National Technical University of Athens
School of Electrical & Computer Engineering
Division of Communication, Electronic and
Information Engineering
Microprocessors and Digital Systems Lab

Design Automation and Synthesis Techniques for Approximate Computing

Ph.D. Thesis
of
Georgios D. Zervakis

Supervisor: **Prof. Kiamal Pekmestzi**

Submitted in School of Electrical & Computer Engineering of
National Technical University of Athens

Athens, November 2018

National Technical University of Athens
School of Electrical & Computer Engineering
Division of Communication, Electronic and Information Engineering
Microprocessors and Digital Systems Lab

Design Automation and Synthesis Techniques for Approximate Computing

Ph.D. Thesis
of
Georgios D. Zervakis

Supervising Committee: Kiamal Pekmestzi
Dimitrios Soudris
Michael Huebner

Approved by the advisory committee on November 23, 2018.

.....	.....	.....
Kiamal Pekmestzi	Dimitrios Soudris	Michael Huebner
Professor N.T.U.A	Professor N.T.U.A	Professor B.T.U

.....	.....
Nectarios Koziris	Dionisios Pnevmatikatos
Professor N.T.U.A	Professor T.U.C

.....	.....
Dimitris Gizopoulos	Muhammad Shafique
Professor U.O.A	Professor T.U.W

Athens, November 2018

This Ph.D. Thesis was cofinanced by the E.C. funded projects H2020 AEGLE, H2020 FabSpace 2.0, FP-7 Swan i-Care. Also, part of this Ph.D. Thesis was partially supported by Hellenic Funds and by the European Regional Development Fund (ERDF) under the Hellenic National Strategic Reference Framework (NSRF) 2007-2013, for the project "THALES-HOLISTIC".

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
& Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής και
Συστημάτων Πληροφορικής
Εργατήριο Μικροϋπολογιστών

Αυτοματοποιημένες Μεθοδολογίες και Τεχνικές Σχεδίασης και Σύνθεσης Προσεγγιστικού Υπολογισμού για Ενεργειακά Αποδοτικούς Επεξεργαστές Υλικού

Διδακτορική Διατριβή
του
Γεώργιου Δ. Ζερβάκη

Επιβλέπων: **Καθ. Κιαμάλ Πεκμεστζή**

Υποβλήθηκε στη Σχολή Ηλεκτρολόγων Μηχανικών
& Μηχανικών Υπολογιστών
του Εθνικού Μετσόβιου Πολυτεχνείου

Αθήνα, Νοέμβριος 2018

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
& Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής και
Συστημάτων Πληροφορικής
Εργατήριο Μικροϋπολογιστών

Αυτοματοποιημένες Μεθοδολογίες και Τεχνικές Σχεδίασης και Σύνθεσης Προσεγγιστικού Υπολογισμού για Ενεργειακά Αποδοτικούς Επεξεργαστές Υλικού

Διδακτορική Διατριβή
του
Γεώργιου Ζερβάκη

Συμβουλευτική Επιτροπή: Κιαμάλ Πεκμεστζή
Δημήτριος Σούντρης
Michael Huebner

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 23η Νοεμβρίου 2018.

.....	.....	.....
Κιαμάλ Πεκμεστζή	Δημήτριος Σούντρης	Michael Huebner
Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ	Professor B.T.U

.....	.....
Νεκτάριος Κοζύρης	Διονύσιος Πνευματικάτος
Καθηγητής ΕΜΠ	Καθηγητής Π.Κ.

.....	.....
Δημήτριος Γκιζόπουλος	Muhammad Shafique
Καθηγητής ΕΚΠΑ	Professor T.U.W

Αθήνα, Νοέμβριος 2018

.....
Γεώργιος Δ. Ζερβάκης
Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Abstract

Since the failure of Dennard scaling, energy efficiency has become a first-class design concern in computer systems. Recently, exploiting the intrinsic error resilience of a large number of application domains, approximate computing has emerged as a design alternative for energy efficient system design, trading accuracy for significant energy gains. In this thesis, we focus on the design of hardware approximate accelerators. Existing hardware approximation techniques mainly apply single-level approximation, limiting thus the potential energy-savings of the approximate computing application. In this dissertation, in order to max out the benefits of approximate computing application, we examine, introduce, and enable multi-level approximation in the design of hardware arithmetic circuits as well as hardware accelerators. In order to enable straightforward and seamless application of approximate computing and multi-level approximation on hardware circuits, we propose four automated frameworks, i.e., VOSsim, Partial Product Perforation framework, HAM, and METHADONE. All the proposed frameworks were extensively experimentally evaluated. We demonstrated their efficiency and optimality through comparisons against exhaustive design space exploration and related state-of-the-art works. Specifically, we showed that all the approximate designs produced by the proposed frameworks, outperform existing state-of-the-art designs in terms of both induced error as well as power reduction.

Keywords: Approximate Computing, Approximate Synthesis, Arithmetic Circuits, Design Automation, Hardware Accelerators, Multi-Level Approximation, Voltage Over-Scaling

Περίληψη

Οι τεχνολογίες πληροφορικής βρίσκονται πλέον σε μια εποχή που για να διατηρηθεί αλλά και να βελτιωθεί η απόδοση και αποτελεσματικότητα των υπολογιστικών συστημάτων είναι αναγκαίες ριζικές αλλαγές συγκριτικά με τις παραδοσιακές τεχνικές. Ο προσεγγιστικός υπολογισμός αποτελεί ένα χαρακτηριστικό πρότυπο ριζικής μεταβολής που έχει εισέλθει στο σχεδιασμό και τη λειτουργία των σύγχρονων συστημάτων. Η διατριβή αυτή επικεντρώνεται στην μελέτη και σχεδίαση προσεγγιστικών επιταχυντών υλικού. Οι υπάρχουσες μεθοδολογίες σχεδίασης προσεγγιστικών κυκλωμάτων εφαρμόζουν κατά κύριο λόγο μονο-επίπεδες προσεγγιστικές τεχνικές, περιορίζοντας έτσι σημαντικά τα ενεργειακά οφέλη από την εφαρμογή του προσεγγιστικού υπολογισμού. Ο στόχος αυτής της διατριβής είναι να μεγιστοποιήσει τα ενεργειακά οφέλη των προσεγγιστικών κυκλωμάτων και να προτείνει συστηματικές μεθοδολογίες για την εφαρμογή του προσεγγιστικού υπολογισμού, έτσι ώστε να επιτρέψει την αξιοποίησή του στον τομέα της σχεδίασης ψηφιακών συστημάτων. Για να μεγιστοποιήσουμε την απόδοση των προσεγγιστικών κυκλωμάτων μελετάμε, προτείνουμε, και καταστούμε δυνατή την εφαρμογή πολυ-επίπεδων προσεγγιστικών τεχνικών για την παραγωγή τόσο προσεγγιστικών αριθμητικών κυκλωμάτων όσο και επιταχυντών υλικού. Τα ανωτέρω επιτυγχάνονται μέσα από τα τέσσερα αυτοματοποιημένα πλαίσια σχεδίασης και σύνθεσης που προτείνουμε: VOSsim, Partial Product Perforation, HAM, και METHADONE. Τα πλαίσια αυτά αξιολογήθηκαν διεξοδικά μέσω εκτενούς πειραματικής διαδικασίας και συγκρίθηκαν με τις εκάστοτε βέλτιστες υπάρχουσες μεθοδολογίες και τεχνικές αποδεικνύοντας την αποδοτικότητά τους.

Λέξεις Κλειδιά: Αριθμητικά Κυκλώματα, Αυτοματοποιημένη Σχεδίαση Κυκλωμάτων, Επεξεργαστής Υλικού, Πολυ-επίπεδος Προσεγγισμός, Προσεγγιστικός Υπολογισμός, Σύνθεση Προσεγγιστικών Κυκλωμάτων, Υπερκλιμάκωση της Τάσης

Extended Abstract

Since the failure of Dennard scaling, energy efficiency has become a first-class design concern in computer systems. Its potential benefits go beyond reduced power demands in servers and longer battery life in mobile devices, since improving energy efficiency has become a requirement due to limits of device scaling and the well-known "dark silicon" or "power wall" problem. Recently, exploiting the intrinsic error resilience of a large number of application domains, approximate computing has emerged as a design alternative for energy efficient system design, trading accuracy for significant energy gains. In this thesis, we focus on the design of hardware approximate accelerators. Approximate hardware circuits, on the contrary to software approximations, offer also transistors reduction, lower dynamic and leakage power, lower circuit delay and opportunity for down-sizing. Existing hardware approximation techniques mainly apply single-level approximation, limiting thus the potential energy-savings of the approximate computing application. Moreover, the increased requirements for verifying the circuit's functionality, as well as operating within the error bounds, greatly increase the design time cycle of approximate hardware accelerators. In this dissertation, in order to max out the benefits of approximate computing application, we examine, introduce, and enable multi-level approximation in the design of hardware arithmetic circuits as well as hardware accelerators. Multi-level approximation refers to applying an approximation technique in every design layer, i.e., the algorithmic, the logic, and the physical ones. However, multi-level approximate architectures exacerbate the design complexity due to the diversity of inexact techniques and their impact on final circuit implementations. In order to enable straightforward and seamless application of approximate computing and multi-level approximation on hardware circuits, we propose four automated frameworks, i.e., VOSsim, Partial Product Perforation framework, HAM, and METHADONE.

VOSsim enables very fast and accurate quantification of the power-error characteristics of approximate circuits under voltage over-scaling. Partial Product Perforation is a generalized technique that can be, out-of-the-box, applied to any multiplier circuit, providing known a priori and bound output error values. HAM exploits Partial Product Perforation and introduces multi-level approximation in the design of approximate multipliers, showing that multi-level approximation, compared to single-level one, delivers more efficient solutions in terms of both power and error. Finally, we present METHADONE an approximate accelerator synthesis framework which enables efficient inexact circuits implementations by leveraging the incorporation of diverse multi-level approximate techniques. METHADONE incorporates all the aforementioned frameworks and given the behavioral description of a hardware accelerator and an error bound, quickly produces its power-optimal multi-level approximate counterpart that satisfies the error bound. METHADONE can be applied to any accelerator circuit and seamlessly extends typical behavioral and/or RTL synthesis tools by operating on the accelerator's scheduled data flow graph. The approximate accelerators produced by METHADONE deliver energy savings that range from 10% (for 1% error bound) to 70% (for 10% error bound). Compared to an exhaustive design space exploration METHADONE produces close to Pareto-optimal multi-level approximate accelerators, while delivering more than 589× speedup in finding the Pareto-front designs. All the proposed frameworks were extensively experimentally evaluated. We demonstrated their efficiency and optimality through comparisons against exhaustive design space exploration and related state-of-the-art works. Specifically, we showed that all the approximate designs produced by the proposed frameworks, outperform existing state-of-the-art designs in terms of both induced error as well as power reduction.

Keywords: Approximate Computing, Approximate Synthesis, Arithmetic Circuits, Design Automation, Hardware Accelerators, Multi-Level Approximation, Voltage Over-Scaling

Εκτεταμένη Περίληψη

Οι τεχνολογίες πληροφορικής βρίσκονται πλέον σε μια εποχή που για να διατηρηθεί αλλά και να βελτιωθεί η απόδοση και αποτελεσματικότητα των υπολογιστικών συστημάτων είναι αναγκαίες ριζικές αλλαγές συγκριτικά με τις παραδοσιακές τεχνικές. Ο προσεγγιστικός υπολογισμός αποτελεί ένα χαρακτηριστικό πρότυπο ριζικής μεταβολής που έχει εισέλθει στο σχεδιασμό και τη λειτουργία των σύγχρονων συστημάτων. Βασίζεται στην ιδέα ότι περιορίζουμε την απόδοση των υπολογιστικών συστημάτων εξαναγκάζοντάς τα να επιτελούν υπολογισμούς μεγαλύτερης ακρίβειας απ' όσο πραγματικά χρειάζεται. Είναι γεγονός ότι ακριβείς απαντήσεις δεν είναι εφικτές/απαραίτητες σε πληθώρα εφαρμογών, όπως για παράδειγμα στους τομείς βαθιάς μάθησης, όρασης υπολογιστών, μηχανικής μάθησης κ.λ.π. Όμως, ο προσεγγιστικός υπολογισμός πρέπει να εφαρμόζεται μέσω μιας αυστηρής και τυπικής μεθοδολογίας. Για το λόγο αυτό είναι αναγκαία υπολογιστικά επίπεδα αφαίρεσης που να επιτρέπουν την μεθοδική μείωση της υπολογιστικής ακρίβειας κερδίζοντας σε άλλες μετρικές όπως επίδοση και ενέργεια.

