DVF has to be kept at zero), it is sufficient to aim at forcing slack to converge to zero.</i>	112
6.2	Variable explanation for the block diagram of Figure 6.4	113
6.3	Default simulation parameters of the setup of Figure 6.6 [227]	116
6.4	Timing and power model used in the simulation setup of Figure 6.6 based on prior art [200], as originally disclosed in a previous publication [227]	117

6.5

Frequency multipliers of target platform: V_{dd} values are
hypothesized, based on assumed f - V_{dd} relationship and publicly
available voltage range of target processor.

121

Chapter 1

Introduction

In this Chapter, we illustrate the high-level connection between the variability observed at various abstraction levels of integrated circuits with the respective reliability implications. Initially, we provide the context within which variability modeling and mitigation is perceived as a valid research goal, especially given its ramifications at various levels of digital system design. We then navigate the abstraction stack, thus substantiating the constraint propagation from functional to parametric violations and vice versa. Finally, the Chapter is completed with an overview of the contributions of the current text.

1.1 Context

Concepts like dependability and reliability are of major importance for all engineering reductions to practice. The progress that is promised by technological development is put to risk when reliability/dependability of the delivered products does not follow suit. Every novel idea, regardless of its performance or energy efficiency, is always put to the reliability test before being made available to the vendor or end user. It is through such constructive doubt that the engineering domain has safeguarded the quality and longevity of its innovations. Only through a systematic and formulated perception of reliability and reproducibility can the engineering disciplines issue guarantees, especially in applications related or connected to human life and safety. In the face of rigorous growth principles and aggressive innovation, reliability is an ever-present checkpoint. In stark contrast to many other domains, people involved in the integrated circuit industry typically cite certain growth principles/laws

in their effort to safeguard their market position, justify their investments and meet customer needs. Readers familiar with the art have definitely heard about Moore’s law [185], different flavors of which have regularly featured in introductory slides, Sections or Chapters of related work. Regardless of how this argument has been reproduced over the years, since its initial articulation in 1965, it has succeeded in capturing the growth race of the integrated circuit industry. Whether this is transistors per chip, frequencies, numbers of cores or threads per chip, it can be historically verified that there has been always something being pushed to increase exponentially in the integrated circuit industry.

Frequency has been following transistor “cramming” and downscaling up to a certain point [84]. Then came the concept of chip multi-processors. In any case, transistors have to be miniaturized even further so that more cores can fit, let alone co-operate seamlessly, on a single silicon die. This required the introduction of novel materials in the gate stack and use of aggressive

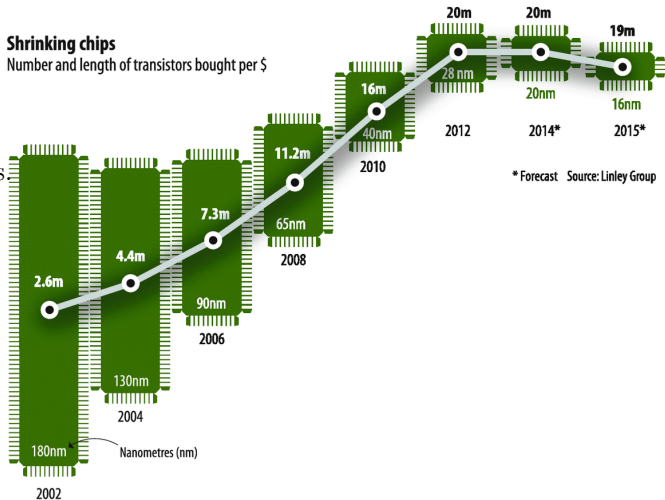


Figure 1.1: Transistor downscaling trends [258]
© The Economist Newspaper Ltd, London

lithography, aiming at maintaining the traditional MOSFET, and later the FinFET, as a vital building block. Such material science and process engineering innovations did not arrive at negligible cost. Figure 1.1 of the Economist uses a rather compound metric (number of transistors per USD) to indicate how non-negligible this cost is. It is also noteworthy that the disparity in our “transistor buying power” is a rather current issue, given that it refers to current and future technology nodes. It is rather logical that efforts in ensuring the feasibility of future technology nodes are of paramount importance. Variability analysis is an enabler, in the sense that it formulates reliability assessment techniques. Variability mitigation looks towards the implementation of systems in a way

that does not deviate from the specifications that the integrated system industry has adhered to for so many decades.

As the deca-nanometer regime is reached, there is a wide variety of process and design parameters that appear to fluctuate, such as the dimensions of the transistors. This requires the development of modeling techniques that capture variability in the internals of integrated circuits or even at the level of discrete components (Chapters 3 and 4). In Section 1.3 we will illustrate that variability is not a prerogative of a certain abstraction level. On the contrary, it strides vertically across the abstraction stack and manifests itself in different ways.

In the “predictable” world of digital systems, governed by clocks, synchronization, (dead)lock elision and deadlines, variability is usually the culprit of erroneous behavior. Hence, it is logical to strive for the mitigation of variability or, at least, its effects. Rarely does such mitigation come for free in digital systems. Area, power or performance costs are typically paid to ensure correct functionality. As a result, variability mitigation is effectively equivalent with shifting the “cost” of erroneous behavior to a side (power, energy, performance) that is of lesser importance.

A near-optimal variability mitigation technique brings about minimal overheads and the cost that is imposed is such that no specification violations are observed. In the current text, performance is placed as the top priority when handling variability. It will be shown that *existing* reliability techniques cause performance to fluctuate (Chapter 5). However, it is equally important to retain dependable performance in a cost-conscious way, namely without extremities in energy consumption. As a result, the problem of dependable performance requires a holistic treatment that accounts for minimal energy overheads. In the current text, a reactive performance boost is proposed in order to mitigate performance variability (Chapter 6).

1.2 Reliability Background

Apart from a conceptual view point, there have been many documented definitions for reliability [141] and the associated threats. Here, we attempt one Definition for the sake of completeness.

Definition 1.1. Reliability violation *is an umbrella term describing any deviation of a digital system from its correct and expected operation. The is regardless of the deviation’s severity, hence it covers minor through fundamental malfunctions. In the context of this text, we only assume physically induced reliability violations.*

of faults are shifts in the threshold voltage of a transistor (ΔV_{th}) or a reduced critical charge (Q_{crit}) [224] of an SRAM cell. A fault remains *inactive* until the respective HW component gets involved in system operation. Once the fault is activated, an error is triggered. A fault may also be naturally *corrected* before it is activated.

Definition 1.5. Reliability Analysis of a digital system, is the monitoring of the system's performance under the presence of reliability violations.

Reliability violations can be *injected* in the system either naturally, through the operating conditions, or artificially, by manipulating the ported SW or regions of the HW. The choice between fault or error injection depends on the complexity of the injection procedure and the target abstraction level. Finally, the direct injection of failures in the digital system would be less useful, since the user-specified constraints are violated upon injection. Monitoring of the system during reliability analysis may address functional and parametric reliability violations, with various degrees of severity. Furthermore, the propagation of a reliability violation to higher abstraction levels is also *detected*. With reliability analysis, we can estimate the portion of the injected faults that become activated (i.e. become errors) within a digital system.

Definition 1.6. Reliability Modeling involves the derivation of a formal expression for a reliability metric of a well defined digital system.

During the reliability analysis phase, we gather data about system performance under a variety of usage profiles. During reliability modeling we will search for and formulate correlations between the system's usage profile (i.e. combination of SW workload and operating conditions) and the quantities measured during the analysis. Here lies an opportunity to use a reliability model, i.e. a formulated correlation between a reliability metric and system usage, for the calibration of reliability analysis at a higher abstraction level. In other words, instead of performing the reliability analysis with real disturbance of the operating conditions, we can create fault or error artifacts using a reliability model that has already been created at lower abstraction levels. As a result, we identify two flows of information:

1. **Major Information Flow:** Reliability analysis data about system performance under a variety of usage profiles are passed to the modeling phase, in search of correlations that will create reliability models.
2. **Minor Information Flow:** A reliability model calibrates a reliability analysis campaign at a higher abstraction level. The calibration may involve the aggression of fault or error injection, in lieu of actual faults or errors occurring in the system.

In view of this interaction, it is important to understand the *constraint propagation* between analysis and modeling. By nature of these two phases, the major information flow occurs from the reliability analysis to modeling. Hence constraints are propagated “horizontally” at each abstraction level. The minor information flow happens in a “vertical way” between abstraction levels, so that it does not cause any cycle or mutual dependence. In Figure 1.3, we illustrate this concept for a wide variety of abstraction levels. Following the same logic as the abstraction level ascends, we can perform consecutive modeling and analysis attempts reaching the entire digital system. In each step, analysis data create models, used to calibrate reliability analysis at higher abstraction levels. Usually, the latter is performed with HW descriptions, as in Technology Computer Aided Design (TCAD) [37], SPICE [190], VHDL [108] or Verilog [109] and SystemC [110] (in ascending abstraction across the Y-chart [85]).

Definition 1.7. *Mitigation of a reliability violation is the alleviation of the latter’s cause or effect in a way that the violation does not propagate to higher abstraction levels.*

Obviously, the complexity of mitigation is tightly coupled to the severity of the corresponding violation (e.g. fault, error or failure). Also, mitigation can be performed at various abstraction levels of either the design or manufacturing process of an integrated circuit.

At this point, we have established the terminology foundations upon which the concepts of the current text will be developed. In the next Section, we elaborate on an interesting distinction between reliability violations, which has direct implications for any mitigation mechanism.

As a far as the modeling of variability, prior art features many samples that describe semiconductor mechanisms using TCAD [13] or SPICE [229] implementations. Additionally, there is significant effort in abstracting reliability

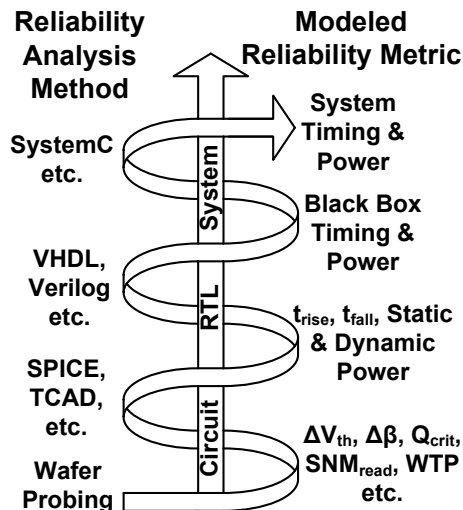


Figure 1.3: Horizontal constraint propagation from analysis to modeling; constraints also propagate vertically across abstraction levels [231]

and variability information at the level of a hardware black box [31]. The basic observation from such techniques is the increased computational requirements for the full exploitation of such models. Simulation typically take a long time, especially when full lifetime reliability analysis is attempted. The current text emphasises on this complexity, in terms of signal compression (Chapter 3) and probability calculation (Chapter 4), and attempts to alleviate it, while keeping reasonable levels of accuracy.

On the mitigation side, prior art features circuit or architectural techniques, whereby errors are detected and corrected. In some cases, signal timing speculation is used [75]. In other cases encoding of the data ensures detectability and correctability [98]. Defective component disabling is also a valid mitigation technique [251]. Additionally, the performance impact of such techniques has already been documented [102, 138]. The current text documents the of a mitigation technique and assesses its impact on application quality of service (Chapter 5). Following such measurable impact, a reformulation of dependable performance is outlined and experimentally verified on a real platform.

1.3 Functional vs. Parametric Reliability Violations

A valid distinction is between *functional* and parametric violations [231].

Definition 1.8. *Functional reliability violation relates to the correctness of the bits that are stored, processed or communicated within and across systems.*

A functional reliability violation is the *soft error* [187], i.e. the binary corruption of a node (e.g. of the bit in an Static Random Access Memory – SRAM, cell). It is clear that a functional violation is translated to a binary random variable, representing the pass/fail eventualities. A valid metric for functional reliability violations is the failure probability of an electronic system (P_{fail}).

Definition 1.9. *A parametric reliability concern refers to operation margins of the digital system, or a black box within the former. These may be related to the timing or quality cost (e.g. total energy) of that system or black box.*

A good example of parametric reliability violations is the fluctuation of a pMOSFET's threshold voltage (V_{th}) due to Negative Bias Temperature Instability (NBTI) [94]. From a reliability metric viewpoint, a good example is the Static Noise Margin (SNM), which illustrates the voltage margin of an SRAM cell, beyond which the cell becomes unstable [242] (i.e. prone to “bit flips”). In complete contrast to functional violations, parametric ones are reflected on a multi-variate (and probably continuous) random variable. Already,

we can see the variability concept being associated with parametric reliability violations. In other words, *a variability issue arises when design parameters that were assumed constant, appear to be multi-variate at a certain stage of the design process*. Already at this point, we can identify a significant difference in the *complexity of detecting* a reliability violation, based on the above distinction: A functional reliability violation typically requires some sort of thresholding among observed values in order to resolve the pass/fail hypothesis. This is a far less complex procedure than statistically handling the observed values, in order to resolve the extend of the parametric reliability violation and derive the underlying distributions.

Even though Definitions 1.8 and 1.9 are strictly complementary, it is important to realize that designating a reliability violation as parametric or functional is highly *context dependent*. We illustrate this concept with the example of Figure 1.4. We start from a simple combinational circuit (Figure 1.4a), which exhibits variable timing (Figure 1.4b) as a result of random transistor variability. This is an example of a parametric reliability violation at the circuit level. When clocked and buffered (Figure 1.4c) this circuit may exhibit timing errors, due to mismatch with the clock frequency of the latches (Figure 1.4d), which is a functional reliability violation occurring with a specific probability. Prior art has devised ways to avoid such errors, using timing speculation and shadow latches [75, 269]. The idea is that the timing error is “sensed” and a rollback is imposed on the sequential circuit (Figure 1.4e). Even though the functional violation is resolved, an increment in the number of clock cycles of the logic operation is imposed, since timing errors add an extra clock cycle to the system’s execution (due to the rollback). The concept of Performance Vulnerability Factor (PVF) has been previously introduced to quantify this timing overhead [102]. Timing speculation leads to performance variability, namely variability in the clock budget utilized by the system, which is clearly a parametric reliability violation: the more probable a rollback is due to a timing error, the larger the extra cycle overhead (or PVF) due to timing speculation. From the above, it is clear that parametric reliability violations at the circuit level cause functional ones at the register-transfer level (RTL). Functional violations cost in performance when detected/mitigated (i.e. parametric violations).

For the sake of completeness, it should be noted that many functional mitigation techniques lead to system-wide parametric reliability violations. The general concept of graceful degradation has long proposed the use of knobs and monitors that guarantee binary correctness at the cost of system performance [251]. In any case, it is clear that modern digital systems exhibit parametric reliability violations at various abstraction levels [85]. In many cases, one can identify a context-specific interplay between functional and parametric ones. As a result, in perceiving the contributions of the current text, the reader should

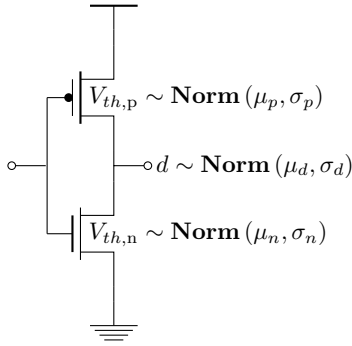
be mindful that variability (i.e. occurrence of parametric violations) should not be restricted to a specific abstraction level. Such a viewpoint restricts the scope of mitigation techniques, especially if we consider the aforementioned inter-dependencies between functional and parametric aspects. On the contrary, a cross-layer view of reliability and variability should be maintained in order to make sure that mitigation techniques are actually beneficial [231, 183].

1.4 Contributions and Text Structure

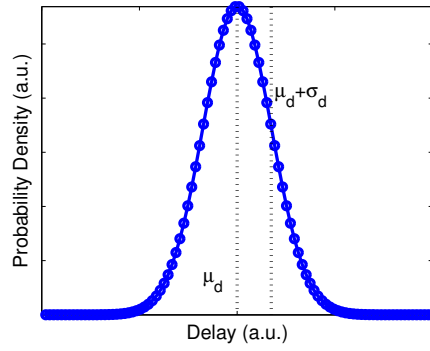
In the previous Sections, we have presented key concepts that relate to the time-dependent variability of digital systems. In Section 1.3, we have illustrate the difference between parametric and functional reliability violations. Additionally, certain examples of variability have been provided and escalation of parameter fluctuation across the abstraction stack has been illustrated (e.g. ΔV_{th} shifts at the transistor level, leading to fluctuations of noise margin or delay at the circuit level). As such, we have identified the interplay between functional and parametric violations: variability at a certain abstraction can be translated in binary errors, which in turn can lead to system-level performance variability. The contributions of the current text are appropriately positioned into the general cross-layer reliability paradigm.

On the reliability analysis and modeling side, our contributions are as follows:

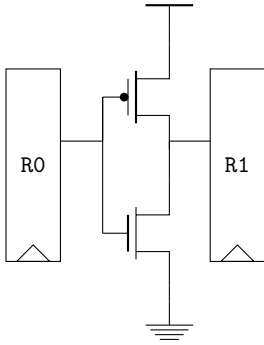
- We provide insight into a transistor-level phenomena that modulate the threshold voltage (V_{th}), namely Bias Temperature Instability (BTI) and Random Telegraph Noise (RTN).
- We propose a signal representation format that allows lifetime inspection of BTI and RTN, without incurring prohibitive execution times. This reliability modeling phase is based on a state-of-the-art atomistic model for the above phenomena. This contribution stands apart from prior art, by attempting to trade-off computation times with proper BTI/RTN evaluation accuracy. *To the best of our knowledge this is the first systematic approach towards signal compression for transistor aging analysis.*
- Starting from V_{th} shifts (ΔV_{th}) caused by BTI/RTN, we implement a reliability analysis methodology to estimate the failure probability of SRAM components. For that purpose, we are leveraging the Most Probable Failure Point (MPFP) methodology, which has been previously used in prior art. We substantiate the impact of ΔV_{th} standard deviation for accurate failure probability calculation.



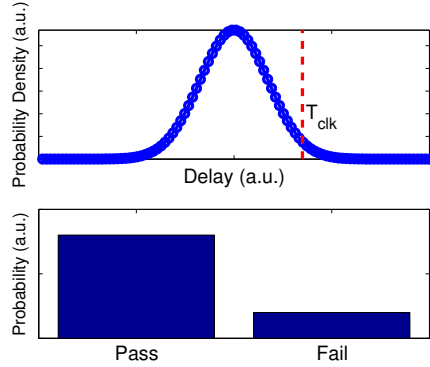
(a) Case of variability-induced combinational delay distribution



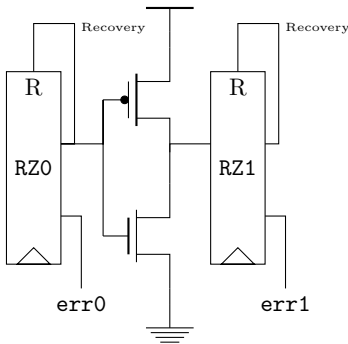
(b) Variable delay indicates possible parametric reliability violations



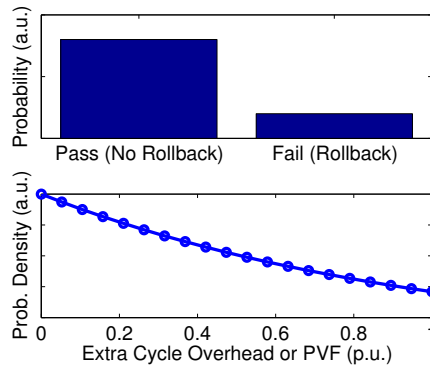
(c) Basic example of logic circuitry, synchronized with two registers



(d) At the synchronization point, the parametric reliability violations are perceived as functional violations



(e) Timing speculation is implemented with shadow latches [75, 269] to avoid errors



(f) With timing speculation, the initial timing error contributes to performance variability, due to rollback overheads

Figure 1.4: Variable circuit timing starts as parametric at the combinational logic, it appears functional when clocked and then as performance variability with timing speculation (all above diagrams are qualitative).

- Additionally, we propose an extension of the MPFP principle to more complex gates and paths of standard cells, as a *novel complement to existing Statistical Static Timing Analysis (SSTA) techniques*.

On the mitigation side, the contributions of the current text are as follows:

- Based on a Mean Time To Failure (MTTF) specification, we implement a functional mitigation techniques for injected transient errors on a research-grade platform and application.
- The implemented mitigation technique comes from prior art and resembles fine-grain rollbacks and application state reconstruction. *In the current text we discuss the suitability of such techniques for distributed memory chip multiprocessors.*
- The temporal impact of the above methodology is reclaimed with a statically resolved frequency boost on the target platform in order to ensure dependable application timing. Extensive measurements create a design space from which we pick the optimal points for performance-conscious, rollback-based transient error mitigation.
- The greater performance impact of Reliability, Availability and Serviceability (RAS) mechanisms is formulated into “cycle noise”. The problem of dependable performance, in view of RAS mechanisms is translated to a traditional control-theory problem and a solution is proposed using a Proportional-Integral-Differential (PID) controller. Enabler for this contribution is related work on performance vulnerability of out-of-order processors.
- The efficacy of the PID controller is evaluated with simple timing/power simulations. *Contrary to prior art, which typically features scheduling or mixed criticality problems, we additionally demonstrate the efficacy of our control-based technique on a real platform and application.*

The current text is organized accordingly to reflect the above contributions. A schematic representation can be seen in Figure 1.5. At a Chapter-level, the current text is structured in the following way:

Chapter 2: This Chapter discusses prior art on the modeling and mitigation of time-dependent variability. As such, it presents both these domains in a systematic way, using a top-down-derived binary tree. In that way, all possible analysis, modeling and mitigation techniques for time-dependent variability are thoroughly covered. Highly specialized related work that

refers specifically to each Chapter is presented in dedicated Sections respectively.

- Chapter 3:** This Chapter starts with a presentation of BTI/RTN primitive concepts. It briefly reviews prior art specifically for these two phenomena and expands on the latest modeling approaches. Then, it presents the proposed signal representation format (i.e. the Compact Digital Waveform – CDW), which is used to speed up reliability analysis/modeling for BTI/RTN.
- Chapter 4:** In this Chapter, we link BTI/RTN variability with the failure probability of system components. We present the applicability of MPFP methodology in order to derive failure statistics of an SRAM cell. We also reflect on the extensibility of this methodology to the greater domain of Statistical Static Timing Analysis (SSTA) for digital standard cells.
- Chapter 5:** Starting from an MTTF specification (which typically originates from the transient or permanent failure occurrence probability), this Chapter discusses the implementation and experimental verification of a fine-grain rollback scheme. The target reliability violations are transient errors, which occur at a specific rate. We show how this functional mitigation scheme causes performance overheads on the target application and we work for dependable performance by appropriately boosting the frequency of the processor.
- Chapter 6:** The performance variability caused, generally, by RAS techniques (such as the individual case of Chapter 5) is formally and experimentally treated in the last technical Chapter of this text. We formulate a control problem for dependable performance and propose a solution using a PID controller for the processor's frequency. We evaluate this idea both with simulations and on an experimental setup.
- Chapter 7:** This Chapter contains concluding remarks and pointers for future work, corresponding to the aforementioned technical contributions of this text.
- Appendix A:** In this Chapter, we outline the prior art classification methodology that is employed to present related work samples in any field of study. The systematic step of this methodology are presented along with some graphical representations.

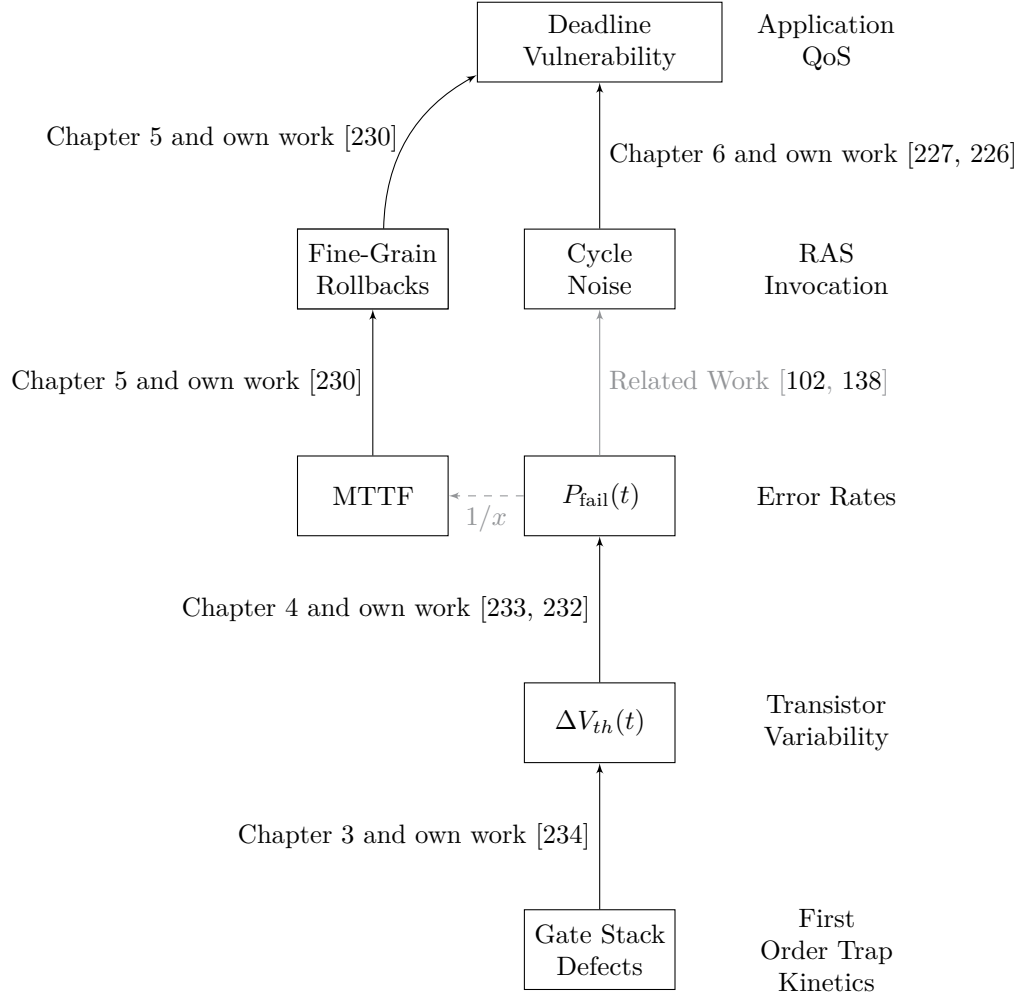


Figure 1.5: Schematic overview of the contributions of the current text

Chapter 2

Prior Art

In this Chapter, we will present prior and related work on the greater domain of time-dependent variability modeling and mitigation. The content presented herein is largely based on two survey project that have been encountered [231, 225]. It is also worth noting that prior art classification attempts have been made for other research domains, as in the case of scheduling techniques [142]. The goal of this Chapter, as is the case for the survey projects, is to provide any interested reader with a guidance framework on prior art samples.

As such, we are based on a systematic classification methodology, the goal of which is to operationalize the handling of prior in any target domain. In line with the scope of the current thesis, we address issues of reliability *modeling, analysis and mitigation*. The latter part is split between intra-package and inter-package solutions, given the dramatic increase in reliability complexity, when one perceives systems of discrete components (i.e. packages). It is also important to note that the mitigation techniques that will be discussed here are *parametric* (see Definition 1.9), in the sense that functional (binary) issues are beyond the scope of time-dependent.

For a detailed description of our classification methodology, the interested reader is encouraged to read Appendix A of the current text. In summary, our way of handling prior art samples is as follows:

(i) Starting from a root topic (e.g. reliability analysis), we perform binary splits in an inductive way. The goal is to partition the target domain in orthogonal categories. For that reason, the criteria that are used to split each node of the tree are strictly complementary. Having derived a sufficiently deep binary tree, we end up with leaves, which constitute complementary categories.

(ii) From the pool of prior art, we map existing samples to leaves base on their similarity in terms of assumptions and/or implementation. That way, the use of the so called “guidance framework” is able to map prior, related and even his own works in such a way that novelty, or research relevance can be resolved.

In the current Chapter we address, the analysis (Section 2.1), modeling (Section 2.2) and mitigation (within and across silicon dies – Section 2.3 and 2.4 respectively) of reliability violations. We apply the aforementioned top-down splitting technique and map samples of prior art in each case. Having derived a complete view of the reliability domain, in Section 2.7 we perform a critical appreciation of mapped prior art samples. We also comment on the assignment of this text’s Chapters to our classification framework.

2.1 Reliability Analysis

At the reliability analysis phase, the digital system activity is monitored under specific SW workload and operating conditions (see Definition 1.5). This Section will address the splits of the analysis branch in a top-down fashion, mirroring the classification framework methodology. The resulting classification and mapping appears in the left branch of Figure 2.1 Reliability violations, namely faults, errors and failures (in ascending order of severity according to Definitions 1.2 through 1.4) are present in the system during the reliability analysis. These violations are either naturally occurring, or artificially injected. In any case, a subset of these violations needs to be detected, in order to provide data for the next modeling phase. As a result, the initial split of the reliability analysis branch is between injection and detection of reliability violations. Each one will be presented in Subsections 2.1.1 and 2.1.2 respectively.

2.1.1 Injection

The first step of reliability analysis requires the creation of reliability violations within the digital system (i.e. faults, errors and failures). In reality, physical mechanisms are the ones that trigger the escalation up to a user-perceived fundamental malfunction. However, it is also acceptable to create artifacts of reliability violations (of varying severity, based on Definitions 1.2 through 1.4) for analysis purposes. This Subsection will cover all possible approaches, with respect to the injection of such violations. The first splitting criterion is based on the availability of the digital system’s actual HW. This yields two branches, one for *HW* and one for the case of a *HW description*.

In case the HW is physically available, we can inject reliability violations either by *directly contacting the HW* under test or *without a physical conduction path*. This distinction yields a pair of branches at a lower abstraction level. In case we opt to physically connect to the target system, we can check for “black boxes”¹ that are *controllable*. The inputs of these black boxes, assuming that they are controllable, can be used for the injection of reliability violations [170]. The alternative option is to implement *injector modules* in the HW of the digital system under test. These additional HW regions are exclusively dedicated to the creation of disturbance that will cause reliability violations in the HW [122]. If we choose to inject reliability violations without a contact, the only remaining components that can be used are its *operating conditions* and *ported SW*. In the former case, we can manipulate the natural interferences that surround the digital system, thus creating the desired level of reliability violations [273]. In the latter case, alterations can occur to generic or service-specific SW that is executed on the digital system [281].

When HW is unavailable, we have to limit ourselves to a HW description. Without it, injection of reliability violations cannot be performed. A HW description is executable and allows *alterations*, in our case, for the injection of reliability violations. Also, a simulation/emulation tool that executes the HW description can be enhanced to enable injection of reliability violations. In case we choose to make HW description alterations for injection purposes, we can either designate *controllable black boxes* [245] or create additional *injection modules* [52]. If we choose to leave the HW description unaltered, the only components that can be used for injection of reliability violations are the *SW executed on the HW description* and the *tool that emulates/simulates the HW description*. In the former case, we manipulate the binary of the ported application in order to emulate reliability violations [167]. In the latter case, we enhance the simulator/emulator tool in order to enable the controllability of the executed HW description. Given that the HW is in a description form, it would be meaningless to manipulate operating condition for injection purposes, since the HW is not physically present. The cases where the HW description is emulated or simulated under artificial operating conditions correspond to the previous branch (namely, manipulation of the HW description). In such cases, artifacts of special operating conditions are created using existing features of the emulator/simulator (e.g. SPICE simulation for a specific temperature) and not by adding new ones [215].

¹A “black box” can be perceived as a module of a system with behaviorally known operation and strictly defined inputs and outputs. However, this term does not encapsulate the inner workings of the black box.

2.1.2 Detection

Having injected faults or errors in a digital system, we need to detect their impact across various abstraction levels. This detection provides information about the degree of propagation and the percentage of the injected violations that lead to user-perceived malfunctions (i.e. failures). That way, we can gather measurements and try to correlate them to the SW workload and operating conditions that have been applied to the digital system under test. The latter stage is equivalent to reliability modeling and will be covered in Section 2.2. The *physical availability of the system's HW* is chosen again as the initial splitting criterion for the detection branches. The HW of the digital system may be physically present (HW) or instantiated as a HW description.

Given that the system is physically implemented, we can turn either to *HW* or *SW* utilities to detect reliability violations. In the HW case we can either assess the *observability of target black boxes* [273, 221] or incorporate *detection modules* within the system [273, 170]. When SW is chosen for detection purposes we can choose between *generic SW utilities* [281] or utilities that are related to a *specific service* [132] delivered by the system.

If we use the HW description for detection purposes, we can either assess the *observability of the design's black boxes* [52] or introduce *additional detection modules* [79, 167, 175]. In the former case, the observability of a design's black boxes depends highly on its complexity as well as the capabilities of the verification tool that is used (e.g. emulator or simulator of the HW description). In case we choose the SW that is ported on the HW description for detection purposes, we can opt for generic SW (e.g. OS utilities) [215, 79] or SW that is connected to a specific service delivered by the system (e.g. device driver) [90].

2.2 Reliability Modeling

In the previous Section, we have presented all possible approaches on the reliability analysis of a digital system. First, we have explored the injection of reliability violations in a system. The impact (e.g. propagation) of these violations is detected during system operation. For example, the percentage of violations that cause user-perceived malfunctions can be tracked. Such information needs to be post-processed in order to create formal correlations between the usage profile applied to the system and its reliability. This is performed during reliability modeling (see also Definition 1.6). The classification framework of prior is shown in the right branch of Figure 2.1.

The initial distinction refers to the source of the reliability violations, namely the *physical mechanism that lies under the target reliability violation*. A valid distinction of that sort is between environmentally inherent mechanisms and the ones that are rooted within the physical boundaries of the system platform. Any interference coming beyond the system packaging is considered environmental. All other mechanisms are assumed to occur within the platform. The former branch will be covered in Subsection 2.2.1 and the latter in Subsection 2.2.2.

2.2.1 Environment-Inherent Reliability Violations

This part of the modeling classification deals with the reliability models for violations that are rooted in the environment of the system. We can further split between *interference from neighboring HW* or *actually natural stimuli*. In the former case, we incorporate all sorts of interaction between systems. In the latter case, we include all mechanisms that happen by nature in the “ecosystem” of the system under test, without originating from another system.

In the case of interference from a neighboring system (a system beyond the packaging boundaries of the system under test), we can check whether any *functional correlation* is present between the system and its neighbors (i.e. an inter-dependence may exist for all systems to be able to operate). In case the system under test is functionally dependent to neighboring systems, an *I/O dependency* can be present, as in the case of systems connected in series [255]. Alternatively, the two systems (the one under test and its neighbor) may be correlated in terms of *resource provision* [131]. The domain of battery-powered embedded systems presents an excellent source from which to draw relevant papers. In such designs, a battery is a separate package than the processor or any Intellectual Property (IP) block. This domain is fully compatible to our formulation of the current leaf. When the neighboring system is not functionally coupled to the system under test, the interferences we can inspect may require a conduction path. The case where a conduction path is required for the interference to reach the digital system narrows down to *heat transfer* [26]. The complementary branch deals with *inter-HW radiation* [48].

Apart from neighboring systems, natural environment may also provide a lot of interference that can cause reliability violations to digital systems. Again, we check whether nature is *functionally coupled* to the system under test. If such a functional coupling exists, the next split can be again between *I/O dependency* [14] (namely, the use of natural stimuli as a source of information) and *resource provision* [112]. If the natural environment is uncorrelated to the system under test, we can further distinguish based on whether a *conduction path* is required for the natural interference to reach the digital system under

test. The branch that requires a conduction path is narrowed down to *heat transfer* [14]. The complementary branch deals with the operation under *particle interference* [192, 24, 93]. To the best of our knowledge, reliability modeling due to heat transfer is always related to specific applications that involve elevated temperatures. Greenwell et al. illustrate the extreme temperatures observed in various aspects of the automotive domain [96] (e.g. HEVs, as discussed earlier for [26]). In the majority of the heat transfer modeling papers, it is neighboring HW that creates the extreme temperatures. We can populate the current leaf with modeling attempts that concentrate on heat transfer from the ambient environment, while agnostic of the neighboring HW or the specific “ecosystem” where the digital system is operating.

2.2.2 Platform-Inherent Reliability Violations

In this domain of reliability modeling, we are dealing with violations rooted within the digital system’s physical boundaries. The impact of the underlying physical mechanism may propagate to the end-user. As a result, a valid criterion for the first split in this sub-domain, is whether the responsible mechanisms are caused by *manufacturing defects* or are occurring because of specific *system utilization*.

We can further split based on the time instance at which the effects of these mechanisms (i.e. the respective faults) become apparent. This distinction yields the *time zero* and *evolving mechanisms* branches. When the platform-inherent physical mechanisms cause reliability violations from time-zero, we can have time-zero device *variability*, which can be either *systematic* [99, 32] or *fully random* [289, 235]. Variability does not make the system completely dysfunctional, however its impact can be perceived from time-zero. In the complementary branch lie the mechanisms, which are triggered during manufacturing and have a perceivable impact on system operation, only as its lifetime progresses. That impact cannot be perceived at time-zero. Some of these mechanisms are *irreversible*, namely their impact is strictly increasing with system lifetime [134, 254]. Other mechanisms are evolving during the system lifetime, but they contain a *partially reversible* feature, i.e. their impact is partially revoked during the system lifetime [119]. These two types create the branches of the current split.

When the utilization of the system is the source of reliability violations, we can check whether the user can have any *control over the way that the system is utilized*. If that is not the case, we can look into HW utilization, as a result of the system’s operation. This branch is static in nature, in the sense that the user has no control over the way that the HW is utilized. The complementary

branch refers to user-controlled HW utilization either by manipulating the ported SW or by configuring the HW utilization options. In other words, there is a set of “knobs” at the user’s disposal which allow HW utilization adjustment (e.g. cutting supply to specific HW black boxes or manipulating the application mapping). In case the user is unable to control system utilization, we have to focus on the HW activity. This requires a black box exploration of the system HW. Then, we can distinguish between the two major issues of HW utilization, namely the existence of *hot spots* [249] across the HW and regions that exhibit a specific *current or voltage activity* [265, 162]. When the utilization of the HW is coupled to or configurable by the user, we assess both the *mapping utilities* [273] of the system or the *actually mapped application* [29, 133, 221]. The former component is generic and the latter is coupled to a specific delivered service.

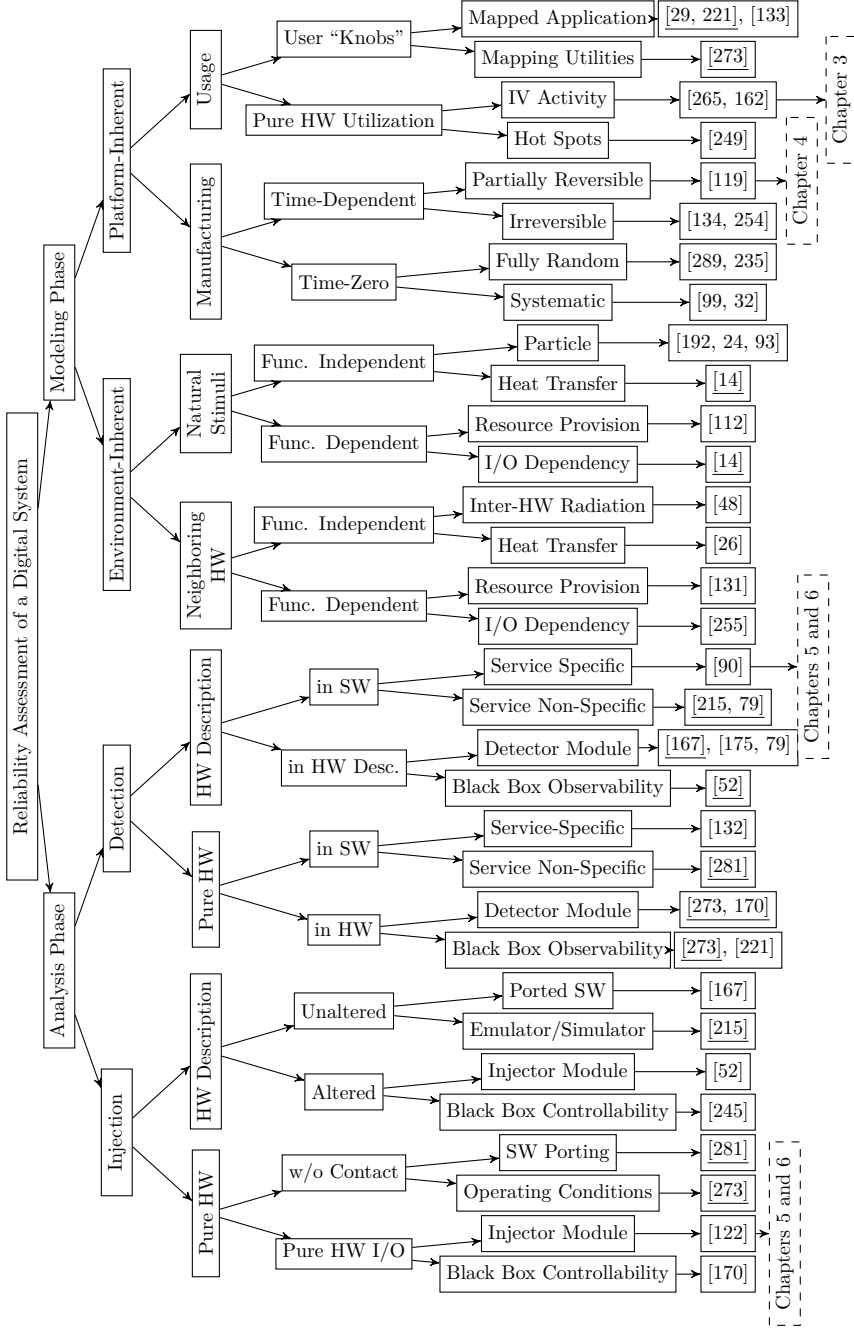


Figure 2.1: Classification framework for analysis and modeling of reliability violations on digital systems

2.3 Parametric Mitigation Within Package

Having covered the analysis and modeling domains, we move towards mitigation techniques against reliability violations. In this Section, we cover techniques that are concentrated within the packaging of the electronic system. The corresponding classification can be seen on the left branch of Figure 2.2. We further split this in order to equally address single-die and multi-die arrangements, which we assume are integrated within the same packaging.

2.3.1 Single Die

Parametric reliability techniques for a single die can be further split between *pre-fabrication* and *post-fabrication*.

Before the die is fabricated and packaged, we can identify parametric mitigation techniques in two complementary domains: either *design related* or related to *process technology*. With respect to the various design stages, which finally lead to the tape-out, parametric mitigation can be applied either by *optimizing the existing design* (without adding new features) [126, 10, 286] or by *enhancing* the design with additional capabilities for parametric reliability [284, 209]. In the category of process technology, we include all mitigation techniques that are employed during the processing of the wafer so that the target design can be taped out with minimal parametric violations. Such techniques may pertain either to the choice of *materials* [266, 82] or the *lithography* techniques [286, 285, 205].

After the die has been fabricated, one option is to compensate *before bringing the die to the market*. Alternatively, techniques are available that can be employed *after the chip has been packaged and brought to the market*. Before the die is marketed, available techniques can be further split between *non-intrusive* [238, 153, 165] and *intrusive* [80, 206, 70]. Reliability techniques may take effect after the die is marketed and can be split between *reactive* [291, 216, 180] and *proactive* [180, 282].

2.3.2 Multiple Dies

In this domain we assume that a set of dies is to be packaged into a single discrete component [63, 202]. We split our classification between the *integration* of dies and the addition of *extra features* to the final System in Package (SiP) [194] (i.e. other than the dies that provide system functionality).

With respect to multi-die integration, we can look into the *wiring* at the various levels or the *stacking* procedure per se. The wiring counterpart can be easily split between the two most available options, i.e. *mechanical* (e.g. wire bonding) [248] and *thermal/chemical* (e.g. soldering) [21, 19, 36]. With respect to die stacking (irrespective of conducting connections), we distinguish between *arrangement* (i.e. deciding on the functionality of each die level) [51, 46, 30] and the *selection* of specific die samples to be combined in the final SiP [28, 81, 89].

So far, we have addressed the issue of multi-die integration in a single package in terms of the interconnection and stacking aspects. Typically, extra components may be added to facilitate the integration process. We split the current branch based on whether an *interposer* is relevant to the reliability of the system [150]. The following two leaves address 3D integration without an interposer (either this component is ignored or not assumed). Through Silicon Vias (TSVs) and cavity cooling are the main focus and we distinguish between *adaptivity* [38, 237] and *redundancy* [130, 105, 38] as a mitigation against parametric violations. In case an interposer is involved in parametric reliability, we distinguish [264] between *active* [21, 262, 252] and *passive* [19, 164, 45] interposers .

2.4 Parametric Mitigation Across Packages

The current Section discusses parametric mitigation for discrete component systems (including a processor, memory and storage periphery). We are drawn outside the semiconductor packaging and inspect either Embedded or HPC systems (running on real or virtualized resources). For instance, reliability issues at a single node may cause system-wide parametric violations. A mitigation technique is then applied to alleviate the consequences of an erroneous component. Solutions tend to aim either at parametric targets (e.g. performance) or parametric headroom (e.g. thermal dissipation or availability). The first split is between *hardware* and *software* solutions.

2.4.1 Hardware Solutions

The current branch will be split based on the node granularity, namely a uniquely identifiable processing element, containing discrete components. Node examples include racks of a data center or an HPC cluster or the control units of an automotive electronic system [241, 267]. We distinguish prior art samples between *intra-node* and *inter-node* techniques.

Computing nodes are split between *general purpose* and *domain specific*, i.e. the ones that are tied to a specific domain (e.g. automotive etc). General purpose techniques are split between those that refer to the *processing* components of the node [222] and those that relate to the *memory* [188, 87]. A basic distinction among domain specific nodes is based on whether an *actuator* is integrated with the node. The thermal headroom and package-enabled dissipation capabilities of on-board flight entertainment computing nodes is discussed by Sarno et al. [239]. Thermal stress of automotive control modules are discussed by Kumar et al., with emphasis on heat spreading options and copper thickness of the printed circuit board (PCB) [148]. Von Tils discusses the parametric reliability of automotive components [275], isolating the semiconductor solutions for the automotive domain and providing examples of thermal-aware discrete components. Thermal boundaries of discrete components for electric traction-drive systems (ETDSs) are discussed by Moreno et al. [186], as well as air-cooling-friendly, wide bandgap electronic materials.

The inter-node branch covers parametric reliability techniques that are applied collectively across all nodes of a system. A distinction is made between policies that affect *operation* of a set of nodes and those that refer to the *management of data* between the nodes. In the former category, we further split between collective node *configuration* [83] or hardware techniques pertaining to the *grouping or partitioning* of nodes [220, 44]. Regarding data management a complementary split yields a distinction between *wired* [198, 88] and *wireless* [113, 160] system interconnection.

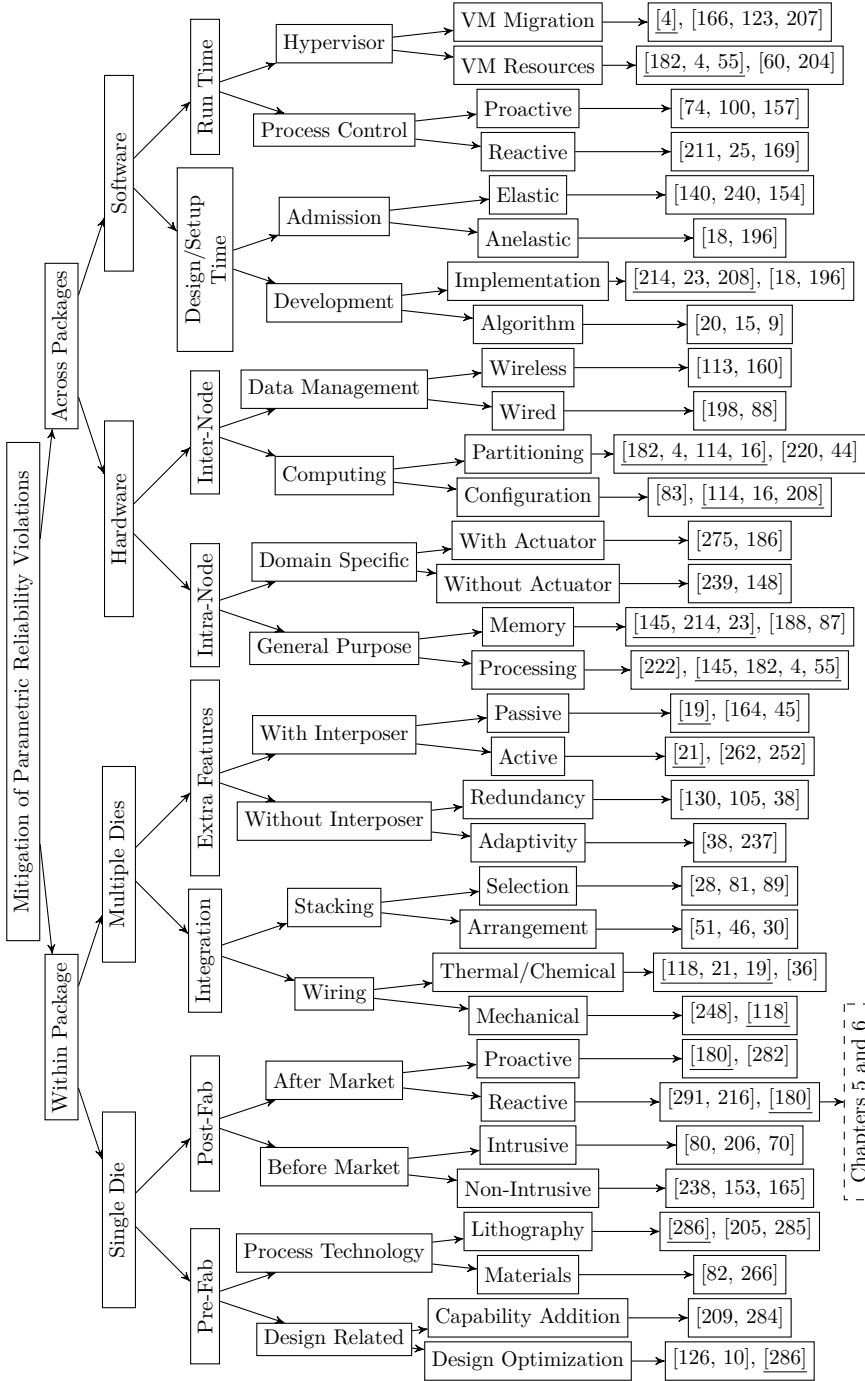
2.4.2 Software Solutions

As the complexity of systems increases, software solutions for parametric reliability become even more prevalent given the wide coverage they can provide. We split between *design or setup time* techniques and those that take effect at *run time*.

Before software is deployed, techniques for parametric reliability can be applied either during software *development* or as policies for the *admission* of software workloads in the system. On the development side, we distinguish between *algorithmic* aspects [20, 15, 9] and code *implementation* techniques that aim at parametric reliability [18, 196]. Admission-related techniques for parametric reliability are split based on the elasticity [140] of the target system. We note that both physical or virtual target systems are considered. Shin et al. discuss fixed priority scheduling in multi-node systems with strict latency constraints [246]. The work of Swaminathan et al. looks into the schedulability of I/O devices for hard real time systems [259]. Konstanteli et al. propose a cost-

conscious admission scheme for elastic cloud services [140]. The work of Schad et al. focuses on the performance variability of the Elastic Computing Cloud from Amazon and correlates availability zones with dependable performance [240]. Lin et al. propose a task scheduling solution for mobile cloud computing (MCC) environments [154].

The runtime domain covers parametric reliability solutions that are employed during system operation. A distinction is made between the *direct control of an operating system process* and *hypervisor-like* utilities. The process control side is further split between *reactive* [211, 25, 169] and *proactive* techniques. On the complementary branch, we look into software that typically handles instances of virtual machines (VMs). We split between techniques that address the *resource provisioning* [60, 204] of VMs and those that deal with VM *migration* [123, 207].



Chapters 5 and 6

Figure 2.2: Classification framework for parametric reliability techniques

2.5 Overview of Representative Groups

Apart from a conceptual breakdown of the target research domain, it also useful to present representative groups in the field of variability modeling and mitigation. It is important to note that the current Section keeps a neutral attitude towards the jury, in the sense that none of the jury's affiliations is presented as a related group. The interested reader can easily substantiate the breadth and depth of the jury's expertise in modeling and mitigation of system variability, ranging from the transistor to the system level. Furthermore, the presentation of representative groups is limited to ten samples that have been coming up very frequently during the research presented in this text. It stands to reason that much more related groups exist, however, for the sake of brevity, the current Section is limited to a sufficient subset. Finally, a lengthy and complete presentation of each group is beyond the scope of the current Section. On the contrary, examples of major contributions are presented, along with the link to details of each group for any interested reader to delve into.

Reliability-Aware Microarchitectures Group, IBM (USA)

This IBM group [107] has contributed in the formulation of modeling and validation concepts used to assess the reliability of digital systems. A case can certainly be made for the concept of Failures-in-Time (FIT) of a representative circuit (FORC), which encapsulates the module-level error rate based on that of a representative subcircuit [31]. Such reliability primitives have enabled the system-level assessment of reliability and the proposal of mitigation techniques to ensure dependability.

Robust Systems Group, Stanford (USA)

The concept of cross-layer reliability [183] has been strongly advocated from this group [256] and many derived solutions cover both the circuit and system level [180]. Also, this group boasts important prior art samples on the modeling and mitigation of variability for beyond-CMOS fabrics [292].

Chair of Dependable Nano Computing, KIT (Germany)

The research of the above group [135] covers both functional and parametric aspects. Also from KIT, the Chair of Embedded Systems [136] is equally noteworthy, featuring the DFG SPP 1500 project on Dependable Embedded Systems [137]. Within this initiative, fundamental dependability trends have been outlined, especially for aggressively downscaled technologies [103].

Robust Integrated Systems Group, TIMA (FR)

This TIMA group [111] is highly active in the modeling and mitigation of

erroneous behavior in digital systems. This group features strong background in error correction and protection techniques [193].

Michigan Integrated Circuits Laboratory, Michigan (USA)

One of the most important contributions of this group [272], which is highly related to RAS overheads as presented in the current text, is the RAZOR processor [75]. This implementation has been extensively discussed and documented. Generally, it relies upon timing speculation to correct errors using fine-grain pipeline rollbacks. Thus, it constitutes a representative example of timing costs being paid to ensure processor functionality. This links directly to the concept of cycle noise as presented in Chapter 6 of the current text.

Institute for Microelectronics, TU Wien (Austria)

This group [270] is highly active on the modeling of transistor variability. It features a wide range of work on the atomistic modeling of defect sites in the gate-oxide interface and gate stack, also emphasizing on novel modeling principles for BTI and RTN [94].

RSim Research Group, UIUC (USA)

Apart from focus on highly parallel hardware and software, this group [271] boasts significant system-level contributions on dependability. One of the most noteworthy samples in is the Reliability-Aware MicroProcessors (RAMP) lifetime reliability model [217].

2.6 Current and Future Reliability Needs

When addressing the issue of time-dependent variability, it is very important to realize the trends and attributes that semiconductor manufacturing is phasing at the currently applicable era. From the Economist graph of Figure 1.1 one can get a historical view about the number of transistors that can be effectively bought with one US dollar at a variety of technology nodes [258]. During previous decades, the community witnessed an aggressive scaling of semiconductor technologies [185]. However, nowadays it is clear that an inflection point is encountered as transistors reach (below) deca-nanometer dimensions.

A similar observation can be derived for the scaling of processor frequencies. The saturation point of single-core performance gave rise to multi- and many-core platforms so that computing performance is effectively leveraged [84]. By analogy to transistor scaling, processor vendors are facing the task of efficiently utilizing the silicon real estate, in order to provide users with the desirable

amount of performance without running into power distribution obstacles or, in general, design feasibility issues [76].

In view of the above trends, it is clear that reliability techniques need to adapt to the environment of electronic design and manufacturing. Apparently, this is a gradually constricting ecosystem, with constraints from material science and design complexity being amplified with time. We shall see, for instance, that semiconductor material variability itself may be perceived as an upper bound for system performance, unless proper techniques are implemented.

As computing penetrates increasingly more aspects of human activity, the degree of integrated processing power is also increasing [72]. Even if process technology may deliver transistors of improved resiliency, the large number of integrated components brings about dependability and reliability as very relevant design aspects of future systems. In an effort to mitigate inflection points in performance and efficiency, the global computing community experiences holistic approaches [195], featuring partnerships between system-level and technology-level bodies [61, 250]. Reliability and dependability are no strangers to holistic approaches, as conveyed by trends such as that of cross-layer resilience [183].

When looking at worldwide research trends in computing, we can isolate many domains where increased system resiliency and robustness are deciding factors. In the domain of *Embedded Systems*, an increase in the number of operations per unit power is seen as a crucial trend for a variety of critical applications of the future [59]. Meanwhile, such power efficient modes of operation typically come with partially compromised resiliency [59]. Hence, it is clear that if power is to be reduced, effort should be invested in keeping the functionality and performance of the hardware intact. Similarly, in the *high performance computing (HPC) domain*, the performance impact of resiliency techniques has already been documented and formulated for the case of the Tianhe2 supercomputer [288]. With an ever increasing number of processing nodes, the concept of the *Reliability Wall* has been clearly outlined as an upper bound in performance due to the invocation of resiliency techniques, such as checkpoint and restart [158]. From the above observations, it is quite clear that international research schemes do emphasize on the reliability and dependability as key features of current and future computing.

The above trend can be easily identified in the context of European projects as well. Figure 2.3 provides a crude schematic timeline of EU projects related to reliability and dependability, whereby the distinction between dependability modeling vs. mitigation is also highlighted.

To begin with, we can identify projects that are quite focused on the reliability of computing fabric as in the case of MODERN [184] and SUPERTHEME

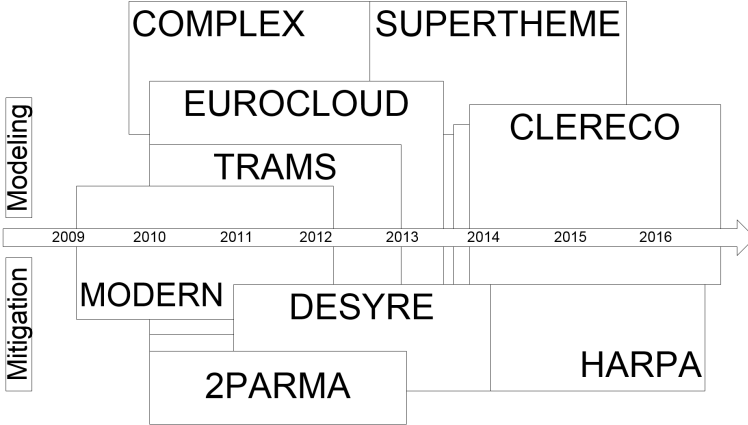


Figure 2.3: Crude timeline of indicative European projects in the vicinity of performance dependability and reliability [228]

[257]. In these cases, the primary focus is on the semiconductor phenomena and insight is provided in making the designers aware of variability and degradation phenomena, by developing novel design and simulation techniques. In other cases, such as TRAMS [268], information about the computing fabric is exploited even further and proposals for by-design reliable systems have been disclosed. EUROCLOUD [77] presents an interesting case of dependability and fault tolerance concepts applied on servers-on-chip for a highly efficient cloud infrastructure. The DESYRE project [65], has also been focused in developing reliable Systems-on-Chip (SoCs) under different types of fault occurrences. Reliability is also treated in the context of design space exploration, as in the case of the COMPLEX project [54]. This approach is fully materialized within the CLERECO project [53], where the goal is to provide solutions for early dependability analysis of computing platforms. Finally, system availability has been treated as a resource management problem in the context of the 2PARMA project [7]. Finally, the HARPA project maintains a cross-layer perception of performance variability (from the transistor to the application level) by featuring partners that are appropriately distributed across the abstraction stack.

2.7 Critical Appreciation and Contributions

In the previous five Sections we have covered the entire domain of reliability modeling and mitigation, with specific emphasis on parametric violations (which represent the target domain of time-dependent variability). Apart from creating

the classification frameworks, we have mapped prior art samples to the leaves of the former, so that the validity of our classification is completely substantiated.

In the current Section, we will reflect on the mapping of this text's next Chapters to the classification frameworks mentioned above. Detailed and specialized prior art listings appear in each of the following technical Chapters. Here, we incorporate the technical contributions of the current text in the overall valley of reliability analysis, modeling and mitigation solutions. Schematically, the correspondence of technical Chapters of this text to the leaves of the framework presented above are shown in Figures 2.1 and 2.2, respectively.

Chapter 3: This Chapter deals with the modeling of BTI and RTN. The focus of this part of the text is the compression of signal activity, in order to enable reliability predictions across extended time intervals (e.g. throughout the circuit lifetime). This modeling Chapter deals with the HW utilization at the level of circuit and especially at the level of the voltage waveform that is applied to the transistor gate. As a result, we classify Chapter 3 to the *IV Activity* branch of Figure 2.1. The novelty of this Chapter lies on the systematic compression. It will be demonstrated that, in comparison to previous approaches, the CDW succeeds in balancing the accuracy vs. computation time in circuit aging analysis. The core of the work presented in Chapter 3 has been previously disclosed by the author in an IEEE TDMR journal paper [234].

Chapter 4: The MPFP methodology is used in this Chapter in order to deliver time-dependent failure probabilities for SRAM components. The target silicon aging phenomena are, again, BTI and RTN. A partial recovery component has long been identified for these phenomena (corresponding to the emission of minority carriers in the gate stack) [119]. As a result, it is valid to classify this Chapter under the leaf *Partially Reversible* of Figure 2.1. This Chapter is based on the well known MPFP and proposes extensions for circuits other than memory cells. It also discusses paths of digital standard cells. The core of the work presented in Chapter 4 has been previously disclosed by the author in two workshops [233, 232].

Chapter 5: This Chapter deals with the implementation of a functional mitigation technique (fine-grain rollbacks) and the performance implications thereof. Static DVFS boosts are proposed to enable performance dependability of the experimental setup. Even though this is clearly a mitigation technique, it is verified with extensive reliability analysis campaigns. Given the scope and implementation of the concepts discussed in this Chapter, we can classify it, from a reliability analysis perspective, under leaf *Injector Module* in Figure 2.1. From a mitigation perspective the proposed technique is

clearly *Reactive*, hence it is classified accordingly in Figure 2.2. The core of the work presented in Chapter 5 has been previously disclosed by the author in an IEEE TVLSI journal paper [230].

Chapter 6: This Chapter deals with the formulation and generalization of the mitigation technique presented in Chapter 5. As such it falls under the same classification leaves as discussed above. The main novelty featured in this Chapter is the casting of the performance vulnerability problem from cycles [138, 102] to time units. This is a much required translation of the problem, since it connects RAS temporal overheads with the overall quality of service provided by a processor. Additionally, the DVFS capabilities that appear in abundance in modern processors are appropriately exploited in the concept presented herein. The formulation and reductions to practice presented in this Chapter have been previously disclosed by the author in one IEEE journal paper and an ACM journal submission [227, 226].

Chapter 3

Compact Digital Waveform for Circuit Aging Modeling

This Chapter discusses the usability of a novel signal representation format for efficient modeling of integrated circuit aging. Especially at aggressively downscaled nodes, semiconductor aging phenomena are getting increasingly complex to predict. In the Chapter, a case is made for Bias Temperature Instability (BTI) and Random Telegraph Noise (RTN). We start from some general background on BTI and RTN and discuss the most prevalent samples of prior art on this domain (Section 3.1) and also focus on the inner workings of atomistic aging models. Then, we present the Compact Digital Waveform (CDW) and discuss its applicability for BTI/RTN modeling (Section 3.3). An extensive verification of this signal representation format is performed in Section 3.4. The CDW's applicability in terms of bias- and temperature-awareness is discussed in Section 3.5. Finally, the compatibility to standard design flows is discussed in Section 3.6. We need to note that the largest part of the current Chapter has been already disclosed in a previous publication [234].

3.1 Background

Bias Temperature Instability (BTI) has been examined for more than 30 years [115] as a major concern for integrated circuit (IC) reliability. It causes the absolute threshold voltage (V_{th}) increase under certain bias and temperature conditions. Typical simulation involves a degradation model assigning V_{th} shifts to each transistor and timing analysis employed to reveal timing fluctuations.

The Reaction-Diffusion (RD) model for BTI has triggered prior art in both modeling [12, 201, 276, 243, 168] and mitigation [147, 43, 146, 201, 33]. This model emphasizes on the breaking and annealing of Si-H bonds at the Si/oxide interface and mainly focuses on pFETs (NBTI). As a pFET is stressed, the model argues that minority carriers (holes) are replacing H atoms in Si-H bonds at the Si/oxide interface, thus causing a shift to the pFET's threshold voltage (V_{th}). Once the stress is removed from the pFET's gate (off state), a subset of the Si-H bonds is annealed, thus partially restoring V_{th} shifts [49]. However, the RD model has been shown to deviate from experimental data at aggressively downscaled technologies. Thus, it is motivated to adapt circuit simulation to the deca-nanometer technology paradigm [94].

Atomistic approaches do move to this direction, additionally capturing the effects of Random Telegraph Noise (RTN) in the same model. Certain transient implementations of the atomistic model [229, 120, 41] are very accurate but are very processing- and memory-intensive, which sometimes causes them to fail. Hence, their computational feasibility *cannot be guaranteed*.

What makes atomistic approaches different is that, instead of having an analytical formula that predicts average aging, such models account for matter discretization in a more detailed manner: Each defect in the gate stack is independently modeled, with its V_{th} impact (ΔV_{th}) and time constants for the capture and emission of minority carriers. When a minority carrier is captured by a defect, then the appropriate ΔV_{th} is superimposed to the transistor's V_{th} . Even though extremely accurate, this technique is very time-consuming given that different transistors have different numbers of different defects. This difference between average RD and atomistic models for BTI is clarified schematically in Figure 3.1.

It is worth noting that, since the initial papers [120, 229] and patent disclosures [41] on atomistic BTI/RTN modeling, an attempt to scale the complexity of this task has been made by embedding the atomistic model's code within the source of code of an open source SPICE implementation and scaling it with a hypervisor [161]. This technique succeeded in removing the memory wall of time-driven, atomistic BTI/RTN simulation, especially in comparison to commercial SPICE solvers [229].

In any case, in order to perform atomistic simulations for the entire lifetime of a typical integrated circuit, it is very important to enable strides over circuit workload. This is exactly the direction of the current Chapter and of the general concept of the Compact Digital Waveform (CDW).

One of the attributes that became rather apparent since the first SPICE-compatible atomistic modeling papers [120, 229] is that such models succeed in

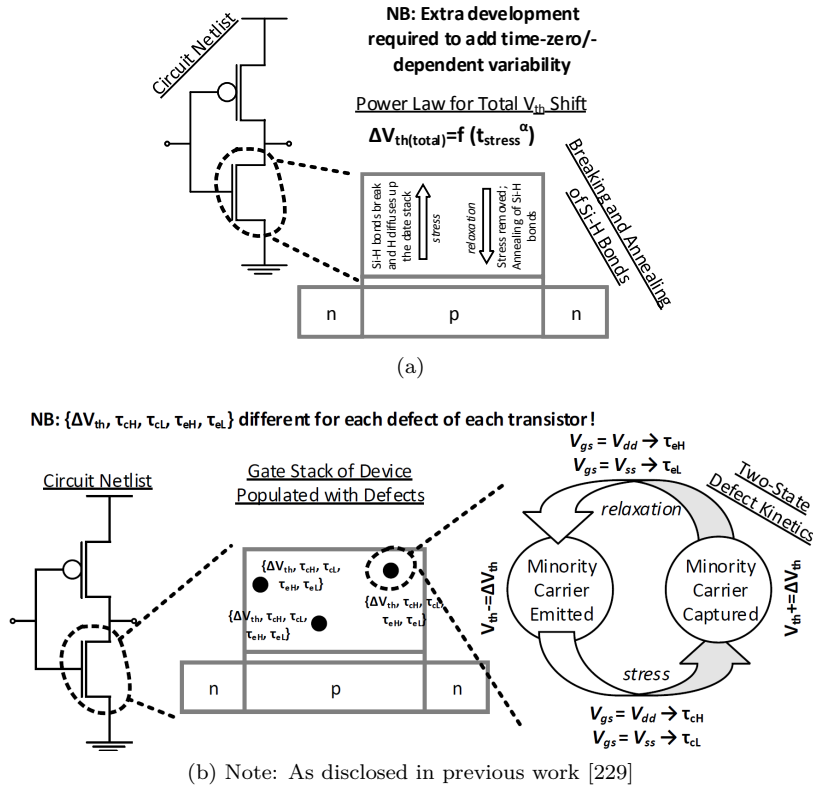


Figure 3.1: Artist's illustration of the differences between the average RD model and the sought-after atomistic modeling schemes for BTI/RTN

maintaining *both* the stochastic nature of minority carrier capture and emission *and* the workload dependent attribute of this phenomenon. These are two aspects of gate stack defect interference which should remain coupled for the BTI/RTN study to be accurate. The former component is rather trivial, in the sense that the capture and emission of minority carriers are quantum-mechanically described, hence absolutely deterministic modeling of these mechanisms is out of the question. However, there is a strong *workload dependency* in BTI/RTN, in the sense that certain V_{gs} signals at the transistor gate favor certain degradation patterns. The combination of these two attributes has been substantiated in the lower levels of the circuit abstraction for simple memory simulations [229]. One of the fundamental goals of this Chapter is to retain the combined stochastic and workload dependent model for BTI/RTN, while reducing its computational complexity.

3.2 The Atomistic BTI Model

The atomistic approach towards BTI modeling focuses on the trapping of charge at gate stack defects. Such an approach maintains an agnostic view regarding defect site creation and rather focuses on the capture and emission of minority carriers. A very important metric in a *charge-trapping* model is the capture probability (P_c) of each simulated gate stack defect.

Definition 3.1. Capture Probability (P_c) is the probability of a minority carrier being captured by a gate stack defect after a signal has been applied to the respective gate (V_{gs}).

The charge-trapping model is also *atomistic*, since it replicates the granularity of charge and matter across the gate stack in a realistic way. This translates to certain model attributes, originating from the random structure of each considered trapping site [13]. Defects capture and emit minority carriers at various rates. This is modeled using *time constants* for the capture and emission processes at a high or low level (τ_{cH} , τ_{cL} , τ_{eH} , τ_{eL}). The impact of each individual defect on the overall V_{th} shift is also variable and is modeled with a signature ΔV_{th} per defect. Starting from measurements on real devices, related work has shown that the τ_{cH} , τ_{cL} , τ_{eH} , τ_{eL} and ΔV_{th} parameters of each defect follow certain distributions [119, 265], thus enabling atomistic modeling of BTI.

P_c depends on the time constants of each defect and the characteristics of the V_{gs} signal, i.e. its frequency f , its duty factor α and its duration Δt . It also depends on the initial value P_{c0} . At each point of the V_{gs} representation, P_c is evaluated and the occupancy of the respective defect is resolved by comparing P_c with a random number r . The defect is assumed occupied by a minority carrier iff $r < P_c$ and its ΔV_{th} is added to the device's V_{th} . According to the V_{gs} signal representation format, we look into different evaluation methods for P_c . The granularity of a signal representation format is coupled to the CPU time required for BTI/RTN reliability analysis, especially if one realizes that P_c evaluation has to be performed for each time point, for each defect of each transistor in the target circuit (see Figure 3.1b). We look into the available and established signal representation formats:

SPICE: This tool has been considered as the industry standard for IC simulation [190]. In the transient simulation mode, signals are adaptively discretized across time. We can assume that this is the most accurate signal representation. In previous work [120, 229], we have seen transient, workload dependent implementations of the atomistic BTI model. The model's core is Equation 3.1. Workload memory is supported when the $P_c(t)$ value of the previous step is included in the formula. Otherwise

(when $P_c(t) = 0$), the entire simulation is treated as a Monte Carlo iteration.

$$P_c(t + t_s) = \frac{\tau_{ei}}{\tau_{ei} + \tau_{ci}} + \left[P_c(t) - \frac{\tau_{ei}}{\tau_{ei} + \tau_{ci}} \right] \exp \frac{-t_s(\tau_{ci} + \tau_{ei})}{\tau_{ci}\tau_{ei}} \quad (3.1)$$

VCD: This representation creates an abstraction of voltage signals to logic levels [3]. We will assume that the VCD format includes only the high (V_{dd}) and low (V_{ss}) levels. Given a VCD file for V_{gs} , we can calculate P_c using Equation 3.2, either for the high ($i = H$) or low ($i = L$) logic level [265]. This option is clearly workload dependent, given that P_c is a function of the duration of constant voltage levels (t_i). It is also a model with memory, given that P_c is a function of its previous value ($P_{ci'}[n-1]$). However, in the course of a system's lifetime, a BTI/RTN reliability analysis using a VCD activity file is computationally prohibitive.

$$P_{ci}[n] = \frac{\tau_{ei}}{\tau_{ei} + \tau_{ci}} + \left\{ P_{ci'}[n-1] - \frac{\tau_{ei}}{\tau_{ei} + \tau_{ci}} \right\} \exp \frac{-t_i(\tau_{ci} + \tau_{ei})}{\tau_{ci}\tau_{ei}} \quad (3.2)$$

SAIF: This format focuses on the switching activity of the nodes in a digital circuit [2]. It contains the number of alterations between logic levels and the percentage of time spent at each logic level. The latter quantity is also referred to as *duty factor* [265] or *signal probability* [147].

Previous work [265] has demonstrated an AC model that estimates P_c from the time zero instance of device lifetime. In order to calculate P_c after n digital pulses have been applied to the device's gate, Equation 3.3 is used, where constants a and b are functions of the f and α characteristics of the V_{gs} pulse series. This model *is indeed workload dependent* but, in its current formulation, it contains *no memory* of previous defect condition. As a result, it serves as a very good tool for initializing defect states [120, 229] but cannot provide an accurate *time dependent* view of defect activity. In Section 3.3 we extend this formulation to systematically incorporate memory of previous defect states.

$$P_c[n] = \frac{b}{a} (1 - e^{-an}) \quad (3.3)$$

A statistical approach, derived from the atomistic model and assuming constant AC stress, involves Capture and Emission Time (CET) Maps [279]. A distribution of V_{th} values is returned, given a single triplet of f , α and Δt values for the entire V_{gs} signal. Similarly to the above model, time zero is assumed as the initial condition. The formulation of Weckx et al. is orthogonal

to approach of digital waveform compression and can be used as the core BTI resolution kernel, instead of the aforementioned P_c formulation. As such, the CET-map-based methodology is left beyond the scope of the current text.

3.3 Proposed Tool Support

In the previous Section it has been thoroughly clarified that the execution time of a BTI model is proportional to the number of points at which V_{th} values are to be evaluated. As a result, any attempt to simplify the complexity of BTI/RTN modeling requires tackling the number of points contained in the utilized signal activity representation. In the current Section, we propose a waveform representation called *Compact Digital Waveform (CDW)*. Being of tunable accuracy, this format exploits optimally the accuracy vs. complexity trade-off. In this Subsection, we will show how this format is ideally positioned between the computationally intensive SPICE or VCD files and the over-simplifying duty factor derived from a SAIF file. This format will be used as a framework for computationally efficient, atomistic BTI simulations.

Let us assume a transient waveform, which in the most detailed case is represented as a SPICE signal. Here, we propose the grouping of consecutive signal regions that feature a similar frequency (f) and duty cycle (α) figures and occupy a duration Δt . Each region of the signal that adheres to this property is represented by a single CDW point with coordinates $(f, \alpha, \Delta t)$. To derive these points, we scan the initial signal, which for the sake of simplicity is represented by a VCD equivalent. Thus, the input can be perceived as a succession of low and high voltage levels, i.e. a sequence of AC pulses that have, in the general case, varying f and α values. As the digital signal is scanned, a buffer accumulates pulses that satisfy Equation 3.4.

When we come across a digital pulse that cannot be appended (i.e. does not satisfy Equation 3.4), the buffer is flushed to a single CDW point and re-initialized to receive the next pulses. ε_f (Hz) and ε_α (p.u.) are user-defined error margins. These margins enable the tunable accuracy of the CDW approximation. The crudest approximation is achieved with $\varepsilon_f \rightarrow +\infty$ and $\varepsilon_\alpha \rightarrow 1$, since all digital pulses are appended to the buffer. Conversely, for the finest approximation we set $\varepsilon_f \rightarrow 0$ and $\varepsilon_\alpha \rightarrow 0$. In that case, each individual AC pulse constitutes a single CDW point.

$$|f_{\text{pulse}} - f_{\text{buffer}}| < \varepsilon_f \quad \wedge \quad |\alpha_{\text{pulse}} - \alpha_{\text{buffer}}| < \varepsilon_\alpha \quad (3.4)$$

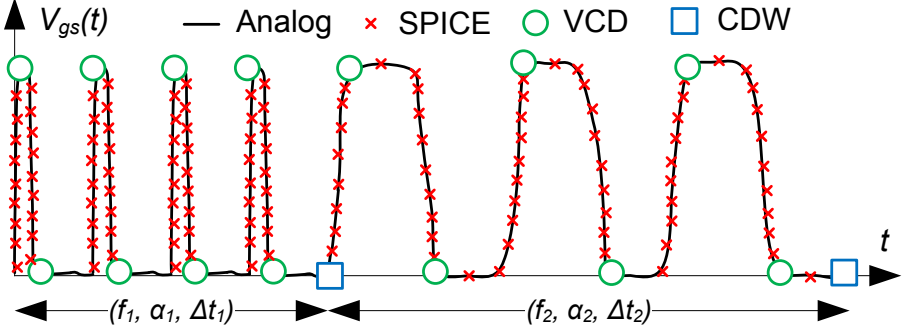


Figure 3.2: Artist's impression of a CDW compared to the SPICE and VCD waveforms: Signal regions with similar f and α are grouped in a single CDW point. This concept alleviates the complexity of atomistic BTI simulations over large time intervals, since it reduces the number of model iterations [234].

The reference values f_{buffer} and α_{buffer} are defined in Equation 3.5 for a buffer containing n pulses and are computed every time a new digital pulse is appended to the buffer.

$$f_{\text{buffer}} = \frac{1}{n} \sum_{i \in \text{buffer}} f_i \quad \wedge \quad \alpha_{\text{buffer}} = \frac{1}{n} \sum_{i \in \text{buffer}} \alpha_i \quad (3.5)$$

We have implemented the above concept in a simple tool, called **wavapprox**, which produces CDW representations given SPICE or VCD waveforms. A qualitative example of the produced CDWs can be seen in Figure 3.2. The functionality of **wavapprox** can be seen in Figure 3.3. In order to assess the accuracy of our tool, we use a random 1 μs bit stream of 1 GHz frequency. The CDW representation is compared against the respective SPICE waveform. We define the error of signal approximation according to Equation 3.6 and produce the mean (Figure 3.4a) and standard deviation (Figure 3.4b) values. Given that the **wavapprox** buffer averages-out the f and α values of the grouped pulses, we expect reduced impact of the tool on the mean error of our approximation (Figure 3.4a). However, the standard deviation (Figure 3.4b) is affected much more, given that a cruder approximation misses both voltage transitions and regions of constant voltage level.

$$e_s(t) = V_{\text{SPICE}}(t) - V_{\text{CDW}}(t) \quad (3.6)$$

Apart from the accuracy of the CDW representation from a signal point of view, it is important to consider the achieved compression in the number of points it

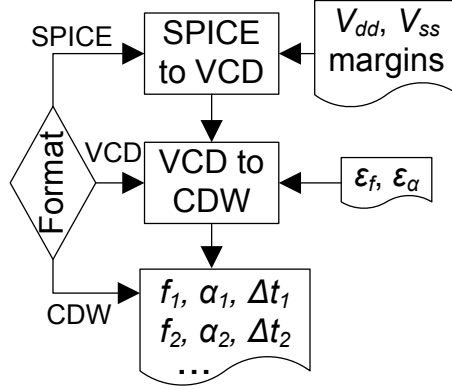


Figure 3.3: Flowchart of `wavapprox` tool: Each V_{gs} signal is handled according to its initial format, finally providing a CDW approximation [234].

uses for representation. In Figure 3.4c we see the number of points produced for various choices of the ε_f and ε_α error margins. The number of points used by any signal representation is directly coupled to the CPU time required for atomistic BTI/RTN simulation. The reason is that each point corresponds to a single iteration of the respective atomistic model. The aggressive reduction in the number of such points is the first step towards a computationally feasible atomistic BTI/RTN simulation. In the next Section, we explore the impact that aggressive CDW compression has on the accuracy of atomistic modeling.

3.4 CDW Verification for BTI Modeling

Given the compression capabilities of a CDW, it is important to explore its usability for our target goal, namely to execute the atomistic BTI/RTN simulation over strides of V_{gs} workload. We will be altering a version of the atomistic BTI model [265, 120] and will reformulate as follows: Starting from the premise of first order kinetics, the capture probability of a gate stack defect follows the ODE of Equation 3.7. The general solution returns P_c , after n V_{gs} pulses of specific f and α have been applied (equivalent duration $\Delta t = n/f$). Parameters a and b are functions of f and α [265, 120].

$$\frac{dP_c}{dt} = aP_c + b \Rightarrow P_c = \left(P_{c0} + \frac{b}{a} \right) e^{an} - \frac{b}{a} \quad (3.7)$$

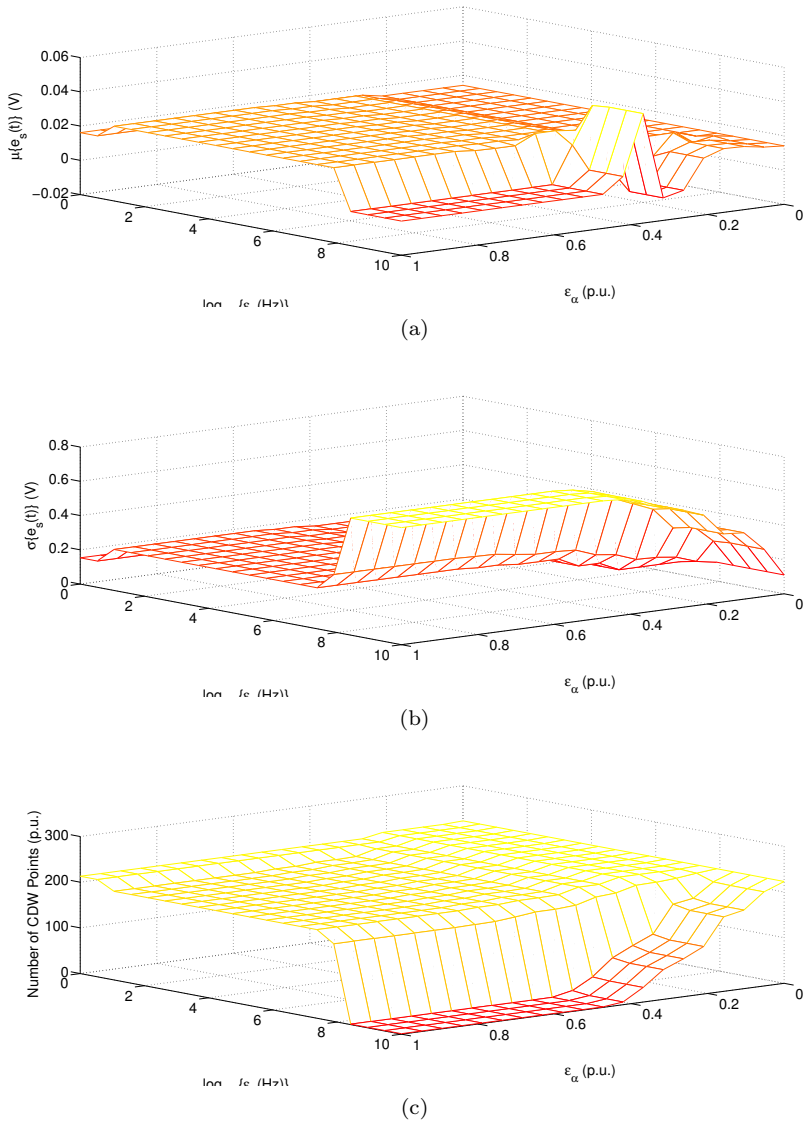


Figure 3.4: Accuracy assessment (signal integrity) of the **wavapprox** tool that produces CDW representations, using a 1 μ s bit stream. It should be noted that SPICE requires 17089 points to represent that signal and a VCD representation uses 521 points [234].

Each iteration of the general solution of Equation 3.7 is used as initial condition for the next model evaluation. The **wavapprox** tool, presented previously, identifies regions of similar f and α in the V_{gs} (i.e. a CDW representation is produced). Then, at each $(f, \alpha, \Delta t)$ point, a single iteration of Equation 3.7 is performed. A CDW representation reduces aggressively the number of model iterations required for the calculation of P_c . That way, time-dependent BTI simulation becomes computationally feasible over long time spans (see Figure 3.2). Given the compression capabilities of the CDW representation (see Figure 3.4c), the number of model iterations is aggressively reduced, thus decreasing the CPU time required for simulation.

In Section 3.3, we have assessed the accuracy of the **wavapprox** from a signal integrity point of view. Given the proposed model of Equation 3.7, we will evaluate the accuracy of a CDW approximation in terms of P_c evaluation. For our experiments we use 100 random bit streams with an average frequency (f_{avg}) of 1 GHz and a duration of 1 μ s. Given each signal, we define an error metric for BTI evaluation, according to Equation 3.8, where $P_{c,SPICE}$ is the capture probability for a defect at the end of the signal based on SPICE evaluation (i.e. using Equation 3.1). $P_{c,CDW}$ is a similar capture probability, using a CDW file (for a specific pair of ε_f and ε_α) and the model of Equation 3.7. We use 5 different cases, ranging from “fast” ($\tau_{cH} = 10^{-11}$ s) to “slow” ($\tau_{cH} = 10^{-3}$ s) defects. That way, we can calculate the mean, standard deviation and maximum of e_p across the 100 signals for various values of $(\varepsilon_f, \varepsilon_\alpha)$ and for the five different defects (Figures 3.5a – 3.5f).

$$e_p = P_{c,SPICE} - P_{c,CDW} \quad (3.8)$$

A direct correlation exists between how “fast” a defect is and the average frequency of the V_{gs} bit stream (f_{avg}). More specifically, the defects that are significantly “slower” in comparison to the average frequency of the bit stream are accurately modeled, regardless of the compression imposed by **wavapprox** (Figures 3.5a through 3.5d). Faster defects (Figures 3.5e and 3.5f) have time constants comparable to the bit stream’s period ($1/f_{avg}$). The proposed model is unable to follow the P_c of these defects, irrespective of ε_f and ε_α values.

The relation between f_{avg} and the defect time constants has been identified earlier [120], however no comprehensive study has been performed for the accuracy limitations of the AC BTI/RTN model [265, 120]. Given that a version of this model, enhanced with workload memory, is the kernel of our simulation methodology, we should exhaustively explore its accuracy limitations for each model iteration. More specifically, it is important to identify *for which defects and under which conditions* is the accuracy of our model acceptable.

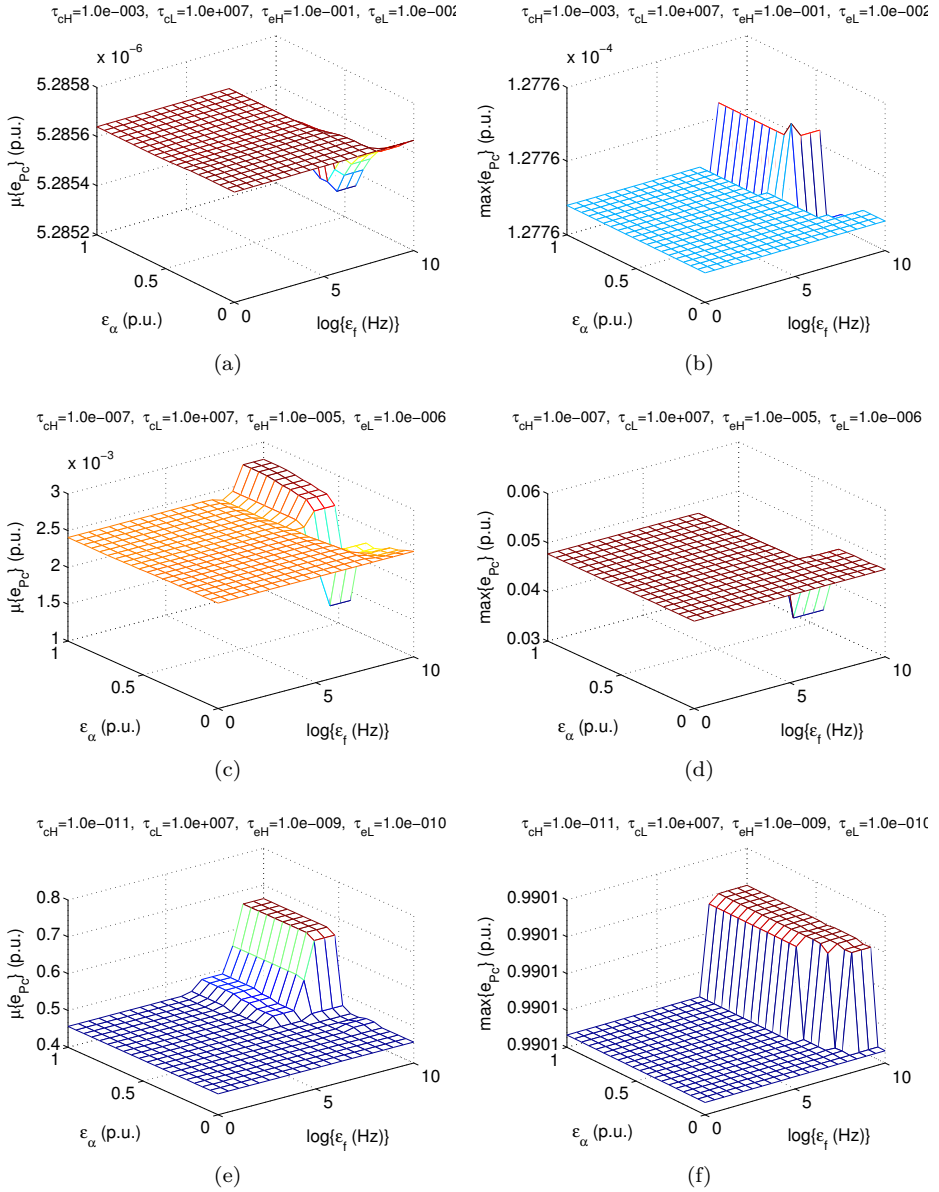


Figure 3.5: Accuracy assessment of the atomistic BTI model of Equation 3.7 for various degrees of CDW-compression [234].

In order to formally conclude on the accuracy limitations of our approach, we first create a set of 400 different defects, with time constants $(\tau_{cH}, \tau_{cL}, \tau_{eH}, \tau_{eL})$ uniformly distributed in the logarithmic scale. The defects are arranged in a two dimensional plane according to $\min\{\tau_{cH}, \tau_{cL}\}$ and $\min\{\tau_{eH}, \tau_{eL}\}$.

We use three sets of bit streams with f_{avg} equal to 1 GHz, 100 MHz and 10 MHz for each set. Each set contains 100 signals, each one with a duration of 1 μ s. Each signal is CDW-approximated with maximum compression ($\varepsilon_f \rightarrow +\infty$ and $\varepsilon_\alpha \rightarrow 1$) and e_p is calculated according to Equation 3.8. That way, we can calculate $\mu\{e_p\}$, $\sigma\{e_p\}$ and $\max\{e_p\}$ for each defect and for each f_{avg} case. The results are shown in Figures 3.6a through 3.6f. It is important to note that, these results represent the accuracy of the proposed model *for a single iteration* of Equation 3.7. However they provide useful insight on the accuracy limitations of the proposed model, when using it for an arbitrary number of iterations.

For all f_{avg} values, we can always find a subset of defects that we are unable to model accurately. As f_{avg} is increasing, this subset is reduced and an acceptable accuracy of our model spreads to more defect time scales. For the case of $f_{avg} = 1$ GHz, our model can accurately simulate all defects that satisfy $\min\{\tau_{cH}, \tau_{cL}\} \geq 10^{-7}$ s and $\min\{\tau_{eH}, \tau_{eL}\} \geq 10^{-7}$ s.