Η διατριβή αυτή επικεντρώνεται στην μελέτη και σχεδίαση προσεγγιστικών επιταχυντών υλικού. Τα προσεγγιστικά κυκλώματα προσφέρουν χαμηλότερη δυναμική αλλά και στατική κατανάλωση ισχύος, μικρότερη κυκλωματική καθυστέρηση, καθώς επίσης και δυνατότητα σμίκρυνσης. Οι υπάρχουσες μεθοδολογίες σχεδίασης προσεγγιστικών κυκλωμάτων εφαρμόζουν κατά κύριο λόγο μονο-επίπεδες προσεγγιστικές τεχνικές, περιορίζοντας έτσι σημαντικά τα πιθανά ενεργειακά οφέλη από την εφαρμογή του προσεγγιστικού υπολογισμού. Επιπροσθέτως, οι υπάρχουσες μεθοδολογίες εφαρμόζουν, ως επί το πλείστον, προσεγγιστικές τεχνικές ειδικευμένες για την εκάστοτε εφαρμογή και με μη αυτοματοποιημένο τρόπο. Η σχεδίαση προσεγγιστικών κυκλωμάτων και οι εφαρμογο-κεντρικές μεθοδολογίες που ακολουθούνται δυσχεραίνουν ακόμα περισσότερο το ήδη εξαιρετικά πολύπλοκο έργο της σχεδίασης ψηφιακών

συστημάτων, καθώς ο σχεδιαστής πρέπει να ελέγχει τόσο τη λειτουργία και τη βελτιστοποίηση του κυκλώματος όσο και να τηρεί τα όρια του σφάλματος. Ο στόχος αυτής της διατριβής είναι αφενός να μεγιστοποιήσει τα ενεργειακά οφέλη των προσεγγιστικών κυκλωμάτων και αφετέρου να προτείνει συστηματικές μεθοδολογίες για την εφαρμογή του προσεγγιστικού υπολογισμού, έτσι ώστε να επιτρέψει την αξιοποίηση αλλά και εγκαθίδρυσή του στον τομέα της σχεδίασης ψηφιακών συστημάτων. Για να μεγιστοποιήσουμε την απόδοση των προσεγγιστικών κυκλωμάτων μελετάμε, προτείνουμε, και καταστούμε δυνατή την εφαρμογή πολυ-επίπεδων προσεγγιστικών τεχνικών για την παραγωγή τόσο προσεγγιστικών αριθμητικών κυκλωμάτων όσο και επιταχυντών υλικού. Με τον όρο πολυ-επίπεδες προσεγγιστικές τεχνικές αναφερόμαστε στην εφαρμογή μιας προσεγγιστικής τεχνικής σε κάθε ένα από τα τρία επίπεδα σχεδιασμού, δηλαδή το αλγοριθμικό, το λογικό, και το κυκλωματικό. Για να αντιμετωπίσουμε την αυξημένη σχεδιαστική δυσκολία των προσεγγιστικών κυκλωμάτων, που δυσχεραίνεται ακόμα περισσότερο στον χώρο των πολυ-επίπεδων προσεγγιστικών επιταχυντών υλικού, προτείνουμε γενικευμένες και αυτοματοποιημένες μεθοδολογίες για τη σχεδίαση και σύνθεση αυτών. Τα ανωτέρω επιτυγχάνονται μέσα από τα τέσσερα αυτοματοποιημένα πλαίσια σχεδίασης και σύνθεσης που προτάθηκαν και υλοποιήθηκαν κατά την εκπόνηση της διατριβής: VOSsim, Partial Product Perforation, HAM, και METHADONE.

Το VOSsim είναι το πρώτο και μοναδικό εργαλείο προσομοίωσης κυκλωμάτων σε υψηλό επίπεδο (επίπεδο πυλών) που λαμβάνει υπόψη του την τάση λειτουργίας και παρέχει τη δυνατότητα αλλαγής της τιμής της. Το VOSsim, λοιπόν, επιτρέπει να προσδιοριστούν πολύ γρήγορα και με μεγάλη ακρίβεια τα χαρακτηριστικά (κατανάλωση, σφάλμα) των κυκλωμάτων υπερ-κλιμακούμενης τάσης. Το Partial Product Perforation είναι ένα αυτοματοποιημένο πλαίσιο παραγωγής προσεγγιστικών πολλαπλασιαστών που εισάγει την εφαρμογή προσεγγισμών στην παραγωγή των μερικών γινομένων. Μπορεί να εφαρμοστεί σε κάθε κύκλωμα πολλαπλασιαστή ανεξάρτητα από την αρχιτεκτονική του και παρέχει εκ των προτέρων γνώση του παραγόμενου σφάλματος. Στη συνέχεια προτείνεται το πλαίσιο HAM, το οποίο χρησιμοποιεί το Product Perforation και εφαρμόζει για πρώτη φορά πολυ-επίπεδες προσεγγιστικές τεχνικές στη σχεδίαση προσεγγιστικών αριθμητικών κυκλωμάτων. Με το HAM, δείχνουμε επίσης και την πολύ υψηλή ενεργειακή αποδοτικότητα του πολυ-επίπεδου προσεγγι-

σμού συγκριτικά με τον παραδοσιακά εφαρμοζόμενο μονο-επίπεδο. Τέλος, προτείνουμε το πλαίσιο METHADONE, ένα αυτοματοποιημένο πλαίσιο σύνθεσης προσεγγιστικών επιταχυντών υλικού το οποίο καθιστά δυνατή την αποδοτική εφαρμογή πολυ-επίπεδου προσεγγισμού στο σχεδιασμό πολύπλοκων κυκλωμάτων. Το METHADONE το επιτυγχάνει αυτό ενσωματώνοντας όλα τα προαναφερθέντα προτεινόμενα πλαίσια. Δεδομένης της συμπεριφορικής περιγραφής ενός επεξεργαστή υλικού (ή αλλιώς επιταχυντή) και ενός ορίου σφάλματος, το METHADONE παράγει πολύ γρήγορα τον αντίστοιχο ενεργειακά-βέλτιστο πολυ-επίπεδο επεξεργαστή υλικού που ικανοποιεί το όριο αυτό. Το METHADONE λειτουργεί πάνω στον γράφο ροής δεδομένων του επιταχυντή και έτσι α) μπορεί να εφαρμοστεί σε κάθε κύκλωμα/επιταχυντή υλικού και β) είναι ικανό να επεκτείνει και να ενσωματωθεί αδιαφανώς στις τυπικά χρησιμοποιούμενες διαδικασίες σύνθεσης και σχεδίασης ψηφιακών συστημάτων. Τα κυκλώματα που παράγονται από το METHADONE χαρακτηρίζονται από πολύ υψηλά ενεργειακά οφέλη. Συγκριτικά με το ορθό κύκλωμα, χαρακτηρίζονται από 10% χαμηλότερη κατανάλωση ισχύος για μόλις 1% σφάλμα και 70% χαμηλότερη κατανάλωση για 10% σφάλμα. Η πειραματική αξιολόγηση δείχνει ότι το METHADONE παράγει πολύ γρήγορα λύσεις (περισσότερο από 589×) οι οποίες βρίσκονται πολύ κοντά στις Pareto-βέλτιστες. Όλα τα προτεινόμενα πλαίσια αξιολογήθηκαν διεξοδικά μέσω εκτενούς πειραματικής διαδικασίας και συγκρίθηκαν με τις εκάστοτε βέλτιστες υπάρχουσες μεθοδολογίες και τεχνικές αποδεικνύοντας την αποδοτικότητά τους.

Λέξεις Κλειδιά: Αριθμητικά Κυκλώματα, Αυτοματοποιημένη Σχεδίαση Κυκλωμάτων, Επεξεργαστής Υλικού, Πολυ-επίπεδος Προσεγγισμός, Προσεγγιστικός Υπολογισμός, Σύνθεση Προσεγγιστικών Κυκλωμάτων, Υπερκλιμάκωση της Τάσης

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Energy Efficiency: "Sine Qua Non"

A prominent example of the importance of energy efficiency are data centers. Data centers are basically computer warehouses that store very large amounts of data and support a multitude of applications, systems, and functions [1] (e.g., streaming media, email, internet content, e-commerce, social networking [2]). Today, these applications are used more and more by our computers, mobile devices, sensors, and networks through the ever-growing cloud [3]. Because of the need to run incessantly [4], data centers require huge amounts of energy to operate (typical power densities of 538-2153 W/m^2, up to 10 KW/m^2 [5,6]). A 2013 example shows that U.S. data centers consumed an estimated 91 billion kWh of electricity [7] (Table 1.1), a trend that's only going to grow to an estimated 140 billion kWh by 2020; to put things into perspective, this translates to the the annual output of 50 power plants, costing American businesses $13 billion annually in electricity bills and emitting nearly 100 million metric tons of carbon pollution per year. In Europe the respective numbers are 72.5 billion kWh for 2010 and a projection of 104 billion kWh for 2020 [6]. If we also consider the amount of data these centers are handling (350m terabytes of data as of 2015 [3]) and the consequent power consumption (over 100TWh of electricity annually by 2020, the amount of public attention power-hungry data centers have drawn comes as no surprise.

Table 1.1: Data centres Energy consumption estimations and projections in TWh from a European, American and Global perspective. Source [6].

Consumption (TWh)	**Reporting Year**
EU consumption	
18.3	2000
41.3	2005
56	2007
72.5	2010
104	2020
US consumption	
91	2013
140	2020
Global consumption	
216	2007
269	2012

In addition to data centers, there is another major domain of the information and communication technology (ICT) sector that exhibits increased energy efficiency demands: embedded systems. An embedded system is an engineering artifact involving computation that is subject to physical constraints. These constraints affect available processor speeds, power, and hardware failure rates [8]. Many embedded systems feature very tight power budget (an order of some Watts) while ultra low-power ones (e.g. wearable systems) only a few milli watts [9]. The importance of energy efficiency for embedded systems becomes apparent if we consider the example of mobile or wearable systems. Although their small size comes with a tight power budget, they need to perform high performance functions, such as 3G/4G. If we add into the mix the ever-growing number (Figure 1.1) of embedded and internet-connected (IoT) devices (15 billion in 2015, est. 75 billion by 2025), it becomes obvious that the systems' ability to perform efficient computations will be absolutely essential.

Lastly, there is also an environmental reason behind the significance of energy efficiency. Close to 2% of the global CO_2 emissions comes from the

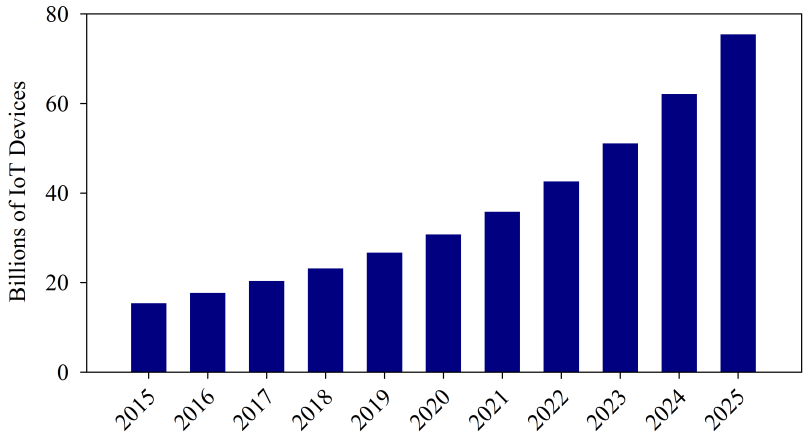


Figure 1.1: Internet of Things (IoT) connected devices (in billions) installed base worldwide from 2015 to 2025 (in billions). Source: IHS white paper https://cdn.ihs.com/www/pdf/enabling-IOT.pdf.