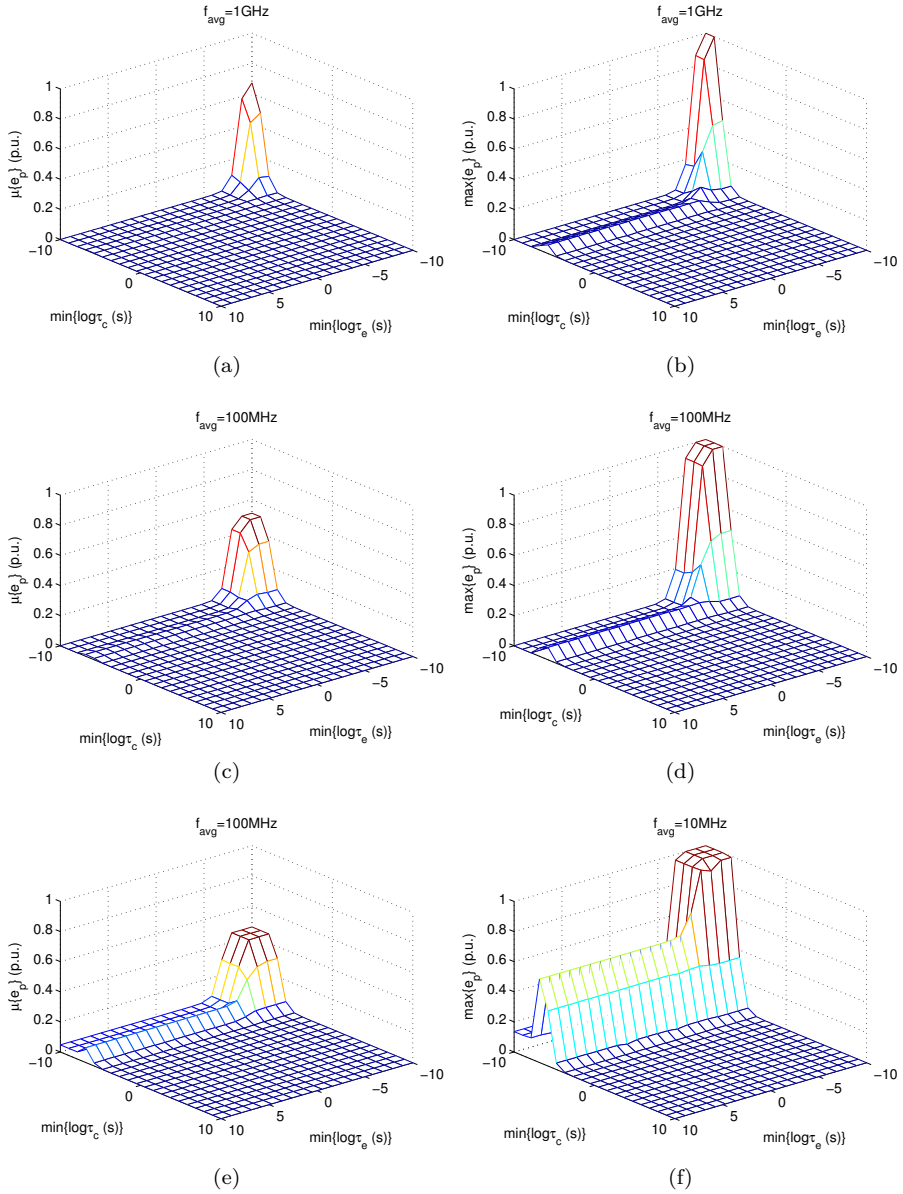
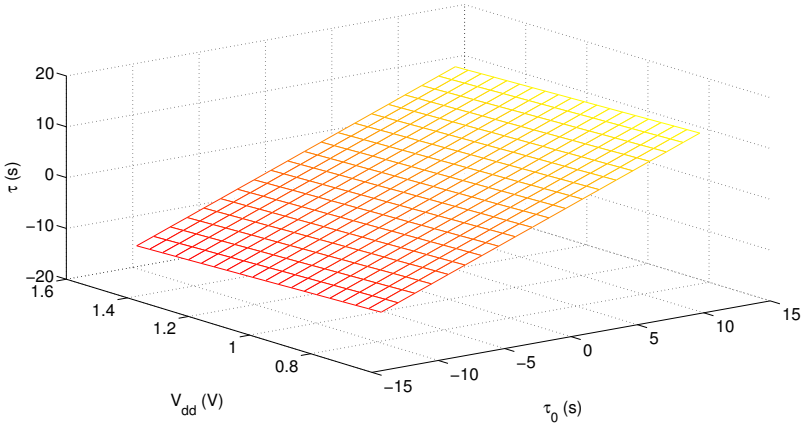
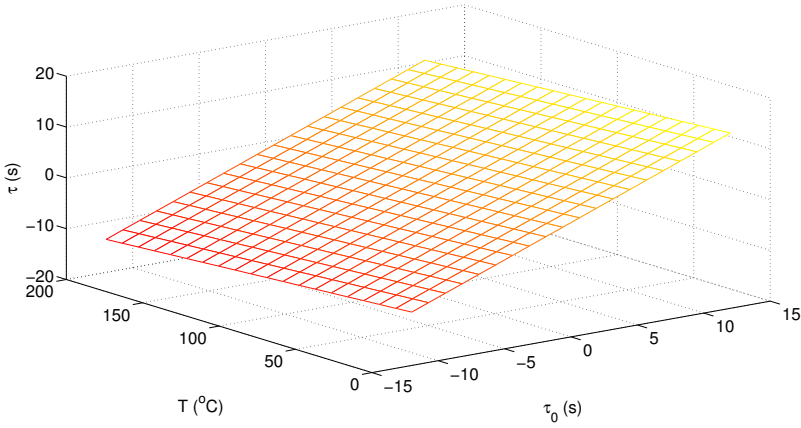


Figure 3.6: Errors from a single iteration of Equation 3.7 on bit streams of different f_{avg} . The error of the proposed model is restricted to small defect subsets, as f_{avg} is increased [234].



(a) Scaling defect time constants for various bias (V_{dd}) conditions, according to Equation 3.9



(b) Scaling defect time constants for various temperature (T) conditions, according to Equation 3.10

Figure 3.7: Equations 3.9 and 3.10 enable full support of bias and temperature variations in the CDW-based BTI simulation framework [234].

3.5 Thermal & Bias Applicability

With Dynamic Voltage and Frequency Scaling (DVFS) being increasingly common among modern ICs [34], it is important to also include the impact of voltage bias on the BTI simulation. Furthermore, aggressive integration may

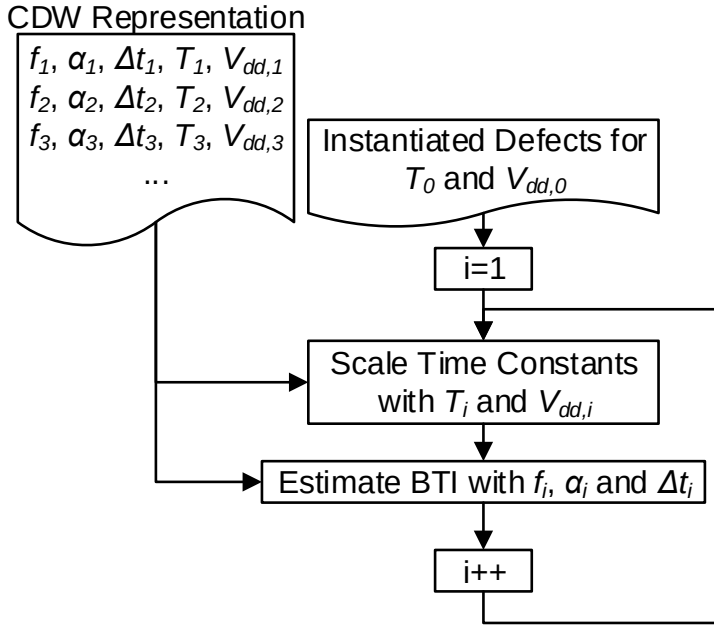


Figure 3.8: Before resolving occupancy of defects with the CDW, time constants need to be scaled [234]

also cause significant thermal gradients across an IC [249]. As a result, the BTI simulator should also be aware of the local temperature conditions. It has been previously demonstrated that the time constants of BTI/RTN defects are affected by the operating bias (V_{dd}) and temperature conditions. Based on state-of-the-art measurements [95], we propose two orthogonal scaling rules of a time constant, in view of temperature and bias variations (see Equations 3.9 and 3.10). These rules constitute a first order approximation of time constant scaling, as observed in measurements from real devices. We can see that in each case, the new time constant (τ) depends on its original value (τ_0), as well as the original and new value of bias ($V_{dd,0}$ and V_{dd}) or temperature (T_0 and T), respectively. The original temperature (T_0) and bias ($V_{dd,0}$) values refer to the conditions under which the time constant measurement was performed on a sample of devices from a specific technology node. The scaling factors (-2 for bias and -1/50 for temperature) are also technology-dependent. In the current text, we conform to related work measurements for these factors [265]. An example of bias and temperature scaling (applied individually) appears in Figures 3.7a and 3.7b respectively, for an initial temperature $T_0 = 125$ °C and an initial bias condition $V_{dd,0} = 1$ V.

$$\mathfrak{B}(\tau_0) = \log \{\tau\} = 2 \times (V_{dd,0} - V_{dd}) + \log \{\tau_0\} \quad (3.9)$$

$$\mathfrak{T}(\tau_0) = \log \{\tau\} = \frac{T_0 - T}{50} + \log \{\tau_0\} \quad (3.10)$$

It is easy to prove that the scaling operators $\mathfrak{B}$ and $\mathfrak{T}$ maintain a commutative composition, thus they can be applied in an arbitrary sequence. Equations 3.9 and 3.10 require an extension of the CDW representation in order to include bias and temperature information. Thus, before the occupancy of the defects is resolved, the time constants should be scaled appropriately (see Figure 3.8). It should be noted that temperature scaling affects all four time constants. Bias affects only the time constants that refer to a non-zero voltage, namely τ_{cH} and τ_{eH} for the nMOS and τ_{cL} and τ_{eL} for the pMOS.

From the results of Figure 3.6 we see that, given the time constants of a defect and the workload of the respective device, we can identify whether this defect is suitable for pseudo-transient simulation.

If that is the case, we can stride over V_{gs} intervals using a CDW approximation. This course of action involves a few iterations of the proposed BTI/RTN model and comes at a very small CPU overhead, compared to traditional atomistic approaches in SPICE or VCD. If the defect is too “fast” for the pseudo-transient model proposed herein, a more detailed simulation (with a SPICE or VCD waveform) is deemed necessary. However, its duration need not cover the entire device lifetime: the impact of fast defects can be seen within a small time interval. For the subset of the “fast” defects, we can derive a margin of V_{th} short-term variation that can be superimposed to the long term BTI/RTN impact derived by the pseudo-transient model (see Figure 3.10a).

In order to verify the claims of this methodology, we test it on a single device. First, we populate this device with defects and arrange them in the $\min \{\tau_{cH}, \tau_{cL}\}$ vs. $\min \{\tau_{eH}, \tau_{eL}\}$ plane (Figure 3.9). A bit stream with $f_{avg} = 1$ GHz is assumed as device workload. Given the results of Figure 3.6, we can assume that 10^{-7} s is the boundary between fast and slow defects. For the “fast” defects, we use a sample SPICE waveform, representative of the device workload (Figure 3.10b). With a brief simulation that covers the “fast” time constants, we receive the short-term impact of the respective defects (Figure 3.11a). For the long-term impact of BTI/RTN, we create a CDW representation of the device lifetime (until 10^8 s) using 100 points. As a result, each point has $f_{avg} = 1$ GHz and $\Delta t = 10^6$ s. What varies between CDW points is the α , which is visualized in Figure 3.10b. The long-term BTI/RTN impact is calculated using

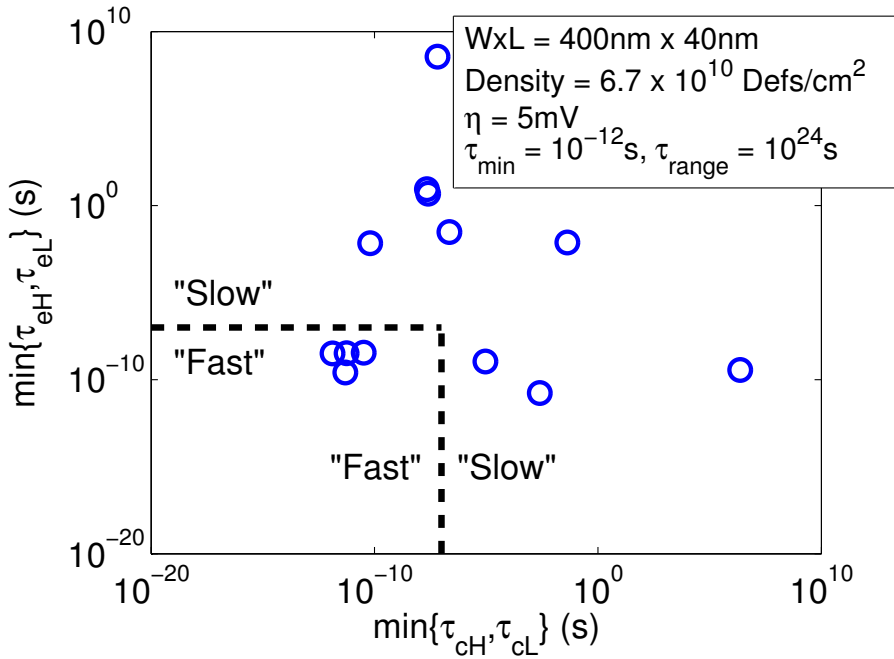


Figure 3.9: Distinction between “fast” and “slow” defects [234]

the model of Equation 3.7 and can be combined with the worst case impact of “fast” defects (Figure 3.11a).

The device workloads presented above are artificial. However, they can be easily derived from industry standard tools when handling realistic circuits and applications. A brief SPICE simulation of the target circuit can produce a representative workload for the “fast” defect simulation. α values for different circuit operation modes can be easily derived using timing analysis tools. The above methodology has no inter-device dependencies and can be generalized to a larger device inventory [161]. This data parallelism is highly compatible with the multi-/many-core paradigm [84] observed in the computing infrastructure of modern design houses [173].

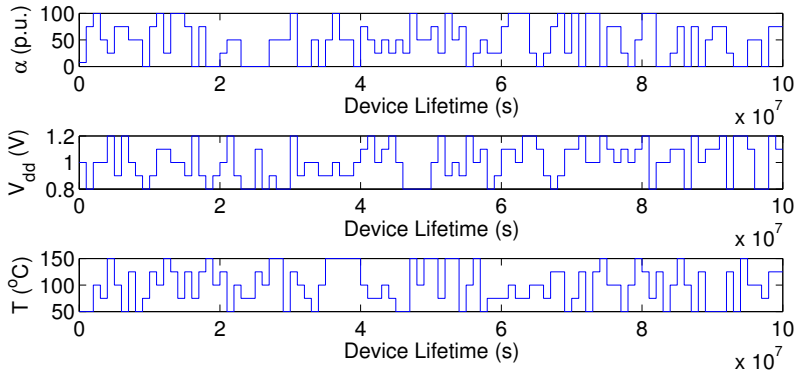
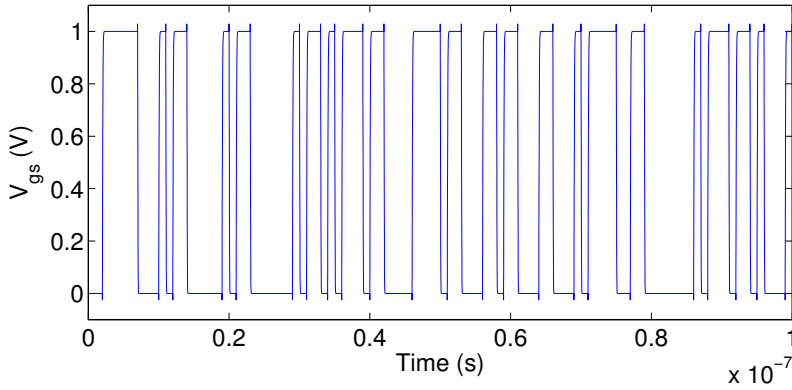
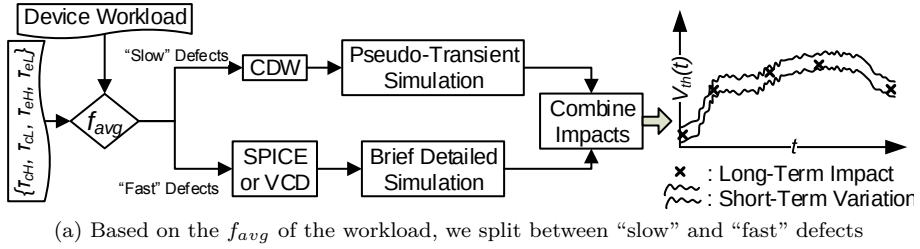
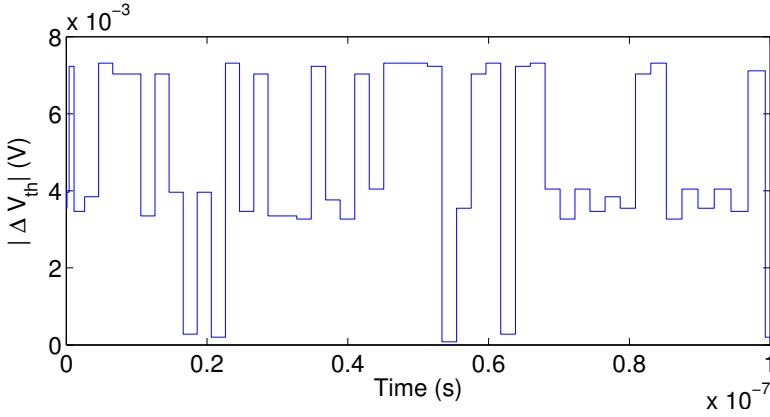
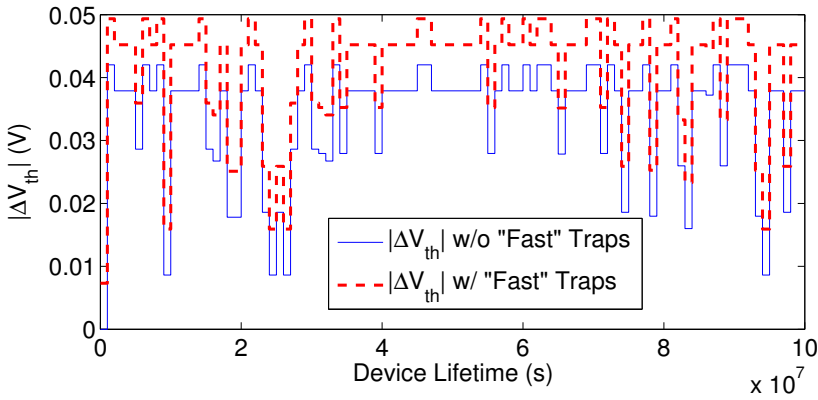


Figure 3.10: Applying the concept of pseudo-transient BTI/RTN simulation on a single device [234].

(a) Short term BTI/RTN impact on the V_{th} 

(b) Long term impact of BTI/RTN and its combination with "fast" defects

Figure 3.11: Applying the concept of pseudo-transient BTI/RTN simulation on a single device: the short- and long-term BTI/RTN impact for the workloads considered in Figure 3.10 [234].

3.6 Design Flow Compatibility

Given that the CDW representation leads to an acceleration of BTI simulation across device lifetime, it is very important to reflect on the usability of the proposed framework for realistic circuits. We look into the accurate estimation of *functional yield* of a circuit over three years of operation.

We use a subset of an embedded Static Random Access Memory (SRAM) [58], isolating a representative path, from a memory cell to the SRAM's input/output ports. This circuit contains 63 transistors and it supports reading and writing of the cell. We define a sample of this design as *functionally correct* iff a zero value can be written and then read successfully (Figure 3.12a). We estimate the *functional yield* of the target circuit, namely the percentage of samples that exhibit correct functionality. Derivation of such metrics is of major importance at design time and many methodologies exist solving this problem [173]. Given the inherent time- and workload-dependency of the proposed model, we can estimate the functional yield at different instances of circuit lifetime.

We perceive the evolution of V_{th} for each device according to Equation 3.11. $V_{th,TZ}$ represents the time-zero component of V_{th} fluctuations, whereas $V_{th,TD}$ the time-dependent counterpart. The former component is constant throughout the device lifetime but varies across device samples. In the context of the current text, the latter component is attributed solely to BTI. $V_{th,TD}$ is a function of time and varies across different device samples. The atomistic model for $V_{th,TD}$ is configured according to Table 3.1 and solved in a pseudo-transient way.

$$\begin{aligned}
 V_{th}(t) &= V_{th,TZ} + \Delta V_{th,TD}(t), \text{ where:} \\
 V_{th,TZ} &\sim \mathbf{Norm}(\mu_0, \sigma_0) \\
 \text{and } \Delta V_{th,TD} &\sim \mathbf{Atomistic Model}
 \end{aligned}
 \tag{3.11}$$

The proposed flow features detailed waveform dependency in aging analysis by utilizing the CDW format. We illustrate this capability by performing a stress-aware reliability analysis of the target circuit. More specifically, we assume five different cases in terms of voltage supply for the test case circuit (Figure 3.12b). Such settings cover the V_{dd} parameter of the CDW. We assume that the test case circuit operates at a 1 GHz frequency, executing iteratively the functionality criterion test of Figure 3.12a (namely the write $0 \rightarrow$ read 0 sequence of operations). Knowledge of this activity can provide the α and f parameters of the CDW for each device of the circuit using the **wavapprox** tools presented in Section 3.3. The inspected lifetime is about 3 years (10^8 s).

	Parameter	Distribution
$V_{th,TZ}$	Time-Zero pFET	$V_{th,TZ} \sim \mathbf{Norm}(-0.339V, -0.033V)$
	Time-Zero nFET	$V_{th,TZ} \sim \mathbf{Norm}(0.397V, 0.04V)$
$\Delta V_{th,TD}(t)$	Defects per pFET	$N_p \sim \mathbf{Pois}(\lambda = 10^{11} \times \text{Area}[\text{cm}^2])$
	Defects per nFET	$N_n \sim \mathbf{Pois}(\lambda = 6.7 \times 10^{10} \times \text{Area}[\text{cm}^2])$
	Time Const. per Defect (s)	$\log_{10} \{\tau_{pV}^*\} \sim \mathbf{Unif}(a = -12, b = 12)$
	ΔV_{th} per Defect (mV)	$\Delta V_{th} \sim \mathbf{Exp}(\eta = 5)$, for pFETs and nFETs alike

* p : process, either capture (c) or emission (e)

V : voltage, either high (H) or low (L)

Table 3.1: Atomistic model configuration for yield analysis of the target SRAM circuit, according to BTI/RTN measurements [265] and the 90 nm Predictive Technology Model [1], as initially disclosed in [234]

and is split in intervals of 2.5×10^7 s (which is used as the Δt parameter of the CDW). Finally, the temperature is fixed at a reference value of 50 °C for all the simulations.

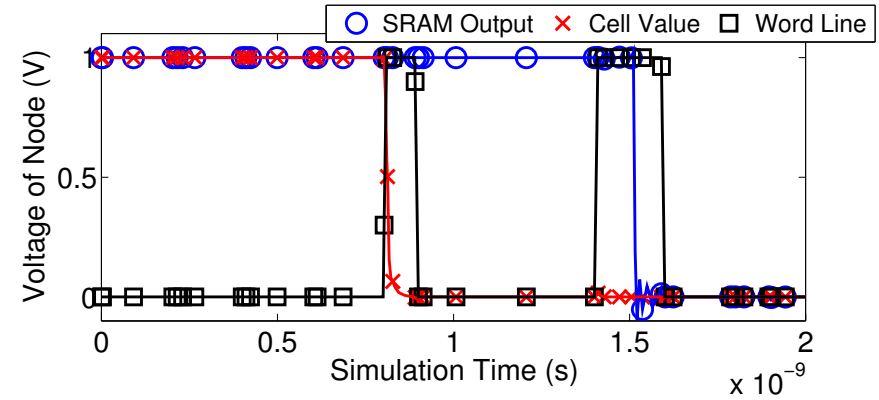
The functional yield of the circuit is estimated for all five V_{dd} cases and the 95% confidence interval [66] is also provided based on 150 Monte Carlo iterations for each case. We have measured the execution time at roughly 3 hours of processing per 150 Monte Carlo iterations on a desktop computer with an Intel® Core™ 2 Quad processor operating at 2.66 GHz. Each iteration starts with initialization of $V_{th,TZ}$ (from the normal distribution) and of the gate stack defects (based on the atomistic model). At each time step, $V_{th,TD}$ is evaluated per device and the circuit’s functionality is tested.

For the sake of brevity, we have ignored the impact of the “fast” defects on the functional yield analysis. Apart from being covered by previous work [229], we have refrained from transient BTI/RTN simulations, since they can be incorporated to the time-zero (and ever present) variations of V_{th} by adding a “safety” margin. Significant work exists in experimentally delivering yield and failure metrics for SRAMs, strictly due to “fast” defect activity [263]. However, until now and to the best of our knowledge, yield analysis of memory circuits has never accounted for time-dependent changes in parameters like V_{dd} stress, temperature or even signal activity. In the results presented herein we substantiate how different the yield profile of the target circuit may look, when V_{dd} stress varies throughout its lifetime.

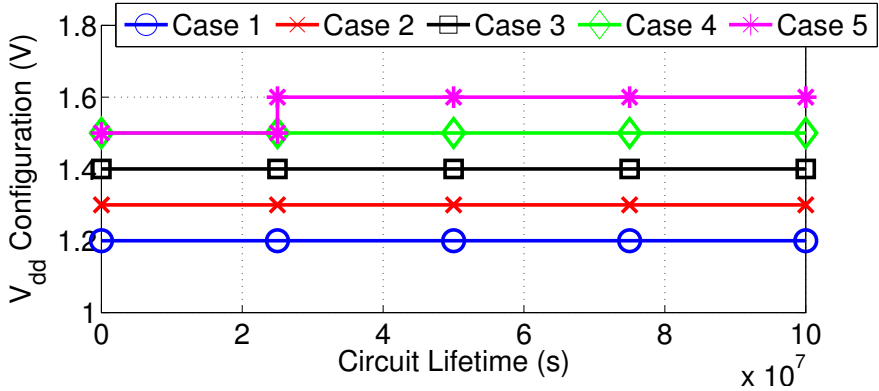
The yield estimations of Figure 3.12c provide useful hints towards time-zero and time-dependent variability mitigation. We observe that at a constant V_{dd} , the target circuit degrades aggressively within the first 2.5×10^7 s lifetime interval.

Afterwards, the yield shows minimal fluctuation. It is evident that constant voltage supply is unable to keep the yield of the circuit at acceptable levels. On the contrary, a dynamic V_{dd} configuration (namely Case 5 of Figure 3.12c) leads to a more acceptable yield profile. Finally, the BTI amplification due to elevated voltage supply appears to be overwhelmed by the reliability enhancement due to increased overdrive.

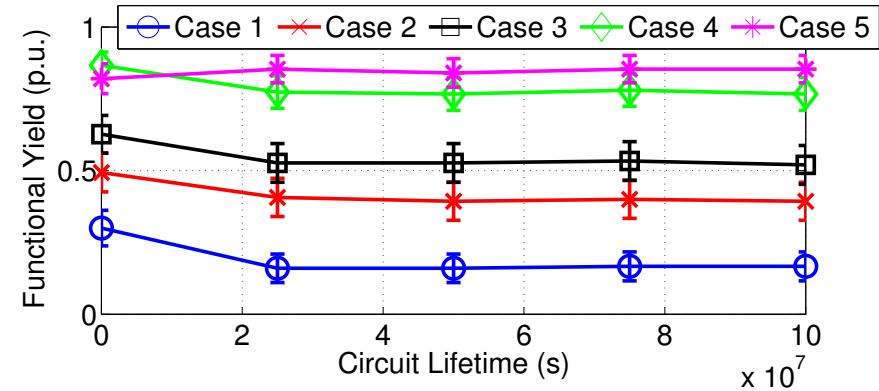
The proposed pseudo-transient, atomistic simulation scheme for BTI is a major enable for a thorough yield analysis, as presented above. By retaining the *time dependency* of the atomistic model, we monitor the evolution of functional yield throughout the circuit lifetime. Based on the model's *workload dependency*, we test dynamic alterations to the circuit's workload and observe their impact on the functional yield. Such capabilities are very important for the design of reliable ICs, especially at aggressively downscaled technologies [173].



(a) Functionality criterion for target circuit



(b) Five different stress conditions of the target circuit



(c) Functional yield of the target memory at various instances of circuit lifetime, for five different stress conditions (static and dynamic V_{dd} configurations)

Figure 3.12: Use of pseudo-transient BTI analysis for the estimation of time-dependent functional yield of an SRAM circuit

Chapter 4

Time-Dependent Variability in the P_{fail}

In the previous Section, we have provided sufficient hints about the propagation of reliability information from the level of gate stack defects to the performance and functionality of a simple memory circuit. In order to solidify this link, it is important to deliver failure metrics, under the influence of material variability, in a way that is consistent with system architecture needs. The contributions of the current Chapter are exactly in-line with this direction. We aim to deliver failure probability (P_{fail}) values for a SRAM cell which is injected with BTI variability, hence we initially revisit major milestones of prior art in this domain (Section 4.1). The Most Probable Failure Point (MPFP) methodology is used as the kernel of P_{fail} estimation (Section 4.2). Two BTI models are compared, the Reaction-Diffusion and a Defect-Centric (Section 4.3) and their impact on P_{fail} is further discussed (Section 4.4). Finally, the current Chapter ends with a discussion about the applicability of the MPFP technique on Statistical Static Timing Analysis (SSTA – Section 4.5).

4.1 Background

The variability of V_{th} is of major importance for the design of memory structures, with the SRAM cell being one of the most studied components. A variety of metrics can be derived to quantify SRAM stability. From an architectural perspective, an aggregate metric such as the failure probability (P_{fail}) is the most

usable. Based on the P_{fail} , architects can take design-time decisions regarding the protection of SRAM banks (as in the case of caches) [64] or predict the impact of SRAM-induced performance degradation [251, 102]. Consequently, for design choices to be optimal (i.e. guaranteeing protection while imposing minimal overheads), P_{fail} should incorporate properly all the semiconductor phenomena that induce V_{th} variability.

P_{fail} values have been previously calculated for various SRAM configurations [125]. P_{fail} estimation has been also performed with specialized Monte Carlo simulations, using the Reaction-Diffusion (RD) model for BTI [128]. However, as we have discussed in Section 3.1, failing to capture the *time-dependent* effects of transistor variability, i.e. the fact that two transistors may age in a different rate and/or at a different degree during their lifetime [265], may lead to significant errors in reliability analysis. A defect-centric model, namely a simplified version of the earlier discussed atomistic version BTI [41], has been used to capture time-zero and time-dependent variability on simple circuit operation parameters [280]. *In the current Chapter, we differentiate from prior art by focusing on the standard deviation of V_{th} shifts for proper P_{fail} calculation. We also demonstrate that average aging models may miscalculate the cell's P_{fail} by more than three orders of magnitude at the Most Probable Failure Point (MPFP).*

The estimation of the failure probability of an SRAM cell can be performed in various ways. Initially, a series of failure criteria are established, which may correspond either to static [27] or dynamic [125] figures of merit. Also, the choice of failure criteria may be such that permanent [218] and/or transient failure modes [86] are studied. Then, cell parameters (transistor dimensions, threshold voltages, etc.) are artificially varied and the union of the failure criteria is tested for each inspected point of the parameter space. A subset of points in this space is inspected, using the general concept of Monte Carlo sampling. From an algorithmic perspective, there are two orthogonal approaches towards sampling the design space:

- (i) *Importance Sampling*: This technique shifts one [50] or more [121] cell parameter distributions, so that extreme cases are sampled. That way, a pessimistically biased P_{fail} is calculated, which is recast to realistic values, based on the initial distribution shift. Systematic ways to control the accuracy of this process have been previously proposed [69].
- (ii) *MPFP*: According to this concept [125, 219, 218, 86], the P_{fail} of the cell is a specific combination of SRAM cell parameters, which must lead to cell failure (according to the considered criteria) *and* should occur with maximum probability (according to the respective distributions). We provide more information on the basics of this method in Section 4.2.

Regardless of the P_{fail} calculation method, SRAM cell parameters may vary for many reasons: *Time-zero* variations are static and a result of imperfect manufacturing [203]. *Time-dependent* aging phenomena cause changes in transistor parameters over their lifetime, as in the case of BTI, whereby $|V_{th}|$ increases during operation, both for p- (NBTI) and n-type (PBTI) transistors. The BTI phenomenon has been inspected in a P_{fail} context using the RD model and a “non-linear regression model” [128]. The defect-centric approach has been used to quantify SRAM cell parameters, but without deriving the overall probability of failure [280]. *In the current Chapter, we leverage the MPFP concept to estimate the impact of BTI on P_{fail} . We use a defect-centric model which properly captures time-dependent variability of BTI in aggressively downscaled nodes [94].*

4.2 The MPFP Methodology

To derive the P_{fail} of an SRAM cell configuration, using the MPFP concept, we identify the combination of involved V_{th} values with a maximum probability of occurrence. This combination should *additionally* be leading to failure of the cell, according to a failure criterion. We will be using the notation $\mathbf{x} = [x_0 \ x_1 \ \dots \ x_{N-1}]$ for the combination of N V_{th} shifts (one vector element per involved transistor). In this work, we use a permanent failure criterion, based on the six-transistor SRAM cell’s Static Noise Margin (SNM) for the hold operation as described by Equation 4.1. Given that we look at the hold operation of the cell and for the sake of simplicity, we emphasize on the four transistors of the cross-coupled inverters, as highlighted in Figure 4.1. More comprehensive criteria can be used [27, 125, 218, 86], however a full reliability analysis is out of scope for the current text.

$$A = \{\mathbf{x} : \text{SNM}(\mathbf{x}) > 0\} \quad (4.1)$$

The SNM is derived with two DC analyses of the SRAM cell netlist, each one sweeping one of the two complementary nodes (Q or QBar) and probing the other. Having derived the curves from the two sweeps (Figure 4.2a), we rotate them by 45 degrees [57] and calculate the number of zero crossings [261] of their difference (Figure 4.2b). If the zero crossings are less than three (i.e. less than the two stable points and the one metastable point that should exist by default), the cell is assumed unstable and a failure is recorded, whereby we use the notation $\mathbf{1}_A(\mathbf{x}) = 0$.

The value of $\mathbf{x}$ which leads to failure with maximum probability (i.e. MPFP) is derived from Equation 4.2. This, effectively, represents an optimization problem:

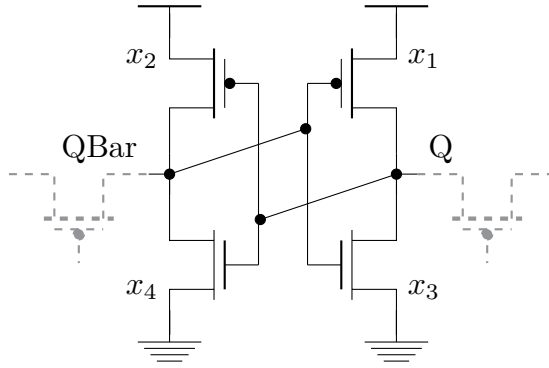


Figure 4.1: Schematic of the target circuit (pass transistors ignored)

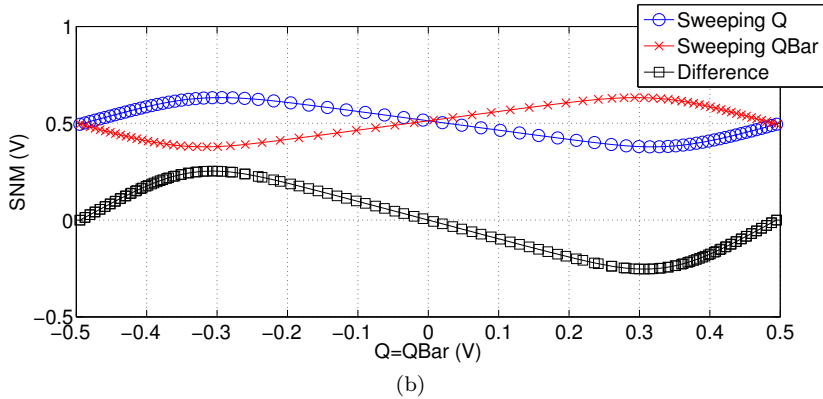
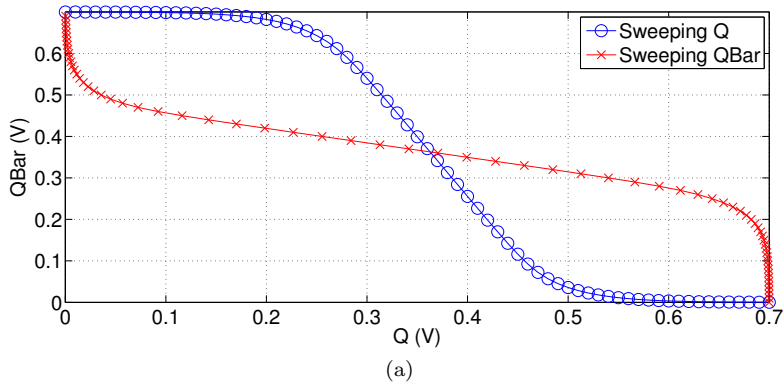


Figure 4.2: Failure criterion analysis based on the SNM hold

We need to maximize the probability of $\mathbf{x}$, as long as the failure criterion is satisfied. For the sake of accuracy in this work, we have performed an exhaustive analysis of the $\mathbf{x}$ space, only bounding it in terms of “reasonable” V_{th} values such as $\|\mathbf{x}\|_\infty \leq 0.5$ V. An algorithmically efficient solution for this space search is beyond the scope of our work, however prior art features many solutions to this problem [125, 219, 218].

$$\max \left\{ P_{\text{fail}} = \prod_{i=0}^{N-1} P(|\Delta V_{th,i}| \geq x_i) \right\} \quad (4.2)$$

such that $\mathbf{1}_A(\mathbf{x}) = 0$

The probability that corresponds to each $\mathbf{x}$ failure point is calculated based on the statistics of the involved V_{th} parameters. We will assume that both for pFETs and nFETs, V_{th} variability (both time-zero and time-dependent) is modeled using Gaussian distributions¹: $\Delta V_{th,p} \sim \mathbf{Norm}(\mu_p, \sigma_p)$ and $\Delta V_{th,n} \sim \mathbf{Norm}(\mu_n, \sigma_n)$. During the exploration, for a certain value of x_i , the equivalent probability is equal to the area highlighted in Figure 4.3, which is equivalent to $P(|\Delta V_{th,i}| \geq x_i)$. As soon as the MPFP is located, it is useful to report the respective probability (which is equal to P_{fail}) and a figure of MPFP “distance” from the default design point, where V_{th} is not variable at all, according to Equation 4.3.

$$\|x\|_F = \sqrt{\sum_{i=0}^{N-1} x_i^2} \quad (4.3)$$

4.3 BTI Variability Models

In Chapter 3, we have proposed the Compact Digital Waveform (CDW): Each interval of system operation is characterized by the clock frequency (f), a vector of duty cycles per gate node (α), the interval’s duration (Δt), the operating temperature (T) and voltage supply (V_{dd}) [234]. The aggregate notation $\mathbf{t}$ will be used for a single CDW point. In the general case, the aggregate V_{th} shift is formulated according to Equation 4.4. The former term reflects time-zero variability and the latter represents time-dependent (aging) mechanisms.

¹Non-Gaussian distributions have been suggested for certain aspects of transistor variability [280]. Our setup supports any type of ΔV_{th} distribution. In this work we use Gaussian distributions, as sufficient first-order approximations.

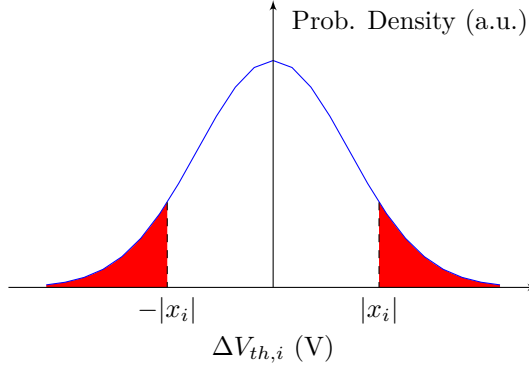


Figure 4.3: Probability of a single failure point coordinate. The MPFP maximizes the coordinate-wide product of probabilities.

$$\Delta V_{th}(\mathbf{t}) = \Delta V_{th0} + \Delta V_{th,\text{aging}}(\mathbf{t}) \quad (4.4)$$

4.3.1 Time-Zero Variability

A valid course of action is to implement the Pelgrom scaling principle [203], which is formulated in Equation 4.5, where W and L are the transistor's width and length and A_{VT} is the technology-dependent Pelgrom constant. This implies that smaller transistors exhibit greater time-zero variability, namely their σ_0 is higher.

$$\Delta V_{th0} \sim (0, \sigma_0), \text{ where } \sigma_0 = \frac{A_{VT}}{\sqrt{WL}} \quad (4.5)$$

4.3.2 Time-Dependent Variability and Aging

In the average case, the time-dependent component of BTI is calculated as a function of $\mathbf{t}$ according to Equation 4.6, with μ given by Equation 4.7 and notation explained in Table 4.1 [276]. Such models are compatible with *additional transistor aging variability, however this requires additional development and may incur increased computation times* [191]. The defect-centric model brings about both average aging and variability in a single compact formulation. In the current paper, we will use the average model and

Parameter	Value	Description
E_a	0.49 eV	Activation Energy
k	8.62×10^{-5} eV / K	Boltzman Constant
E_0	0.335 V/nm	Electrical Field
t_{ox}	1.4 nm	Oxide Thickness
q	1.6×10^{-19}	Electron Charge
ϵ_{ox}	3.9×10^{-11} F/m	SiO ₂ Electrical Permittivity
T_o	10^{-8} (s/nm ²)	Constant Parameter
ξ_1	0.9 (a.u.)	Fitting Parameter
ξ_2	0.5 (a.u.)	Fitting Parameter
K	8×10^{22} (s ^{-0.25} C ^{-0.5} m ^{-0.2})	Constant Parameter
n	0.25 (a.u.)	H ₂ Diffusion Constant

Table 4.1: Model parameter explanation and values used for average BTI aging model

the defect centric, to illustrate the importance of standard deviation of ΔV_{th} for the accurate calculation of P_{fail} .

$$\Delta V_{th}(\mathbf{t}) \sim \mathbf{Norm}[\mu(\mathbf{t}), 0] \quad (4.6)$$

$$C = \frac{\exp(-E_a/kT)}{T_o} \text{ and } \beta_t = 1 - \frac{2\xi_1 t_{\text{ox}} + \sqrt{\frac{\xi_2 C(1-\alpha)}{f}}}{2t_{\text{ox}} + \sqrt{C\Delta t}}$$

$$K_v = \left(\frac{qt_{\text{ox}}}{\epsilon_{\text{ox}}}\right)^3 \frac{K^2 \epsilon_{\text{ox}} (V_{dd} - V_{th}) \sqrt{C}}{t_{\text{ox}}} \exp\left[\frac{2(V_{dd} - V_{th})}{E_o t_{\text{ox}}}\right] \quad (4.7)$$

$$|\mu|(\mathbf{t}) = \left(\frac{\sqrt{\frac{K_v^2 \alpha}{f}}}{1 - \beta_t^{1/2n}}\right)^{2n}$$

From a defect-centric perspective, BTI is predicted to contain a time-dependent variability component. In other words, the mean V_{th} aging across a set of transistors follows a trend similar to the power law of Equation 4.6. However, there is also a time-varying standard deviation. Hence, according to the defect-centric model, two identical transistors that are subjected to the same operating conditions may age in a different way. More specifically, the defect-centric

model assumes that a variable number of different defects resides in the gate stack of each transistor. Each defect can capture and emit minority carriers at a rate governed by its individual *time constants*. The contribution of each capture/emission event is also variable, with each defect contributing by a characteristic step to the overall V_{th} shift occurring to the transistor [234] due to BTI.

Recent defect-centric literature documents that, for downscaled devices, the *average* V_{th} shift caused by BTI follows a power law similar to Equation 4.7 [143]. Furthermore, by fitting experimental data, a correlation has been derived in related work between the standard deviations of V_{th} shift due to time-zero and time-dependent variability [280], which is repeated in Equation 4.8. It is important to note that this Equation features in prior in the context of FinFETs. However, it is reused in our work within a MOSFET context since (i) it is a representative formulation of time-dependent standard deviation of ΔV_{th} and (ii) similar observations have been reported for the case of MOSFETs as well [94]. Hence, we can assume that, from a defect-centric perspective, ΔV_{th} is described by Equation 4.9.

$$\sigma^2 = \frac{\mu}{0.1 \text{ V}} \sigma_0^2 \quad (4.8)$$

$$\Delta V_{th}(\mathbf{t}) \sim \text{Norm}[\mu(\mathbf{t}), \sigma] \quad (4.9)$$

4.4 Simulation Flow and Results

Following the methodology as described in Section 4.2, we have implemented the simulation setup of Figure 4.4. Inputs to this setup are the V_{th} variability distributions for p- and n-type transistors, which, as we have discussed previously, can be generally assumed as time-dependent.

To begin with, we perform an exploration using the framework of Figure 4.4. We sweep two parameters for transistor variability, namely $\mu = |\mu_p| = \mu_n$ and $\sigma = \sigma_p = \sigma_n$. The simulated cell is implemented with a high performance 16 nm Predictive Technology Modelcard [1], i.e. all transistors are dimensioned as 32 nm x 16 nm. Apart from the default V_{dd} (i.e. 0.7 V) we also test two lower voltages (0.6 and 0.65 V).

In Figure 4.5a we see the distance of the respective MPFP from the ideal design point (where $\mu = \sigma = 0$) for the aforementioned sweep of variability parameters. Even if we only have the aggregate distance of the most probable failure point,

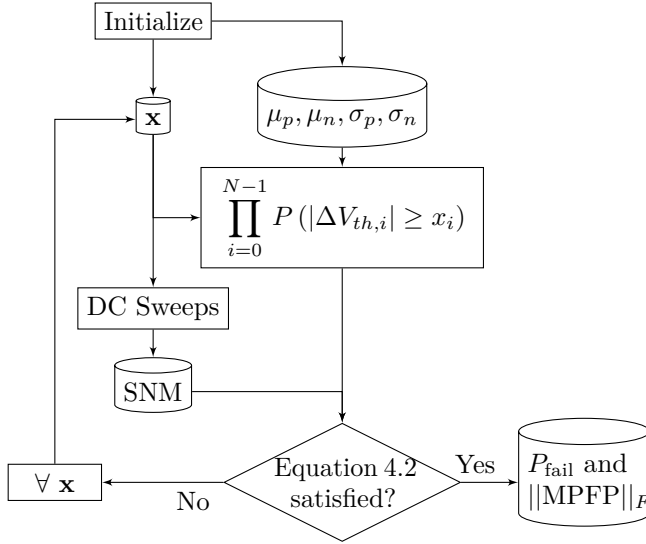


Figure 4.4: MPFP estimation setup used in the current work

it is clear that there is a weak dependence (if any at all) to the mean of V_{th} shifts. Instead, the most definitive parameter is the spread of the ΔV_{th} , i.e. σ . Moreover, we observe that as variability is amplified, the MPFP moves to more “distant” regions of the design space. This is expected, given that, as we amplify variability (by increasing σ), we increase the probability of occurrence for failure points that are otherwise rare. Also, $\|\text{MPFP}\|_F$ is affected by the V_{dd} in a non-regular way: For reduced variability, a higher V_{dd} activates more distant failure points. For aggressive variability, a higher V_{dd} activates failure points closer to the ideal design point.

In Figure 4.5b we see how P_{fail} (i.e. the actual probability of the most probable failure point) changes for various values of μ and σ . *We verify that P_{fail} is a more sensitive to the spread, rather than the mean of V_{th} shifts.* The dependence on μ is pronounced only for the case of reduced σ . This indicates that uniform changes that are applied to all V_{th} values are less definitive for the reliability of the cell. On the contrary, P_{fail} is dictated by the spread of outliers, i.e. the standard deviation of ΔV_{th} . *This observation motivates that, for a proper aging analysis of an SRAM cell, the time-dependent spread of V_{th} should be properly accounted for.* We will further substantiate this claim in the next Subsection. Finally, we notice that as the V_{dd} increases, P_{fail} decreases, which is a consistent observation. We observe a maximum P_{fail} drop by an order of magnitude for a 0.1 V V_{dd} increase (highlighted in Figure 4.5b).

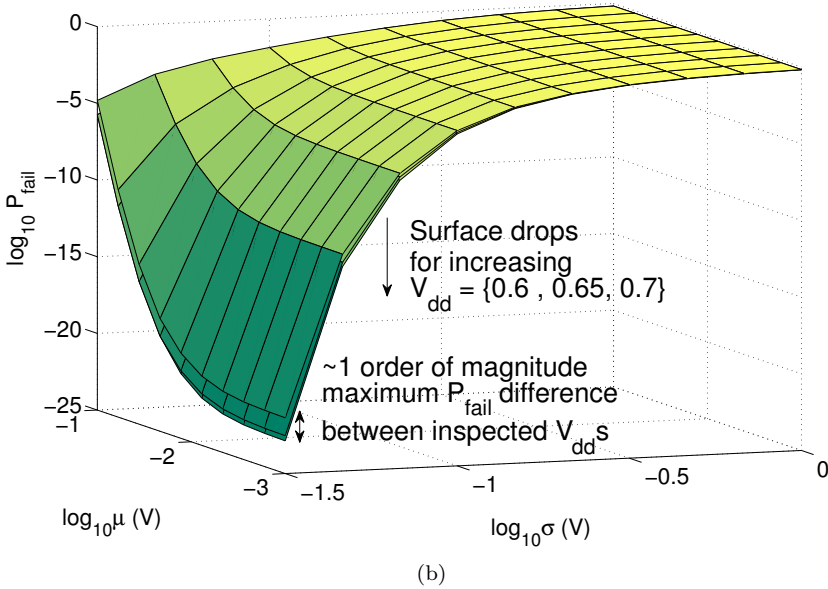
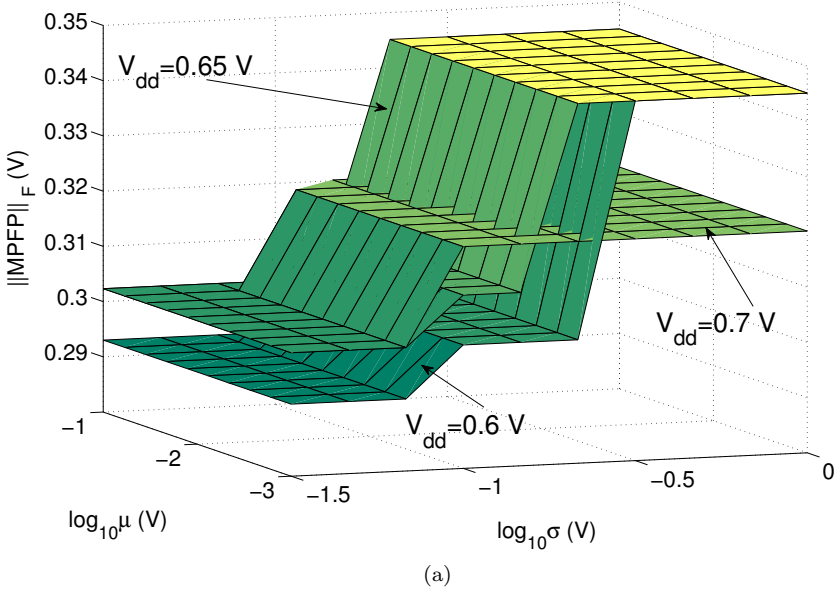


Figure 4.5: The MPFP is more sensitive to the standard deviation of threshold voltage shifts, rather than the mean.

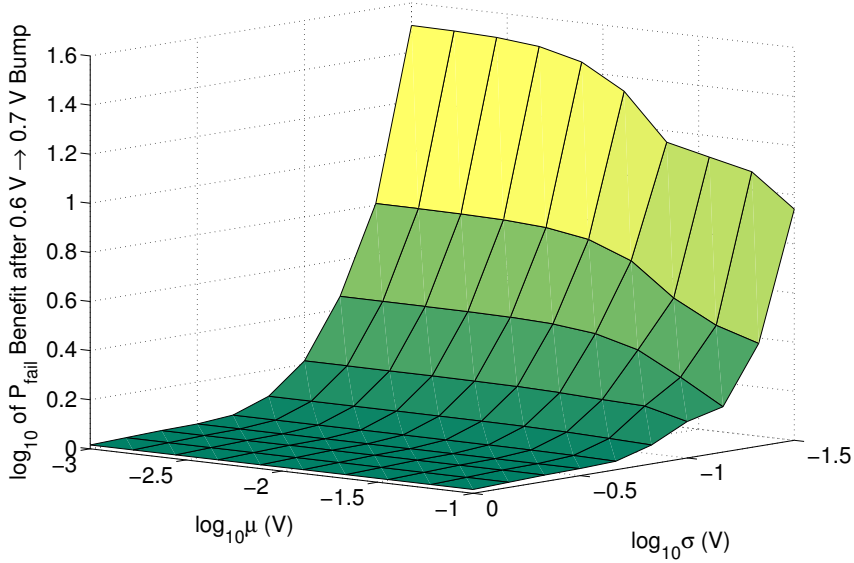
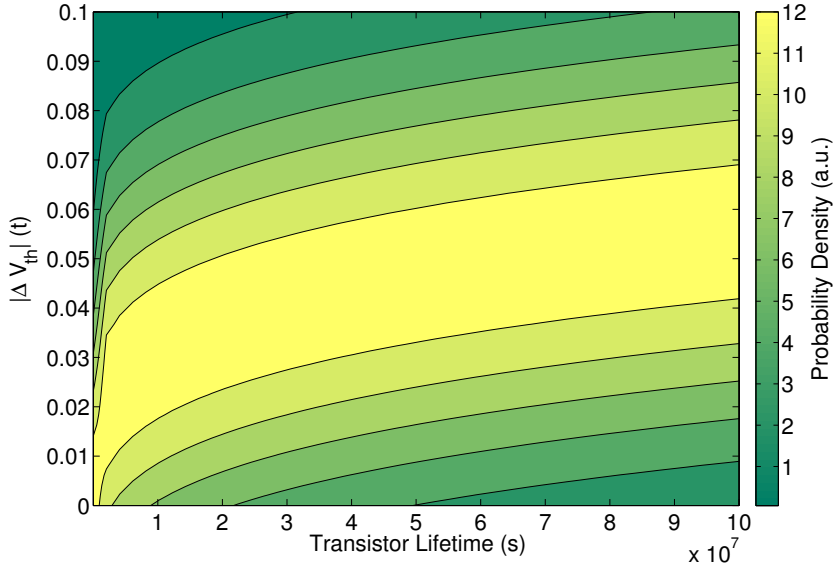


Figure 4.6: The inspected V_{dd} bump is not effective for SRAM cell reliability in case of aggressive variability. This claim remains valid only for the SNM hold criterion used in the current work.

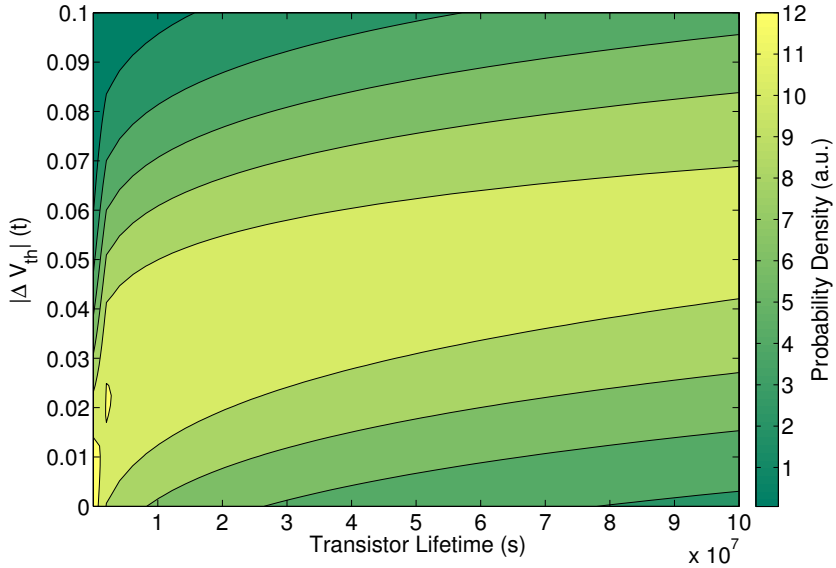
In Figure 4.6 we focus further on the P_{fail} benefit achieved by increasing the V_{dd} from 0.6 V to 0.7 V. It appears that this benefit decreases as variability intensifies within the SRAM cell. This benefit is also more sensitive to σ in comparison to μ . However, it should be stressed that this claim is valid only for the currently inspected failure criterion (non-zero SNM hold). Generalizing this claim towards overall reliability benefits from a V_{dd} increase requires a fully comprehensive reliability analysis, which is beyond the scope of our work.

In a second series of experiments, we evaluate the target SRAM cell across a lifetime of 10^8 seconds, i.e. roughly three years. Assuming aging under BTI and $V_{dd} = 0.7$ V, we implement the two aging models discussed in Section 4.3. In both aging implementations, time-zero variability is present with $\sigma_0 = 30$ mV. Average aging follows Equation 4.10, whereas the defect-centric approach follows Equation 4.11. We assume pFETs and nFETs undergo identical mean aging [143], i.e. $\mu(\mathbf{t}) = |\mu_p(\mathbf{t})| = \mu_n(\mathbf{t})$, according to Equation 4.7, and feature identical standard deviation, i.e. $\sigma = \sigma_p = \sigma_n$.

$$|\Delta V_{th}(\mathbf{t})| \sim \text{Norm}[\mu(\mathbf{t}), \sigma_0] \quad (4.10)$$



(a) Transistor aging using the average BTI model



(b) Transistor aging using a BTI model that includes time-dependent variability

Figure 4.7: The defect-centric model introduces a time-dependent increase of the V_{th} shift standard deviation. This is an additional and desirable feature on top of average aging models. We note that the color bars of the above Figures are presented in arbitrary units, whereas the actual proper unit of measurement for this probability density is Volts per second.

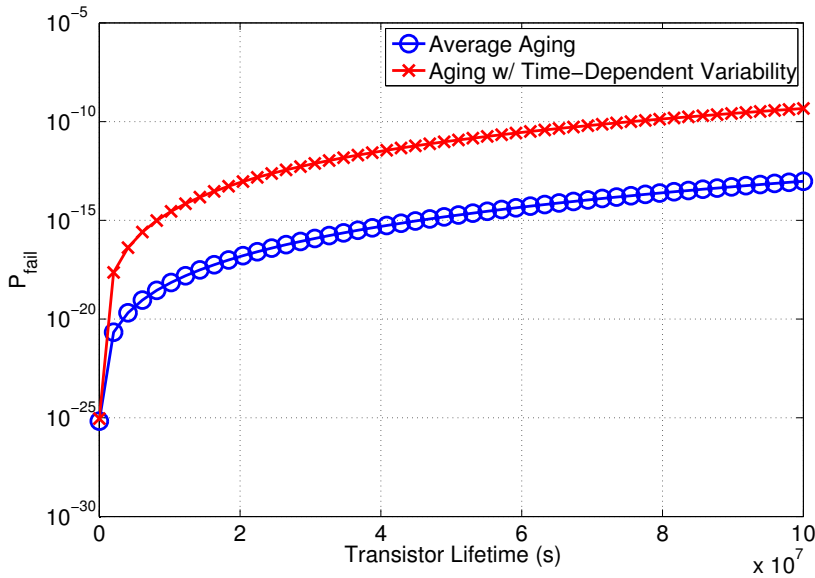


Figure 4.8: The average BTI modeling approach fails to properly capture the impact of time-dependent variability on the P_{fail} . Apparently, this course of action deviates by more than three orders of magnitude if stochastic ΔV_{th} behavior is unaccounted for.

$$|\Delta V_{th}(\mathbf{t})| \sim \mathbf{Norm} \left[\mu(\mathbf{t}), \sigma_0 \sqrt{\frac{\mu(\mathbf{t})}{0.1} + 1} \right] \quad (4.11)$$

In Figure 4.7 we see the transistor-level difference between these two aging models. We plot the probability density function of the average and the time-dependent variability aging model in Figures 4.7a and 4.7b, respectively. The coloring of the Figures indicates the probability density, for each ΔV_{th} value in time. In Figure 4.7a the average V_{th} shift is a function of time and the respective standard deviation is constant and equal to σ_0 . Conversely, in Figure 4.7b *both* the mean and the standard deviation of the total V_{th} shift are functions of time.

Apparently, an average aging model without any stochastic component fails to capture the additional variability that is included in time-dependent approaches, such as the defect-centric one. This can be observed by comparing Figures 4.7a and 4.7b: On the former case, the probability density is concentrated around the mean V_{th} shift trend. On the latter case, probability density is increasingly

dispersed with time. Given our observation on the P_{fail} sensitivity to the standard deviation of the V_{th} shift (Figure 4.5), we evaluate the P_{fail} over a ~ 3 year lifetime, for each of the two above variability models. The results are shown in Figure 4.8. As expected, we observe a substantial difference in the derived failure probabilities.

The average aging approach leads to an optimistic estimation, deviating by more than three orders of magnitude, compared to the approach that encapsulates time-dependent variability. The reason for such deviation is that the non-stochastic average aging model fails to capture the time-dependent standard deviation of V_{th} shifts caused by the BTI effect.

4.5 Discussion and Extensions to SSTA

The delay distribution of a digital circuit path is crucial for the early reliability evaluation of a digital design. As transistors are shrunk to unprecedented dimensions, accurate yet fast estimation of such distributions remains a valid goal. Such distributions may not be provided or are delivered in a heavily abstracted fashion to designers, which reduces the insight into design dependability. In view of the above observations, we propose a technique that approximates the probability density function of a path of digital circuits using a well-known computational kernel, namely the MPFP technique. The output of this concept is the failure probability of a digital circuit path for various target delays. We reformulate the statistics of the MPFP kernel and establish a concise methodology for delay distribution approximation. We present simulations for an inverter and outline projections for more complex gates.

The inherent time-zero and time-dependent variability of semiconductor structures creates challenges for the efficient/accurate modeling of integrated circuit reliability [234]. Statistical Static Timing Analysis (SSTA) has been prevalent in handling delay distributions of standard cells. This is split mainly between depth-first (or path based) [116, 197] and breadth-first (or block based) [274, 155, 11] techniques.

Regardless of SSTA techniques, what is really interesting is the derivation of the delay distributions of standard cells. Accurate derivation requires a large number of simulations or measurements, since rare delay events need to be accounted for. In many cases, complete distributions are not even available and only an approximated view of standard cell delay variability is provided, as in the case of the stage-based on-chip variation (OCV) [71]. In view of the above, it is important to provide an accurate and efficient technique for the approximation of a standard cell's delay distribution.

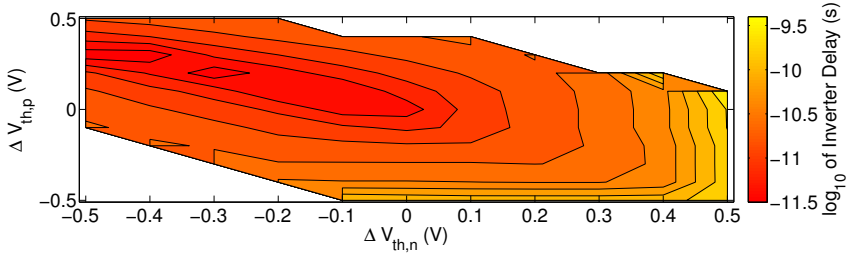
The current work delivers the distribution of an inverter delay, based on the distributions of the involved threshold voltages (V_{th}). The numerical kernel used is the Most Probable Failure Point (MPFP), which has been used for memory cells [125, 86, 233]. We reformulate the MPFP using the χ^2 distribution and use it iteratively to get the probability mass for various inverter target delays. Project and extensions are provided for more complex standard cells and paths.

The manifestation of variability in a circuit can be encapsulated in a vector $\mathbf{x}$ (e.g. threshold voltage shifts of the involved transistors). A performance metric (e.g. delay of cell) y can be evaluated at each $\mathbf{x}$ point as $y(\mathbf{x})$. A failure occurs when the performance metric is larger than a specified target (Y), namely $y > Y$ ². For a specific Y , the MPFP methodology aims to find the failure, i.e. $P_{\text{fail}}(Y) = P(y > Y)$. In order to connect variability with the failure specification, it is important to isolate all the values of $\mathbf{x}$ that satisfy the failure criterion. Let us use F to represent the set of $\mathbf{x}$ values that lead to a failure. The failure probability is $P(\mathbf{x} \in F)$. The challenge posed by MPFP is *both* isolating F and calculating $P(\mathbf{x} \in F)$ in a systematic way. We illustrate this situation with an inverter, which has been profiled using Synopsys NanoTime [260] according to Figure 4.9a. Simulation results have been fitted with MATLAB for simplicity, as shown in Figure 4.9b, where a level set is annotated for a specific Y (roughly 17.5 ps). For all the simulations presented herein, we have used a publicly available high performance 16 nm modelcard [1]. A functionality criterion (in our case $y > Y$) has been previously combined with a product of probabilities to approximate space F [125, 86, 233] and its probability mass, according to Equation 4.2 has been used to approximate F . If we cast the product of probabilities to the delay space (Figure 4.9c), it is clear that *even if the functionality criterion is correct, the probability mass of set F is not totally covered*. To address the above inaccuracy, we propose $P_{\text{fail}}(Y)$ calculation in two separate steps. There exactly lies the novelty of the current work, in reformulating the MPFP using the χ^2 distribution.

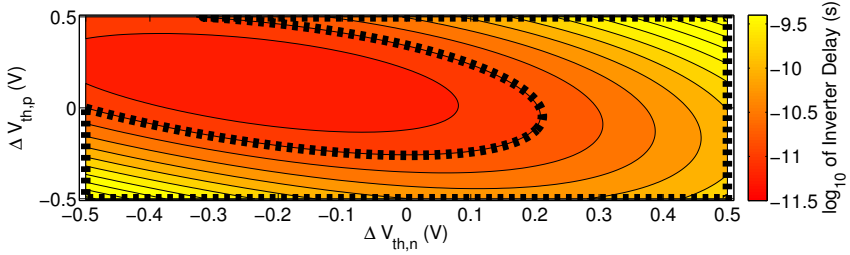
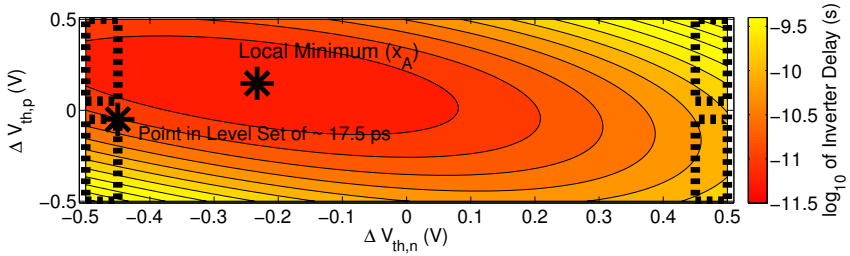
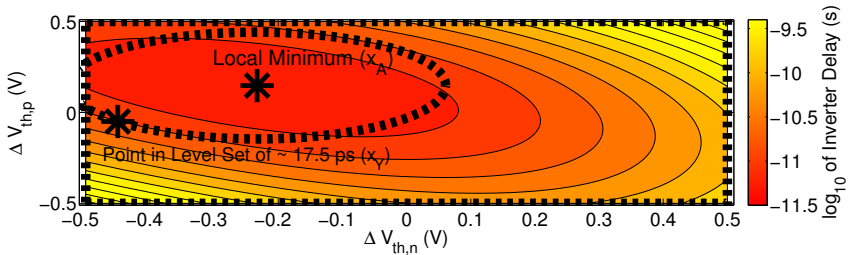
4.5.1 Reformulating the MPFP – Inverter Case Study

The first step is to isolate the point $\mathbf{x}_A$ which leads to minimum delay $y_{\min}$. To achieve this, we solve the optimization problem $\min \{y(\mathbf{x})\}$ for $\mathbf{x}$ and also get the distance of $\mathbf{x}_A$, which we notate as r_A . For the case of the inverter, it is reasonable to expect a unique solution to this optimization problem. This statement has been verified with a series of Synopsys NanoTime [260] simulations (Figure 4.9a) and remains valid when we use a fitted expression for $y(\mathbf{x})$ as in Figure 4.9b.

²In the current Section we will be using lower case letters for random variables and the upper case letters for variants.



(a) Inverter delay as measured by NanoTime

(b) Fitted NanoTime measurements and bounding F with level set of 20 ps(c) F approximation by $\prod_{i=0}^{N-1} P(|\Delta V_{th,i}| \ge x_i)$ as used in Equation 4.2(d) F approximation using a hypersphere around the local minimum of inverter delay; additional probability mass is included and the χ^2 distribution appliesFigure 4.9: Navigating the $\mathbf{x}$ space for accurate approximation of the inverter delay distribution. The χ^2 distribution is useful in bounding failure region F .

The procedure followed in this step is outlined in Figure 4.9d. First, we isolate a direction away from $\mathbf{x}_A$ along which the increase of delay is maximum. Having selected a target delay Y , we move along the above direction, until we reach $\mathbf{x}_Y$, where $y(\mathbf{x}) = Y$ at a distance equal to r_Y from the point $\mathbf{x}_A$. At this point, the *generalized non-central Chi distribution* can be used (χ^2). This will provide the probability mass that corresponds to a hypersphere in the threshold voltage shift space, which is centered at $\mathbf{x}_A$ and has a radius equal to r_Y . By comparing Figures 4.9b and 4.9d, it is clear that the use of the χ^2 distribution is rather pessimistic, since $\mathbf{x}$ points that are faster than the target Y are included in space F . However, this is preferable than the formulation of Equation 4.2, which is optimistic: a huge portion of probability mass is ignored, as we can verify from Figure 4.9c. On the contrary, *the χ^2 -based formulation is guaranteed to contain all the failure points*, provided that point $\mathbf{x}_Y$ is selected based on a greatest ascent algorithm, moving away from $\mathbf{x}_A$. This ensures that the “pass” region only contains “pass” points, even though some are excluded (pessimism). It is important to note that the above statement is correct *regardless of the shape of $y(\mathbf{x})$* .

Looking into the χ^2 distribution, we are specifically searching the probability mass of random variable $\mathbf{z}^2$, as defined in Equation 4.12, where N is the number of involved transistors (2 in the case of the inverter) and σ_i is the standard deviation of the ΔV_{th} per involved transistor. In the current work, we assume that all transistors exhibit the same σ_i . The *non-centrality parameter* of the utilized distribution is calculated according to Equation 4.12 [117]. Apart from the involved σ_i , this parameter considers the distance between point $\mathbf{x}_A$ and the origin of the axis of the $\mathbf{x}$ space, which is encapsulated as the translated mean V_{th} shift per transistor (i.e. μ_i). For the rest of current Section, we assume that these mean values are constant. In case transistor aging is assumed, actual mean V_{th} shifts become non-zero [280] and distance to $\mathbf{x}_A$ can be easily recalculated.

$$\mathbf{z}^2 = \sum_{i=0}^{N-1} \frac{x_i^2}{\sigma_i^2} \text{ and } \lambda = \sum_{i=0}^{N-1} \frac{\mu_i^2}{\sigma_i^2} \quad (4.12)$$

At this point, we need to note that the approximation of F can be improved by utilizing distribution transformations, as advised in prior art [181]. However, in the current work we prefer the directly applicable χ^2 distribution at the small “cost” of more pessimistic approximation of set F . Using the above approach, we end up with the P_{fail} results of Figure 4.10a. As it can be expected, a higher σ of transistor V_{th} shift leads to higher P_{fail} for the same target inverter delays Y . Finally it is clear that, in case $f(\mathbf{x})$ has a single maximum (instead of minimum), we can alternatively start from its maximum and directly bound set

F , instead of its complement. The choice between minimum/maximum depends on the shape of $f(\mathbf{x})$.

A (sufficiently) correct estimation of the failure probability for a target delay Y satisfies Equation 4.13, whereby PDF_y and CDF_y are the respective probability and cumulative density distributions for random variable y (inverter delay in our case).

$$P_{\text{fail}} = P(y > Y) = 1 - \text{CDF}_y(Y) = 1 - \int_{-\infty}^Y \text{PDF}_y dy \quad (4.13)$$

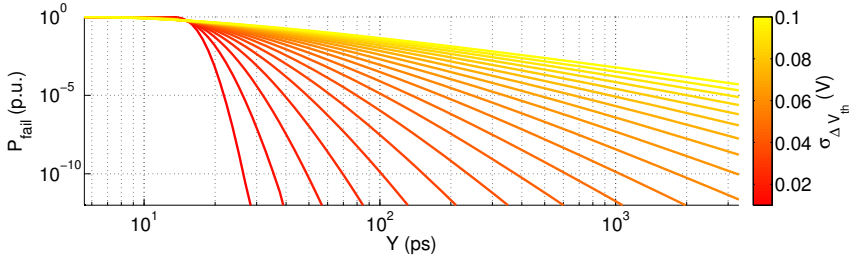
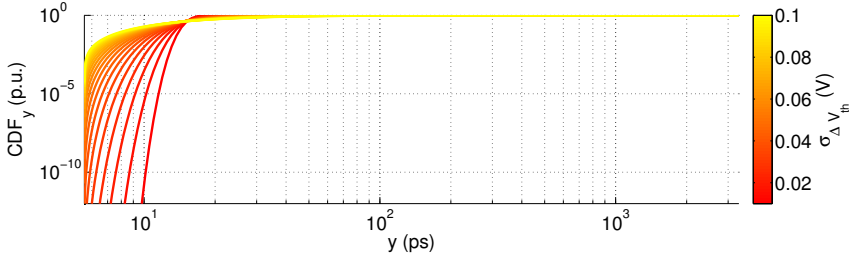
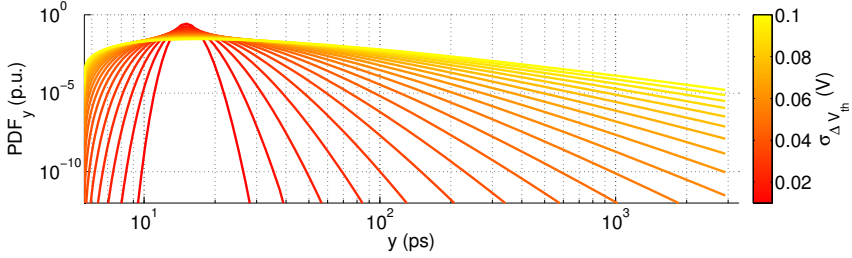
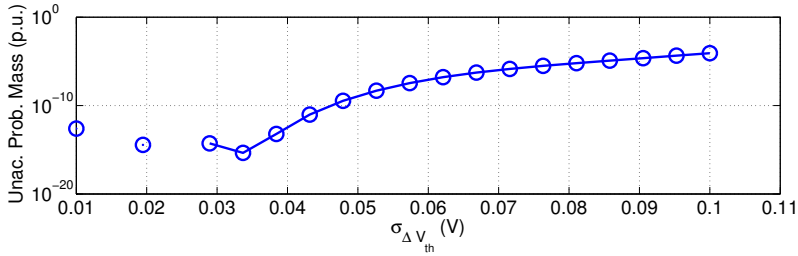
It is clear that by repeating the above process for different values of Y we can isolate the cumulative probability for various delay specifications of the target circuit. Based on the P_{fail} vs. Y relation derived in Figure 4.10a, we easily produce the respective CDF_y data, as illustrated in Figure 4.10b. A simple differentiation yields the corresponding PDF_y . This effectively constitutes the *delay distribution of the inverter*. It is solely based on a NanoTime-compatible description of the inverter and uses values of the standard deviation for V_{th} shifts (multiple values inspected). This being a non-analytical approach, it is important to highlight the probability mass that is unaccounted for. This corresponds to events that are not included in the presented PDF_y , as in the case of very high target delays (which can be safely assumed as negligibly rare). The unaccounted probability mass is illustrated in Figure 4.10d. Evidently, *the higher the spread of V_{th} shifts is, the higher the unaccounted probability mass is using our technique*. However, when considering σ values for ΔV_{th} that are relevant to current technologies [280], our technique behaves with sufficient accuracy.

The delay of the inverter is lower bounded (Figure 4.9b). This means that delay distribution of the inverter does not have a tail on the left. However, as σ increases, the right-hand tail extends uncontrollably.

4.5.2 Generalizing to Complex Gates and Standard Cell Paths

Standard cells with more than one transistor in the pull-up/-down branches, require a systematic way of identifying the delay minimum (of maximum). In any case, we have no information about $y(\mathbf{x})$, which is evaluated with NanoTime for each iteration. We implement coordinate descent [283] according to Algorithm 1 for the case of a NAND gate.

Algorithm 1 has been implemented as Perl wrapper around NanoTime. We initially sweep the $y(\mathbf{x})$ function for the NAND case, in a 100 mV granularity

(a) Inverter failure probability for various values of V_{th} shift spread(b) Cumulative density function of inverter delay for various values of σ (c) Probability density function of inverter delay for various values of σ 

(d) Unaccounted probability mass using our approximation

Figure 4.10: Inverter delay analysis and approximation of set F for P_{fail} extraction

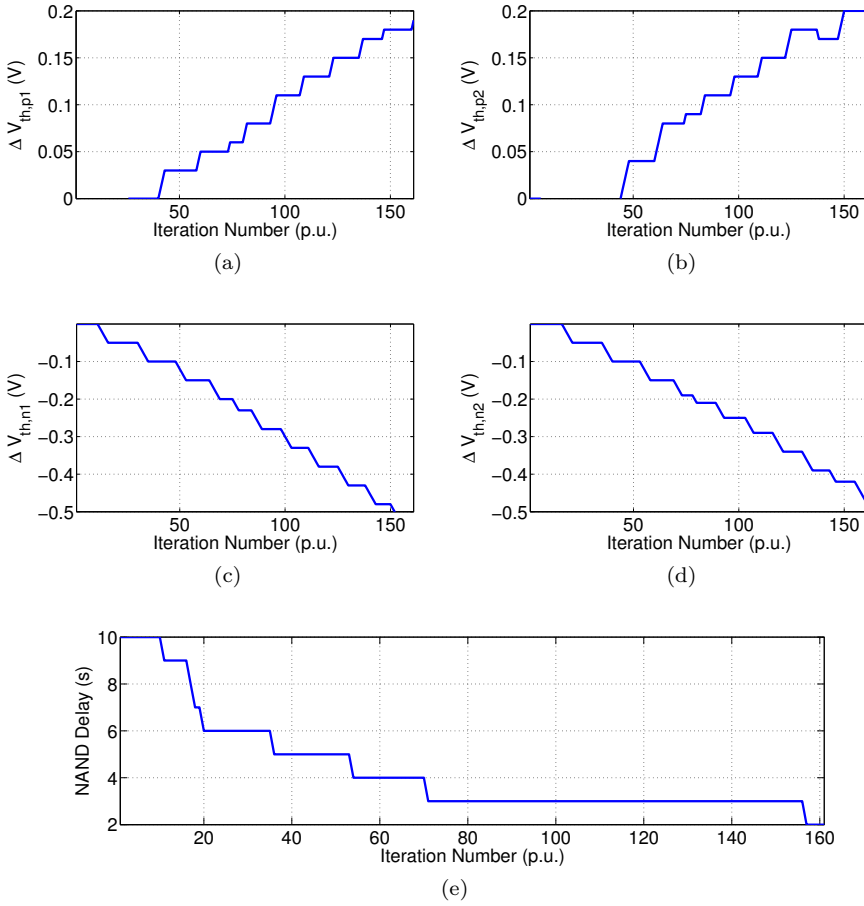


Figure 4.11: Results of the coordinate descent implementation of Algorithm 1. One of the minimum delay points is identified within ~ 160 NanoTime iterations.

Algorithm 1 Coordinate descent used in the current work, based on iteration limit and V_{th} step equal to s

```

while (itNum < Limit) {
  for  $i \in \{0, 1, \dots, N-1\}$  {
     $s = \text{find descending direction for } x_i$ 
    if  $y(x_0, \dots, x_i + s, \dots, x_{N-1}) < \text{prev\_delay}$ 
      {Update  $\mathbf{x}$  with  $x_i + s$ }
    else {Proceed to next transistor}
  }
}

```

($> 14,000$ NanoTime iterations). This is a crude estimation of the minimum value and the value of the corresponding V_{th} shifts (i.e. estimation of $\mathbf{x}_A$). Using Algorithm 1, we succeed in identifying the minimum point in ~ 160 NanoTime iterations (Figure 4.11).

Given the multitude of transistors in the pull-up/-down branches of standard cells, multiple minima may exist for $y(\mathbf{x})$ ($i = 0, 1, \dots, M-1$). Given the symmetry that appears in these cases, we may treat only one of these points ($\mathbf{x}_{Ai}$) with the technique of Subsection 4.5.1. The cumulative probability around $\mathbf{x}_{Ai}$ can be multiplied by M to provide the total probability mass of the “pass” event at delay Y . This is, conceptually, the complement of the F set (“failure” region). By subtracting from one, we get P_{fail} , which is substituted in Equation 4.13. Repeating this for different Y values yields the delay distribution of the standard cell.

Clearly, in case $y(\mathbf{x})$ has a finite number of maxima (instead of minima), a dual approach can be maintained. This leads to bounding set F , instead of the latter’s complement. The choice between the two courses of action can be resolved with a high-level view of $y(\mathbf{x})$, e.g. by crudely sweeping the $\mathbf{x}$ space.

The techniques of minimum and maximum identification, Algorithm 1) and probability mass calculation need to be generalized for an arbitrary number of transistors (N) in the standard cell. The two step technique of Subsection 4.5.1 should account for plateaus in $y(\mathbf{x})$ and non-global minima/maxima. Such enhancements constitute points for future work.

Given a (sufficiently) accurate PDF approximation for the delay distribution of a set of standard cells, it is quite easy to provide the delay distribution for a path of standard cells. Given that we target the sum of the delays of the involved standard cells, the respective distribution is produced with convolution of the delay distributions [97]. In Figure 4.12 we present the results for a chain of four inverters, each one being identical to the one used in Subsection 4.5.1. The chain

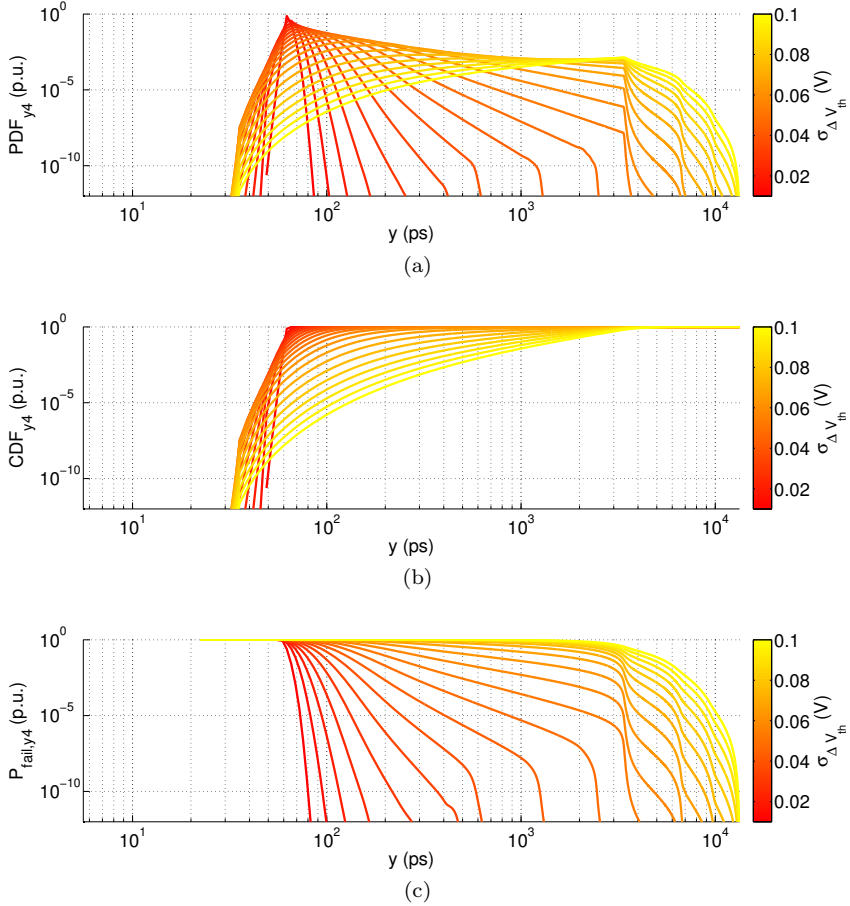


Figure 4.12: Iterative convolution of PDF_y from Figure 4.10c provides the delay PDF, CDF and, eventually, the P_{fail} for a chain of four inverters.

of operations is exactly inverted: we convolute PDF_y the appropriate amount of times (four) and produce Figure 4.12a, namely the delay PDF for the path of inverters (PDF_{y4}). A simple integration yields the CDF_{y4} (Figure 4.12b) and subtraction from one provides the failure probability of the simple four-inverter path for various target delays (Y), as illustrated in Figure 4.12c. We note that the resulting failure probabilities span a wider Y range in comparison to the single-inverter equivalent. Also, there is a general transposition of the nominal delay in comparison to Figure 4.10a, given the connection of inverters in series.

Chapter 5

Dependable Performance under Fine-Grain Rollbacks

In the previous two Chapters, we have shown how variability at the level of gate stack defects is typically manifesting itself at the system level, which can be eventually measured using the component failure probability (P_{fail}). In general, components may be failing at different rates and for different reasons. So far, we have explored at length the issue of BTI and RTN as a valid reason for reliability violations. As semiconductor phenomena are encapsulated into a single P_{fail} , so do higher level reliability violations abstract to the Mean Time to Failure (MTTF). This metric corresponds to the mean time between error or failure events for an integrated (sub)system [187]. Typically, in modern processors, certain actions are taken upon error or failure realization, which generally fall under the Reliability, Availability and Serviceability (RAS) umbrella term [247]. Another reliability metric that is relevant is the Failures in Time (FIT), which the number of failure or error events in one billion device hours (product of devices tested by the duration of the test). It is clear that regardless of the underlying phenomenon (e.g. material aging as discussed in Subsection 2.2.2 or particle interference as discussed in Subsection 2.2.1), the circuit designer can derive a failure probability (based on the failure mechanism featured within the computing fabric), which can then be recast to an MTTF or FIT rate for the respective error occurrence. This manifestation can be either *permanent* (i.e. irreversible) or *transient* (i.e. restricted both in space and time) [290]. The casting from failure probabilities of circuits to MTTF or FIT rates of systems is a procedure which requires extreme care, given that it requires knowledge of the activity patterns on the actual computing fabric. That way, proper consideration

of masking effects or false negatives can be achieved in the reliability analysis.

Regardless of their root, fault events in the computing fabric may propagate to higher abstraction levels, thus causing fluctuation of the system’s operation parameters (e.g. leakage power, delay etc.) or a binary corruption of the system’s output. There exactly lies the distinction between *parametric* and *functional* reliability violations, as articulated in Definitions 1.8 and 1.9. In this Chapter, we will start from a FIT specification that targets the data plane of a streaming application executed on a real many-core chip. Errors are injected as software artifacts so that both the MTTF specification and the binary corruption are properly replicated. This work addresses functional *and* parametric aspects and focuses on transient bit flips in low-level data caches (referred to as “soft” or “transient errors”). Many mechanisms are responsible for such errors, ranging from cosmic particles to supply voltage fluctuations [294]. Manifestation of such errors threatens functional reliability, namely the binary correctness of the application output. In this Chapter, we demonstrate a hybrid technique that guarantees the resiliency of corrupt application data structures. Mitigation of these errors is performed by a demand-driven rollback in application execution, using safe copies of key application data structures. This affects the parametric reliability of the system, i.e. the target latency constraints. The trade-off between output correctness and the time/energy overheads is illustrated. Dynamic Frequency Scaling (DFS) is used to avoid timing degradation, even for extreme error rates.

We note that the content presented herein has been disclosed in a previous publication [230], where the current Chapter presents the major highlights of this reduction to practice. A background on the mitigation of transient errors is presented in Section 5.1, whereas the selected and finally implemented scheme is outlined in Section 5.2. The experimental setup (target platform and application) is presented in Section 5.3 and results are extensively discussed in Section 5.5.

5.1 Background

In the context of Static Random Access Memories (SRAMs), which are mainly used in the cache blocks of modern digital systems, there is a wide variety of mechanisms that lead to transient errors [139]. The transient error is typically manifested as the corruption of the bit value that is stored in an SRAM cell. The reason may be a drop in the power or ground grid of the system [127]. In other cases, cosmic particles may disrupt the value stored in one or more cells of an SRAM array [179] (i.e. soft error). The current work focuses on the data

plane vulnerability to bit-level corruption without connecting it to the physical phenomena that affect SRAM cell stability.

Some mitigation approaches rely on additional HW components to guarantee the correct operation of low-level cache blocks. A popular technique is ECC [101], where codes determine if stored data have been corrupted by a transient error. Modern implementations involve the use of codes both for the data per se and for invariant attributes of the data [98], in what is known as Error Code Tagging (ECT). The addition of extra hardware to support error correcting capabilities may introduce significant area overheads, as in the case of Single-Error-Correction Double-Error-Detection of [212]. In a 64KB SRAM, the associated area overhead is quite extreme [129]. ECC has to be distinguished from the pure use of parity bits, that allow only the detection of erroneous bits within pieces of data. Other purely HW approaches employ redundancy of crucial HW components, as in [172]. Especially in cases where the redundancy scope is nearly global across the HW, as in the case of Field Programmable Gate Arrays (FPGAs) [124], the associated area and power overhead is very unacceptable. Tackling transient errors at the SW level comes with reduced observability of the platform's components. Prior art proposes the use of system memory, in order to protect the contents of registers [151]. This introduces a temporal overhead, as in the case of bringing a value from the memory (where it is protected) to the register file (where it is vulnerable). Other approaches turn to the optimized use of system resources, in order to guarantee the overall system reliability. Such an approach, in the context of many-core architectures, is based on the correlation between the utilization of a physical component and its vulnerability to failures [293]. Even though this a valid assumption, these techniques cannot handle stochastic reliability violations, as in the case of cosmic particle interference, where radiation interference is not coupled to resource utilization.

The above insufficiencies illustrate how HW *and* SW combined should yield the optimal mitigation schemes. Such an approach has been applied to an embedded processor [287], for protection against soft errors, where memories are placed out of scope. In the current Chapter, memories are the main focus and a many-core chip is used so that the overhead of inter-core communication [223] is accounted for within the mitigation scheme. Furthermore, *cross-layer reliability* features a shift to combined HW and SW mitigation techniques [40]. According to this paradigm, designers should “permit devices to fail and compensate [...] at higher [abstraction] levels” of the system stack [62]. Hence, the mitigation scheme instantiated in this Chapter is quite in-line with this design trend [183]. This scheme also allows the exploration of the time redundancy vs. error tolerance and system performance trade-off. A very interesting demonstration of this trade-off is performed in [73], however no optimal points are formally identified.

Finally, the term “performability” has been used to illustrate the probability of both timely and functionally correct operation [73]. In our case study, delay and functional correctness are treated as perfectly independent *cost functions* of system operation. A hybrid (HW-SW) technique for the mitigation of transient errors in data caches has been previously presented [236, 42]. It proposes small, protected, on-chip memory buffers that hold key application data. That way, upon error detection, a successful rollback to a previous functionally correct state is enabled. Here, we demonstrate the full viability of this methodology by studying a realistic streaming application on a real research-grade many-core chip. The Single-Chip Cloud Computer (SCC) experimental processor [104] is a 48-core “concept vehicle” created by Intel Labs as a platform for many-core software research. The streaming application of our choice is an MJPEG decoder. In [236, 42] only a simulation-based study has been performed. Here, we present a case-study that evaluates this technique on a real hardware platform, which incorporates the physical effects of a full system realization. Finally, we explore the performance vs. quality cost trade-off within the application and formally derive optimal solutions.

5.2 Rollback-Based Transient Error Mitigation

Definition 5.1. A *Safe Copy* of a data structure is a copy that resides in a memory area that allows HW detection and mitigation of transient errors, hence it is fully protected.

The proposed mitigation methodology is based on the assumption that low-level data caches (e.g. L1) are *unprotected*, but equipped only with transient error *detection* HW components. An on-chip *protected* memory bank is used to keep safe copies of key data structures (e.g. with a combination of parity checking and ECC). Upon error detection, safe copies are retrieved and the questionable computation is repeated. In prior art [236, 42], we see that use of a protected memory with optimized size leads to full error mitigation without violations of area constraints and with a maximum 22% energy overhead.

Definition 5.2. A *Rollback* is the retrieval of safe copies, which puts the digital system in a previous error-free state.

A rollback is initiated upon error detection in the unprotected memory. The safe copies overwrite the corrupt ones and the execution returns to a point, where the previously erroneous operation can be repeated. This rollback can be implemented with exceptions in the application source code. Here lies an opportunity for compiler optimizations that automatically create these

exceptions, but this concept falls beyond our scope. We stress that this technique targets transient errors in the data plane of a digital system. It is also orthogonal to cache-aware techniques that protect the *control plane* [215, 278]. In order to compartmentalize the application execution we use the *kernel* abstraction. The HW-SW mitigation scheme will be implemented at this granularity.

Definition 5.3. A **Kernel** is a subset of an application, performing a specific operation with strictly defined I/O.

Based on the data flow within a kernel, we distinguish two types of kernels. Each type will require a different maintenance of safe copies, which enable a demand-driven rollback.

Definition 5.4. A **Computation Kernel (CK)** is a kernel that performs a computation directly on its input data.

Definition 5.5. An **Accumulating Kernel (AK)** is a kernel that performs an operation on a set of accumulated data.