Information and Communication Technology (ICT) sector (including data centers), a percentage which is only projected to grow due to technological advances such as the cloud computing, as well as the rapid growth of the use of Internet services and IoT/internet-connected devices [6]. To put it into numbers, the ICT industry is forecasted to use 20% of the world's electricity by 2025, which also translates to 5.5% of the world's carbon emissions by then. Therefore, we can easily support that this is an issue that extends far beyond the ICT sector; it has a global effect on countries and communities. For example, this is well reflected on EU energy policies [10] that aim to improve energy efficiency in virtually every sector of the economy.

1.2 Moore's Law and Dennard's Scaling

Moore's Law [11] has been a fundamental driver of computing for more than four decades [12]. For the last forty years, industry's unrelenting focus on Moore's Law transistor scaling has constantly delivered increased transistor

performance and density. Throughout all these years of technology advancements and computer science evolution, leading researchers and technologists have awaited and forecast the "end of scaling" within one or two next generations. Nevertheless, every time the technology reached the anticipated transistor scaling break off, scaling continued (Figures 1.2 and 1.3) [13]. Inspired and innovative new solutions were developed to further prolong Moore's Law and maintain the transistor scaling roadmap [14].

> *Moore's law is the observation that the number of transistors in a dense integrated circuit doubles approximately every two years. The period is often quoted as 18 months because of Intel executive David House, who predicted that chip performance would double every 18 months. — G. E. Moore, 1965 [11].*

One of the main challenges of doubling the number of transistors on the chip is powering them without melting the chip and incurring excessively expensive cooling costs [12]. Although, considering the Moore's Law, the number of transistors in a dense integrated circuit (Figure 1.3) has im-

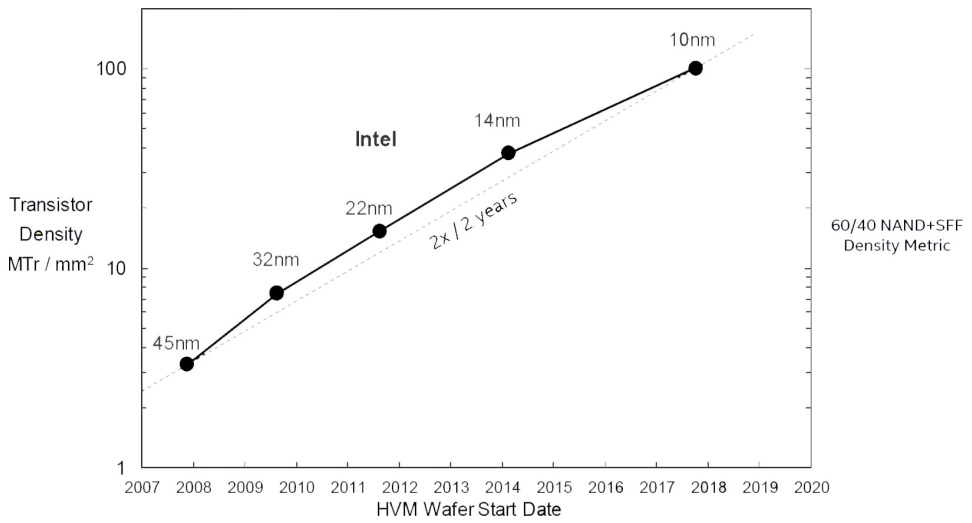


Figure 1.2: Transistor density improvements continue at a rate of doubling every 2 years. Source [15].

mensely increased, in the past 40 years, the chip power consumption has slightly increased. It is remarkable that with the same amount of power two times more transistors can be driven. In 1974, Robert Dennard [16], formulated how the transistor fabrication process technology can provide such physical properties. In fact, Dennard's theory of scaling is the main force behind Moore's Law [12].

> *Dennard scaling is a scaling law originally formulated for MOSFETs (also known as MOSFET scaling). It states, roughly, that as transistors get smaller their power density stays constant, so that the power use stays in proportion with area: both voltage and current scale (downward) with length.— R. H. Dennard 1974 [16].*

With Dennard's scaling rules, the total chip power for a given area size remained the same from process generation to process generation. Hence, a new process technology could double the transistors' count in a fixed chip

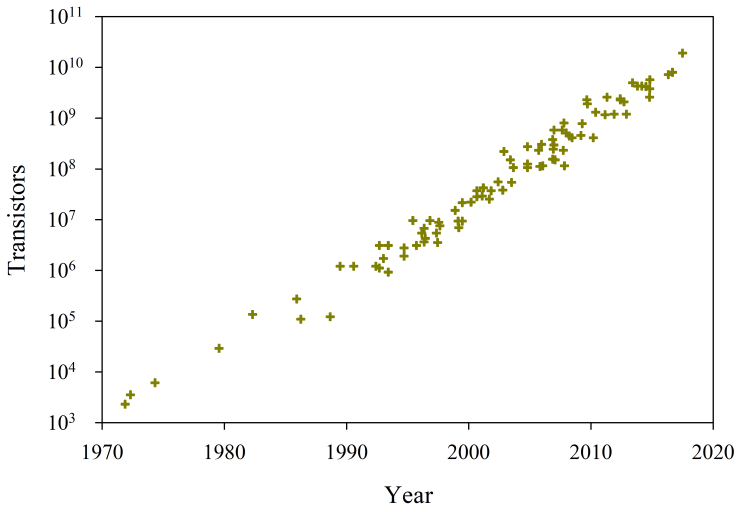


Figure 1.3: Number of transistors with fit into a microprocessor. Source: Karl Rupp, 42 Years of Microprocessor Trend Data https://github.com/karlrupp/microprocessor-trend-data.

size without increasing the power dissipation of the chip. Dennard's formulation [16] provided to our industry a concrete guide to the future, a way for setting goals and expectations for the next process technology generations. This paper, [16], provided a specific transistor scaling formula, needed to continue Moore's Law, which was first articulated by Gordon Moore in 1965 and was in effect being followed by the semiconductor industry since the early 1960's [17]. Moore's and Dennard's papers gave a roadmap to our industry on how to develop new integrated circuits on a constant pace, that deliver regularly improved performance and power. Reducing the critical dimensions while keeping the electrical field constant, yields higher speed and a reduced power consumption of a digital MOS circuits [16]. Every new process technology generation was expected to reduce minimum feature size by approximately 0.7× and it provided roughly a 2× increase in transistor density [17]. Starting in the mid-1990's our industry started introducing new technology generations once every 2 years. The trend of increasing chip size has slowed due to cost constraints, so we have settled into a trend of doubling transistor density and count every 2 years [17].

> *Moore's law gives us more transistors... Dennard scaling made them useful. — Bob Colwell, DAC 2013.*

1.3 The end of Dennard's scaling

Voltage scaling was a crucial component of Dennard's scaling because it maintains constant electric field, which is important for reliability, and it lowers transistor power, which is needed to maintain constant power density [17]. However, voltage scaling has run into lower limits imposed by threshold voltage (V_T) scaling limits [18]. Dennard's scaling law assumed that V_T would scale along with operating voltage, and thus provide improved performance and power [16]. Nevertheless, Dennard's scaling did not consider the impact of sub-threshold leakage. By 1970, sub-threshold leakage was quite small and its contribution to the total power consumption of the chip was negligible. However, by 2005 V_T has scaled to the point where sub-threshold leakage has increased more than 10,000× (from

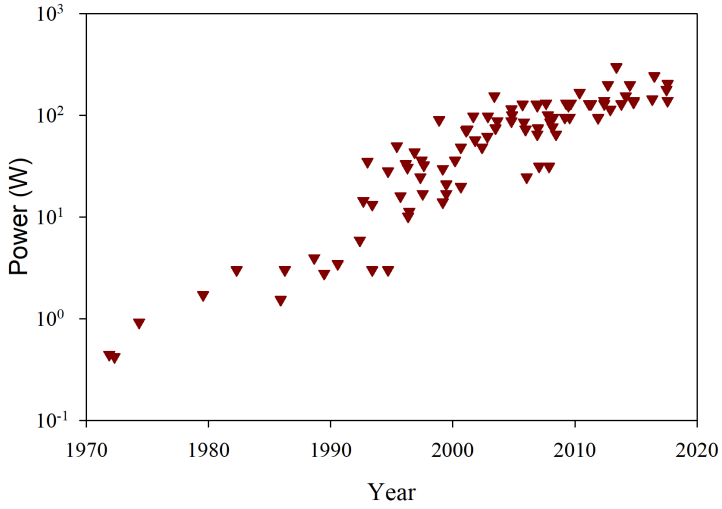


Figure 1.4: Scaling of processor power over years. Source: Karl Rupp, 42 Years of Microprocessor Trend Data https://github.com/karlrupp/microprocessor-trend-data.

$< 10^{-10}$amps/mm to $> 10^{-7}$amps/μm) and further reduction in V_T was not feasible anymore. Therefore, voltage scaling slowed down, since it was no longer possible to scale further the threshold voltage due to rising leakage currents [19]. Furthermore, another limiting factor in Dennard's scaling law was the assumption that the ability to scale gate oxide thickness would be retained. On the contrary, gate oxide thickness scaling reached the point of five atomic layers and the contribution of direct tunnel leakage current to the overall chip power highly increased [20,21]. The underlying cause of the provisioned power growth (Figure 1.4) can be traced to two factors: the fact that we did not scale power supply voltages at the constant field rate [22], and the fact that in our quest for performance, we scaled clock frequencies faster than dictated by constant-field scaling [19].

As long as Dennard's scaling was still in the spotlight, computer science leveraged the transistors' increase to produce higher frequency processors and equip them with more capabilities to further improve their performance [12]. With the end of Dennard's scaling, power densities in today's

integrated circuits are rapidly reaching unmanageable levels. Future technology generations can sustain the doubling of devices every generation, but with significantly less improvement in energy efficiency at the device level. This device scaling trend presages a divergence between energy-efficiency gains and transistor-density increases [23]. Driven by the regularly maintained 2× increase in transistors' count and the failure of Dennard scaling power management and efficiency emerges as a primary issue across most domains of the computing industry and is now essential for practical realizations. Therefore, the computing industry is forced to explore and adopt new computing alternatives in order to maintain the power scaling and increase the energy efficiency of the modern computing systems.

1.4 Approximate Computing

We are at the threshold of an explosion in new data, produced not only by large, powerful scientific and commercial computers, but also by the billions of low-power devices of various kinds [24]. Energy efficiency is now a first-class design constraint in computer systems. Its potential benefits go beyond reduced power demands in servers and longer battery life in mobile devices, since improving energy efficiency has become a requirement due to limits of device scaling and the so called "dark silicon" [25] or "power/utilization wall" problem. Computing and information technologies have entered now the "no-free lunch" era, meaning that radical departures from conventional approaches are needed to sustain and further improve the performance and efficiency of the computing systems.

Guaranteed numerical precision of each elementary step in a complex computation has been the mainstay and the fundamental principles of traditional computing systems for many years. But abstractions with perfect accuracy come at a cost. This era, fueled by Moore's law and the constant exponential improvement in computing efficiency, is at its twilight: from tiny nodes of the Internet-of-Things, to large-scale HPC computing nodes and data-centers, energy efficiency has become the paramount concern in

design of computing systems [26]. To overcome the "power wall", a shift from traditional computing paradigms is now mandatory and imposes the computing society to re-think these principles and investigate new computing alternatives. While precision is crucial for some tasks, many modern applications are fundamentally approximate. Recent research by Intel [27], IBM [28], and Microsoft [29,30] has demonstrated that there is a large body of resource-hungry applications that exhibit an intrinsic resilience to approximation errors and a significant portion of their functions/computations still produce outputs that are useful and of acceptable quality for the users. They observed that, in many modern online services it is acceptable to approximate rather than produce accurate outputs. Such services include search engines (Google, Microsoft Bing, Yandex, and Yahoo search), recommendation systems (Youtube, Facebook, Amazon, and Netflix), speech recognition (Apple Siri and Google voice search), and computer vision (online games). The "correctness" or quality of output of these services is defined as providing good enough or sufficient quality of results for users satisfaction [31]. Perfect answers are unnecessary or even impossible in several application domains [32, 33]. Today's systems waste time, energy, and complexity to provide uniformly precise operations for applications that eventually do not require it. For example, based on the work of Misailovic et al. [34], the preferred quality loss range is between 0-10% for applications such as video decoding. Moreover, another research by Park et al. [35], that recruited 700 users, showed that the level of acceptable quality loss significantly varies across applications, e.g., to satisfy the 90% of the users, 8% quality loss is acceptable for jpeg while they tolerate 26% accuracy loss for audio-enc.