Using the CK and AK abstractions, we can describe a very wide variety of applications. Firstly, the JPEG decoder is a very good example, especially being a major component of our experimental setup. In this *image processing* application, accumulation and computation kernels (CKs and AKs) can be identified as indicated in Figure 5.1a. Apart from the multimedia domain, there is a wide variety of data flow applications, which can be described by a combination of computation and accumulation kernels. In the *wireless radio* domain there are many transmission techniques that can be abstracted with CKs and AKs. A representative example is the Orthogonal Frequency-Division Multiplexing (OFDM), which uses multiple carrier frequencies to transmit digital data [159]. An interpretation of an OFDM transmitter using AKs and CKs can be seen in Figure 5.1b. Another case, where combinations of AKs and CKs apply is that of *scientific computing*. For example, numerical integration of Ordinary Differential Equations (ODEs – for instance: $f(x, y, dy/dx, \dots) = 0$) with the fourth-order Runge-Kutta method is implemented with a combination of computation kernels (for the calculation of the k_1, k_2, k_3 and k_4 parameters) and an accumulation kernel that stores the result of each iteration. The (x_i, y_i) result is used as a starting point for the next iteration, which calculates (x_{i+1}, y_{i+1}) . In case adaptive step size is employed, the result of each iteration is compared with a previous calculation attempt to converge on the optimal step size [210] (visualized in Figure 5.1c). From the examples above, it is shown that the AK and CK abstractions (major component of the demonstrated mitigation methodology) are reusable across a variety of computing applications.

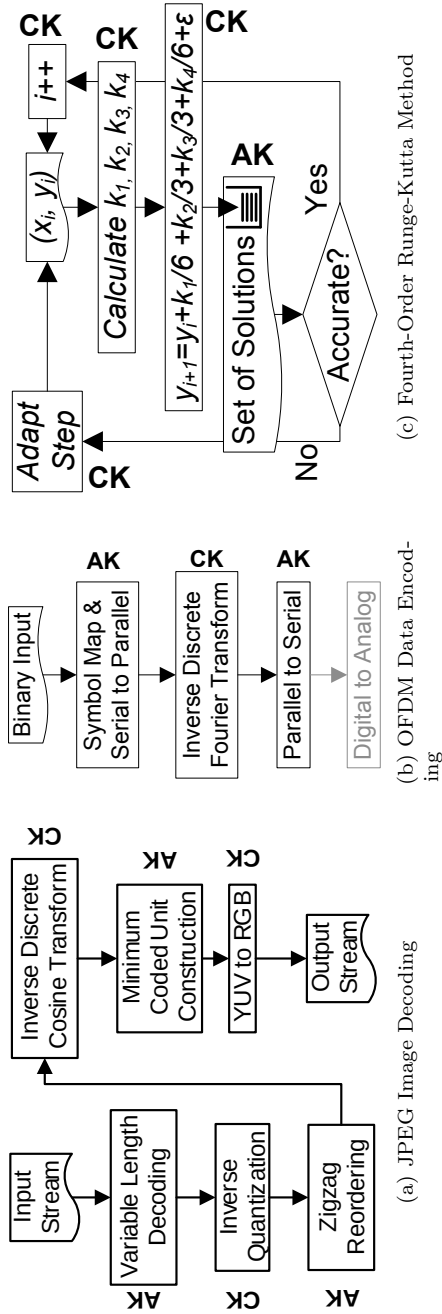


Figure 5.1: Various applications, abstracted using a combination of Computation Kernels (CKs) and Accumulation Kernels (AKs) [230]

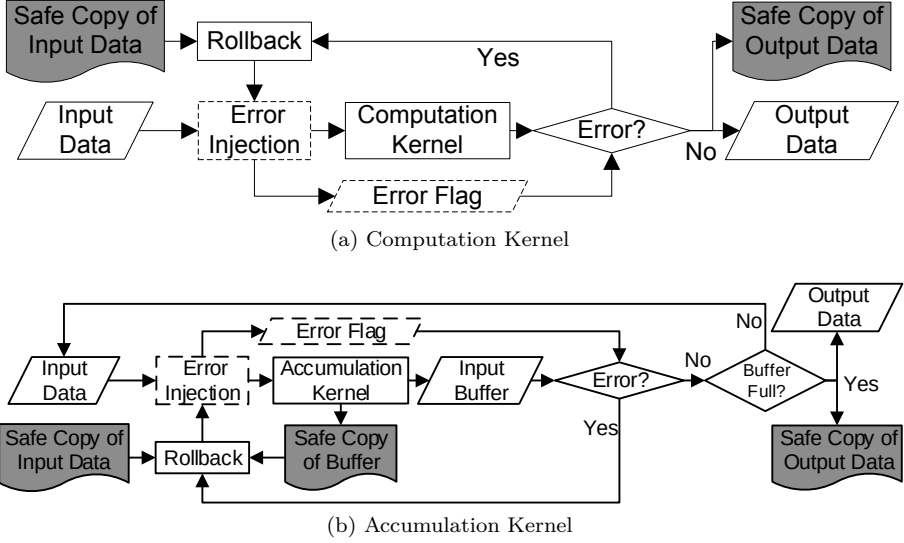


Figure 5.2: Hybrid (HW-SW) mitigation of errors in application data, for the kernel types of Definitions 5.4 and 5.5. Error injection stages are depicted, in lieu of real transient errors [230].

Having substantiated the reusability of the kernel abstraction, we now illustrate how a SW-initiated rollback can be applied in each case, to protect the executing system from transient errors. In the context of a *CK*, we can mitigate a transient error in the data plane by keeping a safe copy of the input data and by repeating the erroneous computation (illustrated in Figure 5.2a). Given the accumulation feature of the *AK* type, it is important to keep a safe copy of the input (or output) accumulation buffer that is used in the AK. Once the buffer is full, we can proceed with the execution and rollback in case a data plane error is captured (illustrated in Figure 5.2b).

With the mitigation semantics established, we now look into a many-core platform, with a series of kernels mapped on each execution node. Our goal is to implement the proposed hybrid (HW-SW) transient error mitigation scheme. We follow the steps illustrated in Figure 5.3, thus enhancing the ported application with demand-driven rollback capabilities. The architectural extensions required to implement the proposed methodology involve a low-level cache (e.g. L1) with detection capabilities. This detection can be implemented using simple parity circuitry, as also proposed in prior art on the mitigation of transient binary corruptions [278]. Furthermore, a small, low-level cache is required to keep safe copies of application data structures. This memory should be

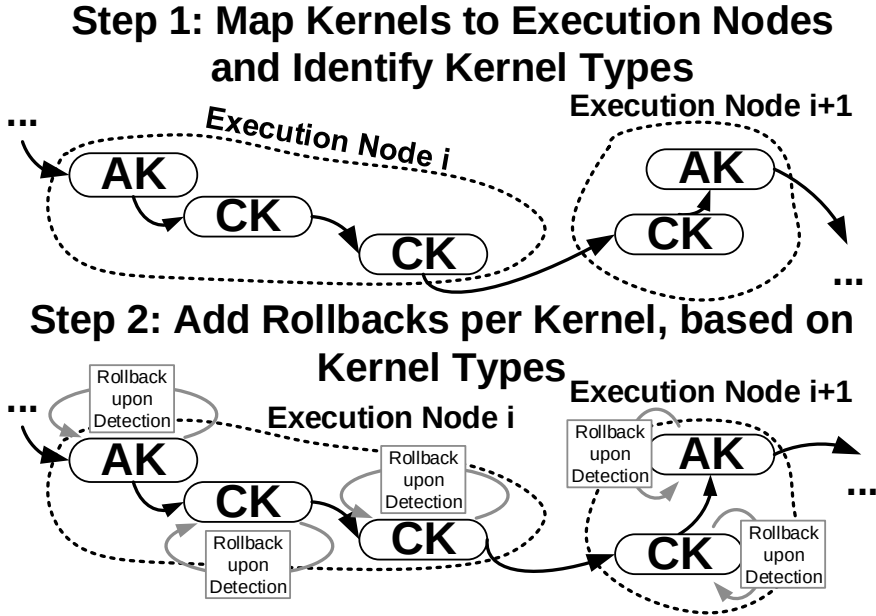


Figure 5.3: Steps of the mitigation methodology for a hypothetical application, being ported on a many-core platform. The concept is illustrated for two execution nodes and their kernels [230].

SW-controlled [22, 47], to enable demand-driven rollbacks. Thus, the proposed methodology is *hybrid (HW-SW) in nature*, since the detection occurs in HW and the mitigation is initiated by SW. A timing penalty is paid only when errors do occur and can be reclaimed if appropriate utilities are available on the target platform (e.g. DFS). The demand-driven nature of the proposed mitigation methodology removes any concerns about *error detection latency*, as reported in other cross-layer approaches [183].

5.3 Experimental Setup

For the demonstration of the HW-SW mitigation methodology, we use the Motion JPEG (MJPEG) decoder [144]. For the first step of the methodology (see Figure 5.3), we group application kernels to execution nodes and identify their types.

Each JPEG frame is stored as a series of Minimum Coded Units (MCUs).

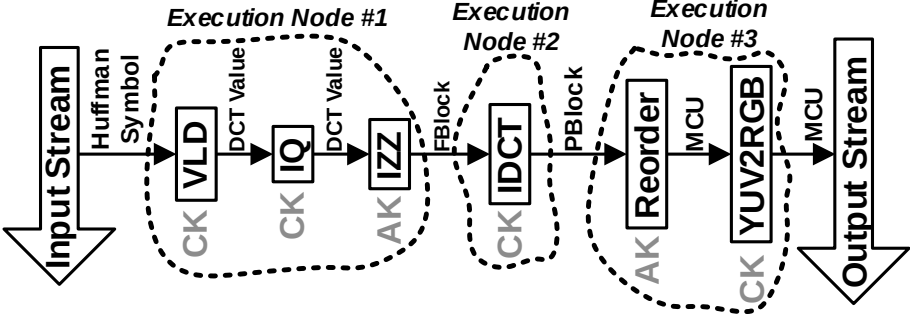


Figure 5.4: Initial step of the proposed mitigation methodology, as outlined in Figure 5.3. Application kernels are mapped to execution nodes and their types are identified [230].

MCUs are processed one-by-one and contain information on the Frequency Space (FBLOCKS) [178]. Every element of each FBlock is Huffman-encoded in the JPEG frame (Input Stream). This value has been transformed according to the Discrete Cosine Transform (DCT) and has to proceed to Variable Length Decoding (VLD). The produced decoded DCT value is dequantized based on a Quantization Table (IQ). Finally, the DCT value is placed in the FBlock, based on the Zigzag order that is enforced during encoding (IZZ). A complete FBlock will be forwarded to the Inverse Discrete Cosine Transform (IDCT), to produce a Pixel Space equivalent (PBlock). PBlocks of each MCU are rearranged (Reorder), transformed from the Luma and Chrominance (YUV) to the Red-Green-Blue (RGB) color space (YUV to RGB) and appended to the output RGB frame (Output Stream).

At this point, we observe a throughput mismatch between the execution nodes. By inspecting Figure 5.4, we see that the data size communicated across kernels of each execution node is varying. For example, in Execution Node 1, the IZZ kernel processes 1 DCT Value, while it is supposed to provide an entire FBlock to the next kernel. In view of this mismatch, we identify the two kernel types of Definitions 5.4 and 5.5, namely CKs and AKs. In order to balance the illustrated throughput mismatch, AKs are bound to exist at various points of each execution node. In Figure 5.2, we have conformed the proposed demand-driven rollback to the two possible types of kernels that appear in any application. That way, we can carry on with the second step of the mitigation methodology (see Figure 5.3), which is described at length in Section 5.4.

5.3.1 Target Platform

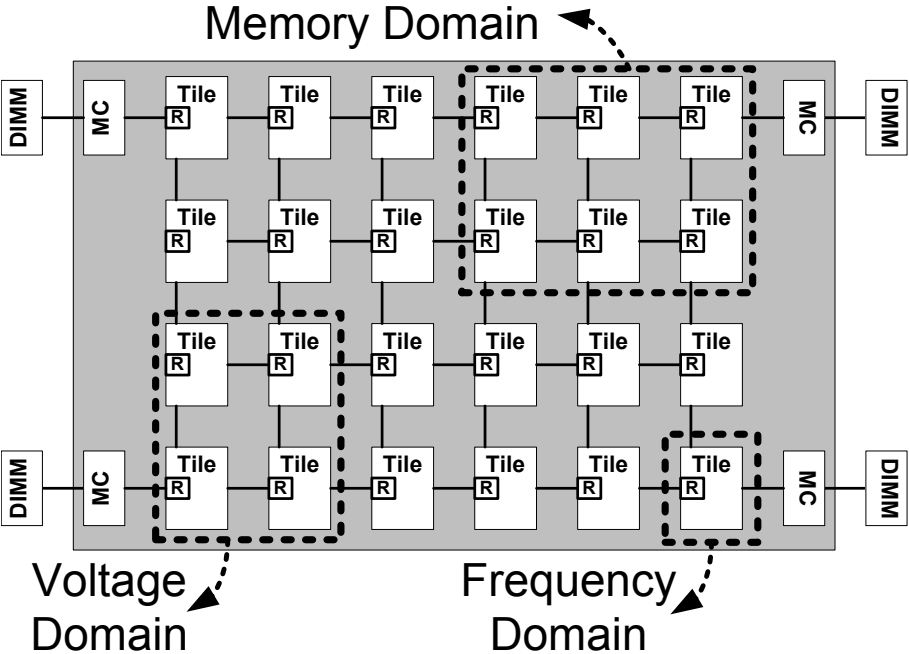
A real SCC chip is used for the demonstration of the hybrid (HW-SW) mitigation technique that was presented in Section 5.2. This processor [104] consists of 48 cores, which are organized in pairs, called tiles. The tiles are inter-connected through a mesh network (Figure 5.5a). Each tile contains two Intel Architecture 32 bit (IA-32) P54C cores, a unified L2 cache memory, a router that connects the tile to the mesh network and a Message Passing Buffer (MPB). The latter is used for the commitment and retrieval of messages between cores, enabling inter-core communication. A portion of the main memory is assigned to each core, thus creating the private memory space of the latter. The chip allows dynamic voltage scaling across a voltage domain of 4 tiles (8 cores), DFS across a single tile and is connected to a Management Console Personal Computer (MCPC). The SCC is programmed in a Single Program, Multiple Data (SPMD) fashion [171]. The source code is compiled on the MCPC and run on the SCC.

Having presented the three execution nodes for each JPEG decoder and the overall architecture of the SCC, we illustrate the setup that will be used for our experiments, a schematic of which is found in Figure 5.5b. Based on the execution node grouping of Figure 5.4 and with a total of 48 cores, we can obviously allow a maximum of 16 JPEG decoders running in parallel. That way, each execution node is mapped to exactly one SCC core. Input to our setup is a series of JPEG frames that have been demultiplexed from an input Motion JPEG (MJPEG) video stream. This pre-processing step is performed at the MCPC. The dispatching of the parallel decoders is also performed from the MCPC, using the software-provided utilities of the SCC, namely the command `rcцерun` [171].

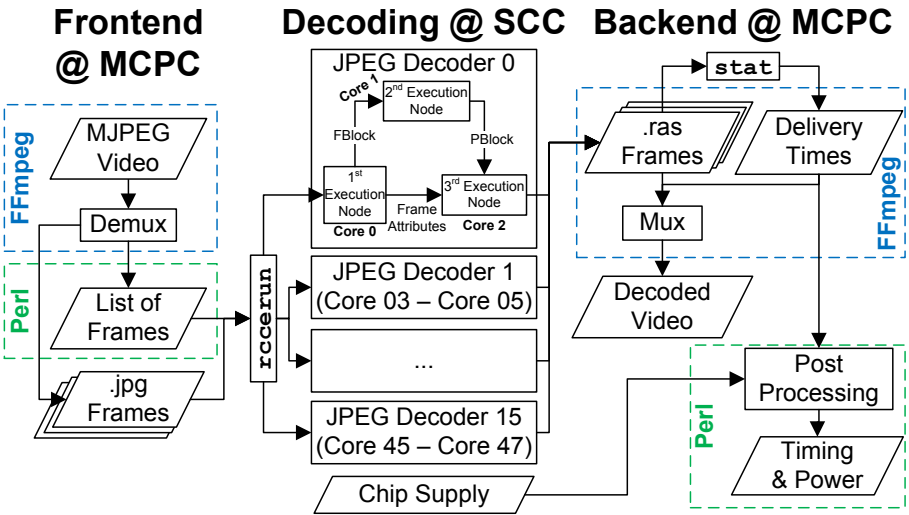
It is important to reflect on the suitability of the SCC for the illustration of the proposed HW-SW transient error mitigation technique. This chip is a representative platform for the many-core paradigm that is materialized in both the High Performance [6] and Embedded Computing [176] domains. It features small, low-level memory banks that are SW-controlled and distributed across the chip (MPBs). This is a major precondition of the proposed mitigation methodology, as outlined in Section 5.2 and in previous work [236, 42]. Furthermore, it features utilities that allow DFS, which can be used for reclaiming application slack that has been invested in error mitigation.

5.3.2 Error Injection & Application Correctness

In lieu of real errors in our experimental setup, we perform *error injection* using commands, in-line with the application source code. The injected errors are

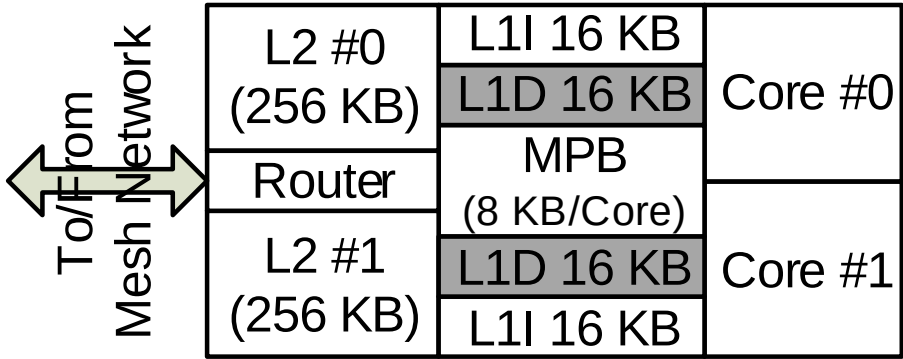


(a) Overview of the SCC system, containing 24 tiles. On-chip areas are highlighted in gray background. view of an SCC tile.

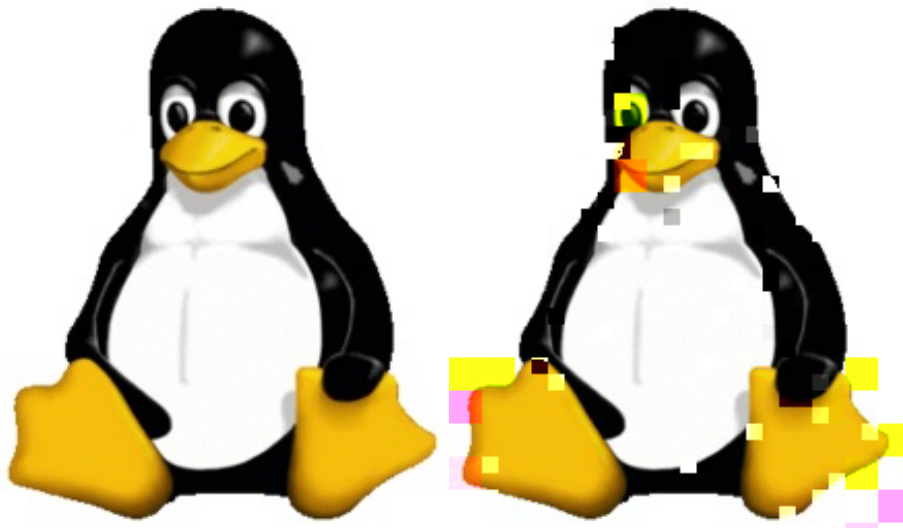


(b) Porting of 16 3-node JPEG decoders on the 48 cores of the SCC. The MCPC is used to dispatch decoders and keep performance metrics.

Figure 5.5: Schematic overview of the SCC layout and of our experimental setup [230]



(a) Artist’s impression of an SCC tile [171]; injected caches are gray-shaded.



(b) $0 \text{ FIT} \Rightarrow \text{SNR} = +\infty \text{ dB}$ (c) $2 \cdot 10^{16} \text{ FIT} \Rightarrow \text{SNR} = 26 \text{ dB}$

Figure 5.6: Scope of the error injection per SCC tile and examples of SNR degradation due to errors in the decoder’s data plane [230].

considered *transient* [290] in the sense that the responsible physical mechanism is typically of finite duration (e.g. an IR drop or a particle interference). However, we refrain from the “adjudged or hypothesized cause” [17] of the injected errors (i.e. the respective *fault*). In the context of our work and according to prior art on bit-level corruption of digital systems, the injected errors can be considered true *Detected Unrecoverable Errors* (DUEs) [187]. Thus, we can identify their occurrence, however they are not corrected in HW. The architectural assumption of parity as a means of detection enables the “fail-stop behavior” in the system [213], which should trigger the proposed SW rollbacks.

The intensity of DUE occurrence is measured in FITs and, in the context of this work, corresponds *only* to the data cache of each of the SCC cores (see Figure 5.6a). We use a probabilistic error injection model that is evaluated at each iteration of each application kernel. We illustrate this injection model with an example: Let us assume a kernel of the MJPEG decoder, for instance the IDCT of a Frequency Block. From application profiling, we know the duration (Δt) of an IDCT iteration. Let us also assume a user-defined constant τ that shows the Mean Time To Failure (MTTF), while for the sake of simplicity, we assume that the Mean Time to Repair (MTTR) is zero [187]. The probability of an error having occurred after one kernel iteration is given by Equation 5.1.

$$P_e = 1 - e^{-\Delta t/\tau} \quad (5.1)$$

For each evaluation of this Equation, a random number $r \in [0, 1]$ is compared to P_e . Iff $r < P_e$, we assume that an error has occurred and a random bit flip is enforced on the data of the respective kernel. Such formulas are evaluated for each iteration of all application kernels, thus providing full error injection coverage across the data plane of the experimental setup. To enable error injection, the user has to provide the appropriate FIT number which is connected with the aforementioned τ constant through Equation 5.2.

$$\text{MTTF (seconds)} = \frac{3600 \cdot 10^9}{\text{FIT}} = \tau \quad (5.2)$$

In order to specify a service failure, we need some metrics that quantify *application-level correctness* [152]. In terms of *latency*, a valid metric is the Frames per Second (FPS) rate. In terms of *output fidelity*, we will be using the average Signal to Noise Ratio (SNR) [92], calculated across all decoded frames. Given specifications for the FPS and average SNR, violation of these specifications corresponds to a service failure. In Figures 5.6b and 5.6c we can see examples of a decoded frame for the non-error-injected and error-injected cases.

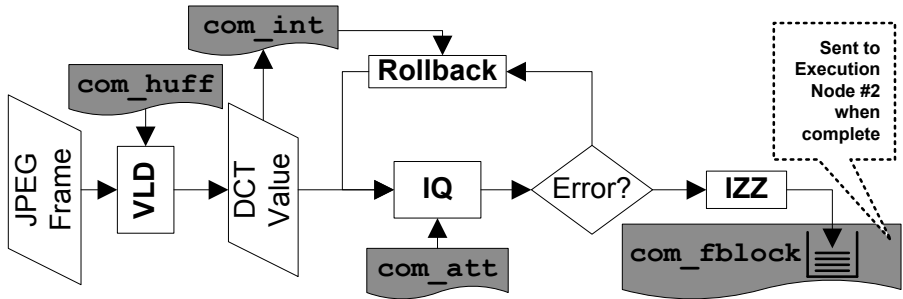
Finally, we comment on the FIT rates used in this work ($\geq 10^{13}$). Related work reports that a typical FIT rate (from radiation interference) is 1.6kFIT/Mb for six transistor SRAM cells [24]. For an L1 data cache of an SCC core we get an cache-wide failure rate of 200 FIT. This value is much smaller than the FIT rates considered in this work (e.g. Figure 5.6c). Let us also assume the Arrhenius relationship used to extrapolate failure rates from elevated temperature measurements [5]. It can be seen that the FIT rates used in this work are elevated by an order of typical acceleration factors, in comparison to FIT rates reported for digital circuits. The reason for this amplification is that we need to illustrate the trade-off between performance, total energy and application quality using simple benchmarks. Thus, we accelerate the injection campaign to receive results in reasonable execution times. This amplification is also following the trend of ascending FIT rates expected for systems in downscaled technologies and operating under reduced voltage supplies [24, 68].

5.4 Rollback Mitigation per Execution Node

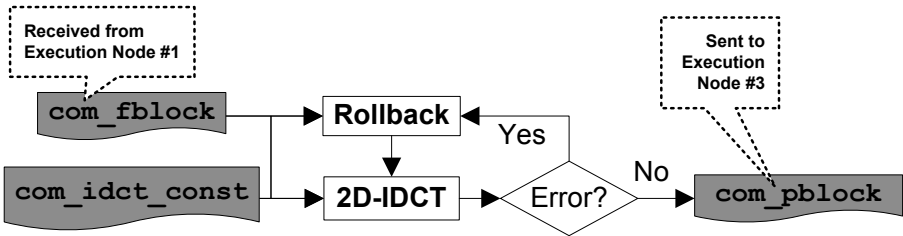
Node #1: The first execution node of the decoder performs the parsing of the JPEG frame. To guarantee the protection of the image attributes we create the `com_att` MPB data structure, which contains frame header information. The Huffman Tables are also protected in the `com_huff` location. The encoded Huffman symbol (that will produce one DCT value, i.e. an element of the FBlock) is kept in the safe location `com_int` until it is decoded. In case the error flag is set before the symbol is decoded, we retrieve the Huffman symbol from `com_int` and repeat the IDQ and IZZ processes. This is a typical instantiation of the mitigation technique for Computation Kernels (Definition 5.4 – Figure 5.2a). Decoded DCT values are accumulated in the FBlock before the latter is sent to the second execution node (Accumulation Kernel – Definition 5.5 – Figure 5.2b). The safe copy of this buffer is `com_fblock`. The above operations are repeated for every element (DCT value) of every FBlock (Figure 5.7a).

Node #2: This node calculates the Inverse Cosine Transform (IDCT) of every FBlock of the input JPEG frame. To find the PBlock, the 8-point, one dimensional IDCT is performed in the rows and columns of the FBlock [156].

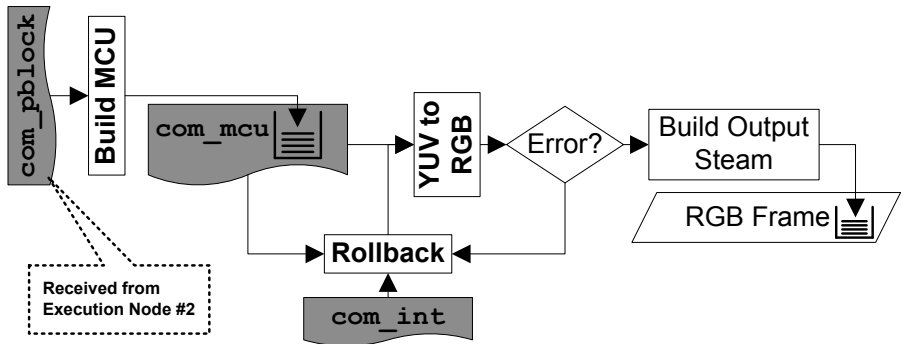
In order to enable rollbacks, we store the current FBlock, as well as the coefficients of the transform operation. The FBlock is communicated from the previous execution node so the safe copy for it (`com_fblock`) exists already in the MPB of the current core. A safe copy is also created for the IDCT constants at the beginning of the execution (`com_idct_const`). If the detection flag is raised after the IDCT of a complete FBlock, we



(a) **First Execution Node:** Safe copies for the Huffman & Quantization Tables are created during frame header parsing.



(b) **Second Execution Node:** A safe copy of the IDCT constants is created at the beginning of the execution.



(c) **Third Execution Node:** A safe copy for the constants of the YUV to RGB Transform is created at the beginning of the execution.

Figure 5.7: Second step of the methodology: implementing the rollbacks on demand for every encountered kernel type [230].

rollback by retrieving these two safe copies from the MPB and repeat the computation. This concept is repeated for every FBlock of the JPEG frame. It is illustrated in Figure 5.7b and perfectly resembles the mitigation scheme of Computation Kernels (Figure 5.2a).

Node #3: As PBlocks arrive to this execution node, they are accumulated in the MCU buffer (`com_mcu`). Before appending the decoded MCU to the output, we convert it from the YUV to the RGB color space, using Equation 5.3.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & \mathbf{359/256} \\ 1 & -\mathbf{11/32} & -\mathbf{183/256} \\ 1 & \mathbf{227/128} & 0 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix} \quad (5.3)$$

This operation resembles a Computation Kernel, so we need to keep a safe copy of the MCU and a safe copy of the constants of Equation 5.3 (highlighted in bold – `com_int`). If the error flag is set after the YUV to RGB transform, we can repeat the transform of the last decoded MCU and guarantee correct application output. The above concept is repeated for every MCU of the frame and is depicted in Figure 5.7c.

Even though the rollbacks presented above are completely demand driven, we need to reclaim the timing overhead that has been invested in the mitigation of bit flips in the application data, especially for aggressive injection rates. The reason is that, apart from output fidelity, we need to respect the timing constraints of the application. To avoid deadline violations, we accelerate the execution using *DFS*. At the beginning of each frame processing, the frequency is initialized at its default value (533 MHz). Slack reclaiming with DFS involves a percentage the execution being performed at high frequency (800 MHz). Increasing the clock frequency for a portion of the execution will lead to a higher power consumption, as illustrated in Figure 5.8. Naturally, we cannot allow more than 100% of the processing performed at high frequency.

Another way to reclaim application slack is to *willingly ignore* injected DUEs. Given that HW detection is communicated to the SW through a simple flag, the application can be informed whether an error has occurred. Also, the rollback mitigation scheme is SW-controlled. We can opt to ignore a portion of the errors that have occurred in the data plane, in order to reclaim application slack. Obviously, we can expect degradation of the output fidelity, which will be referred to as *average frame distortion*.

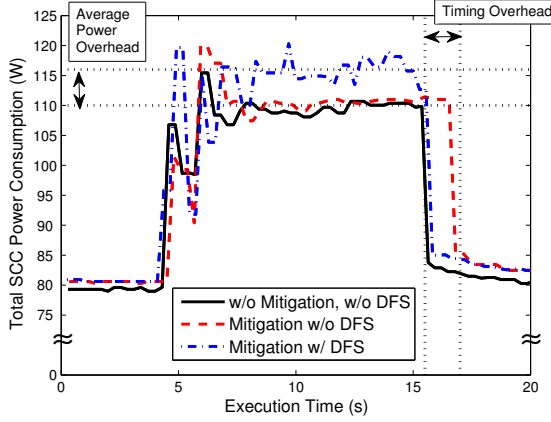


Figure 5.8: When DFS is used along with mitigation, the timing overhead is transformed to an average power equivalent [230].

5.5 Experimental Results

The results presented herein correspond to the decoding of a 640×360 pixel benchmark MJPEG video, encoded at 25 FPS and consisting of 250 frames. We maintain full chip utilization by porting 16 JPEG decoders that work in parallel (see Figure 5.5b). We monitor the power consumption as well as the FPS of the decoder. In a first round of experiments (Subsection 5.5.1), we exhaustively explore the trade-off between *output fidelity*, *application timing* and *application total energy*. In a second round of experiments (Subsection 5.5.2), we place a hard constraint on output fidelity and explore the impact of DFS on application timing and power/energy (quality cost).

The FIT rates employed are substantially increased, as compared to measurements typically reported [24, 5]. We aim to illustrate the effectiveness of the proposed hybrid (HW-SW) mitigation, since future systems are expected to suffer from increasing FIT rates [68]. Hence, we push the hybrid (HW-SW) technique to its boundaries, in order to identify the trade-off between application performance, output quality and the associated quality cost (i.e. power/energy).

5.5.1 Building the Pareto Space

As indicated in Section 5.4, we have devised two ways with which application slack can be reclaimed (will be referred to as “knobs”). DFS for a portion

k_{DFS} (%) of the execution allows speedup of frame decoding and comes at an average power cost. Willingly ignoring injected DUEs for a portion k_{WI} (%) of the execution reclaims application slack but increases average frame distortion. Given that both knobs are quantized as portions of the execution, we represent their configuration by a pair of values $(k_{\text{DFS}}, k_{\text{WI}})$. Each pair is considered as a *design point* for the MJPEG decoder and can be assessed in terms of total decoding duration, total energy consumed and average frame distortion. These metrics are the *cost functions* of the mitigation and slack reclaiming schemes. The first two are obvious, whereas the third is given as the normalized, inverse of the average SNR across all decoded frames.

We consider five possible configurations per knob: $\{0\%, 25\%, 50\%, 75\%, 100\%\}$, thus creating 25 possible design points, i.e. 25 points of the $(k_{\text{DFS}}, k_{\text{WI}})$ format. All design points are assessed for the aforementioned cost functions and for various FIT rates. By implementing a simple algorithm to identify the minima of a set of vectors [149], we can estimate which of the design points represent the *Pareto Optimal Points* [199] of our setup. These optimal points define a front of Pareto Optimality which can be piece-wise linear in two or three dimensions (depending on the number of cost functions we are evaluating against). The results of this analysis can be seen in Figures 5.9a through 5.9i *only* for the cost functions of distortion and delay. In Figures 5.9b through 5.9j we present a view of the Pareto Space when total energy is included. In Table 5.1 we see the Pareto Optimal configurations for each inspected FIT rate in the $(k_{\text{DFS}}, k_{\text{WI}})$ format. From the above experimental results we can draw the following conclusions:

- Depending on the number of cost functions, the number of Pareto Optimal knob configurations is different. This means that configurations that are considered suboptimal when only distortion and delay are considered are actually Pareto Optimal if we considered the total energy. These design points are highlighted in bold in Table 5.1.
- Low FIT rates feature less Pareto Optimal knob configurations. The reason is that for such FIT rates, mitigation interferes minimally with the runtime. Hence adjusting the knobs does not affect the optimality of the execution.
- Execution 100% at high frequency is prevalent in the sets of Pareto Optimal options. This is to be expected, assuming the frequency scaling overhead (i.e. time to signal the frequency dividers of the SCC tiles etc).

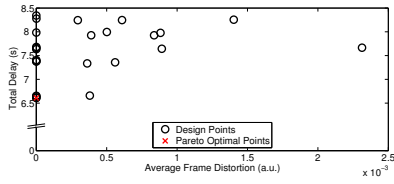
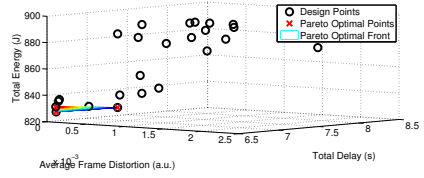
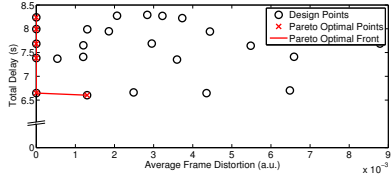
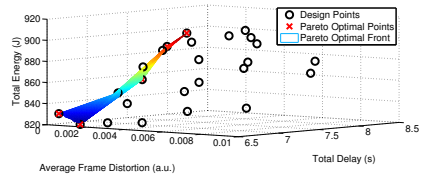
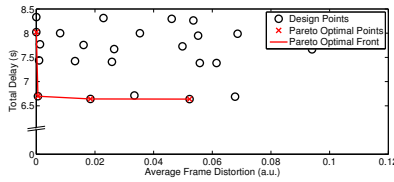
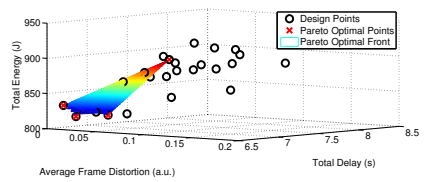
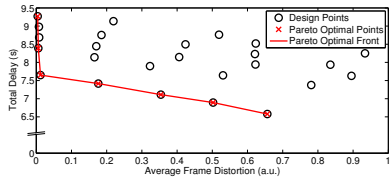
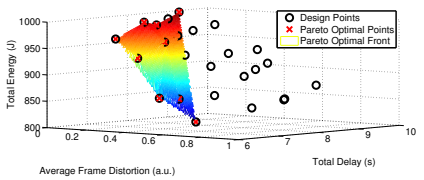
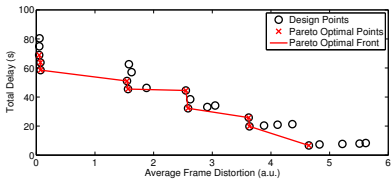
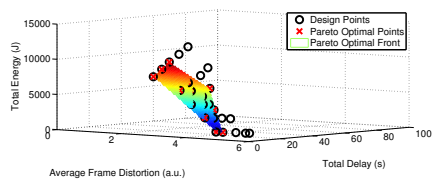
(a) 2D Pareto Front for Injected 10^{13} FIT(b) 3D Pareto front for Injected 10^{13} FIT(c) 2D Pareto Front for Injected 10^{14} FIT(d) 3D Pareto Front for Injected 10^{14} FIT(e) 2D Pareto Front for Injected 10^{15} FIT(f) 3D Pareto Front for Injected 10^{15} FIT(g) 2D Pareto Front for Injected 10^{16} FIT(h) 3D Pareto Front for Injected 10^{16} FIT(i) 2D Pareto Front for Injected 10^{17} FIT(j) 3D Pareto Front for Injected 10^{17} FIT

Figure 5.9: Exploration of the trade-off between the decoder's delay, energy and output fidelity for various rates of DUE injection [230]

Injected FIT	10 ¹³	10 ¹⁴	10 ¹⁵	10 ¹⁶	10 ¹⁷
Cost Functions: – Average Frame Distortion, – Total Delay	(100,100)	(0,0), (25,0), (50,0), (75,0), (100,0), (100,50)	(25,0), (100,0), (100,25), (100,75)	(0,0), (75,0), (100,0), (100, 25), (100,50), (100,75), (100,100)	(0,0), (25,0), (50,0), (25,25), (50,25), (100,50), (75,75), (100,75), (100,100)
Cost Functions: – Average Frame Distortion, – Total Delay and – Total Energy	(75,25), (100,25), (100,100)	(0,0), (25,0), (50,0), (75,0), (100,0), (100,50)	(25,0), (100,0), (100,25), (100,75)	(0,0), (50,0), (75,0), (100,0), (75,25), (100, 25), (100,50), (100,75), (100,100)	(0,0), (25,0), (0,25), (50,0), (25,25), (50,25), (100,50), (75,75), (75,100), (100,75), (100,100)

Table 5.1: Optimal configurations of the slack reclaiming knobs according to the (k_{DFS} %, k_{WI} %) format [230]

Operation Region	1	2	3
Injected FIT	$\leq 10^{14}$	$\in (10^{14}, 10^{16}]$	$> 10^{16}$
Application Fidelity	100% Correct Output	100% Correct Output	100% Correct Output
Application Throughput	Guaranteed	Guaranteed w/ DFS	Not Guaranteed w/ DFS
Comments	No need for DFS	$k_{\text{DFS}} \propto$ Injected FIT	Burnout (always at 800MHz)

Table 5.2: Guarantees on application correctness and timing based on the proposed mitigation and slack reclaiming with DFS [230]

5.5.2 Experiments with a Hard Constraint

In this series of experiments, we impose a hard constraint on the output fidelity, namely we cannot utilize the knob of willingly ignored DUEs (i.e. $k_{\text{WI}} = 0$). In the presence of the other slack reclaiming knob (DFS), we can make different timing guarantees for different injected FIT rates. We define three different *regions of operation* for the MJPEG decoder, depending on the aggression of the error injection (Table 5.2).

For small FIT rates ($\leq 10^{14}$; **Region 1**), the CPU time invested in mitigation is negligible, so we do not need to reclaim it with DFS. For $\text{FIT} \in (10^{14}, 10^{16}]$ (**Region 2**), we implement DFS in a percentage of the CPU time, proportional to the injected FIT rate (see Figure 5.10). In the previous two regions, we can always guarantee *both* the functional correctness of the output *and* its timely arrival. However, for $\text{FIT} > 10^{16}$ (**Region 3**), the slack reclaiming is performed in a “burnout” fashion, with the decoder operating always at high frequency. Even though the functional correctness of the output is guaranteed, we cannot guarantee a constant FPS rate.

Timing & Power

First, we get the timing of a *baseline* MJPEG decoder implementation, which lacks the mitigation and slack reclaiming features. Naturally, the timing and power of this implementation will not be affected by the injected FIT rate. Then,

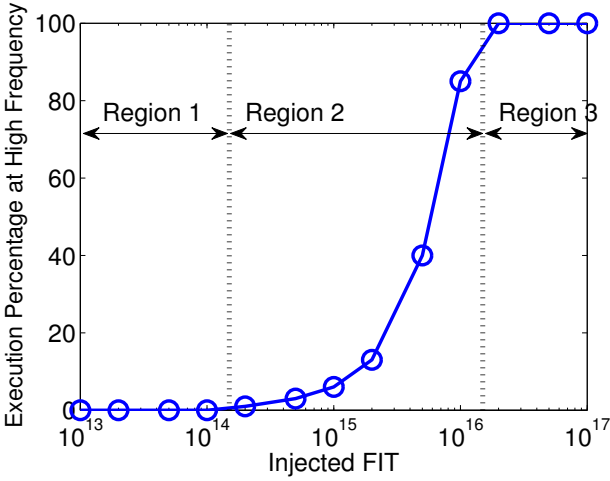


Figure 5.10: As the injected FIT rate increases, a more aggressive DFS is required to keep timely arriving video frames [230].

we execute the MJPEG decoder *with (w/) enabled error mitigation, but without (w/o) DFS*. We immediately witness the effect of the injected FIT rate on the FPS rate of the video decoder. When *DFS is employed* in accordance with the injected FIT rate (Figure 5.10), the FPS rate of the implementation remains constant in Regions 1 and 2. In Region 3, the slack reclaiming technique cannot keep up, even though the decoder operates continuously at high frequency. We can make similar observations for the total processing time of the MJPEG decoder (Figure 5.11b). From the baseline implementation, we identify the minor impact of the mitigation and slack reclaiming features on the application’s timing.

The SCC allows monitoring of the chip’s transient power consumption, as illustrated in Figure 5.8. For the inspected FIT rates, we extract the average power strictly for the execution interval (Figure 5.11c). Then, we can calculate the total energy invested per decoding session, as depicted in Figure 5.11d, using the processing time of Figure 5.11b. We can confirm the translation of the temporal overhead for mitigation purposes, to a power equivalent, when DFS is used. When the upper bound of the injected FIT rate is surpassed, the video is decoded entirely at 800MHz. Thus, the average power remains constant during decoding, at its highest value (Region 3 of Figure 5.11c).

Voltage Scaling Exploration

The results of Figure 5.11 show that our methodology mitigates *all* injected errors when $k_{WI} = 0$, regardless of the injected FIT rate. This is expected, given that the implemented rollbacks are fully demand driven. DFS reclaims application slack successfully up to an upper FIT rate bound.

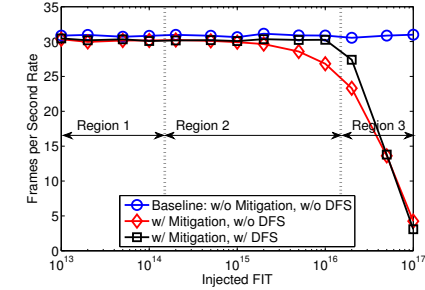
In systems that support Dynamic Voltage *and* Frequency Scaling (DVFS), like the SCC, the voltage is usually adjusted at the lowest possible value that can support the desired frequency [56]. In our setup, we deviate from this practice because runtime changes in the SCC voltage cannot happen as fast as changes to its frequency without the chip becoming unstable [56]. To estimate the dynamic power and energy if voltage could follow the frequency adjustments, we employ the square law for power consumption [253]. For operation at low frequency ($f_L = 533$ MHz), the minimum voltage is $V_{dd,L} = 0.8$ V. For the high frequency mode ($f_H = 800$ MHz), the minimum voltage is $V_{dd,H} = 1.1$ V [56]. That way, if k_{DFS} (p.u.) is the fraction of processing performed at high frequency (see Figure 5.10), we can estimate the average dynamic power consumption using Equation 5.4.

$$P = c [k_{DFS} V_{dd,H}^2 f_H + (1 - k_{DFS}) V_{dd,L}^2 f_L] \quad (5.4)$$

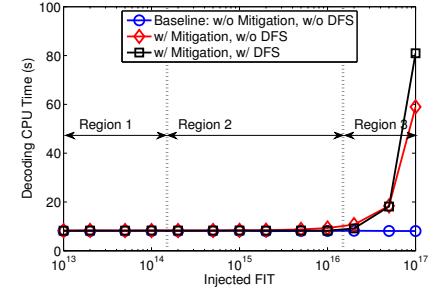
The c constant of Equation 5.4 reflects chip utilization and switching capacitance, which cannot be properly accounted for in our setup. Hence, the average dynamic power and energy estimations will contain a multiplier's uncertainty. Using Figure 5.11b, we estimate the average dynamic power consumption and the consequent total energy for the hypothetical case of DVFS. These estimations are displayed in Figures 5.11e and 5.11f using arbitrary values because of the c multiplier uncertainty.

In the out-lier cases of excessive error injection, the mitigation cost in average power surpasses the $\times 2$ limit. However, output fidelity is *always* guaranteed. From this series of experiments we can draw the following conclusions:

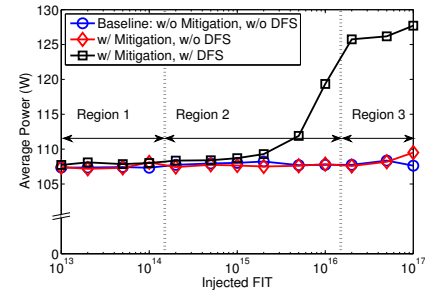
- Placing a hard constraint on output fidelity restricts the available degrees of freedom in terms of slack reclaiming. In case we could tolerate frame distortion, we could recreate a portion of the slack that DFS fails to reclaim.
- A more responsive voltage regulator could allow significant energy savings. That way, we could manipulate the voltage along with the frequency and gain heavily in energy, as indicated from Figures 5.11e and 5.11f.



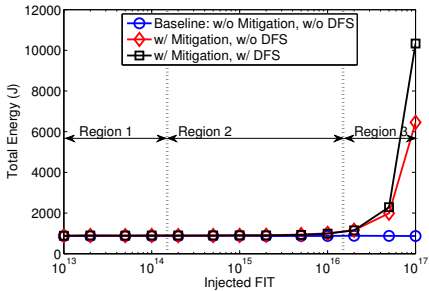
(a) As the FIT rate increases, mitigation consumes significant streaming application slack, which can be only reclaimed with DFS.



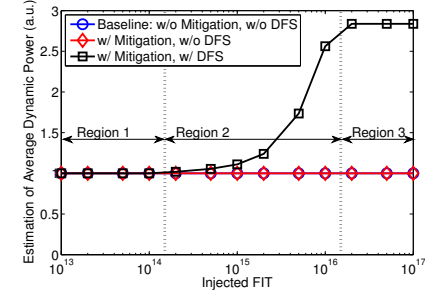
(b) When employed, DFS keeps the execution time of the decoder constant, up to the upper bound of the slack reclaiming technique.