Approximate computing forms a radical paradigm shift in systems design and operation, based on the idea that we are hindering computer systems' efficiency by demanding too much accuracy from them [36]. Approximate computing trades accuracy of computation for savings in execution time and/or energy by leveraging the error tolerance of the respective applications and by exploiting approximation opportunities across the computing stack [37]. Computing workloads with intrinsic error resilience are all around us, both in the embedded and cloud worlds, and they include digital signal processing, multimedia processing (image, video, audio), net-

work processing, wireless communications, web search and recognition and data mining [38]. As discussed in [38], the error forgiving nature of these applications may be attributed to a variety of factors (Figure 1.5), such as:

- Applications that process data from real world (e.g., inputs from sensors) feature and intrinsic error resilience. The input of those applications is innately noisy, and thus, they are designed in such a way to encompass and tolerate this noise in the performed tasks. Hence, they can be also resilient to inaccuracies in their computations.

- Applications that deal with large input data that feature high redundancy. The nature of such applications enables them to accommodate errors in their calculations without deteriorating their output quality.

- Applications that feature several satisfactory outputs. In such applications a perfect output, i.e., single unique result, is not mandatory or even impossible and multiple results are equivalently adequate.

- Applications that produce outputs for human consumption, e.g., audio and video media applications. The perceptual systems of the human exhibit a limited ability in detecting slight degradation in the digital content produced by such applications. As a result, small inaccuracies in their calculations produce results of acceptable quality.

- Applications that perform statistical and/or probabilistic computations. Such applications can mainly tolerate errors in their numerical calculations due to the nature of the implemented algorithms.

- Applications that perform self-healing computations. Such applications implement iterative computations where the result in every iteration is refined until meeting a certain threshold or satisfaction test. In these applications errors occurred in early computations can be corrected/refined in the later ones.

In the past years significant research activities have been performed in the

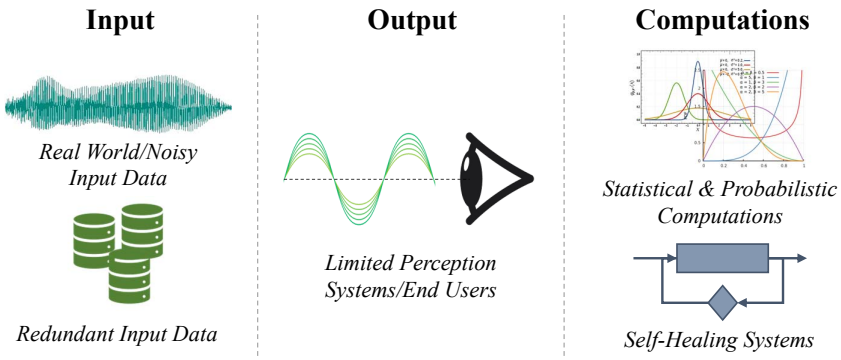


Figure 1.5: Various sources of inherently error resilient applications.

field of Stochastic/Probabilistic computing [39, 40]. However, approximate computing should not be confused with them. Stochastic computing uses random binary bit streams for computation and Probabilistic computing exploits the intrinsic probabilistic behavior of the underlying circuit fabric. The distinctive feature of Approximate computing is that it does not involve assumptions on the stochastic nature of any underlying processes implementing the system. It does, however, often utilize statistical properties of data and algorithms to trade quality for energy reduction. Approximate computing, hence, employs deterministic designs that produce imprecise results [37].

> *Our present treatment of error is unsatisfactory and ad hoc... Error is viewed (in this work), therefore, not as an extraneous and misdirected or misdirecting accident, but as an essential part of the process under consideration—J. von Neumann [40]*

Resilient applications are not, however, a license for computers to abandon predictability in favor of arbitrary errors. We need abstractions that incorporate approximate operation in a disciplined way. Applications and runtime systems should be able to exploit these richer abstractions to treat accuracy as a resource and trade it of for more traditional resources such as

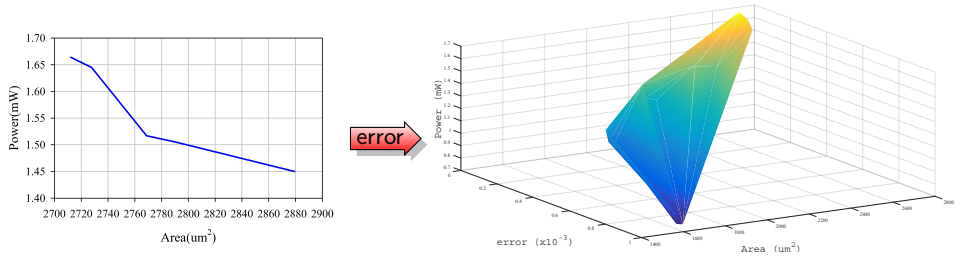


Figure 1.6: Adding the error dimension in the Pareto front of systems design. The produced Power-Area-Error Pareto front of the 16-bit approximate hardware multipliers is depicted (Chapter 4).

time, space, or energy. Different applications feature varying error resilience and exhibit different quality requirements [35]. Approximation needs to be done carefully as it can lead to unacceptable outputs and/or system failure. Hence, a primary target of approximate computing is to determine what degrees of approximations are feasible so that the delivered results are acceptable, albeit possibly different from those obtained using precise computation [24]. Approximate computing adds a new dimension in the modern systems design, i.e., the one of the error (Figure 1.6). As shown in Figure 1.6, the power-area trade-off changes with respect to the error bound. Under iso-area (iso-power) conditions, increasing the error value results in a decrease in the power consumption (area complexity). The addition of this extra dimension induces an extra overhead to the already increased complexity of efficient systems design, since the design space may increase exponentially [24,41]. Moreover, the designers have to find the Pareto front points that optimize the efficiency but also guarantee that the output quality constraints are satisfied. Therefore, systematic approaches with predictable (and bound) error characteristics are mandatory in order to enable and expand the application of approximate computing.

The potential benefits of approximate computing, as previously discussed, have attracted significant research interest in almost all the computer science domains. Notably, approximate computing research targets programming languages, compilers, runtime systems, software applications [30,34,42–55], hardware circuits/accelerators [56–79], and processor micro-architectures [29,48]. Hardware level approximation mainly targets arithmetic units, such

as adders and multipliers and/or accelerator synthesis. Approximate hardware circuits, contrary to software approximations, offer transistors reduction, lower dynamic and leakage power, lower circuit delay and opportunity for down-sizing. Moreover, another significant research domain in approximate computing is the Approximate Software–Hardware Co-design, i.e., how approximations in the hardware level can be exposed at the software level and how this synergetic nature of software and hardware approximations can be efficiently exploited.

1.5 Thesis Scope

In this thesis, we focus on the design of hardware approximate accelerators. In addition to the increased benefits of approximate circuits, it is also shown that using higher software accuracy and approximate hardware leads to more efficient solutions in terms of both energy consumption and delivered output accuracy [80]. With this work, we aim to address the main inefficiencies and limitations in the field of approximate computing circuits. The majority of related hardware approximate computing works provide ad-hoc (per circuit and/or application) solutions, limiting, thus, the exploitation of these techniques in several circuit implementations and/or application domains. Moreover, almost all of the proposed techniques do not leverage the full spectrum of approximate computing. As a result, the potential gains that could be delivered are limited. Finally, another important factor in approximate computing is the satisfaction of the application's quality constraints. Increased requirements for verifying functionality, as well as operating within the error bounds greatly increase design time cycle of approximate hardware accelerators.

The scope of this thesis is to provide generalized approximation techniques to enable the automatic generation of efficient hardware approximate accelerators that satisfy provided error requirements. Moreover, we aim to utilize the full potential of approximate computing and, hence, maximize the delivered gains while considering the specified error bound. Specifically, in this dissertation, we employ voltage over-scaling as an ubiquitous approxi-

mation technique and present a framework to enable very fast and accurate quantification of the power-error characteristics of voltage over-scaled approximate circuits. Targeting the generation of efficient approximate arithmetic circuits, we focus on the exploration of generalized and architecture-independent techniques for the generation of approximate multipliers (in contrast to research activities that mainly target approximate adders). In order to leverage the full spectrum of approximate computing, we examine multi-level approximation and build an arithmetic library comprising of multi-level approximate multipliers and adders/subtractors. The proposed circuits apply approximation techniques at the algorithmic, logic, and circuit levels, i.e., all the design layers of hardware circuits. In order to enable and automate the generation of such a library, we present a design framework that uses an heuristic algorithm along with power-error proxies to rapidly produce close to Pareto-optimal multi-level approximate arithmetic circuits. Finally, on the basis of the proposed simulator and the multi-level approximate arithmetic library, we present an automated synthesis framework that produces multi-level approximate hardware accelerators in a time efficient manner. The proposed framework seamlessly extends typical behavioral and/or RTL synthesis tools and, given an error constraint, generates approximate accelerators that closely follow the power-error Pareto front. Detailed information regarding the major contributions of this thesis can be found in Section 2.1.

1.6 Thesis Organization

The rest of this thesis is organized as follows:

- Chapter 2 briefly summarizes the major contributions of this thesis and reviews the related state-of-the-art approximate computing research works.

- Chapter 3 presents VOSsim, a framework that enables high level and very fast simulation of voltage over-scaled approximate computing circuits for rapidly quantifying their power and error characteristics.

- Chapter 4 introduces efficient approximate hardware multiplier designs by applying for the first time approximations on the partial product generation.

- Chapter 5 demonstrates the efficiency of multi-level approximation in the design of approximate circuits and presents HAM, a framework for generating very fast close to optimal multi-level approximate multipliers.

- Chapter 6 extends the multi-level approximation for arithmetic circuits as introduced in Chapter 5 and, by exploiting the techniques proposed in Chapters 3 and 4, presents METHADONE, a synthesis framework for generating multi-level approximate hardware accelerators that satisfy a given error bound.

- Chapter 7 concludes this thesis by summarizing the presented results and discusses the future extensions of this work.

Chapter 2

Contribution

In this chapter an overview of the respective research activities on approximate computing are presented as well as the major contribution of the proposed solutions is discussed.

2.1 Ph.D. Thesis Contribution

Two major principles govern the work presented in this thesis for designing hardware approximate accelerators. These principles that guided our research activities are as follows:

1. Maximize the energy savings delivered by the approximate computing application while satisfying the error constraints.

2. Propose generalized approximation techniques and automated approximate design frameworks that enable the straightforward application of approximate computing and require minimum user effort for their exploitation.

In order to achieve our first goal (principle 1), we examine, demonstrate, and exploit the efficacy of multi-level approximate accelerators, which are designs that simultaneously apply approximation techniques to all the design

layers, i.e., the accelerator's algorithm, the logic implementing the accelerator (truth table), and the circuit implementation of the accelerator (voltage supply). In regards to our second objective (principle 2), we present the following automated frameworks (Figure 2.1):

- **VOSsim**: A very fast high-level voltage-aware simulator that extends industry strength tools. VOSsim can be seamlessly integrated in any RTL design flow and enable simulation of Voltage Over-scaled approximate circuits at gate-level.

- **Partial Product Perforation**: A generalized technique for generating approximate multipliers that can be easily and out-of-the-box applied to any multiplier architecture. In addition to this technique we propose a framework that, given an error constraint, extracts the power-area Pareto-front comprising the multiplier's architecture and perforation configuration.

- **Hybrid Approximate Multipliers (HAM):** A novel design framework that exploits multi-level approximation for designing power optimized hybrid approximate multiplier architectures. Given a multiplier architecture and an error bound, HAM generates the power-optimal multi-level configuration requiring a small number of syntheses and simulations. HAM applies the proposed partial product perforation at the algorithmic level, approximate 4:2 compressors at logic level, and Voltage Over-Scaling at circuit level. HAM is then extended to support adders and subtractors, creating thus, a multi-level approximate library for arithmetic circuits.

- **Multi-lEvel approximaTe HArDware acceleratOr syNthEsis (METHADONE):** An approximate accelerator synthesis framework that, given the behavioral description of the hardware accelerator and an error bound, quickly produces its power-optimal approximate counterpart that satisfies the error bound. METHADONE operates on the scheduled data-flow graph (DFG) graph of the accelerator and applies multi-level approximation to the arithmetic nodes of the DFG to build the final inexact accelerator by exploiting the proposed multi-level ap-

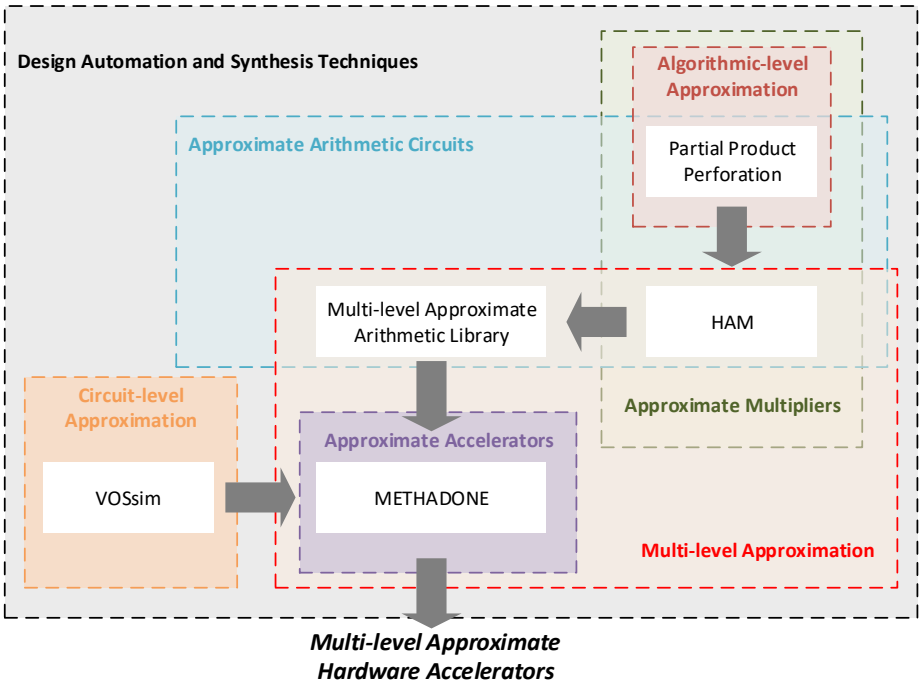


Figure 2.1: Taxonomy of the design automation and synthesis approximate frameworks presented in this thesis.

proximate library (HAM) and VOSsim. Moreover, METHADONE efficiently tackles the Voltage Islands Grouping problem that can occur when applying voltage over-scaling in such a fine granularity.

Figure 2.2 presents in a qualitative and visual manner the positioning of this thesis' contributions in the space of approximate hardware design. In this figure it is shown how the proposed frameworks advance the respective research domain and how this thesis overcomes the deficiencies of the existing state-of-the-art research works. The major contributions of this thesis are summarized in the following subsections.

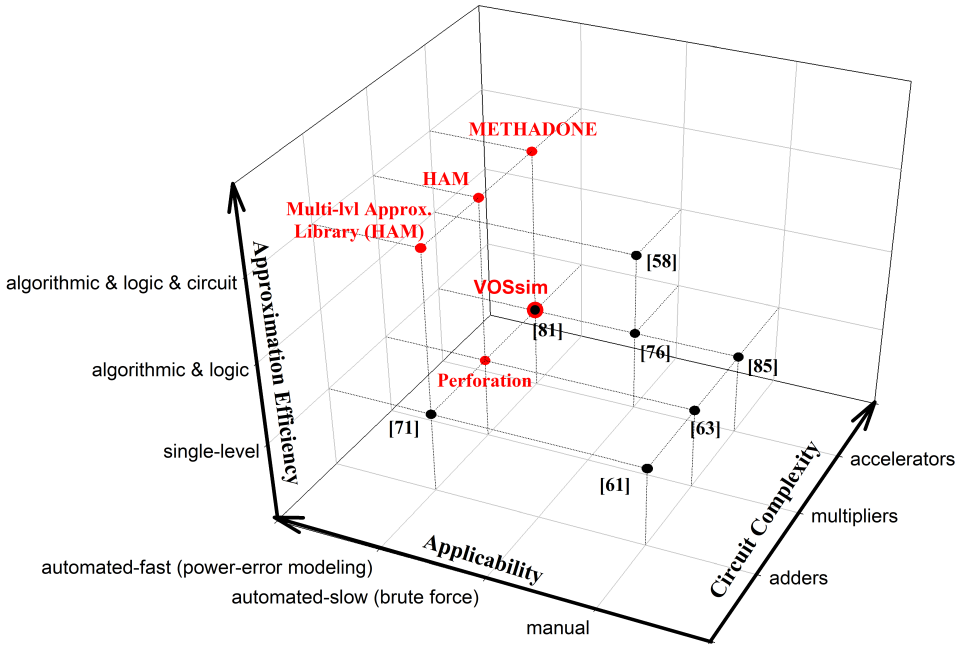


Figure 2.2: Thesis positioning in the space of approximate hardware design. The black dot refer to existing state-of-the-art works and the red ones refer to the contributions of this thesis. Next to every black dot a representative example is cited.

2.1.1 VOSsim

Addressed problem: *Increased complexity of quantifying the power-error characteristics of circuits under VOS.*

The increased complexity of performing Spice-level voltage-aware simulations as well as the high simulation time required, prohibit researchers from incorporating Voltage Over-Scaling in their approximate design works, thus nullifying the exploitation of its benefits. In order to tackle this inefficiency, we propose VOSsim.

VOSsim:

- seamlessly extends typical digital design flows operating over commonly used industry strength tools.
- operates at gate-level and hence, it relieves the designer from the increased complexity of performing SPICE-level simulations.
- is experimentally evaluated under stressed operating conditions, i.e., pipelined designs and operation at critical path delay, in order to examine the accuracy of proposed framework at a worst-case scenario.
- attains 99.2% output and 98.4% power accuracy on average, with an average speedup of 32× in simulation time compared to high precision SPICE simulations.

2.1.2 Partial Product Perforation

Addressed problem: *Limited research on approximate multipliers.*

Motivated by the limited research on approximate multipliers and explicitly by the fact that approximate multipliers apply logic-level approximations and target the accumulation of partial products, we introduce approximations in the partial product generation and propose the Partial Product Perforation method that is applied at the algorithmic level of the multiplier.

Partial Product Perforation:

- can be seamlessly applied to any multiplier architecture.
- is the first technique that applies approximations at the partial product generation.
- delivers optimized design solutions regarding the power–area–error trade-offs.

- features predictable and a priori known output error that does not depend on the multiplier's architecture and is analyzed in a mathematically rigorous manner. Such a rigorous error analysis for approximate arithmetic circuits is presented for a first time and enables precise error estimation over input data distributions.

- is extensively evaluated over several multiplier schemes exposing its power–area impact on different architectures. This is the first time that such an exploratory analysis over different approximate multiplier architectures is offered to the designer.

- and the presented framework enable the selection of the optimum architecture–perforation configuration for given error constraints.

- outperforms, in terms of power consumption and error, related state-of-the-art works that target logic-level and circuit-level approximations.

- outperforms related state-of-the-art works that apply approximations at the partial product generation and were published beyond the publication of partial product perforation.

2.1.3 HAM

Addressed problem: *Enable and exploit muli-level approximation in the design of approximate arithmetic circuits.*

Aiming to maximize the power savings derived by the approximate computing application on the design of hardware circuits, we examine multi-level approximation. With the HAM framework we target approximate multiplier designs that, given the error bounds, push power gains to the limits. However, applying multi-level approximation heavily increases the design complexity since the number of possible configurations grows exponentially. HAM tackle this limitation by using an heuristic algorithm and employing power-error proxies.

HAM:

- introduces multi-level approximation in the design of approximate arithmetic circuits by incorporating voltage reduction as an approximation method.

- demonstrates the efficiency of multi-level approximation. By performing an exhaustive design space exploration, it is shown that applying multiple techniques simultaneously produces more power-efficient approximate multiplier designs for same error values.

- can be applied to any multiplier architecture.

- produces multi-level approximate multipliers that closely follow the power-error Pareto-front derived by the exhaustive design space exploration.

- requires only a small number of syntheses and simulations, delivering a speedup of more than 140× compared to the exhaustive exploration.

- is compared with state-of-the-art approximation techniques when applied in isolation, showing that it outperforms them in terms of power consumption, area complexity, and output error.

2.1.4 METHADONE

Addressed problem: *Enable and exploit muli-level approximation in the design of complex circuits, e.g., accelerators.*

Despite the increased benefits of multi-level approximation, as demonstrated in HAM for arithmetic circuits, the production of multi-level approximate accelerators that satisfy a given error bound has not been efficiently addressed in the related literature. The latter is verified by the limited research in multi-level approximate accelerator synthesis (especially the lack of voltage over-scaling application) and is partially justified by the vast size

of the respective design space. In order to address this issue, we present METHADONE.

METHADONE:

- can seamlessly extend typical behavioral and/or RTL synthesis tools by operating on the accelerator's DFG.
- applies for the first time multi-level approximation in very fine granularity in complex hardware accelerator synthesis.
- produces close to power-error Pareto-optimal solutions with respect to the exhaustive design space exploration.
- provides multi-level approximate accelerators in a time efficient manner. It is noteworthy that, compared to the exhaustive design space exploration, it delivers more than 589× speedup.
- outperforms single-level state-of-art approximate techniques when applied in isolation and demonstrates the efficacy of multi-level approximation in the design of approximate hardware accelerator circuits.