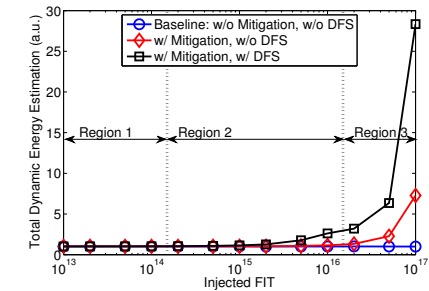
(c) When DFS is employed, the power increases proportionally to the FIT rate as long as slack reclaiming succeeds.



(d) The amount of energy invested in mitigation is effectively the same for distinct FIT rates, regardless of slack reclaiming.



(e) If DVFS could be employed, the dynamic power would be increased as the injection of errors becomes more aggressive.



(f) Without DVFS, the energy mirrors the mitigation overhead. If we use DVFS, the energy increases following the voltage.

Figure 5.11: Performance evaluation of the MJPEG decoder runtime with a hard constraint on output fidelity. DFS successfully reclaims slack, up to a certain upper bound of injected FIT [230].

Chapter 6

DVFS-Assisted Performance Dependability

In the previous Chapter, we have seen an example whereby a functional event (i.e. an error in the data plane of the MJPEG application), when mitigated, is turned to a parametric violation with respect to application timing (due to the timing overhead required by the application rollback). We have attempted to remove the possibility of timing violations by statically decided frequency boosts (as illustrated in Figure 5.10). It is clear that under the presence of RAS mechanisms that cost in timing (as is the rollback demonstrated in Chapter 5 a more automated solution should exist, so that application timing can be guaranteed. Exploring and formulating the limits of timing/performance dependability by a certain platform is also an attractive proposition.

It has to be clarified that the intervention of RAS events with application timing is not a straightforward process. In Chapter 5 the situation was pretty straightforward, since the rate of error events (measured in FITs) was identical to the rate of RAS mechanism invocation (i.e. the demand-driven rollback). In reality, and especially when reflecting on the wealth of RAS techniques, one understands that the situation is rather complicated. For instance, assuming the MTTF (or FIT) of permanent SRAM cell error (e.g. due to material aging). This rate is much more different than the rate of ECC invocation to alleviate the binary effects of such permanent malfunction. Actually, two preconditions must apply: **(i)** A specific cell (or block) should be defective and **(ii)** The very cell should be read, so that the permanent error can manifest itself. In this very case, it is the rate of defective cell (or block) read access that dictates the rate of ECC overhead interference (for specific ECC implementations).

With the above simplified example in mind, it is clear that for dependable performance in view of RAS overheads, one needs to know the rate at which time-consuming RAS techniques are invoked. This information is highly technique-specific and deriving it is a currently active area of research. Providing this very information is indicated in Figure 1.5 as gray-shaded arrow, indicating that is beyond the scope of the current text. The interested reader may wish to look into prior art such as the work of Hardy et al. [102] and Klokkanis [138].

Assuming that sufficient information is present about the rate of time-consuming RAS invocation, the open issue of ensuring dependable performance remains. This is exactly the purpose of the current Chapter. Initially, we present some background on how time-consuming can existing RAS techniques be (Section 6.1). Then, we formulate the problem of dependable performance in view of timing interference (which in our case is attributed to RAS events) and propose a solution based on traditional control theory. This is achieved in Section 6.2 of the current Chapter. Then, in Section 6.3 we perform a simulation-based evaluation of the proposed performance dependability scheme. A series of measurements from a real platform and application are presented and discussed in Section 6.4.

6.1 Background

To understand the performance implications of RAS mechanisms (and the need to absorb these overheads), one needs to reflect on RAS samples in the literature. Most of these techniques typically aim at functional violations and the mitigation thereof. Figure 6.1 contains a brief categorization of prior art in the domain of RAS mechanisms.

At the lowest abstraction level (i.e. circuit), we find techniques like *timing speculation*. In this case, a frequency value is speculated and upon functional error detection, the frequency is relaxed (or the voltage increased) at the cost of cycles executed at mis-predicted frequency [269, 75]. Moving on to the component level (e.g. IP blocks or memory macros), two main RAS techniques are used. One refers to correction of functional errors, as in the case of Error Correcting Codes (ECC – forward correction) [277] or fine-grain rollbacks in the instruction stream (backward correction) [42]. The former has an implementation-specific cycle overhead, whereas the latter costs in performance that is proportional to the rollback rate and granularity [230]. An alternative solution is the disabling of components in the processor, which appear to gradually malfunction. The design principle of *graceful degradation* is highly resonating with this specific trend [8]. It is noteworthy that, for practical

purposes, the invocation criterion is parametric: *only after a certain threshold of correctable errors is the responsible component disabled*. One of these approaches has appeared more than 20 years ago and mainly refers to cache memories [251]. Many years later, it appears that processor vendors accept the disabling of (almost) any component that is in sufficient abundance in a microprocessor system [174].

At the system abstraction, we can now expect a noticeable performance degradation, assuming that some of the above mechanisms are in place within a microprocessor. An initial attempt to quantify this overhead has been made using the *Performance Vulnerability Factor (PVF)*. Further on, effort has been put in quantifying *at runtime* the performance overhead of disabled cache blocks [138].

While cycle-accurate modeling of RAS-induced performance degradation remains an active field of study, there has been significant effort in ensuring system-level performance dependability. Using instruction stream quanta, optimization problems have been formulated, so that sufficient slack is available at all times during execution [189]. These being optimization problems, they require (at least) integer-linear programming solvers, which prevent their seamless integration in a run time software or, let alone, hardware fashion. One can also draw possible solutions from the domain of mixed criticality [35]. For instance, task scheduling in the presence of transient errors has been formally converted to a mixed criticality problem [106]. This work bounds the increment of execution time due to re-execution with worst case values (multiples of the default wall clock time). However, when reflecting on the variety of existing RAS mechanisms it is clear that the associated performance overhead is actually a continuous random variable [102], the bounding of which may call for extensive over-designing.

In view of prior art on RAS techniques, their performance impact and attempts to reclaim the latter, it is clear that an adaptive scheme is required in order to ensure dependable performance. The rate and intensity of RAS scheme invocation leads to increasing performance variability in digital systems. The current Chapter is positioned exactly in this direction: **(i)** We formulate dependable performance as a traditional control theory problem and we solve it with a PID-actuated DVFS. **(ii)** We construct an analytical criterion to resolve whether an arbitrary platform can cope with RAS-induced performance degradation, based on the platform's DVFS capabilities. **(iii)** All our concepts are verified experimentally on a real platform executing a spectrum sensing workload, where timing interference is injected in-line with the target application.

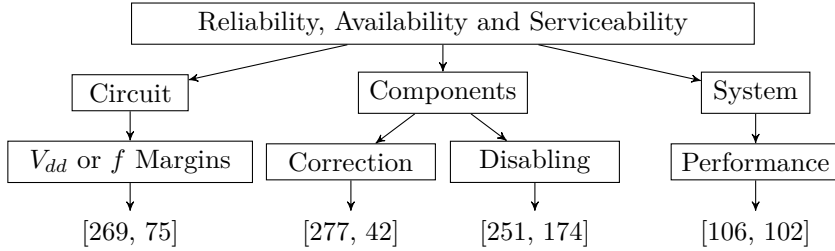


Figure 6.1: Synoptic classification of representative prior art samples on RAS techniques, their performance implications and attempts to reclaim the latter in a systematic way [226]

6.2 Problem Statement

We start the problem of dependable performance by perceiving a processor workload as a series of discrete and atomic operations, using concepts from related work.

Definition 6.1. *Thread Node (TN)* is a sequence of instructions with deterministic behavior (e.g. the body of a loop) that is executed in an atomic way [163].

As a result, a processor workload can be perceived as a series of TNs, the very of which depends on the control plane of the platform [226].

Definition 6.2. *Frequency multipliers (m)* reflect the available configurations of DVFS of a processor. They are normalized values around a reference frequency f .

Definition 6.3. *Cycle budget (N)* is the number of clock cycles for the execution of a TN in the absence of any RAS-related temporal overhead. A **timing budget** equivalent can also be defined, i.e. T_{ref} , for the reference frequency multiplier $m_{ref} = 1$.

Definition 6.4. *Cycle noise (x)* is a random variable for the number of cycles that are the result of RAS mechanism invocation during the TN's execution. In time units, the timing noise is added to the timing budget giving the real time per TN, i.e. T_{real} .

Cycle noise may intervene with the execution in a variety of ways, based on the invoked RAS mechanism in each case. For example, a disabled cache line [251] may increase capacity misses, signaling an intermittent of cache disabling with system performance (each time a miss is caused due to the disabled line) [138].

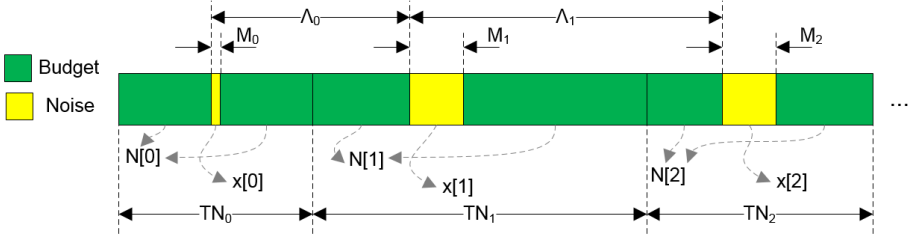


Figure 6.2: Interplay between cycle noise and budget for a series of TNs [226]

Given the deterministic behavior *within* each TN, we can easily formulate their timing characterization and also estimate it at design time. As a result, any workload can be abstracted by a series of TNs, each with its own cycle budget ($N[n]$), cycle noise ($x[n]$) and frequency multiplier ($m[n]$), as illustrated quantitatively in Figure 6.2. At this point, we highlight two important metrics, namely the rate and amplitude of RAS invocation. $\Lambda[0], \Lambda[1], \dots$ correspond to the time between RAS events that cost in performance. $M[0], M[1], M[2], \dots$ is the corresponding impact. For the respective averages (across the entire sequence of executed TNs), we use the notation λ and μ respectively. Both the rate of RAS event invocation and their impact are measured in clock cycles (cc). Naturally, in case there is no inter-TN noise, we can assume that $\mu[i] = x[i]$, $\forall i$. We note that in previous work [227], we used Mean Time to Failure (MTTF) and Mean Time to Repair (MTTR) to refer to rate and amplitude of cycle noise. MTTF will only be a consistent metric if it refers to the rate of transient errors or RAS-recovered errors. The MTTF of quasi-permanent errors (e.g. due to aging) may be much greater than the rate of their RAS invocation. To maintain the generality of our claims, we will use the $\Lambda[i]$, $M[i]$ and their average λ and μ .

Using the TN abstraction we can partition the instruction stream. A TN is the workload quantum of a program's critical region: That way a series of instructions can be split to atomic components. As illustrated in previous work [230], such quanta can be distinguished based on whether they perform computation or accumulation of data. Alternative partitioning may use intervals of catastrophic miss events [39], however such partitioning is out of our scope.

Similar to cycle noise, there are many events that intervene with execution at runtime. A case has certainly been made for "disruptive miss events" [78]. Such penalties must never be labeled as cycle noise, otherwise a highly pessimistic approach is maintained. We can connect cycle noise, cycle budget and the inherent performance attributes of a TN with L instructions using the Cycles per Instruction metric according to Equation 6.1. An arbitrary margin PVF_{limit}

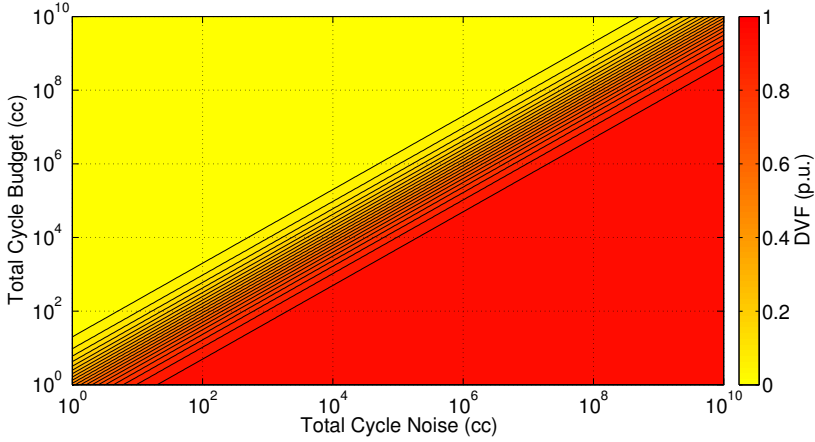


Figure 6.3: DVF for various cycle noises and budgets, assuming a constant reference frequency multiplier. In this case, PVF follows exactly the same behavior [226].

can added to widen the definition of cycle budget, by including other non-deterministic events.

$$N[n] = (1 + \text{PVF}_{\text{limit}}[n]) \times \text{CPI}[n] \times L[n] \quad (6.1)$$

Definition 6.5. *Performance Vulnerability Factor (PVF)* has been introduced in related work [102] as the per unit increase in number of clock cycles required for a specific workload, as a result of RAS technique invocation. After the n -th TN has been executed, PVF can be calculated according to Equation 6.2.

$$\text{PVF}[n] = 1 - \frac{\sum_{i=0}^n N[i]}{\sum_{i=0}^n N[i] + x[i]}, \text{ where } n = 0, 1, 2, \dots \quad (6.2)$$

Definition 6.6. In order to provide a temporal equivalent to PVF, we have introduced the **Deadline Vulnerability Factor (DVF)**. This is the per unit increase in execution time (T) required for an instruction stream, following the temporal overhead of invoked RAS mechanisms: $\text{DVF} = 1 - T_{\text{ref}}/T_{\text{real}}$.

By definition, for a constant reference frequency multiplier, DVF and PVF are identical. Then, DVF can be perceived through Figure 6.3: When the total cycle noise is less compared to cycle budget, DVF is scarcely affected. Our work contributes to cases where cycle noise is sufficient enough to lead to a DVF increment, namely RAS-induced performance degradation. This corresponds to the 45° band of intermediate DVF values in Figure 6.3. In order to approach the issue of dependable performance, one needs to have a run time indication of deadline vulnerability. This requires a recursive expression for DVF (in order to incorporate system “memory”), which is attempted in Equation 6.3 based on Definition 6.6 and where $n = 0, 1, 2, \dots$, $x[0] = 0$ and $m[0] = m_{\text{ref}}$.

$$\text{DVF}[n] = 1 - \frac{\sum_{i=0}^n T_{\text{ref}}[i]}{\sum_{i=0}^n T_{\text{real}}[i]} = 1 - \frac{\sum_{i=0}^n \frac{N[i]}{m_{\text{ref}}}}{\sum_{i=0}^n \frac{N[i] + x[i]}{m[i]}} = 1 - \frac{\sum_{i=0}^n \frac{N[i]}{m_{\text{ref}}}}{\frac{\sum_{i=0}^{n-1} \frac{N[i]}{m_{\text{ref}}}}{1 - \text{DVF}[n-1]} + \frac{N[n] + x[n]}{m[n]}} \quad (6.3)$$

Even if mathematically sound, this expression is rather complicated and may incur significant arithmetic computation overheads. An alternative metric is proposed below.

Definition 6.7. Slack (s) is the cumulative divergence from the cycle budget, assuming default (m_{ref}) and current ($m[n]$) frequency multipliers. We formulate slack according to Equation 6.4, where $n = 0, 1, 2, \dots$, $s[0] = x[0] = 0$ and $m[0] = m_{\text{ref}}$.

$$s[n] = T_{\text{ref}}[n] - T_{\text{real}}[n] + s[n-1] = \frac{N[n]}{m_{\text{ref}}} - \frac{N[n] + x[n]}{m[n]} + s[n-1] \quad (6.4)$$

It is clear that slack requires a simpler recursive computation when compared to DVF. A “register” is supposed to hold the previous slack value and the current cycle budget, noise and frequency multiplier are required (assuming that $m_{\text{ref}} = 1$). It is easy to prove that $\text{DVF}[n] = 0 \Leftrightarrow s[n] = 0$, $\forall n$ using perfect induction (see Table 6.1).

Proved Statement	$s[n] = 0 \Rightarrow \text{DVF}[n] = 0, n \in \mathbb{N}$	$\text{DVF}[n] = 0 \Rightarrow s[n] = 0, n \in \mathbb{N}$
Base Case	$s[1] = 0 \Rightarrow \frac{N[1]}{m_{\text{ref}}} = \frac{N[1]+x[1]}{m[1]} \quad (6.5)$ $\text{DVF}[1] = 1 - \frac{m[1]N[1]}{m_{\text{ref}}(N[1]+x[1])} \quad \text{Equation 6.5} \quad 0$	$\text{DVF}[1] = 0 \Rightarrow 1 - \frac{m[1]N[1]}{m_{\text{ref}}(N[1]+x[1])} = 0 \quad (6.6)$ $s[1] = \frac{N[1]}{m_{\text{ref}}} - \frac{N[1]+x[1]}{m[1]} \quad \text{Equation 6.6} \quad 0$
Inductive Hypothesis	$s[n] = 0 \Rightarrow \text{DVF}[n] = 0 \quad (6.7)$	$\text{DVF}[n] = 0 \Rightarrow s[n] = 0 \quad (6.8)$
Proof for $n+1$	$s[n+1] = 0 \Rightarrow$ $\frac{N[n+1]}{m_{\text{ref}}} - \frac{N[n+1]+x[n+1]}{m[n+1]} = 0 \quad (6.9)$ $\text{DVF}[n+1] =$ $= 1 - \frac{\frac{T_{\text{ref}}[n]}{1-\text{DVF}[n]} + \frac{N[n+1]+x[n+1]}{m[n+1]f}}{\frac{T_{\text{ref}}[n+1]}{1-\text{DVF}[n]} + \frac{N[n+1]+x[n+1]}{m[n+1]f}} \quad \text{Equation 6.7} \quad =$ $= 1 - \frac{T_{\text{ref}}[n+1]}{T_{\text{ref}}[n]} = \frac{N[n+1]+x[n+1]}{m[n+1]} \quad \text{Equation 6.9} \quad =$ $= 1 - \frac{T_{\text{ref}}[n+1]}{T_{\text{ref}}[n] + \frac{N[n+1]}{m_{\text{ref}}}} = 0$	$\text{DVF}[n+1] = 0 \Rightarrow$ $1 - \frac{T_{\text{ref}}[n+1]}{\frac{T_{\text{ref}}[n]}{1-\text{DVF}[n]} + \frac{N[n+1]+x[n+1]}{m[n+1]f}} = 0 \quad \text{Equation 6.8} \quad \Rightarrow$ $1 - \frac{T_{\text{ref}}[n+1]}{T_{\text{ref}}[n] + \frac{N[n+1]+x[n+1]}{m[n+1]f}} = 0 \Rightarrow$ $\frac{N[n+1]}{m_{\text{ref}}} = \frac{N[n+1]+x[n+1]}{m[n+1]} \quad (6.10)$ $s[n+1] = \frac{N[n+1]}{m_{\text{ref}}} - \frac{N[n+1]+x[n+1]}{m[n+1]} \quad \text{Equation 6.10} \quad 0$

Table 6.1: Based on Definition 6.6 and Equation 6.4, a zero target slack should be (intuitively) equivalent to a zero target DVF, namely $s[n] = 0 \Leftrightarrow \text{DVF}[n]$. We prove this statement using perfect induction, assuming that $s[0] = \text{DVF}[0] = 0$, which can be expected at the beginning of execution for a series of TNs. Hence, *when performance variability needs to be mitigated (i.e. DVF has to be kept at zero), it is sufficient to aim at forcing slack to converge to zero.*

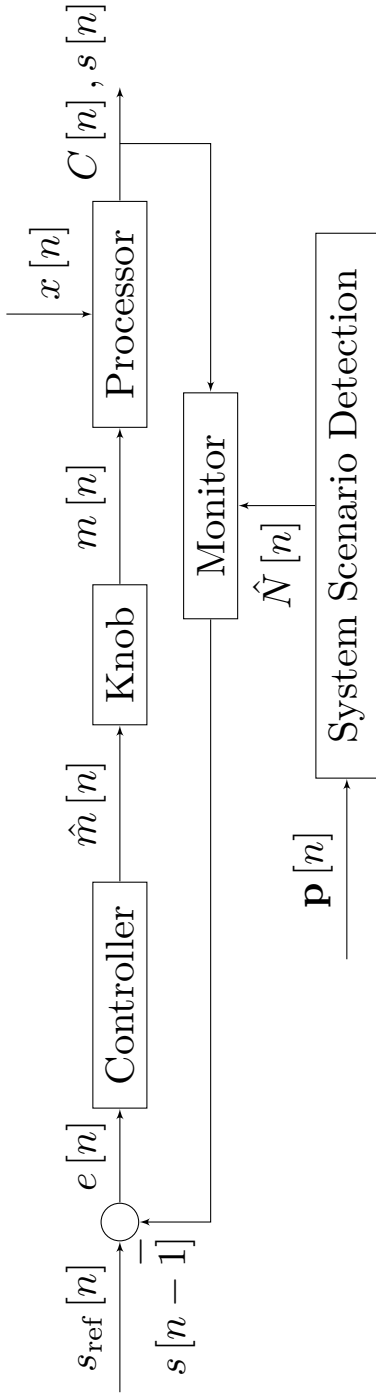


Figure 6.4: Block diagram of the proposed performance dependability scheme [226]

$s_{\text{ref}}[n]$	The target slack specification (0 unless otherwise stated)	$x[n]$	Timing noise interfering with current TN execution
$s[n-1]$	Slack at the end of previous TN	$C[n]$	Actual timing budget of currently executed TN
$e[n]$	Error signal $s_{\text{ref}}[n] - s[n-1]$	$s[n]$	Slack at the end of current TN
$\hat{m}[n]$	Proposed frequency multiplier	$\hat{N}[n]$	Default timing budget estimation for currently executed TN
$m[n]$	Applied frequency multiplier	$\mathbf{p}[n]$	System RTS parameters

Table 6.2: Variable explanation for the block diagram of Figure 6.4

The proposed DVFS policy is shown in Figure 6.4, whereas the involved variables are explained in Table 6.2. The general idea behind our principle is summarized in the following sentence: “one count of cycle noise can be reclaimed with two counts of cycle budget in twice the frequency”. Towards that direction, we propose a PID controller that responds to the divergence of actual slack $s[n]$ from its reference value s_{ref} . The controller suggests a frequency multiplier $\hat{m}$ and the closest available $m[n]$ is selected from the Operating Performance Point (OPP) list of processor [177].

A major precondition of the above control principle is the accurate knowledge of the cycle budget at run time. For that purpose, we leverage the system scenario methodology in order to identify the cycle budget of target TNs [91]. Having clustered and stored this information *at design time*, we can identify the cycle budget of the currently executed TN, based on its run time situation parameters ($\mathbf{p}[n]$).

An alternative solution would be to classify cycles as RAS-related or non-RAS-related, by extending previous monitors that bin cycles into performance-hampering events within the pipeline [78]. Even if promising, this scheme will remain outside our scope.

In order to bound the slack reclaiming capabilities of a platform, we need to model cycle noise. That is achieved using the mean number of cycles per RAS event, i.e. μ and the mean time between them, i.e. λ . The average extra cycles due to RAS events ($\bar{x}$) that interfere with an arbitrary cycle budget N can be given by Equation 6.11. This is a first order estimation, assuming no overlap between RAS events or dependencies between them. The total number of cycles ($\bar{C}$) is computed as a result.

$$\bar{x} = \frac{\bar{N}}{\lambda} \mu \Rightarrow \bar{C} = N + \bar{x} = N + \frac{N}{\lambda} \mu = N \left(1 + \frac{\mu}{\lambda} \right) \quad (6.11)$$

Let us assume that a DVFS policy is in place during the execution of the N cycles, in order to reclaim temporal overheads of RAS events. Across all the frequency multipliers selected during execution, we can calculate an equivalent average frequency multiplier m_{avg} . The goal here is to find the limits in m_{avg} that guarantee dependable performance in view of the $\bar{x}$ average cycle noise interference. The upper bound criterion is formulated in Equation 6.12, using the maximum frequency multiplier m_{max} .

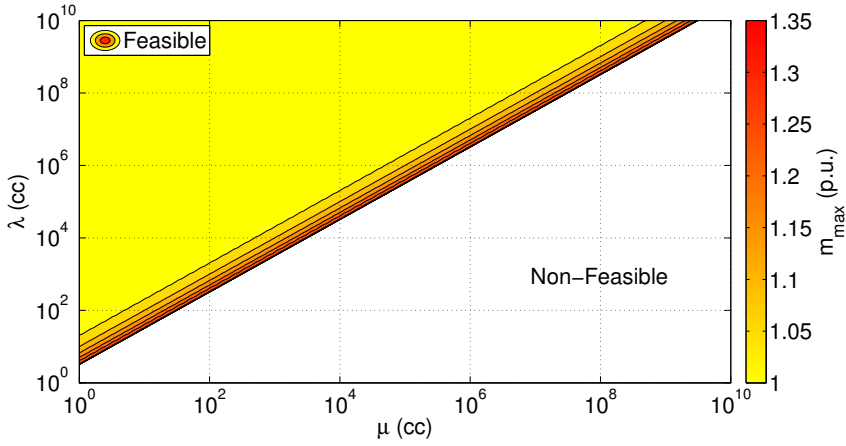


Figure 6.5: Numerical illustration of the average performance dependability criterion of Equation 6.12 [226]

$$\begin{aligned}
 T_{\text{ref}} - T_{\text{real}} \geq 0 &\stackrel{\text{Equation 6.11}}{\Rightarrow} \frac{N}{f \times m_{\text{ref}}} - \frac{N}{f \times m_{\text{avg}}} \left(1 + \frac{\mu}{\lambda}\right) \geq 0 \Rightarrow \\
 \frac{1}{m_{\text{ref}}} - \frac{1}{m_{\text{avg}}} \left(1 + \frac{\mu}{\lambda}\right) &\geq 0 \Rightarrow m_{\text{avg}} \geq m_{\text{ref}} \left(1 + \frac{\mu}{\lambda}\right) \stackrel{m_{\text{ref}}=1}{\Rightarrow} \quad (6.12) \\
 m_{\text{avg}} &\geq 1 + \frac{\mu}{\lambda} \stackrel{\text{Upper Bound}}{\Rightarrow} m_{\text{max}} \geq 1 + \frac{\mu}{\lambda}
 \end{aligned}$$

If the condition of Equation 6.12 is not satisfied for the maximum frequency multiplier then, regardless of the DVFS control principle applied, it is not feasible for the processor to guarantee dependable performance (Figure 6.5). Generally, this condition relates to the relative distance between the maximum and reference frequency multiplier.

6.3 Simulation-Based Verification

The proposed DVFS scheme can be applied to any set of cores, each with its own DVFS island, controller and monitor. We use the setup of Figure 6.6 to simulate a sequence of TNs executed by a single-core processor, with identical

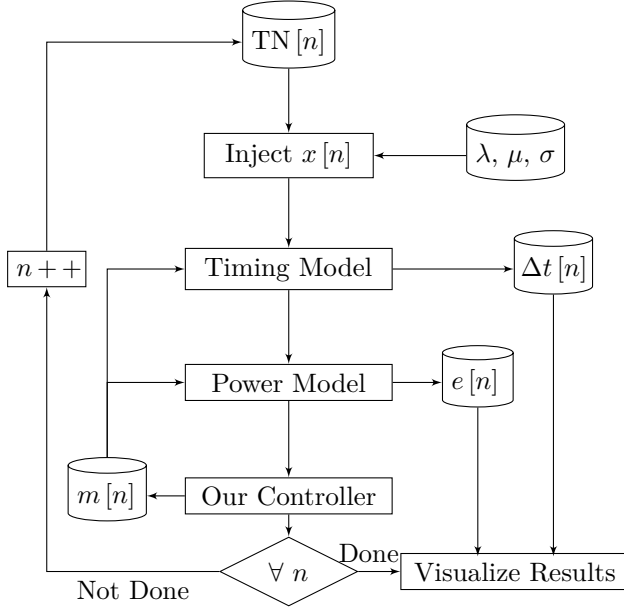


Figure 6.6: Setup simulating the execution of a TN sequence [227]

Parameter	Value	Description
s_{ref} (cc)	0	Target run time slack
N	10^5 cc	Cycle budget per TN
λ	5×10^{-4} s	Mean rate of cycle noise bursts
Norm (μ, σ)	$(10^3 \text{ cc}, 10^2 \text{ cc})$	Cycle noise amplitude statistics
k	10^{-5}	Controller gain ($k_p = k_i = k_d$)

Table 6.3: Default simulation parameters of the setup of Figure 6.6 [227]

cycle budgets N for each TN. We employ the power and DVFS model for an Intel® Core2 Duo E6850 processor, as presented in the literature [200]. For every TN, we calculate the duration ($\Delta[n]$) and energy ($e[n]$) including DVFS and cycle noise. For the incorporation of the latter, we use a cycle noise injection model, following the rationale of Section 6.2: a random number y is created for the processed TN and compared to a probability P derived from **Weib** ($\lambda, 1$). If $y < P$ cycle noise is injected to the respective TN. Unless stated otherwise, simulation parameters follow Table 6.3. The DVFS levels, overheads and corresponding frequency multipliers used in our simulations are shown in Table 6.4 and have been extracted from a prior art sample [200].

$\Delta t[n] = \frac{x[n] + N}{f[n]} + \Delta t_{\text{DVFS}}[n] \Rightarrow e[n] = P_{\text{steady}}[n] \times \Delta t[n] + e_{\text{DVFS}}[n]$					
Available DVFS Levels					
m	V_{dd} (V)	f (Ghz)	P_{steady} (W)		
1.2	1.3	3.074	57.2501		
1.1	1.25	2.852	49.1877		
1.0	1.2	2.588	41.2037		
0.9	1.15	2.281	33.3839		
0.7	1.1	1.932	25.8277		
0.6	1.05	1.540	18.6014		
DVFS Timing ($\Delta t_{\text{DVFS}} - \mu\text{s}$) / Energy ($e_{\text{DVFS}} - \mu\text{J}$) Overheads					
m	1.2	1.1	1.0	0.9	0.7
1.2	0 / 0	5 / 0	5 / 100	5 / 600	5 / 100
1.1	9.77 / 500	0 / 0	5 / 400	5 / 1100	5 / 400
1.0	17.29 / 800	10.95 / 500	0 / 0	5 / 0	5 / 700
0.9	27.29 / 1200	19.69 / 800	12.33 / 400	0 / 0	5 / 100
0.7	39.81 / 1600	31.47 / 1200	32.33 / 900	14.43 / 400	0 / 0
0.6	62.68 / 1700	54.50 / 2200	36.89 / 1200	33.35 / 800	17.14 / 400
					0 / 0

Table 6.4: Timing and power model used in the simulation setup of Figure 6.6 based on prior art [200], as originally disclosed in a previous publication [227]

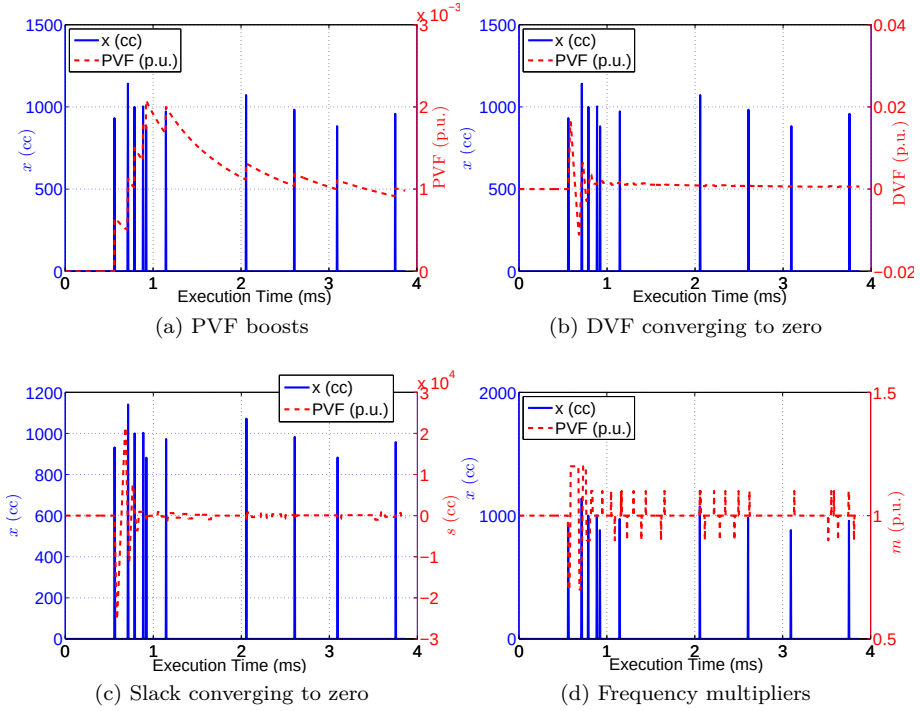


Figure 6.7: Step response of the proposed PID scheme [227]

Initially, we look into the transient behavior of the proposed scheme: We inject cycle noise with $\lambda = 0.5$ ms and $x[n] \sim \mathbf{Norm}(10^3 \text{ cc}, 10^2 \text{ cc})$, causing a PVF boost for every cycle noise burst occurrence (Figure 6.7a). The proposed Controller configures the frequency multiplier (Figure 6.7d) according to the observed slack, following the PID principle of Equation 6.13. This leads both DVF (Figure 6.7b) and slack (Figure 6.7c) to converge to zero.

$$\hat{m}[n] = \underbrace{e[n]k_p}_{\text{proportional}} + \underbrace{m_i[n-1] + e[n]k_i}_{\text{integral}} + \underbrace{(e[n] - e[n-1])k_d}_{\text{differential}} \quad (6.13)$$

At this point, we introduce the r parameter, which effectively represents a mod counter indicating how often (in TNs) we invoke the DVFS of the platform. Slack may be monitored after each TN, however DVFS alterations may be signaled every $r = 1, 2, 5, 10, \dots$ TNs. In Figure 6.8 we compare static V_{dd} and f (i.e. static frequency multipliers) against our scheme for a target execution time

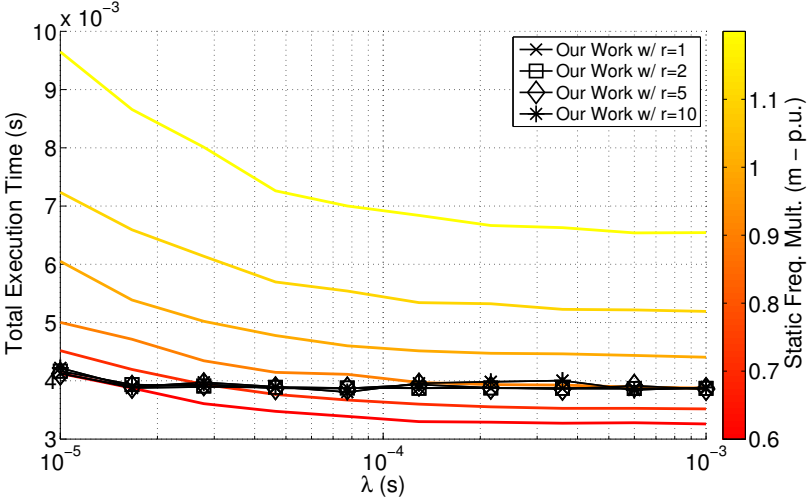


Figure 6.8: Static multipliers may miscompensate cycle noise [227]

of roughly 4 ms. Without the proposed technique, dependable performance cannot be guaranteed, especially for varying λ values. This is shown by the wide divergence of colored lines in Figure 6.8. We also explore the energy cost for dependable performance. Setting $r = 1$ means that V_{dd} and f are bound to change more frequently (after each TN processing, in the worst case), which incurs energy and timing costs (see Table 6.4). If we allow only a portion of frequency multiplier suggestions to be forwarded to the processor (i.e. increase r), then the total energy of the execution drops significantly as indicated in Figure 6.9. We quantify the energy overhead of our technique, with respect to the reference configuration, i.e. static $m = 1$ (see Table 6.4) with cycle noise. The most energy efficient case is for $r = 10$, where the maximum energy overhead is roughly 10%, in the price of negligible deviations from the target total execution time, as shown in Figure 6.10.

6.4 Experimental Verification

For the experimental verification of the proposed PID scheme, we used an Intel⁶ Xeon⁶ X3470 processor. For the results presented herein, we isolate the target application to one of the processor's cores (`cpu0`) using Linux control groups [244]. The target system runs a Linux operating system and DVFS is enabled with the `acpi-cpufreq` driver [67]. The available DVFS points of the processor

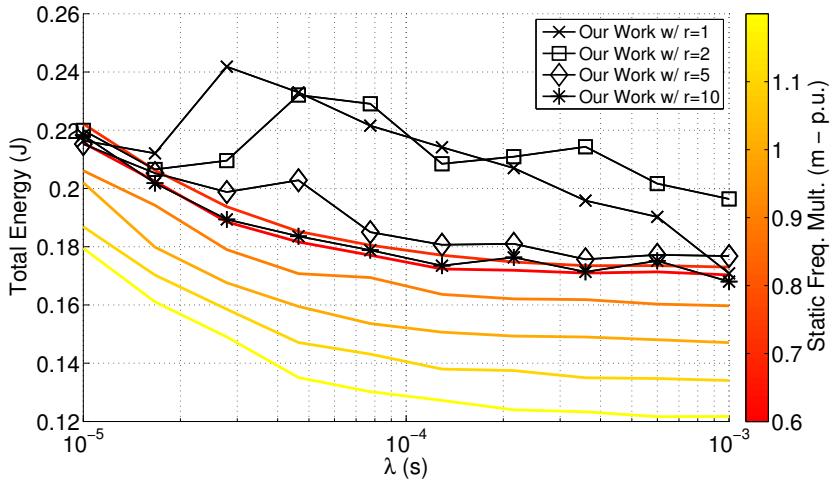


Figure 6.9: The r parameter can improve energy efficiency [227]

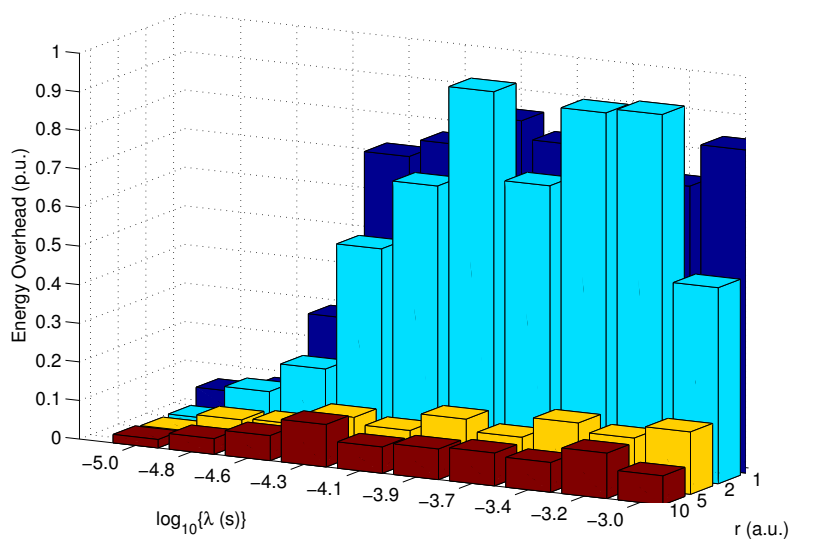


Figure 6.10: Increased r values lead to energy savings [227]

m (p.u.)	f (GHz)	V_{dd} (V)	m (p.u.)	f (GHz)	V_{dd} (V)
0.5625	1.197	0.6500	1.0000	2.128	1.0538
0.6250	1.330	0.7077	1.0625	2.261	1.1115
0.6875	1.463	0.7654	1.1250	2.394	1.1692
0.7500	1.596	0.8231	1.1875	2.527	1.2269
0.8125	1.729	0.8808	1.2500	2.660	1.2846
0.8750	1.862	0.9385	1.3125	2.793	1.3423
0.9375	1.995	0.9962	1.3750	2.926	1.4000

Table 6.5: Frequency multipliers of target platform: V_{dd} values are *hypothesized*, based on assumed f - V_{dd} relationship and publicly available voltage range of target processor.

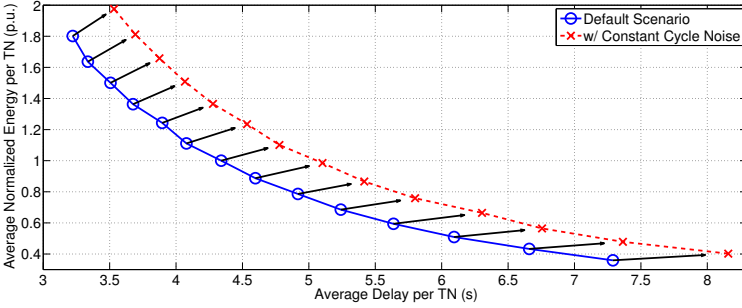


Figure 6.11: The default system scenario is defaced in the presence of cycle noise [226].

are shown in Table 6.5. Monitoring and actuation of DVFS can be performed by accessing the directory `/sys/devices/system/cpu/cpu0/cpufreq`.

The target application performs spectrum sensing which, in our context, will be assumed as an iterative task, corresponding to a TN. In Figure 6.11 we see the design time scenario characterization for this TN in the energy vs. delay trade-off space [91]. We also plot the scenario that corresponds to the injection of a constant, arbitrary value of cycle noise during TN execution. The injection of noise is done using the `usleep` command in-line with the target application, whereby $\Lambda[n] \sim \mathbf{Weib}(\lambda, 1)$ and $M[n] \sim \mathbf{Norm}(\mu, 0.1 \times \mu)$.

The relative distance between the scenarios corresponds to the amplitude of noise and the rate of switching between them corresponds to the rate of noise

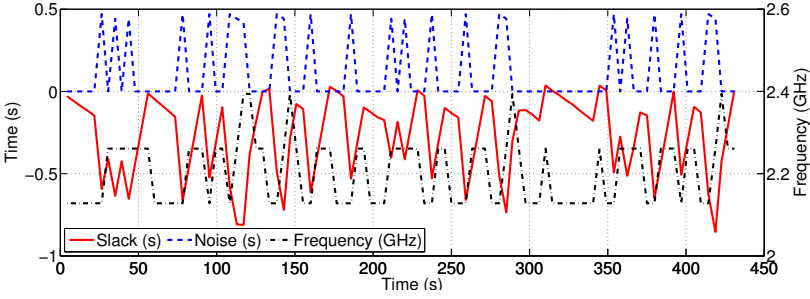


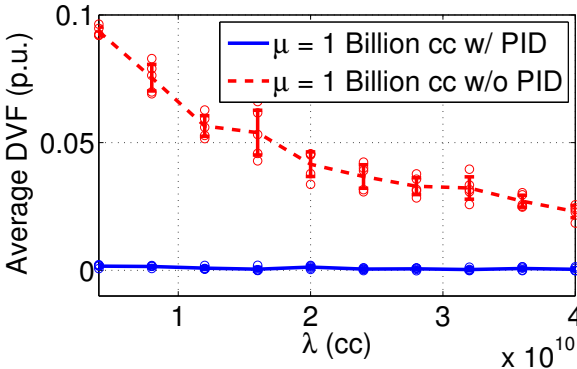
Figure 6.12: Transient view of the PID control functionality on the X3470 [226]

interference. Also, it is noteworthy that the impact of noise is more balanced between delay and energy at high frequencies. At low frequencies, the cost paid in delay is larger than the energy cost (by analogy to the high frequency case). Our goal is to detect the switch from the default to any “noisy” scenario (achieved through slack computation) and strive to absorb the cycle noise overhead with a PID-controlled DVFS boost.

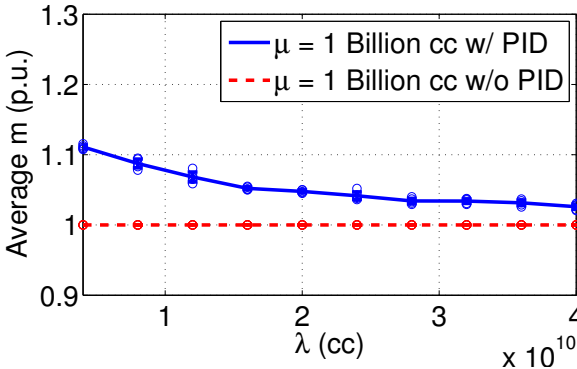
In Figure 6.12 we present a transient view of the proposed PID-controlled DVFS boosting. As cycle noise is injected, slack falls sharply and the our scheme reacts accordingly. In Figure 6.13 we explore performance dependability for a variety of λ values with and without the proposed DVFS technique. We illustrate the average DVF across 10 runs per λ value and the individual DVFs at the end of each run. It is clear that the performance overhead measured without our scheme is translated to an average frequency (and, thus, energy) increment when the PID controller is invoked. Even though controlling for slack (since DVF is not considered in the scheme of Figure 6.4), it is also clear the DVF is forced to zero in the presence of our DVFS policy.

Using Equation 6.12 we can directly bound the slack reclaiming capabilities of the target platform, by the characteristics of RAS events in terms of rate and impact. For the case of the target platform, and assuming the frequency multipliers of Table 6.5, the cases of cycle noise that can be reclaimed by the available multipliers on the processor are shown in Figure 6.14. Again, as expected, combinations of high μ and low λ make dependable performance a non-feasible target. For the same range of μ and λ , we perform runs of the target application on the target platform with the PID controller enabled.

The measurements of average DVF and the respective average frequency multiplier (both across 10 runs per μ and λ combination) are shown in Figures 6.15a and 6.15b, respectively. It is clear that for the region of μ and λ combinations predicted by Figure 6.14, our PID scheme succeeds in



(a)



(b)

Figure 6.13: Evaluating the PID controller with the target application on one core of an X3470, for a variety of MTTFs. Average DVF across 10 runs are presented within a 95% confidence interval. Circular markers show DVF at the end of each run [226].

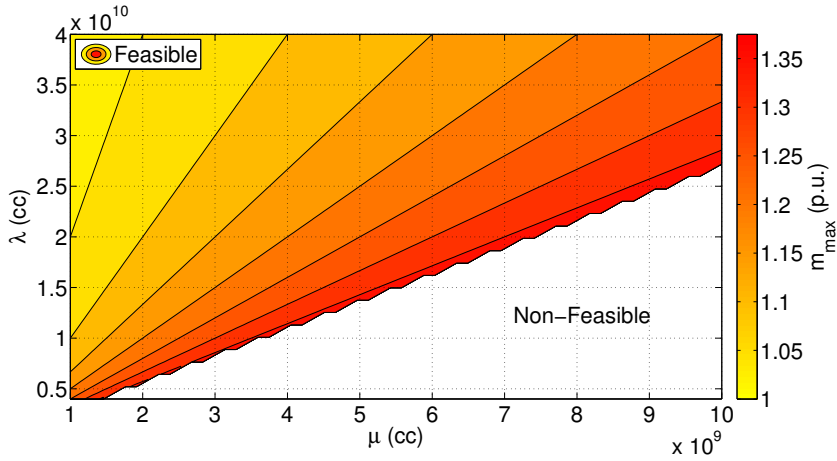
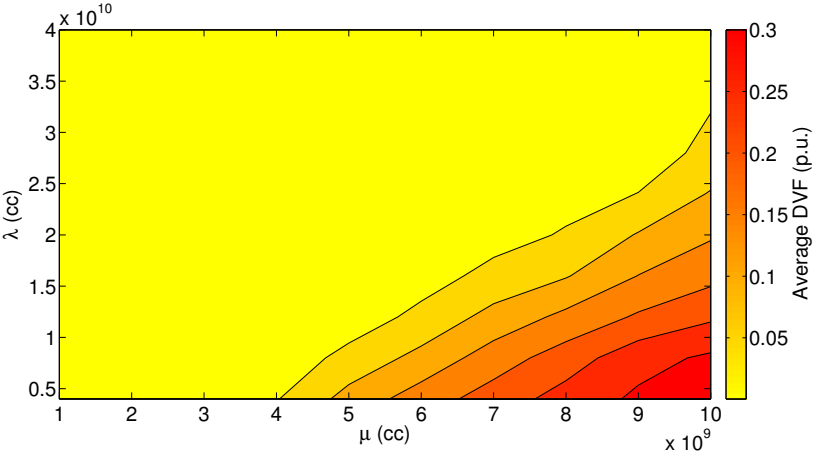
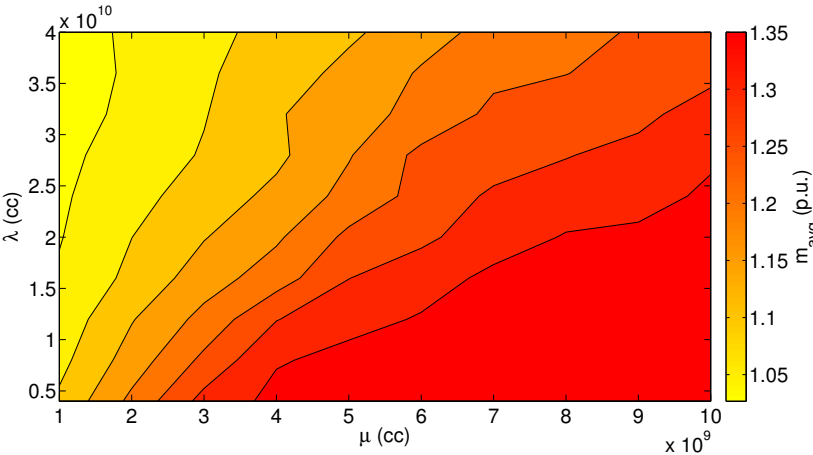


Figure 6.14: Illustration of the mean performance dependability criterion of Equation 6.12, for the range of μ , λ and $m_{\max}$ values encountered in our experiments [226]

achieving dependable performance, since the DVF is well below 5%. A small mismatch is observed for reduced μ values, however this can be attributed to the fact that the condition of Equation 6.12 is formulated at the worst case ($m_{\max}$ is used instead of m_{avg}). Overall, we can safely claim that our proposed DVFS-assisted performance dependability scheme succeeds in its goal. Also, our simple analytical model used to predict the capabilities of our methodology also works with significant accuracy.



(a)



(b)

Figure 6.15: Experimentally derived boundaries for slack reclaiming capabilities of our proposed performance dependability setup. The feasibility region predicted by Equation 6.12 and shown in Figure 6.14 is successfully verified through our measurements [226].

Chapter 7

Conclusion

7.1 General Remarks

With the technical contributions of the current text presented, we can now reflect on the milestones of the overall work in the direction of the modeling and mitigation of time-dependent variability. Even from the engagement with prior art, it is clear that as digital system complexity is increasing reliability considerations should no longer be concentrated at certain abstraction levels. The vision of the so called “cross-layer resilience” is increasingly stressed as reliability challenges become complex and their sources more diverse. In the current text we have verified the above observation, both in terms of modeling and mitigation, especially for the case of time-dependent variability.

We have started from the transistor and have exhaustively studied the impact of gate stack defects on the threshold voltage. State-of-the-art atomistic models have been a great help in realizing the time-dependent variability of phenomena like Bias Temperature Instability (BTI) and Random Telegraph Noise (RTN). In order to leverage the desirable features of such, otherwise impeccable models, a huge amount of engineering effort has to go in reducing their computational complexity. The current text contributes towards this direction with the Compact Digital Waveform (CDW). This signal representation format is used to isolate regions of signal activity that have waveform-level similarities. This enables strides over circuit/system workload, so that reliability analysis can be performed across the circuit’s/system’s lifetime. The technical steps presented in Chapter 3 are sufficient to substantiate the feasibility of CDW usage in standard design flows. However, full automation and seamless integration with

industrial flows are goals that surpass the scope of the current document.

Having provided sufficient hints towards computationally feasible reliability modeling of circuits, it is important to abstract miniaturized information, such as gate stack defect activity. This requires a link of variability phenomena (such as BTI and RTN) with metrics that reside at higher levels of abstraction, in order to enable reliability/variability-aware design. The failure probability (P_{fail}) of circuit components has been selected next, as we scale the abstraction stack (Chapter 4). We have implemented the Most Probable Failure Point (MPFP) technique in order to provide reasonable estimations for the time-dependent impact of gate stack defects on the reliability specifications of an SRAM cell. In doing so, we have compared to models for BTI, the Reaction-Diffusion and a Defect-Centric one. The results substantiate that inaccurate treatment of threshold voltage standard deviation may lead to surprising miscalculation of the P_{fail} .

The rate of component failure have been engaged in the next of step of this work. We have used the FIT rate (Failures in Time) to formulate error injection in the data plane of an experimental setup, using in-line software artifacts. A mitigation technique from prior art has been implemented on a research-grade many-core chip in order to mitigate the injected errors (Chapter 5). This technique involves fine-grain rollbacks and distributed memory banks used for inter-core communication have been used to implement the technique on the target setup. However, engagement of this technique costs in performance. A frequency boost decided at design-time is used to iron out the performance increment of rollbacks. Alternatively, ignoring a portion of injected errors has been explored. Three cost functions have been posed for this experimental setup: the correctness of the application (in view of injected errors), its performance and the total energy consumed. We have exhaustively visited all possible points of this space and have concluded on the optimal solutions.

In realizing the overall performance impact of Reliability, Availability and Serviceability (RAS) techniques (much like the rollbacks discussed previously), there is a clear need for a run time for dependable performance. This has been the last milestone of the current text, whereby the problem of performance dependability, in view of RAS overheads is formulated and a simple control-theoretic solution is proposed (Chapter 6). The proposed concept has been verified over a series of simple simulations and through the injection (and reclaim) of timing interference on a real processing platform.

7.2 Future Work

In spite of the technical contributions of the current text, it is clear that improvements and optimizations can increase the technology readiness of the concepts discussed herein. In the following list, we propose future work items, corresponding to the technical Chapters of the current text:

- Chapter 3: It is clear that the CDW can reduce the number of aging model iterations, while maintaining a certain degree of workload dependency. However, for the CDW to be fully deployed effort must be placed in handling bundles of signals. In that respect, a dominating signal, from the entire bundle, should be assigned the role of “heartbeat” so that grouping of regions is done based on its waveform characteristics. Additionally, it is important to enable a user-defined number of CDW points, so that `wavapprox` is forced to conform to this specification. Moreover, coding `wavapprox` in a more versatile implementation is also a valid course of future work. As described in Chapter 3, the CDW approximation is based on the derivation of periodic workloads that are, more or less, equivalent to the actual windowed signal. As a result, it is very important to generalize this technique and allow the derivation and analysis of arbitrary V_{gs} signals.
- Chapter 4: Regarding the MPFP, this is another domain that invites meaningful future contributions. Already, Section 4.5.2 contains a sufficient amount of pointers towards future work. Initially, the optimization problem that is solved by the MPFP kernel should be improved, so that the proper amount of probability mass is accounted for. Also, the solution of the optimization problem is in itself an area of inviting research, given the algorithmic enhancements that can go into developing optimization heuristics.
- Chapter 5: This Chapter has been developed on top of an existing functional mitigation technique. A real implementation of the proposed rollbacks could pave the way for fully realistic characterization of their associated performance overheads. Then, the proposed DVFS technique, as implemented herein for the SCC, could be verified in order to additionally guarantee dependable performance. Compiler annotations for the distinction of data structures that can be stored as safe copies is another valid course of future work regarding this Chapter. It is important to note that the rollback technique is a strictly functional mitigation technique. In Chapter 5 we have also explored the parametric impact of this technique as well. As a result, it is very logical to move towards a fully integrated functional *and* parametric solution.

Chapter 6: In order to solidify the claims of the proposed PID-based DVFS method for dependable performance, there are two main issues to be addressed. First, the link between component P_{fails} and rate of RAS invocation has to be additionally quantified. As already mentioned in the Introduction of the current text (Figure 1.5), this path is currently being explored by active research. However, it is the view of the author that additional effort has to be invested in order to clarify the extend at which current error rates cause (through RAS mitigation) performance degradation. Also, as outlined in Chapter 6, it is important to work towards on-chip monitors for cycle noise. We should note that in Chapter 6 the clock cycles of the processor (regardless of the applied frequency) have been considered as the timing quanta for dependable performance. As a result, it is interesting to explore the issues of slack and deadline vulnerability within the time horizon of a single cycle.

Appendix A

Guidance Framework Methodology

The classification framework of Figures 2.1 and 2.2 are the core tool with which we handle prior art on parametric reliability analysis and the respective mitigation techniques. Similar classifications have been previously made for domains such as design-time global scheduling techniques [142]. Figure A.1 provides an illustration of the classification methodology that is employed in the current survey, composing of two steps:

(i) The target research topic is treated as the root of a tree. Binary splits are decomposing the initial topic into subcategories. The split at each node is initiated by a criterion that yields two *strictly complementary* children. The splitting procedure is continued until a sufficient depth of the classification is reached. This inductive way, leads to a large number of categories, which are both orthogonal and sufficiently granular to completely cover the target research topic.

(ii) After the classification framework is created in the form of a binary tree, prior art samples are *mapped* to its leaves. Starting from the root of the tree, each work is evaluated against the splitting criteria and finally classified to the most suitable leaf. The suitability is resolved by the similarity of contents/claims of each mapped sample of prior art. During the mapping, one can highlight hybrid and cross-layer works, by instantiating them in more than one leaf of the classification.

Typically, for each leaf of the framework, a number of representative papers are

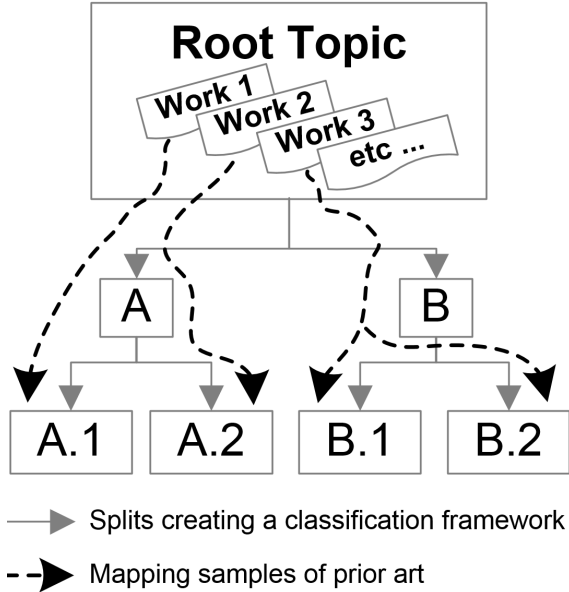


Figure A.1: Binary splits leading to the classification framework used to map prior art. In the above example, “Work 3” is a hybrid, given that it is mapped both in “B.1” and “B.2”.

instantiated and briefly commented upon, in order to motivate the mapping. We note that typical surveys tend to exhaustively list papers related to a somewhat restricted target domain. In most cases, cited work is evaluated against some criteria and trends are discussed. In contrast to typical surveys, the classification method used in the current text has the following unique features: (i) It enables a more *granular* categorization with much more classes than typical surveys. (ii) The classification presented herein is also *extensible*, in the sense that interested readers can induce further splits for any region of the framework they wish to explore further. (iii) It is also highly *reusable*, since future papers that come in the domain of parametric reliability can be seamlessly mapped on the framework by following the mapping principle of Figure A.1. (iv) Finally, it enables comparison of existing techniques in an unambiguous way, since each split has unique left/right branch decision outcomes for every characteristic of the parametric reliability considered.

Assuming a universe set U of *possible approaches* on a specific topic, the classification methodology proposes a split of this set into two complementary sets A and B based on a specific criterion p . For every possible approach x of the universe set, $p(x)$ can be either 1 or 0, depending on whether the criterion is

satisfied by the possible approach or not. That way, each x is placed under the correct A or B subset. No x can belong in both A and B . The formalization of a single split, as proposed by our methodology is presented in Equation A.1. With such consecutive, top-down splits, we end up with a classification framework, the branches of which are created by various splitting criteria. In other words, we perform the same splitting procedure in each of the derived sets A , B and carry on in an inductive way [231].

The criterion chosen for each split results in strictly complementary branches. The lowest level branches, or *leaves*, of the framework are the derived categories of the initial domain. These categories are not independent. Constraints are propagated between each pair in a split, which determines how to proceed in each of the two branches. So, that also implies an ordering to use them. In that respect we can see them as “constraint orthogonal” to each other. The levels of splitting define the *abstraction levels* of the framework. Each *state-of-the-art work* is mapped to the framework leaf, where it bears most resemblance in terms of assumptions and/or implementation [231].

$$\begin{aligned}
 U &= \{x : x \text{ is a possible approach}\} \\
 A &= \{x \in U : p(x) = 1\} \\
 B &= \{x \in U : p(x) = 0\} \\
 A \cap B &= \emptyset \text{ and } A \cup B = U
 \end{aligned}
 \tag{A.1}$$

The completeness of the classification framework is based on the property $A \cap B = \emptyset$ of Equation A.1, which refers to categorization of *possible approaches* on the topic in question. This property propagates to the leaves of the classification framework. It is obvious that *real approaches* cannot comply entirely with the above property. Apparently, it is possible to identify state-of-the-art works that can be placed under more than one leaf of the classification framework. This is acceptable, since research involves assumptions that relax the complementarity constraint to all other *possible approaches* on the topic in question. When categorizing such a *hybrid work* in a verbose way, we will analyze the aspect of that work that conforms to the appropriate framework leaf. In the current text, and specifically in Figures 2.1 and 2.2, we have only highlighted hybrid papers by underlining them in the respective classification trees [231].

A simple illustration of the splits performed for the creation of a classification framework can also be seen in Figure A.2. In this example, we use criteria p , q and r to create the framework of Figure A.2a. The partitioning of all the possible approaches into subsets can be seen in Figures A.2b through A.2d. Once we have reached the leaves of the framework, we populate it with state-of-the-art works. Some of these works may be hybrid, as illustrated in Figure A.2d, and are suitable for more than one leaf of the framework [231].

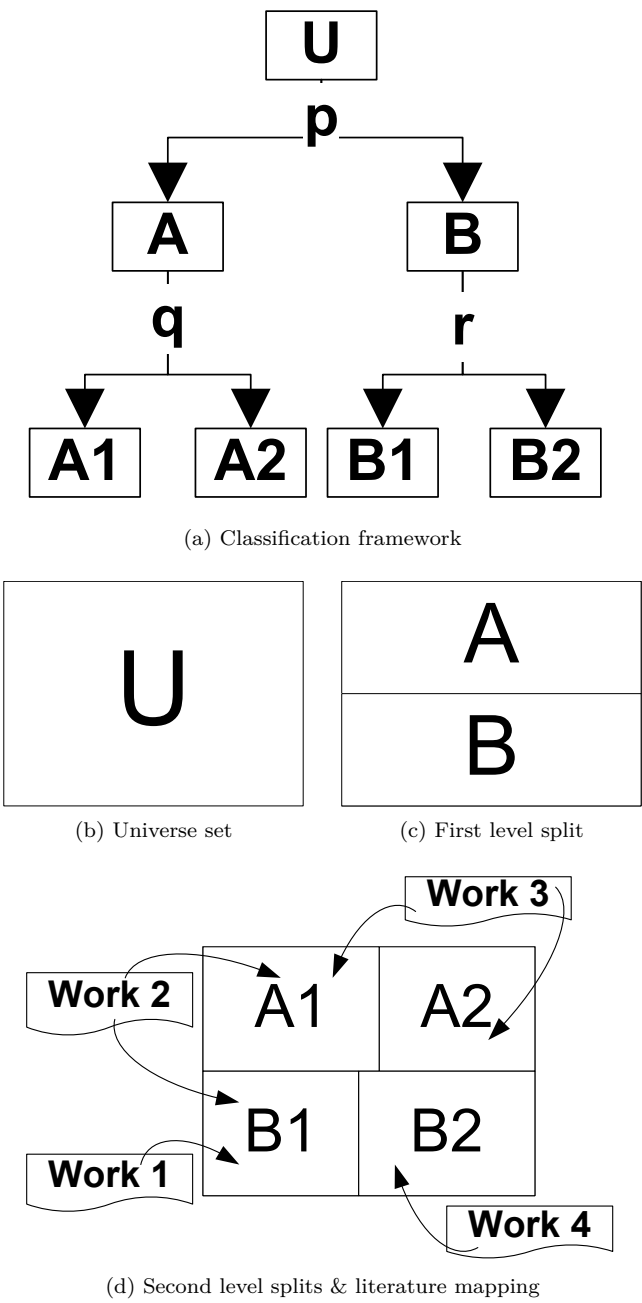


Figure A.2: Example of our classification framework methodology [231]

Bibliography

- [1] Predictive technology model (ptm). <http://ptm.asu.edu/>.
- [2] Switching activity interchange format language reference manual. Tech. Rep. 10/02, Synopsys Inc., 2002.
- [3] Ieee standard for verilog hardware description language. *IEEE Std 1364-2005 (Revision of IEEE Std 1364-2001)* (2006), 0_1–560.
- [4] Sparc enterprise m3000, m4000, m5000, m8000, and m9000 server architecture flexible, mainframe-class compute power. White paper, Oracle, April 2011.
- [5] Device reliability report. Tech. Rep. UG116 (v9.4), Xilinx Inc., 2013.
- [6] Intel xeon phi coprocessor – system software developers guide. Tech. Rep. SKU#328207-001EN, Intel Corporation, 2013.
- [7] 2PARMA CONSORTIUM. Parallel paradigms and run-time management techniques for many-core architectures. [Online]. Available: <http://2parma.microlab.ntua.gr>, 2010.
- [8] ABBOTT, W. R. Graceful degradation reliability. *Reliability, IEEE Transactions on R-26*, 1 (April 1977), 69–69.
- [9] ABDALLAH, R., AND SHANBHAG, N. Reducing energy at the minimum energy operating point via statistical error compensation. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 22, 6 (June 2014), 1328–1337.
- [10] ABELLA, J., VERA, X., AND GONZALEZ, A. Penelope: The nbti-aware processor. In *IEEE/ACM MICRO* (Dec 2007), pp. 85–96.
- [11] AGARWAL, A., BLAAUW, D., ZOLOTOV, V., AND VRUDHULA, S. Computation and refinement of statistical bounds on circuit delay. In

- Design Automation Conference, 2003. Proceedings* (June 2003), pp. 348–353.
- [12] ALAM, M., KUFLUOGLU, H., VARGHESE, D., AND MAHAPATRA, S. A comprehensive model for {PMOS} {NBTI} degradation: Recent progress. *Microelectronics Reliability* 47, 6 (2007), 853 – 862. Modelling the Negative Bias Temperature Instability.
 - [13] AMOROSO, S., GERRER, L., AND ASENOV, A. 3d tcad statistical analysis of transient charging in bti degradation of nanoscale mosfets. In *Simulation of Semiconductor Processes and Devices (SISPAD), 2013 International Conference on* (Sept 2013), pp. 5–8.
 - [14] ANDREI, A., MALHAIRE, C., , BRIDA, S., AND BARBIER, D. Reliability study of alti/tiw, polysilicon and ohmic contacts for piezoresistive pressure sensors applications. In *Sensors, 2004. Proceedings of IEEE* (oct. 2004), pp. 1125 – 1128 vol.3.
 - [15] ARROBO, G., AND GITLIN, R. Improving the reliability of wireless body area networks. In *IEEE EMBC* (Aug 2011), pp. 2192–2195.
 - [16] AVELAR, V. Comparing availability of various rack power redundancy configurations. Tech. rep., APC, 2003.
 - [17] AVIZIENIS, A., LAPRIE, J.-C., RANDELL, B., AND LANDWEHR, C. Basic concepts and taxonomy of dependable and secure computing. *IEEE TDSC* 1, 1 (Jan 2004), 11–33.
 - [18] AZEEM, B., AND HELAL, M. Performance evaluation of checkpoint/restart techniques: For mpi applications on amazon cloud. In *INFOS* (Dec 2014), pp. PDC–49–PDC–57.
 - [19] BAE, H.-C., CHOI, K.-S., EOM, Y.-S., LIM, B.-O., SUNG, K.-J., JUNG, S., KIM, B.-G., KANG, I.-S., AND MOON, J.-T. 3d sip module using tsv and novel solder bump maker. In *Electronic Components and Technology Conference (ECTC), 2010 Proceedings 60th* (June 2010), pp. 1637–1641.
 - [20] BALOUCHESTANI, M., RAAHEMIFAR, K., AND KRISHNAN, S. Increasing the reliability of wireless sensor network with a new testing approach based on compressed sensing theory. In *International Conference on Wireless and Optical Communications* (May 2011), pp. 1–4.
 - [21] BANIJAMALI, B., RAMALINGAM, S., LIU, H., AND KIM, M. Outstanding and innovative reliability study of 3d tsv interposer and fine pitch solder micro-bumps. In *IEEE ECTC* (May 2012), pp. 309–314.

- [22] BATHEN, L., AND DUTT, N. Software controlled memories for scalable many-core architectures. In *Embedded and Real-Time Computing Systems and Applications (RTCSA), 2012 IEEE 18th International Conference on* (Aug 2012), pp. 1–10.
- [23] BATHEN, L., DUTT, N., NICOLAU, A., AND GUPTA, P. Vamv: Variability-aware memory virtualization. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2012* (March 2012), pp. 284–287.
- [24] BAUMANN, R. Radiation-induced soft errors in advanced semiconductor technologies. *Device and Materials Reliability, IEEE Transactions on* 5, 3 (sept. 2005), 305 – 316.
- [25] BELLASI, P., MASSARI, G., AND FORNACIARI, W. Effective runtime resource management using linux control groups with the barbequerm framework. *ACM Trans. Embed. Comput. Syst.* 14, 2 (Mar. 2015), 39:1–39:17.
- [26] BHARATHAN, D., AND KELLY, K. An assessment of air cooling for use with automotive power electronics. In *Thermal and Thermomechanical Phenomena in Electronic Systems, 2008. ITherm 2008. 11th Intersociety Conference on* (May 2008), pp. 37–43.
- [27] BHAVNAGARWALA, A., KOSONOCKY, S., RADENS, C., STAWIASZ, K., MANN, R., YE, Q., AND CHIN, K. Fluctuation limits amp; scaling opportunities for cmos sram cells. In *Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International* (Dec 2005), pp. 659–662.
- [28] BICKFORD, J., AND FOREMAN, E. Selective voltage binning within a three-dimensional integrated chip stack, Aug. 14 2014. US Patent App. 13/766,276.
- [29] BISWAS, A., RACUNAS, P., CHEVERESAN, R., EMER, J., MUKHERJEE, S. S., AND RANGAN, R. Computing architectural vulnerability factors for address-based structures. In *International Symposium on Computer Architecture* (2005), pp. 532–543.
- [30] BLACK, B., ANNAVARAM, M., BREKELBAUM, N., DEVALE, J., JIANG, L., LOH, G., MCCAULEY, D., MORROW, P., NELSON, D., PANTUSO, D., REED, P., RUPLEY, J., SHANKAR, S., SHEN, J., AND WEBB, C. Die stacking (3d) microarchitecture. In *IEEE/ACM MICRO* (Dec 2006), pp. 469–479.
- [31] BOSE, P., HU, Z., RIVERS, J., SHIN, J., AND ZYUBAN, V. Predicting microprocessor lifetime reliability using architecture-level structure-aware techniques, Jan. 8 2009. US Patent App. 12/189,416.

- [32] BOSSUNG, J. W. Projection printing characterization. In *Developments in Semiconductor Microlithography II, Proc. of SPIE* (August 1977), vol. 100, pp. 80–84.
- [33] BSOU, A., MANJIKIAN, N., AND SHANG, L. Reliability- and process variation-aware placement for fpgas. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010* (March 2010), pp. 1809–1814.
- [34] BURD, T., AND BRODERSEN, R. Design issues for dynamic voltage scaling. In *Low Power Electronics and Design, 2000. ISLPED '00. Proceedings of the 2000 International Symposium on* (2000), pp. 9–14.
- [35] BURNS, A., AND DAVIS, R. I. Mixed criticality systems - a review. Tech. rep., Department of Computer Science – University of York, August 2015.
- [36] BUSBY, J., DANG, B., GRUBER, P., HAWKEN, D., SHAH, J., WEISMAN, R., PERFECTO, E., RUHMER, K., AND BUCHWALTER, S. C4np lead free solder bumping and 3d micro bumping. In *Advanced Semiconductor Manufacturing Conference, 2008. ASMC 2008. IEEE/SEMI* (May 2008), pp. 333–339.
- [37] BUTURLA, E. The use of tcad in semiconductor technology development. In *Custom Integrated Circuits Conference, 1991., Proceedings of the IEEE 1991* (May 1991), pp. 23.1/1–23.1/7.
- [38] CAMAROTA, R. Method and apparatus for self-annealing multi-die interconnect redundancy control, Sept. 17 2013. US Patent 8,539,420.
- [39] CARLSON, T. E., HEIRMAN, W., EYERMAN, S., HUR, I., AND EECKHOUT, L. An evaluation of high-level mechanistic core models. *ACM Transactions on Architecture and Code Optimization (TACO)* (2014).
- [40] CARTER, N., NAEIMI, H., AND GARDNER, D. Design techniques for cross-layer resilience. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010* (March 2010), pp. 1023–1028.
- [41] CATTHOOR, F., KACZER, B., RODOPOULOS, D., VALDUGA, A., AND BANDHU, M. Time and workload dependent circuit simulation, Oct. 10 2012. EP Patent App. EP20,120,163,404.
- [42] CATTHOOR, F., SABRY, M., MA, Z., AND ALONSO, D. Method and system for real-time error mitigation, Nov. 14 2013. US Patent App. 13/467,758.
- [43] CHAN, T.-B., SARTORI, J., GUPTA, P., AND KUMAR, R. On the efficacy of nbtI mitigation techniques. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2011* (March 2011), pp. 1–6.

- [44] CHARRON, F. Air flow management system for an internet data center, Jan. 6 2004. US Patent 6,672,955.
- [45] CHAWARE, R., NAGARAJAN, K., NG, K., AND PAI, S. Assembly process integration challenges and reliability assessment of multiple 28nm fpgas assembled on a large 65nm passive interposer. In *IEEE IRPS* (April 2012).
- [46] CHE, F., LIM, S., CHAI, T., AND ZHANG, X. Structure design optimization and reliability analysis on a pyramidal-shape three-die-stacked package with through-silicon via. *IEEE TREL* 12, 2 (June 2012), 201–208.
- [47] CHEN, C., MANZANO, J., GAN, G., GAO, G., AND SARKAR, V. A study of a software cache implementation of the openmp memory model for multicore and manycore architectures. In *Euro-Par 2010 - Parallel Processing*, P. D'Ambra, M. Guarracino, and D. Talia, Eds., vol. 6272 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2010, pp. 341–352.
- [48] CHEN, F., HÜGEL, B., AND DRESSLER, F. Realistic simulation and experimental validation of adjacent-channel interference in planning of industrial wireless networks. In *Proceedings of the 8th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks* (New York, NY, USA, 2011), PE-WASUN '11, ACM, pp. 97–104.
- [49] CHEN, G., CHUAH, K., LI, M.-F., CHAN, D., ANG, C., ZHENG, J., JIN, Y., AND KWONG, D. Dynamic nbtI of pmos transistors and its impact on device lifetime. In *Reliability Physics Symposium Proceedings, 2003. 41st Annual. 2003 IEEE International* (March 2003), pp. 196–202.
- [50] CHEN, G., SYLVESTER, D., BLAAUW, D., AND MUDGE, T. Yield-driven near-threshold sram design. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 18, 11 (Nov 2010), 1590–1598.
- [51] CHONG, D., LIU, H., LIM, B., WIN, P., WANG, C., TAN, H., AND SUN, A. Design and development of stacked die technology solutions for memory packages. In *EPTC* (Dec 2005).
- [52] CIVERA, P., MACCHIARULO, L., REBAUDENGO, M., REORDA, M., AND VIOLANTE, M. Exploiting circuit emulation for fast hardness evaluation. *Nuclear Science, IEEE Transactions on* 48, 6 (Dec 2001), 2210–2216.
- [53] CLERECO CONSORTIUM. Cross-layer early reliability evaluation for the computing continuum. [Online]. Available: <http://www.clereco.eu/>, 2013.

- [54] COMPLEX CONSORTIUM. Complex - codesign and power management in platform-based design space exploration. [Online]. Available: <https://complex.offis.de/>, 2009.
- [55] CORPORATION, D. C. G. I. Intel® xeon® processor e7 family: Reliability, availability, and serviceability. Tech. rep., 2011.
- [56] CORPORATION, I. The scc programmer's guide - revision 0.75.
- [57] COSEMANS, S. *Variability-Aware Design of Low Power SRAM Memories*. PhD thesis, 2009.
- [58] COSEMANS, S., DEHAENE, W., AND CATTLOOR, F. A 3.6 pj/access 480 mhz, 128 kb on-chip sram with 850 mhz boost mode in 90 nm cmos with tunable sense amplifiers. *IEEE JSSC* 44, 7 (July 2009), 2065–2077.
- [59] CROSS, J. Power efficiency revolution for embedded computing technologies (perfect). Tech. rep., Defense Advanced Research Projects Agency (DARPA), United States of America, 2013.
- [60] DAWOUD, W., TAKOUNA, I., AND MEINEL, C. Elastic vm for cloud resources provisioning optimization. In *Advances in Computing and Communications*, A. Abraham, J. Lloret Mauri, J. Buford, J. Suzuki, and S. Thampi, Eds., vol. 190 of *Communications in Computer and Information Science*. Springer Berlin Heidelberg, 2011, pp. 431–445.
- [61] DEBENEDICTIS, E. Rebooting supercomputing. Tech. rep., IEEE Rebooting Computing Initiative, April 2015.
- [62] DEHON, A., QUINN, H., AND CARTER, N. Vision for cross-layer optimization to address the dual challenges of energy and reliability. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010* (March 2010), pp. 1017–1022.
- [63] DENG, Y., AND MALY, W. 2.5-dimensional vlsi system integration. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 13, 6 (June 2005), 668–677.
- [64] DESSET, C., MACQ, B., AND VANDENDORPE, L. Computing the word-, symbol-, and bit-error rates for block error-correcting codes. *IEEE Transactions on Communications* 52, 6 (June 2004).
- [65] DESYRE CONSORTIUM. Desyre: on-demand system reliability. [Online]. Available: <http://www.desyre.eu/>, 2010.
- [66] DEVORE, J. L. *Probability and Statistics for Engineering and the Sciences*. Cengage Learning, January 2011.

- [67] DIEFENBAUGH, P., AND BRODOWSKI, D. ACPI Processor P-States Driver, 2006.
- [68] DIXIT, A., HEALD, R., AND WOOD, A. Trends from ten years of soft error experimentation. In *IEEE SELSE Workshop* (2009).
- [69] DOLECEK, L., QAZI, M., SHAH, D., AND CHANDRAKASAN, A. Breaking the simulation barrier: Sram evaluation through norm minimization. In *IEEE/ACM ICCAD* (Nov 2008), pp. 322–329.
- [70] DONELON, J., DOYLE, J., HURST, J., OPRYSKO, M., ROSSNAGEL, S., AND VON GUTFELD, R. Laser methods for circuit repair on integrated circuits and substrates, Jan. 26 1993. US Patent 5,182,230.
- [71] DUNSMOOR, A., AND AO GEADA, J. Applications and use of stage-based ocv. *EE Times* (May 2012).
- [72] DURANTON, M., BOSSCHERE, K. D., COHEN, A., MAEBE, J., AND MUNK, H. Hipec vision 2015. Tech. rep., FP7 HiPEAC Network of Excellence, 2015.
- [73] EIJALI, A., AL-HASHIMI, B., SCHMITZ, M., ROSINGER, P., AND MIREMADI, S. Combined time and information redundancy for seu-tolerance in energy-efficient real-time systems. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 14, 4 (April 2006), 323–335.
- [74] EL-SAYED, N., AND SCHROEDER, B. To checkpoint or not to checkpoint: Understanding energy-performance-i/o tradeoffs in hpc checkpointing. In *IEEE CLUSTER* (Sept 2014), pp. 93–102.
- [75] ERNST, D., KIM, N. S., DAS, S., PANT, S., RAO, R., PHAM, T., ZIESLER, C., BLAAUW, D., AUSTIN, T., FLAUTNER, K., AND MUDGE, T. Razor: a low-power pipeline based on circuit-level timing speculation. In *IEEE/ACM MICRO* (Dec 2003), pp. 7–18.
- [76] ESMAEILZADEH, H., BLEM, E., ST. AMANT, R., SANKARALINGAM, K., AND BURGER, D. Dark silicon and the end of multicore scaling. In *Proceedings of the 38th Annual International Symposium on Computer Architecture* (2011), pp. 365–376.
- [77] EUROCLOUD CONSORTIUM. Energy-conscious 3d server-on-chip for green cloud. [Online]. Available: <http://www.eurocloudserver.com>, 2010.
- [78] EYERMAN, S., EECKHOUT, L., KARKHANIS, T., AND SMITH, J. A top-down approach to architecting cpi component performance counters. *Micro, IEEE* 27, 1 (Jan 2007), 84–93.

- [79] FEI, Y. Microarchitectural support for program code integrity monitoring in application-specific instruction set processors. In *Design, Automation Test in Europe Conference* (April 2007), pp. 1–6.
- [80] FELDBAUMER, D., BABCOCK, J., MERCIER, V., AND CHUN, C. Pulse current trimming of polysilicon resistors. *IEEE TED* 42, 4 (Apr 1995), 689–696.
- [81] FERRI, C., REDA, S., AND BAHAR, R. Strategies for improving the parametric yield and profits of 3d ics. In *IEEE/ACM ICCAD* (Nov 2007), pp. 220–226.
- [82] FRANCO, J., KACZER, B., CHO, M., ENEMAN, G., GROESENENKEN, G., AND GRASSER, T. Improvements of nbtI reliability in sige p-fets. In *IEEE IRPS* (May 2010), pp. 1082–1085.
- [83] FRATERNALI, F., BARTOLINI, A., CAVAZZONI, C., TECCHIOOLI, G., AND BENINI, L. Quantifying the impact of variability on the energy efficiency for a next-generation ultra-green supercomputer. In *ISLPED* (New York, NY, USA, 2014), ACM, pp. 295–298.
- [84] FULLER, S., AND MILLETT, L. Computing performance: Game over or next level? *Computer* 44, 1 (Jan 2011), 31–38.
- [85] GAJSKI, D., AND KUHN, R. Guest editors’ introduction: New vlsi tools. *Computer* 16, 12 (Dec 1983), 11–14.
- [86] GANAPATHY, S., CANAL, R., ALEXANDRESCU, D., COSTENARO, E., GONZALEZ, A., AND RUBIO, A. Informer: An integrated framework for early-stage memory robustness analysis. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014* (March 2014), pp. 1–4.
- [87] GAO, S., HE, B., AND XU, J. Real-time in-memory checkpointing for future hybrid memory systems. In *ICS* (2015), pp. 263–272.
- [88] GAONKAR, S., ROZIER, E., TONG, A., AND SANDERS, W. Scaling file systems to support petascale clusters: A dependability analysis to support informed design choices. In *IEEE DSN* (June 2008), pp. 386–391.
- [89] GARG, S., AND MARCULESCU, D. System-level process variability analysis and mitigation for 3d mpsoCs. In *DATE* (April 2009), pp. 604–609.
- [90] GEIST, A. Development of naturally fault tolerant algorithms for computing on 100,000 processors. Tech. rep., Oak Ridge National Laboratory, 2002.

- [91] GHEORGHITA, S. V., PALKOVIC, M., HAMERS, J., VANDECAPPELLE, A., MAMAGKAKIS, S., BASTEN, T., EECKHOUT, L., CORPORAAL, H., CATTHOOR, F., VANDEPUTTE, F., AND BOSSCHERE, K. D. System-scenario-based design of dynamic embedded systems. *ACM Trans. Des. Autom. Electron. Syst.* 14, 1 (Jan. 2009), 3:1–3:45.
- [92] GONZALEZ, R. C., AND WOODS, R. E. *Digital Image Processing*. Addison-Wesley Publishing Company, September 1993.
- [93] GORDON, M., GOLDHAGEN, P., RODBELL, K., ZABEL, T., TANG, H., CLEM, J., AND BAILEY, P. Measurement of the flux and energy spectrum of cosmic-ray induced neutrons on the ground. *Nuclear Science, IEEE Transactions on* 51, 6 (Dec 2004), 3427–3434.
- [94] GRASSER, T., KACZER, B., GOES, W., REISINGER, H., AICHINGER, T., HEHENBERGER, P., WAGNER, P.-J., SCHANOVSKY, F., FRANCO, J., LUQUE, M., AND NELHIEBEL, M. The paradigm shift in understanding the bias temperature instability: From reaction–diffusion to switching oxide traps. *Electron Devices, IEEE Transactions on* 58, 11 (Nov 2011), 3652–3666.
- [95] GRASSER, T., REISINGER, H., WAGNER, P.-J., SCHANOVSKY, F., GOES, W., AND KACZER, B. The time dependent defect spectroscopy (tds) for the characterization of the bias temperature instability. In *Reliability Physics Symposium (IRPS), 2010 IEEE International* (May 2010), pp. 16–25.
- [96] GREENWELL, R., MCCUE, B., ZUO, L., HUQUE, M., TOLBERT, L., BLALOCK, B., AND ISLAM, S. Soi-based integrated circuits for high-temperature power electronics applications. In *Applied Power Electronics Conference and Exposition (APEC), 2011 Twenty-Sixth Annual IEEE* (March 2011), pp. 836–843.
- [97] GRINSTEAD, C. M., AND SNELL, J. L. *Introduction to Probability*. 2003.
- [98] GUMPERTZ, R. H. Combining tags with error codes. In *Proceedings of the 10th ISCA* (New York, NY, USA, 1983), ACM, pp. 160–165.
- [99] GUPTA, P., AND HENG, F.-L. Toward a systematic-variation aware timing methodology. In *Design Automation Conference, 2004. Proceedings. 41st* (July 2004), pp. 321–326.
- [100] HACKER, T. J., AND MEGLIICKI, Z. Using queue structures to improve job reliability. In *HPDC* (New York, NY, USA, 2007), ACM, pp. 43–54.
- [101] HAMMING, R. W. Error detecting and error correcting codes. *The Bell System Technical Journal* 29, 2 (April 1950), 147–160.

- [102] HARDY, D., SIDERIS, I., LADAS, N., AND SAZEIDES, Y. The performance vulnerability of architectural and non-architectural arrays to permanent faults. In *MICRO* (2012).
- [103] HENKEL, J., BAUER, L., DUTT, N., GUPTA, P., NASSIF, S., SHAFIQUE, M., TAHOORI, M., AND WEHN, N. Reliable on-chip systems in the nano-era: Lessons learnt and future trends. In *Proceedings of the 50th Annual Design Automation Conference* (New York, NY, USA, 2013), DAC '13, ACM, pp. 99:1–99:10.
- [104] HOWARD, J., DIGHE, S., VANGAL, S., RUHL, G., BORKAR, N., JAIN, S., ERRAGUNTLA, V., KONOW, M., RIEPEN, M., GRIES, M., DROEGE, G., LUND-LARSEN, T., STEIBL, S., BORKAR, S., DE, V., AND VAN DER WIJNGAART, R. A 48-core ia-32 processor in 45 nm cmos using on-die message-passing and dvfs for performance and power scaling. *Solid-State Circuits, IEEE Journal of* 46, 1 (Jan 2011), 173–183.
- [105] HSIEH, A.-C., HWANG, T., CHANG, M.-T., TSAI, M.-H., TSENG, C.-M., AND LI, H.-C. Tsv redundancy: Architecture and design issues in 3d ic. In *DATE* (2010), European Design and Automation Association, pp. 166–171.
- [106] HUANG, P., YANG, H., AND THIELE, L. On the scheduling of fault-tolerant mixed-criticality systems. In *Design Automation Conference* (New York, NY, USA, 2014), ACM, pp. 131:1–131:6.
- [107] IBM RESEARCH. Reliability-aware microarchitectures group. [Online]. Available: http://researcher.watson.ibm.com/researcher/view_group.php?id=2260, 2015.
- [108] IEEE. Ieee standard vhdl language reference manual. *IEEE Std 1076-2000* (2000), i –290.
- [109] IEEE. Ieee draft standard for system verilog–unified hardware design, specification, and verification language. *IEEE P1800/D5, February 2012* (22 2012), 1 –1304.
- [110] IEEE. Ieee standard for standard systemc language reference manual. *IEEE Std 1666-2011 (Revision of IEEE Std 1666-2005)* (9 2012), 1 –638.
- [111] iROC TECH. The Standard for Soft Error Prevention – IROC Technologies. [Online]. Available: <http://www.iroctech.com>, 2015.
- [112] JALALI, F., KHODADOUSTAN, S., AND EIJLALI, A. Error control schemes in solar energy harvesting wireless sensor networks. In *Communications and Information Technologies (ISCIT), 2012 International Symposium on* (Oct 2012), pp. 979–984.

- [113] JAN, H., PAUL, A., MINHAS, A., AHMAD, A., JABBAR, S., AND KIM, M. Dependability and reliability analysis of intra cluster routing technique. *Peer-to-Peer Networking and Applications* (2014).
- [114] JANUSZEWSKI, R., GILLY, L., YILMAZ, E., AUWETER, A., AND SVENSSON, G. Cooling – making efficient choices. Tech. rep., Partnership for Advanced Computing in Europe, 2012.
- [115] JEPPSON, K. O., AND SVENSSON, C. M. Negative bias stress of mos devices at high electric fields and degradation of mnos devices. *Journal of Applied Physics* 48, 5 (1977).
- [116] JESS, J., KALAFALA, K., NAIDU, S., OTTEN, R., AND VISWESWARIAH, C. Statistical timing for parametric yield prediction of digital integrated circuits. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 25, 11 (Nov 2006), 2376–2392.
- [117] JOHNSON, N. L., KOTZ, S., AND BALAKRISHNAN, N. *Continuous Univariate Distributions, Volume 2*. May 1995.
- [118] JUANJUAN, L., ZHENQING, Z., AND LEE, J. Wire bonding performance and solder joint reliability investigation on enepig finish substrate. In *ICEPT-HDP* (Aug 2010), pp. 240–245.
- [119] KACZER, B., GRASSER, T., ROUSSEL, P., FRANCO, J., DEGRAEVE, R., RAGNARSSON, L.-A., SIMOEN, E., GROESENENKEN, G., AND REISINGER, H. Origin of nbtj variability in deeply scaled pfts. In *Reliability Physics Symposium (IRPS), 2010 IEEE International* (May 2010), pp. 26–32.
- [120] KACZER, B., MAHATO, S., DE ALMEIDA CAMARGO, V., TOLEDANO-LUQUE, M., ROUSSEL, P., GRASSER, T., CATTHOOR, F., DOBROVOLNY, P., ZUBER, P., WIRTH, G., AND GROESENENKEN, G. Atomistic approach to variability of bias-temperature instability in circuit simulations. In *Reliability Physics Symposium (IRPS), 2011 IEEE International* (April 2011), pp. XT.3.1–XT.3.5.
- [121] KANJ, R., JOSHI, R., AND NASSIF, S. Mixture importance sampling and its application to the analysis of sram designs in the presence of rare failure events. In *ACM/IEEE DAC* (2006), pp. 69–72.
- [122] KARLSSON, J., AND FOLKESSON, P. Application of three physical fault injection techniques to the experimental assessment of the mars architecture. In *IEEE International Working Conference Dependable Computing for Critical Applications* (1995), pp. 150–161.

- [123] KASHYAP, S., DHILLON, J., AND PURINI, S. Rlc - a reliable approach to fast and efficient live migration of virtual machines in the clouds. In *IEEE CLOUD* (June 2014), pp. 360–367.
- [124] KASTENSMIDT, F., STERPONE, L., CARRO, L., AND REORDA, M. On the optimal design of triple modular redundancy logic for sram-based fpgas. In *Design, Automation and Test in Europe, 2005. Proceedings* (March 2005), pp. 1290–1295 Vol. 2.
- [125] KHALIL, D., KHELLAH, M., KIM, N.-S., ISMAIL, Y., KARNIK, T., AND DE, V. {SRAM} dynamic stability estimation using {MPFP} and its applications. *Microelectronics Journal* 40, 11 (2009), 1523 – 1530. International Conference on MicroelectronicsDigital and Mixed-Signal Circuits and Systems.
- [126] KHAN, S., AND HAMDIOUL, S. Modeling and mitigating nbtI in nanoscale circuits. In *IEEE IOLTS* (July 2011), pp. 1–6.
- [127] KHELLAH, M., KHALIL, D., SOMASEKHAR, D., ISMAIL, Y., KARNIK, T., AND DE, V. Effect of power supply noise on sram dynamic stability. In *VLSI Circuits, 2007 IEEE Symposium on* (June 2007), pp. 76–77.
- [128] KHOSROPOUR, A., KASHANI-GHARAVI, S., ASADPOUR, R., AND AFZALI-KUSHA, A. Process variation tolerant sram cell design using additive model considering nbtI effect. In *Quality Electronic Design (ASQED), 2012 4th Asia Symposium on* (July 2012), pp. 46–53.
- [129] KIM, J., HARDAVELLAS, N., MAI, K., FALSAFI, B., AND HOE, J. Multi-bit error tolerant caches using two-dimensional error coding. In *Microarchitecture, 2007. MICRO 2007. 40th Annual IEEE/ACM International Symposium on* (Dec 2007), pp. 197–209.
- [130] KIM, J., WANG, F., AND NOWAK, M. Method and apparatus for providing through silicon via (tsv) redundancy, Nov. 25 2010. US Patent App. 12/468,908.
- [131] KIM, S., DICK, R., AND JOSEPH, R. Power deregulation: Eliminating off-chip voltage regulation circuitry from embedded systems. In *Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2007 5th IEEE/ACM/IFIP International Conference on* (Sept 2007), pp. 105–110.
- [132] KIM, W.-Y., HOON CHOI, J., SHIM, J.-M., AND CHOI, W. Evergreen: A fault-tolerant application streaming technique. In *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on* (Feb 2009), vol. 03, pp. 2302–2307.