2.2 Related Work

2.2.1 Software-level Approximate Computing

A wide research has been conducted in the field of software approximate techniques, comprising approximate computing languages, compilers, algorithmic transformations as well as quality-aware runtime systems. However, all the existing state-of-the-art works exhibit intrinsic limitations that prohibit them from fully exploiting the benefits of the approximate computing application. Approximate programming languages such as EnerJ [42] and Rely [43] expose approximation to the programmer through language syn-

tax. EnerJ and Rely let a programmer annotate their program with approximate type information. In EnerJ, programmer must explicitly delineate flow from approximate data to precise data, while in Rely the compiler tries to reason statically about how an approximate type flows through to others. EnerJ automatically maps approximate variables to low-power storage and operations. Rely is an imperative language that enables developers to specify and verify quantitative reliability specifications for programs that allocate data in unreliable memory regions and incorporate unreliable arithmetic/logical operations. Approximate compilers use approximations and algorithm analyses, to transform and the semantics of programs to trade accuracy for performance and/or energy consumption. Loop perforation is a software-only technique that modifies the algorithm loops and skips the execution of some iterations [34]. A compiler can also automate the placement of operations that execute on approximate hardware [44]. Moreover, [45,46] reduce the bit width of floating-point operations at compile increasing the program's performance. In [47] the authors lift the semantics of a program from its concrete operations on values to operations over distributions of values. In [48] the most intensive part of the algorithm is replaced by an artificial neural network. The compiler automatically performs the neural transformation and replaces this part with an invocation of a neural hardware accelerator. However, all these works apply do not exploit the synergetic cooperation of multiple approximation techniques. They either operate over unreliable hardware, lower the operations voltage, simplify the computations complexity, or reduce the operations' precision. Moreover, they are statically applied at compile time and no run-time quality guarantees are proposed. Targeting to tackle this inefficiency, several software approximate techniques are proposed that incorporate a runtime system to monitor the output quality. In [49] data approximation is proposed and Spark streaming is extended to operate over representative small samples of the input data. Similarly, [50] approximates Hadoop MapReduce by operating over samples of data and/or by not executing some tasks. Nevertheless, these frameworks are specific to Spark streaming and Hadoop MapReduce. The Green system [51] supports energy-conscious programming using loop and function approximations, but it is targeted for streaming applications in which the system is given a sequence of inputs such as a sequence of video frames, and the results from processing an input can be used to ad-

just the approximation settings for succeeding inputs. Uncertain<T> [30], provide abstractions that encapsulate approximate data within standard object-oriented programming languages and propagates approximate data through a program's variables at runtime. When a program needs to act on that approximate data (i.e., at a conditional) the Uncertain<T> runtime uses hypothesis tests to make statistically correct branch choices. Sage [52] produce approximate CUDA kernels with varying levels of approximation for applications running on GPUs, by applying selective discarding of atomic operations, data packing, and thread fusion. Similarly, Paraprox [53] produces approximate kernels of varying accuracy for OpenCL or CUDA parallel applications kernels by substituting common computation idioms found in data-parallel programs with approximate ones. At runtime Sage and Paraprox check the output quality once in every N invocations of the approximate kernel, in which they compare the quality between approximate computation and exact computation and adjust the approximate modes for the subsequent computations accordingly. However, checking the accuracy every N cycles may lead to unacceptable errors in the meantime, a proper value of N is application dependent and its selection is not comprehensively examined, remaining unclear. [54] proposed the fuzzy memoization technique, which records previous inputs and outputs of a code segment and predicts the output of its current execution with respect to the past executions with similar inputs. Finally, Capri [55] uses machine learning to learn cost and error models for a program, and uses these models to determine, for a desired level of approximation, knob settings that optimize metrics such as running time or energy usage. The error and cost behaviors are substantially different for different inputs. Nevertheless, although these approximate runtime systems deliver better output quality and with higher confidence levels the delivered power/performance are still limited since they apply software-only techniques and do support execution over approximate hardware and/or voltage over-scaled systems. Moreover, apart from Capri, the decisions made by these systems are not input-driven, limiting even further the potential benefits originated by the approximate computing application.

2.2.2 Hardware-level Approximate Computing

In this section, related research in the field of hardware approximate computing is discussed [56–69, 76–79, 81–87]. Both general-purpose approximation techniques [56–58] applied to any arithmetic circuit, as well as circuit-specific approximation either to adder [59–61, 82–84] or multiplier designs [62–66, 86, 87], have been presented. Finally, approximate design automation frameworks [58, 76–79, 81] are also proposed targeting to facilitate the generation approximate circuits that satisfy the quality constraints.

Regarding to the general approximation techniques, VOS [57, 67, 85] and truncation [56, 68, 69] have been proposed. VOS is applied in any circuit by lowering the supply voltage below its nominal value. Decreasing the supply voltage reduces the circuit's power consumption, but produces errors caused by the number of paths that fail to meet the delay constraints [67]. In [68], the authors proposed an automated generation of large precision floating point multipliers in FPGAs, using sophisticated truncation over underutilized DSPs. In [69], a truncated multiplier with a constant correction term is proposed, significantly decreasing the error imposed by typical truncation. [56] proposed a truncated multiplier with variable correction that outperforms [69] in terms of error.

Extensive research has been conducted targeting the implementation of approximate adders and multipliers. In [59], the authors developed a probability proof, estimating that the longest carry chain in an n-bit adder is log$n$, and produced a fast inexact adder limiting the carry propagation. Similarly, in [82], limiting the carry propagation and exploiting several subadder, the authors propose GeAR a quality configurable adder, along with its associated error probability model, that allows dynamic approximation according the expected output quality. In [60], approximation is performed by decomposing the addition circuit in an accurate and an approximate inaccurate part. In [61] and [83], the authors build imprecise full adder cells, requiring fewer transistors, by approximating their logic function and then use them to build imprecise adders. Targeting the generation of approximate adders optimized for FPGAs, [84] applies logic approximation at the

full adder's truth table and proposes a generic methodology to design approximate adder architectures by analyzing the architectural features and resources of the target FPGA. Approximate adders are generated in [71] by reducing the carry chains and then decreasing accordingly the voltage value. Although the authors propose the use of such adders targeting to build approximate multipliers, it is not clear how they can be used in different tree architectures and how their error scales in the case of multi-operand addition. Targeting the creation of approximate multipliers, [62,86,87] proposed a simplified approximate 2×2 multiplier cells and then use them as building blocks in larger designs. [63] presented two approximate 4:2 compressors by modifying the respective accurate truth table, which were then used to build two approximate multipliers outperforming [62]. The approximate compressors of [63] are used in Dadda tree with 4:2 reduction. However, different multiplier architectures were not explored. Based on an approximate adder that limits the carry propagation, [64] presented a fast and low-power multiplier scheme with higher error than [63]. However, in all the aforementioned approaches, the imposed error cannot be predicted as it depends on carry propagation and the circuits' implementation and requires simulations over all possible inputs in order to be calculated. The authors in [72] proposed a multiplier that rounds the input operands into the nearest exponent of two. [73] replaces the floating-point operations with fixed-point ones, and by applying the proposed stochastic rounding, achieve good accuracy results in training deep neural networks, while delivering high energy savings by limiting the data precision representation. [65, 66] proposed the use of $m \times m$ multipliers to perform an $n \times n$ multiplication (with $m < n$). In [65] the authors statically split the multiplicand in three m-bit segments and perform the multiplication utilizing the segment containing the most significant '1' (leading one). However, as stated in [66], m needs to be at least $n/2$ to attain acceptable accuracy, thus limiting the energy savings and the scalability of this approach. In [66] the authors extended the idea of leading-one segments to enable dynamic range multiplication and added a correction term. Although [66] delivers higher accuracy designs than [65] using smaller values for m, their approach requires the allocation of extra complex circuitry, i.e. two leading one detectors, two complex multiplexers for segment selection, one $\log(n)$-bit comparator, a $\log(n)$-bit adder, and one $2n$-bit barrel shifter. These extra components are expected to highly

increase the circuit's complexity introducing non trivial delay, area, and energy overheads that may considerably decrease the approximation benefits [65]. This is expected to be more evident in designs targeting very small error values, in which the need of larger m values is required. The modified Booth encoding is commonly used in signed multipliers [74, 75]. Jiang et al. [74] propose an approximate radix-8 booth multiplier that uses an approximate adder for producing $3 \times A$, and combine this idea with the truncation method. Liu et al. [75] designed approximate modified Booth encoders by modifying its K-Map, and combined them with the approximate compressors of [63].

Several recent research works have proposed techniques for automating the generation of approximate circuits. Probabilistic pruning and logic minimization techniques have been presented in [58], using a greedy approach to generate approximate circuits. These techniques systematically eliminate circuit's components and simplify logic complexity according to the circuit's activity profile and output significance. Both techniques heavily depend on the application's characteristics, and in addition the induced approximation error are not rigorously bounded. In [76] and [77] the authors systematically synthesize logic approximate circuits by exploiting the "don't care" conditions. [78] applies several approximate transformation operators on the circuit's behavioral description and through a greedy approach identifies their optimal combination. The authors in [79] propose approximate accelerator synthesis through precision scaling using an integer linear programming problem formulation. [81] extends [79] and incorporates also voltage scaling to leverage the critical path delay reduction. However, in [71,81] voltage reduction is not used as an approximation method, mainly due to the increased complexity of modeling-quantifying errors due to VOS, limiting thus, the potential energy savings obtained by aggressively decreasing the voltage value. All the aforementioned state-of-the-art works exhibit several limitations since they are either very time consuming and/or do not leverage the full spectrum of approximate computing techniques. They mainly focus on the application of a single type of approximation to avoid design complexity, neglecting, thus, the potential benefits originated by the synergetic incorporation of multiple approximation techniques to structure the final accelerator circuit, e.g., none of these works incorporates logic and

algorithmic approximations as well as VOS.

Chapter 3

High-Level and Fast Voltage-Aware Simulation for Voltage Over-Scaled Approximate Computing Circuits

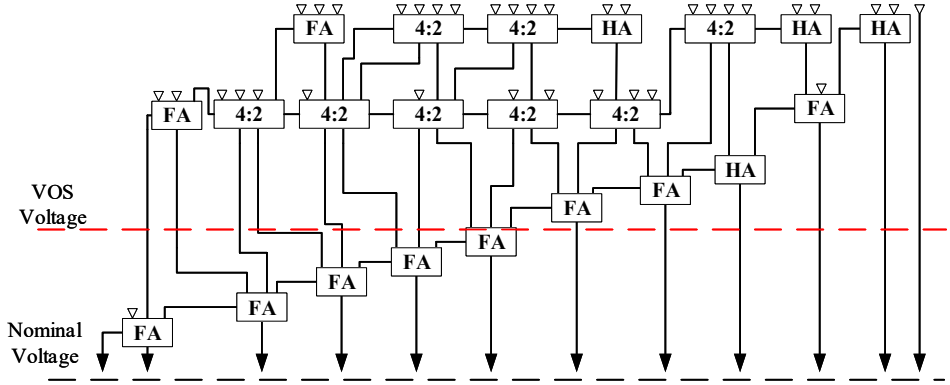
Approximate computing emerges as a new design paradigm for generating energy efficient computing systems. Voltage Over-scaling (VOS) forms a very promising technique to generate approximate circuits and its application in cooperation to other approximate techniques is proven to lead to more efficient solutions. However, the existing design tools fail to provide effective voltage-aware simulation for early exploration of power-error approximate design trade-offs. In this chapter, we present VOSsim, a framework that extends state-of-art industry strength tools to enable fast and accurate simulations of voltage over-scaled circuits. We extensively evaluate VOSsim showing that it attains 99.2% output and 98.4% power accuracy, with an average speedup of 32× in simulation time compared to high precision SPICE simulations, i.e., the only available solution today for VOS-aware simulation. This chapter is based on our publication in [88].

3.1 Introduction

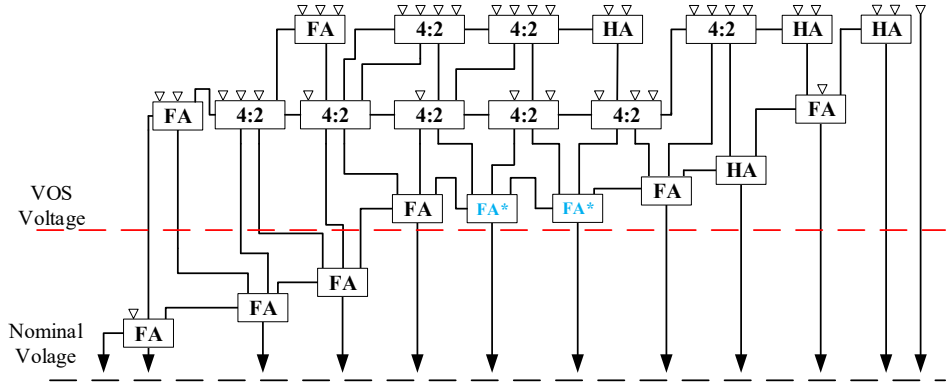
Since the failure of Dennard scaling [23], power consumption has become a first class concern in the design of integrated circuits. Recently, approximate computing has emerged as a design alternative for energy efficient system design, trading accuracy for significant energy gains [89]. Exploiting the intrinsic error resilience of a large number of application domains, e.g., DSP, statistical or probabilistic computations, and applications related to the limited human perception [38], approximate computing aggressively decreases power consumption by relaxing the computations' numerical correctness.

In hardware design, approximate computing can be applied in three distinct layers [41, 58, 86], i.e., the algorithmic level, e.g., omit computations [58], the logic level, e.g., truth table altering [63], and the circuit level, e.g., Voltage Over-scaling [85]. Voltage Over-Scaling (VOS) is one the most widely used techniques to generate energy efficient approximate circuits. VOS is applied to any circuit by keeping the operating frequency constant and decreasing the supply voltage below its nominal value [41]. The voltage decrease reduces the power consumption significantly, but erroneous outputs can be generated due to the circuit paths that fail to meet the time requirements [85].

Driven by the potential of approximate computing for energy reduction, high interest is shown in the design of hardware approximate accelerators [41, 58, 76–79, 86]. Recent works [41, 86, 89] indicate the benefits originated by the synergistic incorporation of multiple approximate techniques to construct the final accelerator circuit. Notably, it is demonstrated that applying VOS in cooperation to other approximate techniques leads to more efficient solutions. When applying approximations from the logic and algorithmic layers the circuit becomes simpler and usually the number of its critical paths is reduced and/or its critical path delay decreases [41,58,71,86,89]. Hence, by operating the approximate circuit at the frequency of the respective accurate one, decreasing the voltage value, can further reduce the power consumption, at the cost of a small (or zero) error increase, as fewer paths are affected by this voltage decrease [41, 71, 77]. Figure 3.1 depicts an in-



(a) Circuit w/o logic approximation. 6 bits are affected by VOS.



(b) Circuit w/ logic approximation. 4 bits are affected by VOS.

Figure 3.1: Qualitative example of applying Voltage Over-Scaling to a circuit with and without logic approximation.

tuitive example of this synergetic cooperation of VOS with approximation techniques from the other layers. In Figure 3.1a the accumulation tree using 4:2 compressors and final addition of a 6×6 multiplier is presented. In this case all the full adders (FA) are accurate ones. As shown, when the voltage value is decreased, six output bits are affected. On the other

hand, Figure 3.1b depicts the same circuit but two of the full adders are replaced with approximate ones (FA*) from [61]. In these FA* the output carry is equal to the full adders input A, therefore the delay of the output carry is zero. Hence, when the same voltage decrease is applied, as qualitatively shown in Figure 3.1b, only four output bits are affected by the voltage decrease.

In order to maximally exploit VOS capabilities, i.e., achieve the highest power reduction for given error constraint, the power-error trade-off has to be computed for every circuit architecture and every voltage assignment. Typically, time consuming Spice simulations are performed in order to compute the circuits' power consumption and the output errors when applying VOS [77, 85, 90]. The errors due to VOS *are deterministic in nature: if the very same computation is performed in the very same context, the very same faulty result will be produced* [91]. In other words, if an error occurs due to VOS under exactly the same context it is expected to obtain the same output. The same context refers to all the parameters that may affect the output value, i.e., voltage value, voltage threshold value, frequency, current and previous inputs, state of the registers, temperature as well as noise originated by external sources etc. Therefore, in order to have a good estimation of a circuit's error statistics when VOS is applied, large input datasets have to be used trying to cover all the possible input combinations and transitions (errors due to VOS errors depend on both the current and the previous inputs). However, the increased time needed to perform Spice simulations and the vast size of the input datasets make VOS simulation to quickly become a bottleneck in approximate design tool flows, especially during the design phase where several architectures with varying configurations are explored. Moreover, the required designer effort to perform Spice simulations increases as the system complexity scales up [92].

In [57, 92], the authors perform statistical analysis and apply probabilistic models to calculate computation errors due to VOS, obtaining moderate accuracy results. However, they do not provide a power estimation and cannot directly extend existing hardware design flows. Furthermore, these models fail to capture errors originated by other approximation techniques and thus, cannot be used in multi-level approximation. In the approximate computing

community, it is more than evident that there is a lack of a VOS specific methodology that rapidly quantifies VOS errors and power savings, thus limiting the potential benefits of standalone VOS as well as its cooperation with other approximation techniques [76, 78, 79].

In this chapter, we propose VOSsim, a framework that seamlessly extends typical hardware design flows, enabling for the first time, very fast (an order of magnitude) and accurate voltage-aware circuit simulation at gate-level. VOSsim operates over widely used industry strength tools, i.e., Design Compiler, PrimeTime, and Questasim, and performs voltage dependent gate level timing simulations, extending Questasim, to produce the circuit's output and power consumption at the desired voltage value. However, the proposed framework is not bound to these tools and with slight modifications can be adapted to any tool with similar functionality. Extensive experimental evaluation compared to high precision SPICE simulations, showed that for 1%-20% voltage decrease, the proposed VOSsim framework achieves 99.2% output and 98.4% power accuracy on average, while achieving an average speedup of 32× in simulation time. Hence, VOSsim can be used as an alternative to the traditionally performed SPICE simulations, to extract very fast but with high confidence the power-error characteristics of circuits under VOS.

The rest of this chapter is organized as follows: in Section 3.2 we discuss the limitations of the existing hardware design industry tools that inherently limit high level voltage-aware simulations. Next, in Section 3.3 the proposed VOSsim framework is described in detail and a discussion on how it tackles the aforementioned limitations is presented. In addition, this Section presents an evaluation of the flip flop behavior in the case of timing violations with respect to varying voltage values. Section 3.4 provides a qualitative comparison regarding the complexity of performing voltage-aware simulations with VOSsim against performing SPICE-level ones. In Section 3.5 the proposed VOSsim framework is experimentally evaluated by examining i) its accuracy in estimating a circuits power and error values and ii) the attained simulation time speedup compared to the traditionally performed SPICE-level simulations. Finally, Section 3.6 concludes this chapter.

3.2 Limited VOS Support in RTL Design Flows

In this section we identify and discuss the limitations of the existing state-of-art hardware design tools, that inherently limit non-SPICE VOS simulations.

The existing circuit synthesis tools, e.g., Synopsys Design Compiler and Cadence Encounter RTL compiler, given the hardware description of a circuit and the design constraints (e.g., frequency and voltage value) synthesize the circuit and produce its gate level netlist based on a provided technology library. However, altering the voltage supply before synthesizing the circuit can lead to different synthesis implementation and as a result, the voltage decrease must be applied after the synthesized netlist of the circuit is produced.

The existing circuit simulators, e.g., Mentor Questasim and Synopsys VCS, perform behavioral circuit simulations at RTL-level and timing accurate simulations at gate-level (post-synthesis). However, by default, these tools do not support the option of voltage (over-)scaling, i.e., only frequency scaling is supported. Therefore, the circuit's cell delays when VOS is applied, have to be pre-computed and provided to them before performing a post-synthesis timing simulation.

Finally, another limiting factor for enabling effective VOS simulation at gate-level is the flip flop timing violations. These violations occur when the flip flop input value is not stable during its setup and hold times around the clock positive edge (Figure 3.2). When applying VOS, the circuit becomes slower and the paths' delays increase. Therefore, it is possible that timing violations occur at the circuit flip flops. In this case, the circuit simulators will report a timing violation, the flip flop output value will become unknown and the simulator will produce a "x". These unknown states cannot be further interpreted by the simulator and the "x" will be propagated to all following computations that depend on the flip flop output. The "x" propagation affects both the simulation output and the switching activity (e.g., '0' to '1' transitions) and thus, the circuit power consumption calculation. Hence, during the simulations, special care must be provisioned for

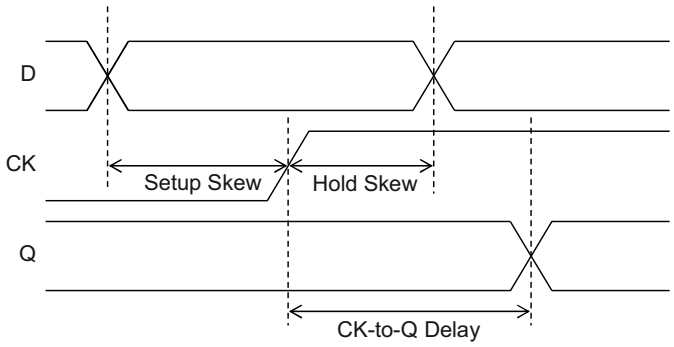


Figure 3.2: Flip flop setup and hold times. Source [93].

the setup and hold violations.

3.3 High-level VOS-Aware Simulator

3.3.1 The proposed VOSsim Framework

In this section, the proposed framework is described and we demonstrate how the aforementioned limitations are tackled and how typical hardware design tools (that initially do not support this functionality) are extended to enable "precise enough" VOS simulations at gate-level. Note that, all the tools in our framework are industry strength tools and are widely used in standard hardware design flows. However, the proposed methodology is tool agnostic and can straightforwardly be adapted to any RTL synthesis and gate-level simulation tool-flows.

The proposed framework is illustrated in Figure 3.3 and operates as follows: the circuit is synthesized at the desired frequency and nominal voltage, and then its cells delays are calculated for the desired voltage supply. Next, the circuit's gate-level netlist and the obtained voltage over-scaled path delays are fed to the circuit simulator and a violation-aware post-synthesis timing simulation is performed. Hence, the circuit's outputs and switching activity

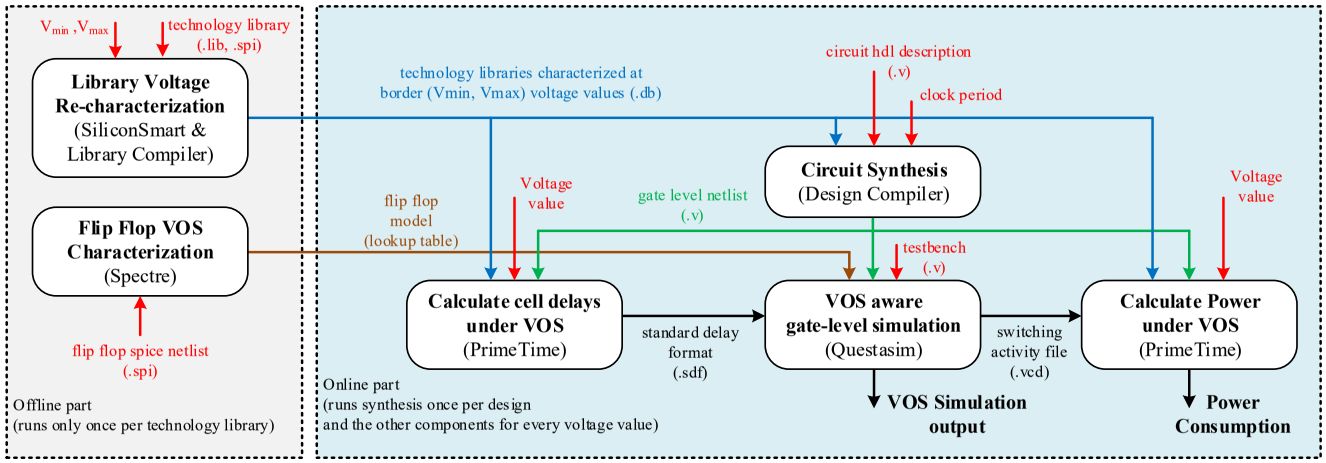


Figure 3.3: The Proposed framework flow diagram. The elements colored in red are the framework's inputs. At every component of the framework the major tool used is reported inside parentheses. The offline part is executed only once per library and the online part performs the VOS-aware simulation.

results, when VOS is applied, are produced. Lastly, using the synthesized netlist and the obtained switching activity, the circuit's power consumption for the respective voltage value is calculated. The proposed framework is divided in two parts, i.e., the offline and the online. The offline part comprises the library re-characterization and the flip flop characterization components and is executed only once per library. The overall VOSsim flow is described in detail next:

Circuit synthesis: This module is responsible for synthesizing the circuit and producing its gate-level netlist that is required by the other components. Given the circuit's hardware RTL description and a technology library (characterized at its nominal voltage value), it uses Synopsys Design Compiler (M-2016.12) to synthesize the circuit at the desired clock period.