- [133] KIM, Y., AND JOHN, L. Automated di/dt stressmark generation for microprocessor power delivery networks. In *Low Power Electronics and Design (ISLPED) 2011 International Symposium on* (Aug 2011), pp. 253–258.
- [134] KIMURA, M. Field and temperature acceleration model for time-dependent dielectric breakdown. *Electron Devices, IEEE Transactions on* 46, 1 (Jan 1999), 220–229.
- [135] KIT. Chair of dependable nano computing (cdnc). [Online]. Available: <http://cdnc.itec.kit.edu/>, 2015.
- [136] KIT. Chair of embedded systems. [Online]. Available: <http://ces.itec.kit.edu>, 2015.
- [137] KIT. Dependable embedded systems. [Online]. Available: <http://spp1500.itec.kit.edu/>, 2015.
- [138] KLOKKARIS, G. Offline and online estimation of performance degradation due to aging faults in caches. master thesis, University of Cyprus, November 2014.
- [139] KONG, J.-T. Cad for nanometer silicon design challenges and success. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 12, 11 (Nov 2004), 1132–1147.
- [140] KONSTANTELI, K., CUCINOTTA, T., PSYCHAS, K., AND VARVARIGOU, T. Admission control for elastic cloud services. In *IEEE CLOUD* (June 2012), pp. 41–48.
- [141] KOREN, I., AND KRISHNA, C. M. *Fault-Tolerant Systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007.
- [142] KRITIKAKOU, A., CATTHOOR, F., KELEFOURAS, V., AND GOUTIS, C. A systematic approach to classify design-time global scheduling techniques. *ACMCSUR* 45, 2 (Mar. 2013).
- [143] KUKNER, H., WECKX, P., FRANCO, J., TOLEDANO-LUQUE, M., CHO, M., KACZER, B., RAGHAVAN, P., JANG, D., MIYAGUCHI, K., BARDON, M., CATTHOOR, F., VAN DER PERRE, L., LAUWEREINS, R., AND GROESENEKEN, G. Scaling of bti reliability in presence of time-zero variability. In *Reliability Physics Symposium, 2014 IEEE International* (June 2014), pp. CA.5.1–CA.5.7.
- [144] KUMAR, A. *Analysis, Design and Management of Multimedia Multiprocessor Systems*. PhD thesis, TU/e, 2009.

- [145] KUMAR, M., DEMSHKI, M., SHIVELEY, R., AND BASSMAN, R. Advanced reliability for intel[®] xeon[®] processors on dell[™] powerededge[™] servers. Tech. rep., May 2010.
- [146] KUMAR, S., KIM, C., AND SAPATNEKAR, S. Impact of nbtI on sram read stability and design for reliability. In *Quality Electronic Design, 2006. ISQED '06. 7th International Symposium on* (March 2006), pp. 6 pp.–218.
- [147] KUMAR, S., KIM, C., AND SAPATNEKAR, S. NbtI-aware synthesis of digital circuits. In *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE* (June 2007), pp. 370–375.
- [148] KUMAR, S., WARN, S., AND LEE, R. Led thermal management of an automotive electronic control module with display. In *Electronics Packaging Technology Conference (EPTC), 2012 IEEE 14th* (Dec 2012), pp. 764–769.
- [149] KUNG, H. T., LUCCIO, F., AND PREPARATA, F. P. On finding the maxima of a set of vectors. *J. ACM* 22, 4 (Oct. 1975), 469–476.
- [150] LAU, J. H. The most cost-effective integrator (tsv interposer) for 3d ic integration system-in-package (sip). In *ASME Pacific Rim Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Systems* (July 2011).
- [151] LEE, J., AND SHRIVASTAVA, A. A compiler optimization to reduce soft errors in register files. In *Proceedings of the 2009 ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems* (New York, NY, USA, 2009), ACM, pp. 41–49.
- [152] LI, X., AND YEUNG, D. Application-level correctness and its impact on fault tolerance. In *HPCA* (Feb 2007), pp. 181–192.
- [153] LICHTENSTEIGER, S., AND BICKFORD, J. Using selective voltage binning to maximize yield. *IEEE TSM* 26, 4 (Nov 2013), 436–441.
- [154] LIN, X., WANG, Y., XIE, Q., AND PEDRAM, M. Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment. *IEEE TCS* 8, 2 (March 2015), 175–186.
- [155] LIOU, J.-J., CHENG, K.-T., KUNDU, S., AND KRSTIC, A. Fast statistical timing analysis by probabilistic event propagation. In *Design Automation Conference, 2001. Proceedings* (2001), pp. 661–666.

- [156] LOEFFLER, C., LIGTENBERG, A., AND MOSCHYTZ, G. S. Practical fast 1-d dct algorithms with 11 multiplications. In *Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on* (May 1989), pp. 988–991 vol.2.
- [157] LOFSTEAD, J., ZHENG, F., LIU, Q., KLASKY, S., OLDFIELD, R., KORDENBROCK, T., SCHWAN, K., AND WOLF, M. Managing variability in the io performance of petascale storage systems. In *SC* (Nov 2010), pp. 1–12.
- [158] LU, K., WANG, X., LI, G., WANG, R., CHI, W., LIU, Y., TANG, H., FENG, H., AND GAO, Y. Iaso: An autonomous fault-tolerant management system for supercomputers. *Front. Comput. Sci.* 8, 3 (June 2014), 378–390.
- [159] LUO, F.-L. *Digital Front-End in Wireless Communications and Broadcasting: Circuits and Signal Processing*. Cambridge Univ. Press, 2011.
- [160] LUO, H., TAO, H., MA, H., AND DAS, S. Data fusion with desired reliability in wireless sensor networks. *IEEE TPDS* 22, 3 (March 2011), 501–513.
- [161] LYRAS, G., RODOPOULOS, D., PAPANIKOLAOU, A., AND SOUDRIS, D. Hypervised transient spice simulations of large netlists & workloads on multi-processor systems. In *Proceedings of the Conference on Design, Automation and Test in Europe* (San Jose, CA, USA, 2013), EDA Consortium, pp. 655–658.
- [162] MA, J., AND TEHRANIPOOR, M. Layout-aware critical path delay test under maximum power supply noise effects. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 30, 12 (Dec 2011), 1923–1934.
- [163] MA, Z., MARCHAL, P., SCARPAZZA, D. P., YANG, P., WONG, C., GOMEZ, J. I., HIMPE, S., YKMAN-COUVREUR, C., AND CATTLOOR, F. *Systematic Methodology for Real-Time Cost-Effective Mapping of Dynamic Concurrent Task-Based Systems on Heterogenous Platforms*. Springer, The Netherlands, 2007.
- [164] MADDEN, L., WU, E., KIM, N., BANIJAMALI, B., ABUGHARBIEH, K., RAMALINGAM, S., AND WU, X. Advancing high performance heterogeneous integration through die stacking. In *ESSCIRC* (Sept 2012), pp. 18–24.
- [165] MANHAEVE, H. Current testing: Dead or alive? In *IEEE ETS* (2013).

- [166] MANN, V., VISHNOI, A., IYER, A., AND BHATTACHARYA, P. Vmpatrol: Dynamic and automated qos for virtual machine migrations. In *Network and service management (cnsm), 2012 8th international conference and 2012 workshop on systems virtualization management (svm)* (Oct 2012), pp. 174–178.
- [167] MAO, S., AND WOLF, T. Hardware support for secure processing in embedded systems. *Computers, IEEE Transactions on* 59, 6 (June 2010), 847–854.
- [168] MARICAU, E., AND GIELEN, G. Computer-aided analog circuit design for reliability in nanometer cmos. *Emerging and Selected Topics in Circuits and Systems, IEEE Journal on* 1, 1 (March 2011), 50–58.
- [169] MARS, J., TANG, L., HUNDT, R., SKADRON, K., AND SOFFA, M. L. Bubble-up: Increasing utilization in modern warehouse scale computers via sensible co-locations. In *MICRO* (2011).
- [170] MARTINEZ, R., GIL, P., MARTIN, G., PEREZ, C., AND SERRANO, J. Experimental validation of high-speed fault-tolerant systems using physical fault injection. In *Dependable Computing for Critical Applications* (Nov 1999), pp. 249–265.
- [171] MATTSON, T., VAN DER WIJNGAART, R., RIEPEN, M., LEHNIG, T., BRETT, P., HAAS, W., KENNEDY, P., HOWARD, J., VANGAL, S., BORKAR, N., RUHL, G., AND DIGHE, S. The 48-core scc processor: the programmer’s view. In *High Performance Computing, Networking, Storage and Analysis (SC), 2010 International Conference for* (Nov 2010), pp. 1–11.
- [172] MAY, M., ALLES, M., AND WEHN, N. A case study in reliability-aware design: A resilient ldpc code decoder. In *Design, Automation and Test in Europe, 2008. DATE '08* (March 2008), pp. 456–461.
- [173] MCCONAGHY, T. E. A. *Variation-Aware Design of Custom Integrated Circuits: A Hands-on Field Guide*. Springer, September 2013.
- [174] MCNAIRY, C., HECKMAN, D., MAYFIELD, J., AND HOYT, S. Error framework for a microprocessor and system, June 27 2013. WO Patent App. PCT/US2011/066,658.
- [175] MEIXNER, A., BAUER, M., AND SORIN, D. Argus: Low-cost, comprehensive error detection in simple cores. In *IEEE/ACM MICRO* (Dec 2007), pp. 210–222.

- [176] MELPIGNANO, D., BENINI, L., FLAMAND, E., JEGO, B., LEPLEY, T., HAUGOU, G., CLERMIDY, F., AND DUTOIT, D. Platform 2012, a many-core computing accelerator for embedded socs: Performance evaluation of visual analytics applications. In *Proceedings of the 49th Annual Design Automation Conference* (New York, NY, USA, 2012), DAC '12, ACM, pp. 1137–1142.
- [177] MENON, N. Operating performance points (opp) library. Tech. rep., Free Electrons – Embedded Linux Experts, 2010.
- [178] MIANO, J. *Compressed image file formats: JPEG, PNG, GIF, XBM, BMP*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999.
- [179] MILLER, J., HECHT, H., AND MORRIS, S. Accounting for soft errors in memory reliability prediction. In *Reliability and Maintainability Symposium, 1989. Proceedings., Annual* (Jan 1989), pp. 197–200.
- [180] MINTARNO, E., SKAF, J., ZHENG, R., VELAMALA, J., CAO, Y., BOYD, S., DUTTON, R., AND MITRA, S. Self-tuning for maximized lifetime energy-efficiency in the presence of circuit aging. *IEEE TCAD* (May 2011).
- [181] MIRANDA, M., ROUSSEL, P., AND BRUSAMARELLO, L. Response characterization of an electronic system under variability effects, Aug. 24 2011. EP Patent App. EP20,100,189,434.
- [182] MITCHELL, J., HENDERSON, D., AHRENS, G., AND VILLAREAL, J. Ibm power platform reliability, availability and serviceability (ras). Tech. rep., IBM Corporation, June 2009.
- [183] MITRA, S., BRELSFORD, K., AND SANDA, P. Cross-layer resilience challenges: Metrics and optimization. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010* (March 2010), pp. 1029–1034.
- [184] MODERN CONSORTIUM. Modern and design of reliable, process variation-aware nanoelectronic devices, circuits and systems. [Online]. Available: <http://www.eniac-modern.org>, 2009.
- [185] MOORE, G. Cramming more components onto integrated circuits. *Proceedings of the IEEE* 86, 1 (Jan 1998), 82–85.
- [186] MORENO, G., NARUMANCHI, S., BENNION, K., WAYE, S., AND DEVOTO, D. Gaining traction: Thermal management and reliability of automotive electric traction-drive systems. *Electrification Magazine, IEEE* 2, 2 (June 2014), 42–49.

- [187] MUKHERJEE, S. *Architecture Design for Soft Errors*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.
- [188] MUKUNDAN, J., HUNTER, H., KIM, K.-H., STUECHELI, J., AND MARTÍNEZ, J. F. Understanding and mitigating refresh overheads in high-density ddr4 dram systems. In *ISCA* (2013), pp. 48–59.
- [189] MUNAGA, S., AND CATTHOOR, F. Method and system for operating in hard real time, Oct. 7 2014. US Patent 8,856,791.
- [190] NAGEL, L. W., AND PEDERSON, D. Spice (simulation program with integrated circuit emphasis). Tech. Rep. UCB/ERL M382, EECS Department, University of California, Berkeley, Apr 1973.
- [191] NAPHADE, T., GOEL, N., NAIR, P., AND MAHAPATRA, S. Investigation of stochastic implementation of reaction diffusion (rd) models for nbtI related interface trap generation. In *Reliability Physics Symposium (IRPS), 2013 IEEE International* (April 2013), pp. XT.5.1–XT.5.11.
- [192] NASEER, R., BOULGHASSOUL, Y., DRAPER, J., DASGUPTA, S., AND WITULSKI, A. Critical charge characterization for soft error rate modeling in 90nm sram. In *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on* (May 2007), pp. 1879–1882.
- [193] NICOLAIDIS, M., ANGHEL, L., ZERGAINOH, N.-E., ZORIAN, Y., KARNIK, T., BOWMAN, K., TSCHANZ, J., LU, S.-L., TOKUNAGA, C., RAYCHOWDHURY, A., KHELLAH, M., KULKARNI, J., DE, V., AND AVRESKY, D. Design for test and reliability in ultimate cmos. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2012* (March 2012), pp. 677–682.
- [194] O’CONNOR, I., COURTOIS, B., CHAKRABARTY, K., AND HAMPTON, M. Heterogeneous systems on chip and systems in package. In *DATE* (April 2007), pp. 1–6.
- [195] OF ELECTRICAL, I., AND (IEEE), E. E. Ieee rebooting computing. <http://rebootingcomputing.ieee.org/>, 2015.
- [196] ORAL, S., WANG, F., DILLOW, D. A., MILLER, R., SHIPMAN, G. M., AND MAXWELL, D. Reducing application runtime variability on jaguar xt5. In *Cray User Group Conference* (May 2010).
- [197] ORSHANSKY, M., AND KEUTZER, K. A general probabilistic framework for worst case timing analysis. In *Design Automation Conference, 2002. Proceedings. 39th* (2002), pp. 556–561.

- [198] OUYANG, X., MARCARELLI, S., AND PANDA, D. Enhancing checkpoint performance with staging io and ssd. In *SNAPI* (May 2010), pp. 13–20.
- [199] PARETO, V. *Manual of Political Economy*. Augustus M. Kelley Publishers, 1971.
- [200] PARK, S., PARK, J., SHIN, D., WANG, Y., XIE, Q., PEDRAM, M., AND CHANG, N. Accurate modeling of the delay and energy overhead of dynamic voltage and frequency scaling in modern microprocessors. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 32, 5 (May 2013), 695–708.
- [201] PAUL, B., KANG, K., KUFLUOGLU, H., ALAM, M., AND ROY, K. Temporal performance degradation under nbtI: Estimation and design for improved reliability of nanoscale circuits. In *Design, Automation and Test in Europe, 2006. DATE '06. Proceedings* (March 2006), vol. 1, pp. 1–6.
- [202] PAVLIDIS, V. F., AND FRIEDMAN, E. G. *Three-dimensional Integrated Circuit Design*. Morgan Kaufmann Inc., San Francisco, CA, USA, 2009.
- [203] PELGROM, M., DUINMAIJER, A. C., AND WELBERS, A. Matching properties of mos transistors. *Solid-State Circuits, IEEE Journal of* 24, 5 (Oct 1989), 1433–1439.
- [204] PHAM, C., ESTRADA, Z., CAO, P., KALBARCZYK, Z., AND IYER, R. Reliability and security monitoring of virtual machines using hardware architectural invariants. In *Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on* (June 2014), pp. 13–24.
- [205] POLIAKOV, P., BLOMME, P., PRET, A., CORBALAN, M., GRONHEID, R., WIAUX, V., VERSLUIJS, J., VERKEST, D., VAN HOUDT, J., AND DEHAENE, W. Spacer-defined euv lithography reducing variability of 12nm nand flash memories. In *IEEE IMW* (May 2012), pp. 1–4.
- [206] POLSTYANKIN, A., AND GRIDCHIN, V. Pulse current trimming of polysilicon piezoresistors in pressure sensor. In *APEIE* (Oct 2012), pp. 15–19.
- [207] POLZE, A., TRÖGER, P., AND SALFNER, F. Timely virtual machine migration for pro-active fault tolerance. In *IEEE ISORCW* (March 2011), pp. 234–243.
- [208] PORTERFIELD, A., FOWLER, R., BHALACHANDRA, S., ROUNTREE, B., DEB, D., AND LEWIS, R. Application runtime variability and power optimization for exascale computers. In *ROSS* (2015).

- [209] POUYAN, P., AMAT, E., BARAJAS, E., AND RUBIO, A. Impact of adaptive proactive reconfiguration technique on vmin and lifetime of sram caches. In *ISQED* (March 2014).
- [210] PRESS, W. H., TEUKOLSKY, S. A., VETTERLING, W. T., AND FLANNERY, B. P. *Numerical Recipes in C - The Art of Scientific Computing*, 2nd edition ed. Cambridge University Press, 1992.
- [211] PUSUKURI, K. K., GUPTA, R., AND BHUYAN, L. N. Thread tranquilizer: Dynamically reducing performance variation. *ACM TACO* 8, 4 (Jan. 2012), 46:1–46:21.
- [212] PYO, S.-S., LEE, C.-H., KIM, G.-H., CHOI, K.-M., JUN, Y.-H., AND KONG, B.-S. 45nm low-power embedded pseudo-sram with ecc-based auto-adjusted self-refresh scheme. In *Circuits and Systems, 2009. ISCAS 2009. IEEE International Symposium on* (May 2009), pp. 2517–2520.
- [213] QUINN, H., BLACK, D., ROBINSON, W., AND BUCHNER, S. Fault simulation and emulation tools to augment radiation-hardness assurance testing. *Nuclear Science, IEEE Transactions on* 60, 3 (June 2013), 2119–2142.
- [214] QURESHI, M., KIM, D. H., KHAN, S., NAIR, P., AND MUTLU, O. Avatar: A variable-retention-time (vrt) aware refresh for dram systems. In *DSN* (2015).
- [215] RAGEL, R., AND PARAMESWARAN, S. Impres: integrated monitoring for processor reliability and security. In *Design Automation Conference, 2006 43rd ACM/IEEE* (July 2006), pp. 502–505.
- [216] RAGHAVAN, A., LUO, Y., CHANDAWALLA, A., PAPAETHYMIU, M., PIPE, K., WENISCH, T., AND MARTIN, M. Designing for responsiveness with computational sprinting. *IEEE Micro* 33, 3 (May 2013), 8–15.
- [217] RAMACHANDRAN, P., ADVE, S., BOSE, P., AND RIVERS, J. Metrics for architecture-level lifetime reliability analysis. In *Performance Analysis of Systems and software, 2008. ISPASS 2008. IEEE International Symposium on* (April 2008), pp. 202–212.
- [218] RANA, M., AND CANAL, R. Reem: Failure/non-failure region estimation method for sram yield analysis. In *Computer Design (ICCD), 2014 32nd IEEE International Conference on* (Oct 2014), pp. 36–41.
- [219] RANA, M., AND CANAL, R. Ssfb: A highly-efficient and scalable simulation reduction technique for sram yield analysis. In *DATE* (2014), pp. 1–6.

- [220] RASMUSSEN, N. Improving rack cooling performance using airflow managementtm blanking panels. Tech. rep., APC, 2008.
- [221] REDDI, V., KANEV, S., KIM, W., CAMPANONI, S., SMITH, M., WEI, G.-Y., AND BROOKS, D. Voltage noise in production processors. *Micro, IEEE* 31, 1 (Jan 2011), 20–28.
- [222] RESEDAHL, T. On chip controller (occ) overview. Tech. rep., 2015.
- [223] ROBINSON, W., ALLES, M., BAPTY, T., BHUVA, B., BLACK, J., BONDS, A., MASSENGILL, L., NEEMA, S., SCHRIMPF, R., AND SCOTT, J. Soft error considerations for multicore microprocessor design. In *Integrated Circuit Design and Technology, 2007. ICICDT '07. IEEE International Conference on* (May 2007), pp. 1–4.
- [224] ROCHE, P., PALAU, J.-M., BRUGUIER, G., TAVERNIER, C., ECOFFET, R., AND GASLOT, J. Determination of key parameters for seu occurrence using 3-d full cell sram simulations. *Nuclear Science, IEEE Transactions on* 46, 6 (Dec 1999), 1354–1362.
- [225] RODOPOULOS, D., CATTHOOR, F., NOLTSIS, M., AND SOUDRIS, D. Classification of prior art on parametric reliability techniques for digital systems. In *IEEE Transactions on Reliability (submitted)* (2015).
- [226] RODOPOULOS, D., CATTHOOR, F., AND SOUDRIS, D. Run time slack creation for processor performance variability. *ACM TODAES (under review)* (2015).
- [227] RODOPOULOS, D., CATTHOOR, F., AND SOUDRIS, D. Tackling performance variability due to ras mechanisms with pid-controlled dvfs. *IEEE Computer Architecture Letters* 14, 2 (July 2015), 156–159.
- [228] RODOPOULOS, D., CORBETTA, S., MASSARI, G., LIBUTTI, S., CATTHOOR, F., SAZEIDES, Y., NICOPOULOS, C., PORTERO, A., CAPPE, E., VAVRIK, R., VONDRÁK, V., SOUDRIS, D., SASSI, F., FRITSCH, A., AND FORNACIARI, W. Harpa: Solutions for dependable performance under physically induced performance variability. In *IEEE SAMOS* (2015).
- [229] RODOPOULOS, D., MAHATO, S., DE ALMEIDA CAMARGO, V., KACZER, B., CATTHOOR, F., COSEMANS, S., GROESENENKEN, G., PAPANIKOLAOU, A., AND SOUDRIS, D. Time and workload dependent device variability in circuit simulations. In *IC Design Technology (ICICDT), 2011 IEEE International Conference on* (May 2011), pp. 1–4.
- [230] RODOPOULOS, D., PAPANIKOLAOU, A., CATTHOOR, F., AND SOUDRIS, D. Demonstrating hw–sw transient error mitigation on the single-chip

- cloud computer data plane. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 23, 3 (March 2015), 507–519.
- [231] RODOPOULOS, D., PSYCHOU, G., SABRY, M. M., CATTHOOR, F., PAPANIKOLAOU, A., SOUDRIS, D., NOLL, T. G., AND ATIENZA, D. Classification framework for analysis and modeling of physically induced reliability violations. *ACMCSUR* 47, 3 (Feb. 2015), 38:1–38:33.
- [232] RODOPOULOS, D., ROUSSEL, P., CATTHOOR, F., SAZEIDES, Y., AND SOUDRIS, D. Approximating standard cell delay distributions using the most probable failure point. In *Workshop on Early Reliability Modeling for Aging and Variability in Silicon Systems Colocated with IEEE/ACM DATE* (2016).
- [233] RODOPOULOS, D., SAZEIDES, Y., CATTHOOR, F., NICOPOULOS, C., AND SOUDRIS, D. Sensitivity of sram cell most probable snm failure point to time-dependent variability. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)* (2015).
- [234] RODOPOULOS, D., WECKX, P., NOLTSIS, M., CATTHOOR, F., AND SOUDRIS, D. Atomistic pseudo-transient bti simulation with inherent workload memory. *Device and Materials Reliability, IEEE Transactions on* 14, 2 (June 2014), 704–714.
- [235] ROMANESCU, B. F., BAUER, M. E., OZEV, S., AND SORIN, D. J. Tech. rep.
- [236] SABRY, M., ATIENZA, D., AND CATTHOOR, F. A hybrid hw-sw approach for intermittent error mitigation in streaming-based embedded systems. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2012* (March 2012), pp. 1110–1113.
- [237] SABRY, M., COSKUN, A., ATIENZA, D., ROSING, T., AND BRUNSCHWILER, T. Energy-efficient multiobjective thermal control for liquid-cooled 3-d stacked architectures. *IEEE TCAD* 30, 12 (Dec 2011), 1883–1896.
- [238] SANZ, C., PRIETO, M., GÓMEZ, J. I., PAPANIKOLAOU, A., MIRANDA, M., AND CATTHOOR, F. Combining system scenarios and configurable memories to tolerate unpredictability. *ACM Trans. Des. Autom. Electron. Syst.* 13, 3 (July 2008), 49:1–49:7.
- [239] SARNO, C., AND TANTOLIN, C. Integration, cooling and packaging issues for aerospace equipments. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010* (March 2010), pp. 1225–1230.

- [240] SCHAD, J., DITTRICH, J., AND QUIANÉ-RUIZ, J.-A. Runtime measurements in the cloud: Observing, analyzing, and reducing variance. *Proc. VLDB Endow.* 3, 1-2 (Sept. 2010), 460–471.
- [241] SCHMITT, M. Power, noise and reliability analysis for automotive electronic systems. In *ASWC* (October 2012).
- [242] SEEVINCK, E., LIST, F., AND LOHSTROH, J. Static-noise margin analysis of mos sram cells. *Solid-State Circuits, IEEE Journal of* 22, 5 (Oct 1987), 748–754.
- [243] SEYAB, AND HAMDIOUI, S. Nbti modeling in the framework of temperature variation. In *DATE* (2010), pp. 283–286.
- [244] SEYFRIED, S. Resource management in linux with control groups. In *Linux-Kongress* (September 2010).
- [245] SHAFIK, R., ROSINGER, P., AND AL-HASHIMI, B. Systemc-based minimum intrusive fault injection technique with improved fault representation. In *On-Line Testing Symposium, 2008. IOLTS '08. 14th IEEE International* (July 2008), pp. 99–104.
- [246] SHIN, Y., AND CHOI, K. Power conscious fixed priority scheduling for hard real-time systems. In *DAC* (1999), pp. 134–139.
- [247] SIEWIOREK, D. P., AND SWARZ, R. S. *Reliable Computer Systems (3rd Ed.): Design and Evaluation*. A. K. Peters, Ltd., Natick, MA, USA, 1998.
- [248] SINGH, I., QIN, I., XU, H., HUYNH, C., LOW, S., CLAUBERG, H., CHYLAK, B., AND ACOFF, V. Pd-coated cu wire bonding technology: Chip design, process optimization, production qualification and reliability test for high reliability semiconductor devices. In *IEEE ECTC* (May 2012).
- [249] SKADRON, K., STAN, M. R., SANKARANARAYANAN, K., HUANG, W., VELUSAMY, S., AND TARJAN, D. Temperature-aware microarchitecture: Modeling and implementation. *ACM Trans. Archit. Code Optim.* 1, 1 (Mar. 2004), 94–125.
- [250] SOCIETY, I. C. Ieee rebooting computing launches initiative to rethink the computer. Tech. rep., Institute of Electrical and Electronics Engineers, 2015.
- [251] SOHI, G. Cache memory organization to enhance the yield of high performance vlsi processors. *Computers, IEEE Transactions on* 38, 4 (Apr 1989), 484–492.

- [252] SONG, J., KIM, S., BAE, B., KIM, H., KIM, J., KIM, D.-H., AND KIM, J. Analysis of wireless power transfer system design on active silicon interposer for low voltage applications in 3d-ic. In *IEEE EPEPS* (Oct 2014), pp. 223–226.
- [253] SOUDRIS, D., PIGUET, C., AND GOUTIS, C., Eds. *Designing CMOS Circuits for Low Power*. Springer, 2002.
- [254] SRINIVASAN, J., ADVE, S., BOSE, P., AND RIVERS, J. The impact of technology scaling on lifetime reliability. In *Dependable Systems and Networks, 2004 International Conference on* (June 2004), pp. 177–186.
- [255] SRIVAREE-RATANA, C., AND SMITH, A. Estimating all-terminal network reliability using a neural network. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on* (Oct 1998), vol. 5, pp. 4734–4739 vol.5.
- [256] STANFORD UNIVERSITY. Robust systems group. [Online]. Available: http://cgi.stanford.edu/~group-rsg_csl/twiki/bin/view.pl/Public/WebHome, 2015.
- [257] SUPERTHEME CONSORTIUM. Circuit stability under process variability and electro-thermal-mechanical coupling. [Online]. Available: <http://www.supertHEME.eu>, 2012.
- [258] SUTHERLAND, B. No moore? – a golden rule of microchips appears to be coming to an end. *The Economist – The World In 2014* (January 2013).
- [259] SWAMINATHAN, V., AND CHAKRABARTY, K. Pruning-based, energy-optimal, deterministic i/o device scheduling for hard real-time systems. *ACM TECS* 4, 1 (Feb. 2005), 141–167.
- [260] SYNOPSYS, INC. Nanotime – transistor-level static timing analysis solution for custom designs. Tech. rep., 2010.
- [261] SYSTEMS, H. I. P. Count zero crossings (matlab), 2012.
- [262] TAKAYA, S., NAGATA, M., SAKAI, A., KARIYA, T., UCHIYAMA, S., KOBAYASHI, H., AND IKEDA, H. A 100gb/s wide i/o with 4096b tsvs through an active silicon interposer with in-place waveform capturing. In *IEEE ISSCC* (Feb 2013), pp. 434–435.
- [263] THIRUVENGADAM, S., TOH, S., AN, H., KOSONOCKY, S., KASPRAK, K., AND MULIG, J. Impact of rtn on vmin elevation of 6-t bitcell designs. In *IEEE SELSE Workshop* (2012).

- [264] THONNART, Y., AND ZID, M. Technology assessment of silicon interposers for manycore socs: Active, passive, or optical? In *IEEE/ACM NoCS* (Sept 2014), pp. 168–169.
- [265] TOLEDANO-LUQUE, M., KACZER, B., ROUSSEL, P., GRASSER, T., WIRTH, G., FRANCO, J., VRANCKEN, C., HORIGUCHI, N., AND GROESENENKEN, G. Response of a single trap to ac negative bias temperature stress. In *IEEE IRPS* (April 2011), pp. 4A.2.1–4A.2.8.
- [266] TONEGAWA, T., HIROI, M., MOTOYAMA, K., FUJII, K., AND MIYAMOTO, H. Suppression of bimodal stress-induced voiding using high-diffusive dopant from cu-alloy seed layer. In *IEEE IITC* (June 2003), pp. 216–218.
- [267] TOTANI, S., AND OKADA, M. Improving reliability of automotive electronics. In *IEEE VTC* (Sept 1980), vol. 30, pp. 80–85.
- [268] TRAMS CONSORTIUM. Terascale reliable adaptive memory system project. [Online]. Available: <https://trams-project.upc.edu/>, 2010.
- [269] TSCHANZ, J., BOWMAN, K., LU, S.-L., ASERON, P., KHELLAH, M., RAYCHOWDHURY, A., GEUSKENS, B., TOKUNAGA, C., WILKERSON, C., KARNIK, T., AND DE, V. A 45nm resilient and adaptive microprocessor core for dynamic variation tolerance. In *IEEE ISSCC* (Feb 2010).
- [270] TU WIEN. Institute for Microelectronics. [Online]. Available: <http://www.iue.tuwien.ac.at/>, 2015.
- [271] UNIVERSITY OF ILLINOIS AT URBANA CHAMPAIGN. RSim Research Group. [Online]. Available: <http://rsim.cs.illinois.edu/>, 2015.
- [272] UNIVERSITY OF MICHIGAN. Michigan Integrated Circuits Laboratory. [Online]. Available: <http://www.eecs.umich.edu/micl/>, 2015.
- [273] VELAZCO, R., KAROUI, S., CHAPUIS, T., BENEZECH, D., AND ROSIER, L. Heavy ion test results for the 68020 microprocessor and the 68882 coprocessor. *Nuclear Science, IEEE Transactions on* 39, 3 (Jun 1992), 436–440.
- [274] VISWESWARIAH, C., RAVINDRAN, K., KALAFALA, K., WALKER, S., NARAYAN, S., BEECE, D., PIAGET, J., VENKATESWARAN, N., AND HEMMETT, J. First-order incremental block-based statistical timing analysis. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 25, 10 (Oct 2006), 2170–2180.
- [275] VON TILS, V. Design requirements for automotive reliability, September 2006.

- [276] WANG, W., YANG, S., BHARDWAJ, S., VRUDHULA, S., LIU, F., AND CAO, Y. The impact of nbtI effect on combinational circuit: Modeling, simulation, and analysis. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 18, 2 (Feb 2010), 173–183.
- [277] WANG, Z. Hierarchical decoding of double error correcting codes for high speed reliable memories. In *ACM/EDAC/IEEE Design Automation Conference (DAC)* (May 2013), pp. 1–7.
- [278] WEAVER, C., EMER, J., MUKHERJEE, S., AND REINHARDT, S. Techniques to reduce the soft error rate of a high-performance microprocessor. In *Computer Architecture, 2004. Proceedings. 31st Annual International Symposium on* (June 2004), pp. 264–275.
- [279] WECKX, P., KACZER, B., TOLEDANO-LUQUE, M., GRASSER, T., ROUSSEL, P., KUKNER, H., RAGHAVAN, P., CATTHOOR, F., AND GROESENENKEN, G. Defect-based methodology for workload-dependent circuit lifetime projections - application to sram. In *Reliability Physics Symposium (IRPS), 2013 IEEE International* (April 2013), pp. 3A.4.1–3A.4.7.
- [280] WECKX, P., KACZER, B., TOLEDANO-LUQUE, M., RAGHAVAN, P., FRANCO, J., ROUSSEL, P., GROESENENKEN, G., AND CATTHOOR, F. Implications of bti-induced time-dependent statistics on yield estimation of digital circuits. *Electron Devices, IEEE Transactions on* 61, 3 (March 2014), 666–673.
- [281] WEI, J., AND PATTABIRAMAN, K. Blockwatch: Leveraging similarity in parallel programs for error detection. In *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on* (June 2012), pp. 1–12.
- [282] WON, J.-Y., CHEN, X., GRATZ, P., HU, J., AND SOTERIOU, V. Up by their bootstraps: Online learning in artificial neural networks for cmp uncore power management. In *IEEE HPCA* (Feb 2014), pp. 308–319.
- [283] WRIGHT, S. J. Coordinate Descent Algorithms. *ArXiv e-prints* (Feb. 2015).
- [284] WU, D., VENKATARAMAN, G., HU, J., LI, Q., AND MAHAPATRA, R. Dicer: distributed and cost-effective redundancy for variation tolerance. In *IEEE/ACM ICCAD* (Nov 2005), pp. 393–397.
- [285] WURM, S., SEIDEL, P., PESKI, C. V., HE, L., HAN, H., KEARNEY, P., AND CHO, W. Euv mask infrastructure challenges. In *EMLC* (Jan 2007), pp. 1–12.

- [286] XU, X., CLINE, B., YERIC, G., YU, B., AND PAN, D. Pd14 self-aligned double patterning aware pin access and standard cell layout co-optimization. *IEEE TCAD* (2015).
- [287] YANG, S., KHURSHEED, S., AL-HASHIMI, B., FLYNN, D., AND IDGUNJI, S. Reliable state retention-based embedded processors through monitoring and recovery. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 30, 12 (Dec 2011), 1773–1785.
- [288] YANG, X., WANG, Z., XUE, J., AND ZHOU, Y. The reliability wall for exascale supercomputing. *Computers, IEEE Transactions on* 61, 6 (June 2012), 767–779.
- [289] YE, Y., LIU, F., CHEN, M., NASSIF, S., AND CAO, Y. Statistical modeling and simulation of threshold variation under random dopant fluctuations and line-edge roughness. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on* 19, 6 (June 2011), 987–996.
- [290] YU, Q., AND AMPADU, P. Transient and permanent link errors co-management. In *Transient and Permanent Error Control for Networks-on-Chip*. Springer New York, 2012, pp. 65–79.
- [291] ZANINI, F., ATIENZA, D., JONES, C. N., BENINI, L., AND DE MICHELI, G. Online thermal control methods for multiprocessor systems. *ACM TODAES* 18, 1 (Jan. 2013), 6:1–6:26.
- [292] ZHANG, J., PATIL, N., HAZEGHI, A., AND MITRA, S. Carbon nanotube circuits in the presence of carbon nanotube density variations. In *Design Automation Conference, 2009. DAC '09. 46th ACM/IEEE* (July 2009), pp. 71–76.
- [293] ZHANG, W., AND LI, T. Managing multi-core soft-error reliability through utility-driven cross domain optimization. In *Application-Specific Systems, Architectures and Processors, 2008. ASAP 2008. International Conference on* (July 2008), pp. 132–137.
- [294] ZHAO, C., BAI, X., AND DEY, S. Evaluating transient error effects in digital nanometer circuits. *Reliability, IEEE Transactions on* 56, 3 (Sept 2007), 381–391.

Curriculum Vitae

Dimitrios Rodopoulos was born in 1989 in Athens, Greece. He graduated from the School of Electrical and Computer Engineering (ECE) of the National Technical University of Athens (NTUA) in 2012. Until 2015, he served as a PhD Researcher at MicroLab-ECE-NTUA and as a Research Associate for the Institute of Communication and Computer Systems (ICCS) of ECE-NTUA. He is a Graduate Student Member of the IEEE and the ACM and a member of the Technical Chamber of Greece. Dimitrios has co-authored more than 20 papers published in international peer reviewed conferences, journals and workshops, as well as three patent applications. At ICCS, Dimitrios served as Principle Investigator on the Intel Single-Chip Cloud Computer and as a Communications Contact for the ICCS membership to the Open Power Foundation. During his PhD studies he designed and supervised more than 10 NTUA Thesis projects and has been involved in projects FP7-612029-HARPA, FP7-248789-TRAMS, and FP7-245244-MOSART of the European Commission as well as the Erasmus Brain Project. In 2015 he was invited in expert capacity by the European Commission to attend, report and present his work in the International Nanotechnology Conference (INC) held in Fukuoka, Japan. Since February 2016, he is serving as RnD Engineer at imec, Belgium in the Process Technology unit. His research interests include: circuit variability and aging for deca-nanometer nodes, functional and parametric processor reliability and optimization of olivocerebellar neuron modeling.

List of Publications

Journals

- [JP-1] **D. Rodopoulos**, M. Noltsis, N. Zompakis, F. Catthoor, and D. Soudris. Run Time Slack Creation for Processor Performance Variability with System Scenarios. *ACM Transactions on Design Automation of Electronic Systems*, (submitted) 2016.
- [JP-2] **D. Rodopoulos**, F. Catthoor, M. Noltsis and D. Soudris. Classification of Prior Art on Parametric Reliability Techniques for Digital Systems. *IEEE Transactions on Reliability*, (submitted) 2015.
- [JP-3] D. Stamoulis, K. Tsoumanis, **D. Rodopoulos**, B. H. Meyer, K. Pekmestzi, D. Soudris, and Z. Zilic. Efficient Variability Analysis of Arithmetic Units using Linear Regression Techniques. *Analog Integrated Circuits and Signal Processing*, 87(2):249–261, 2016.
- [JP-4] **D. Rodopoulos**, Francky Catthoor, and Dimitrios Soudris. Tackling performance variability due to ras mechanisms with pid-controlled dvfs. *IEEE Comput. Archit. Lett.*, 14(2):156–159, July 2015.
- [JP-5] **D. Rodopoulos**, A. Papanikolaou, F. Catthoor, and D. Soudris. Demonstrating hw–sw transient error mitigation on the single-chip cloud computer data plane. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 23(3):507–519, March 2015.
- [JP-6] **Rodopoulos, D.**, Psychou, G., Sabry, M. M., Catthoor, F., Papanikolaou, A., Soudris, D., Noll, T. G., and Atienza, D. Classification Framework for Analysis and Modeling of Physically Induced Reliability Violations. *ACMCSUR* 47, 3 (Feb. 2015), 38:1–38:33.
- [JP-7] **Rodopoulos, D.**, P. Weckx, M. Noltsis, F. Catthoor, and D. Soudris. Atomistic Pseudo-Transient BTI Simulation With Inherent Workload

Memory. *Device and Materials Reliability, IEEE Transactions on*, 14(2):704–714, June 2014.

- [JP-8] K. Siozios, **Rodopoulos, D.**, and D. Soudris. On Supporting Rapid Thermal Analysis. *Computer Architecture Letters*, 10(2):53–56, 2011.

Patents

- [P-1] Francky Catthoor, **Rodopoulos, Dimitrios**, and Dimitrios Soudris. Method of Managing the Operation of an Electronic System with a Guaranteed Lifetime. United States Provisional Patent Application; Applicant: IMEC 3001, Leuven, Belgium US 14/975142, United States Patent and Trademark Office, (submitted) 2016.
- [P-2] Pieter Weckx, **Rodopoulos, Dimitrios**, Ben Kaczer, and Francky Catthoor. Complexity-Reduced Simulation of Circuit Reliability. United States Provisional Patent Application; Applicant: IMEC 3001, Leuven, Belgium US 62/139267, United States Patent and Trademark Office, (submitted) 2015.
- [P-3] Francky Catthoor, **Rodopoulos, Dimitrios**, and Dimitrios Soudris. Method of Managing the Operation of an Electronic System with a Guaranteed Lifetime. European Patent Application; Applicant: IMEC 3001, Leuven, Belgium EP14198876.6, European Patent Office, (submitted) 2014.
- [P-4] Francky Catthoor, Ben Kaczer, **Rodopoulos, Dimitrios**, Vinicius Valduga de Almeida Camargo, and Swaraj Bandhu Mahato. Time and Workload Dependent Circuit Simulations. European Patent Application; Applicant: IMEC 3001, Leuven, Belgium EP 2 509 011 A1, European Patent Office, (published) 2012.

Conferences

- [CP-1] Luca Matti, Vassilios Gerousis, Praveen Raghavan, Peter Debacker, **Dimitrios Rodopoulos**, and Mladen Berekovic EDA Role in the Design-Technology-Co-Optimization Towards n7 (and n5). In *Cadence User Conference (to appear)*, May, 2016.
- [CP-2] Georgios Chatzikostantis, **Dimitrios Rodopoulos**, Sofia Nomikou, Christos Strydis, Chris I. De Zeeuw and Strydis, and Dimitrios Soudris

First Impressions from Detailed Brain Model Simulations on a Xeon-Xeon Phi Node. In *ACM Computing Frontiers (accepted)*, 2016.

- [CP-3] Georgios Smaragdos, Georgios Chatzikostantis, Sofia Nomikou, **Dimitrios Rodopoulos**, Ioannis Sourdis, Dimitrios Soudris, Chris I. De Zeeuw and Strydis, and Christos Strydis Performance Analysis of Accelerated Biophysically-Meaningful Neuron Simulations. In *IEEE ISPASS*, 2016.
- [CP-4] Apostolis Kokolis, Alexandros Mavrogiannis, **Dimitrios Rodopoulos**, Christos Strydis, and Dimitrios Soudris Runtime Interval Optimization and Dependable Performance for Application-Level Checkpointing. In *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2016.
- [CP-5] **Dimitrios Rodopoulos** et al. HARPA: Solutions for Dependable Performance under Physically Induced Performance Variability. In *IEEE SAMOS*, 2015.
- [CP-6] Dimitrios Stamoulis, **Dimitrios Rodopoulos**, Brett H. Meyer, Dimitrios Soudris, Francky Catthoor, and Zeljko Zilic. Efficient Reliability Analysis of Processor Datapath using Atomistic BTI Variability Models. In *GLSVLSI*, 2015.
- [CP-7] Dimitrios Stamoulis, **Dimitrios Rodopoulos**, Brett H. Meyer, Dimitrios Soudris, and Zeljko Zilic. Linear Regression Techniques for Efficient Analysis of Transistor Variability. In *IEEE International Conference on Electronics, Circuits and Systems*, 2014.
- [CP-8] **Dimitrios Rodopoulos**, Giorgos Chatzikonstantis, Andreas Pantelopoulou, Dimitrios Soudris, Chris I De Zeeuw, and Christos Strydis. Optimal mapping of inferior olive neuron simulations on the Single-Chip Cloud Computer. In *International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIV)*, pages 367–374, July 2014.
- [CP-9] **D. Rodopoulos**, D. Stamoulis, G. Lyras, D. Soudris, and F. Catthoor. Understanding timing impact of BTI/RTN with massively threaded atomistic transient simulations. In *IC Design Technology (ICICDT), 2014 IEEE International Conference on*, May 2014.
- [CP-10] Grigorios Lyras, **Dimitrios Rodopoulos**, Antonis Papanikolaou, and Dimitrios Soudris. Hypervised Transient SPICE Simulations of Large Netlists and Workloads on Multi-Processor Systems. In *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2013.
- [CP-11] **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 *IC Design Technology (ICICDT), 2011 IEEE International Conference on*, 2011.

Workshops

- [WP-1] Simone Corbetta, Wim Meeus, **Dimitrios Rodopoulos**, Etienne Cappe, Francky Catthoor, and Agnès Fritsch. Approximating Standard Cell Delay Distributions using the Most Probable Failure Point. In *Silicon Errors in Logic - System Effects (SELSE) Workshop*, 2016.
- [WP-2] **Dimitrios Rodopoulos**, Philippe Roussel, Francky Catthoor, Yannakis Sazeides, and Dimitrios Soudris. Approximating Standard Cell Delay Distributions using the Most Probable Failure Point. In *Workshop on Early Reliability Modeling for Aging and Variability in Silicon Systems Colocated with IEEE/ACM DATE (accepted)*, 2016.
- [WP-3] **Dimitrios Rodopoulos**, Yanos Sazeides, Francky Catthoor, Chrysostomos Nicopoulos, and Dimitrios Soudris. Software Mitigation of Transient Errors on the Single-Chip Cloud Computer. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*, 2015.
- [WP-4] **Dimitrios Rodopoulos**, Antonis Papanikolaou, Francky Catthoor, and Dimitrios Soudris. Software Mitigation of Transient Errors on the Single-Chip Cloud Computer. In *IEEE Workshop on Silicon Errors in Logic - System Effects (SELSE)*, 2012.
- [WP-5] Kostas Siozios, **Dimitris Rodopoulos**, and Dimitrios Soudris. Quick_Hotspot: A Software Supported Methodology for Supporting Run-Time Thermal Analysis at MPSoC Designs. In *2nd PARMA Workshop on Parallel Programming and Run-Time Management Techniques for Many-core Architectures*, 2011.

NATIONAL TECHNICAL UNIVERSITY OF ATHENS
SCHOOL OF ELECTRICAL & COMPUTER ENGINEERING
MICROPROCESSORS & DIGITAL SYSTEMS LABORATORY
9 Heroon Polytechniou
GR-15780 Athens



FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING - ESAT
10 Kasteelpark Arenberg
B-3001 Leuven