Calculate cell delays under VOS: After synthesis, the gate-level netlist is obtained and the path delays have to be calculated for the desired voltage value. In order to evaluate the delay of all the circuit's paths when lowering the voltage supply, we utilize Synopsys Composite Current Source (CCS) timing model [94]. CCS model is current-based and it is designed with the intention to be scalable for voltage, temperature and process. It is proven to deliver very high accuracy (within 2%) compared to HSPICE. Having a technology library characterized at two different voltage values, using the CCS model we can sweep at any voltage value in between and extract timing and power information for the library cells [94]. For more information regarding the CCS model refer to Section 3.3.3. So, by characterizing the technology library at the minimum and maximum voltage values that will be used in the VOS analysis, using the CCS model we obtain the respective information for any voltage in between. The *Library voltage re-characterization* component performs this task by using Synopsys SiliconSmart (L-2016.06) to re-characterize a technology library at the desired voltage value and Synopsys Library Compiler (L-2016.06) to compile it. During the re-characterization procedure we instruct SiliconSmart to use the CCS model for the output library. Hence, having available a technology library characterized at two border voltage values, V_{max} and V_{min}, this component uses Synopsys PrimeTime (M-2016.12) and

Algorithm 1 Flip Flop VOS Characterization

1: **for all** $v \in [V_{min}, V_{max}]$ # all voltage values
2: **for all** $F \in \{Library's\ Flip\ Flops\}$ # all flip flop types
3: **for all** $TT \in [0, 1]$ # all transition types (i.e., 0→1 and 1→0)
4: **for all** $t \in$[-*setuptime*, *holdtime*] # all violation times
5: x = enforceTimingViolation(TT →(1-TT), t)
6: (FF_{out}, *Clk-to-Q*) = SpectreSimulation(F, v, x)
7: lookupTableModel(v, F, t, TT) = (FF_{out}, *Clk-to-Q*)
8: **return** lookupTableModel

by exploiting the scalability of CCS model, performs a timing analysis on the synthesized netlist and calculates the circuit's cell delays at the selected voltage value. The `define_scaling_lib_group`, `set_voltage`, and `update_timing commands` are used, to create the voltage-dependent scaling space, select the desired voltage value, and extract the circuit's timing information at the respective value. Finally, the Unified Power Format (UPF) is used to express the voltage related circuit parameters and the cells' VOS timing information is stored in Standard Delay Format (SDF), in order to be used in the circuit simulation.

Flip flop VOS characterization: In Section 3.2, the significance of special consideration for the flip flop timing violations, during the circuit simulation, is discussed. Hence, a core module of the proposed framework is the characterization of the flip flop behavior when these violations occur. Algorithm 1 is used to model the flip flop's output in case of setup and hold violations. When a timing violation occurs, it is not guaranteed that the flip flop will latch the new value and the clock to output (clock-to-Q) time increases. Therefore, by modeling the flip flop we refer to estimating both the value that the flip flop will produce and the clock to output time. We model the flip flop output with respect to the voltage value, the flip flop input transition type ('0'→'1' or '1'→'0'), and the time that the violation occurred. As time, we refer to the relative time with reference to the clock positive edge. Therefore, the setup violations have negative relative time. In Algorithm 1, for every library's flip flop, for each examined voltage value, and for every transition type, we produce timing violations every 1ps in the

range of [-*setup time*, *hold time*] and using Cadence Spectre (14.1.0), i.e., SPICE-level accuracy, we compute the flip flop output and the time that it is produced. The outcome of Algorithm 1 is a model in terms of a lookup table that given the voltage value, the transition type and the relative time of the violation, returns the flip flop's output value as resulted by the Spectre simulation. The produced model is used in VOS simulation to guide Questasim to produce an appropriate value for the replacement of the "x", i.e., unknown values that the simulator generates by default in case of a timing violation. Studying flip flop meta-stability issues is out of the scope of VOSsim and other flip flop models can be also seamlessly adopted by our framework. Using Algorithm 1, we characterize the flip flop's behavior for clock periods from 0.1ns up to 1ns. We observed that for periods higher than 0.25ns we obtained similar results regarding the flip flop output value and the time that it is produced. Hence, since clock periods less than 0.25ns are considered very small, we use the results obtained at 0.25ns as our flip flop model. However, this component can also be used in the online part and extract the flip flop model for a specific period and voltage value with a negligible time overhead. In general, this component can be moved in the online part and it can be executed in parallel to the Calculate cell delays under VOS component. The only reason that the Flip Flop VOS characterization component is added in the offline part is based on our observation that for periods higher than 0.25ns we obtained similar results. Therefore, without any loss of optimality, the Flip Flop VOS characterization component can used in the online part. The user can run the Flip Flop VOS characterization component for the desired voltage value and period and extract the respective model with respect to those parameters. Then this model can be straightforwardly used in the VOS aware gate-level simulation performed by the proposed framework. The execution of the Flip Flop VOS characterization component for a voltage value and a specific period requires about 25 seconds. As a result the induced time overhead is insignificant and equal to:

$$\left\lceil \frac{\text{number of different flip flop types in the design}}{\text{number of CPU threads}} \right\rceil \times 25s.$$

VOS aware gate-level simulation: This component performs the actual circuit simulation, at the desired voltage value, based on the outputs of the

previous components. In the proposed framework, VOS aware simulation is conducted at gate-level by performing post-synthesis timing simulation using Questasim (10.4c_5) and a user provided testbench. Questasim simulates the gate netlist produced by the circuit synthesis component and uses as its timing information the SDF file produced by the module calculate cell delays under VOS. This SDF file is produced for the desired voltage value, enabling Questasim to simulate the circuit's behavior at this voltage. Moreover, the performed simulation uses the lookup table produced by the flip flop VOS characterization module, in order treat accordingly the flip flop timing violations. Before performing the timing simulation, this component parses the gate-level netlist and finds all the flip flop instances. For every flip flop instance, it also parses the SDF file and extracts its setup and hold times. A tcl script is automatically generated that monitors (at every period) each flip flop's input value in the segment [-*setup time*, *hold time*] (around the clock positive edge) every 1ps and detects if a timing violation occurs based on the input's previous and current values. In the case of a violation, it estimates the flip flop's output value using the previously computed flip flop lookup table. The flip flop type, the input transition type and the relative time of the violation are used as arguments to fetch a value from the lookup table. Then, this component sets the flip flop's output value to the one obtained from the lookup table, overriding the "x" value that Questasim would otherwise produce. To monitor a flip flop's input value we use the `examine` command and to set its output the `force` one. Using `force`, we can specify the time that the change will take effect. The generated script replaces the Questasim `run` command and therefore, it seamlessly extends Questasim timing simulation and no testbench modifications are required. This script is circuit and voltage specific, and is re-generated every time one of them changes. Having produced the circuit's timing information under VOS and the aforementioned script that monitors for setup-hold violations, we perform a post-synthesis timing simulation with Questasim. Hence, the circuit's VOS output and its switching activity for the respective voltage value are produced. The simulation is normally performed, with the exception of the [-*setup time*, *hold time*] time segments where the flip flops are monitored.

Calculate power under VOS: The final step of the proposed framework es-

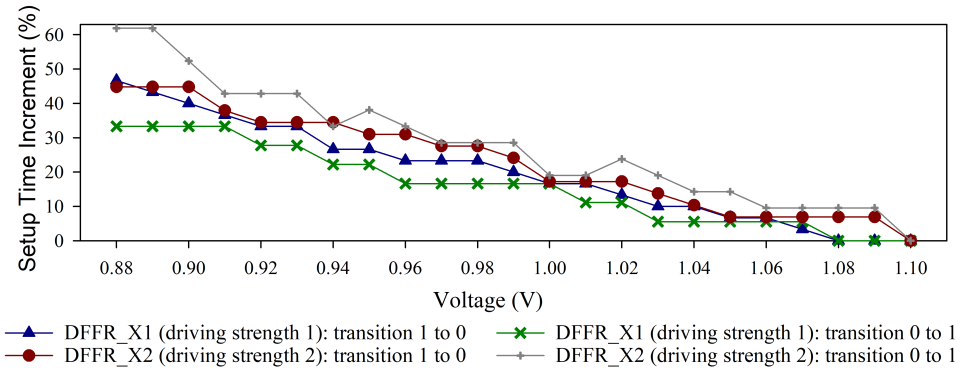
timates the circuit's power consumption when VOS is applied. This component uses Synopsys PrimeTime and the CCS model to calculate, at the desired voltage value, the cells' power consumption of the previously synthesized netlist. Then it reads the switching activity from the conducted post-synthesis simulation and performs a power analysis to calculate the circuit's power consumption at this voltage value.

The proposed framework is divided in two parts, i.e., the online and offline ones. The online part consists of: i) the circuit's synthesis, ii) the re-evaluation of the cells delays when VOS is applied, iii) the VOS aware post-synthesis timing simulation, and iv) the circuit's power consumption calculation. Among these online components the circuit's synthesis (i) is performed only once per circuit and the rest (ii-iv) are executed for every desired voltage value. The offline part comprises the (optional) library re-characterization and the flip flop VOS characterization modules and has to be executed only once per technology library. The cell re-characterization at a voltage value can be executed independently for every library cell. Similarly, the flip flop VOS characterization can run in parallel for every flip flop type and/or voltage value. Hence, the Library Re-characterization and the Flip Flop Characterization components are scalable with respect to the number of executors (CPU threads). Therefore, the time required by the offline part is determined by i) the number of the available CPU threads, ii) the library size, iii) the number of different flip flops types in the library, and iv) the voltage decrease range. In this evaluation we assigned 16 CPU threads to every one of these components. The re-characterization of the Nangate 45nm library required 7 minutes, while the characterization of all the Nangate's flip flops (16 in total) for 20 voltage values at a specific clock period required about 8 minutes. In our evaluation we executed the flip flop characterization for 10 clock periods, resulting in a total time of 80 minutes. Concluding, note that the time required for these processes can scale down if more CPU threads are available. Moreover, this time is insignificant compared to the increased time required to perform a SPICE-accurate simulation, e.g., the SPICE-accurate simulation of the smallest benchmark (MAC) examined in Section 3.5 required 508 minutes. Finally, detailed information regarding the time requirements of the online part are presented in Section 3.5.

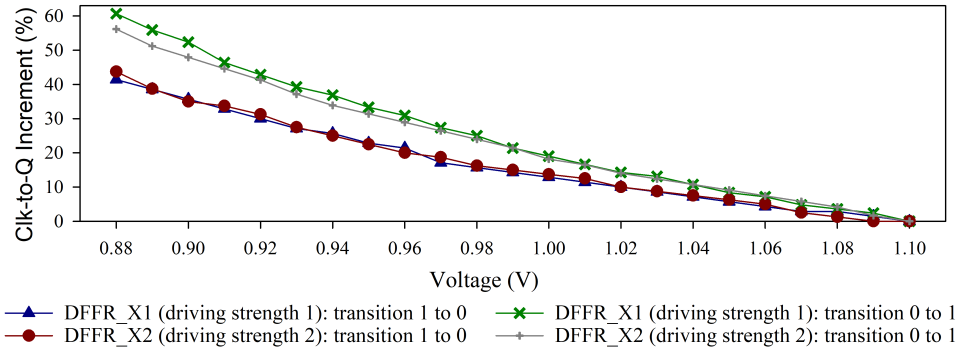
3.3.2 Evaluation of Flip Flops under VOS

In this Section the Flip Flop VOS characterization component of VOSsim is used to explore the flip flop's operation and behavior when the voltage supply is decreased as well as when timing violations occur. The flip flop setup and hold times depend on the supply voltage, however the setup time is more sensitive to it and the hold time at nominal voltage is reasonably pessimistic [93, 95].

In Figure 3.4, targeting to examine the impact of the voltage supply on the flip flop operation, the setup time and Clock-to-Q delay variation with respect to the voltage value is presented. In order to measure the setup time and the clock-to-Q delay for different voltage values, we follow the approach described in the HSPICE Applications Manual and referenced as "pushout" of the Clock-to-Q delay and state [96]. For every voltage value, a flip flop simulation is performed where the D input value is altered half clock cycle before the positive clock edge and the Clock-to-Q delay is measured. The obtained value is considered to be the nominal Clock-to-Q delay for the respective voltage value. Following, the input transition time is moved closer and closer (every 1ps) to the clock's positive edge. As a result, a Clock-to-Q delay increase is observed. At the point where the Clock-to-Q delay becomes 5% greater than the previously measured nominal value, the segment from the input transition time to the clock's positive edge is considered to be be the flip flop's setup time. In Figure 3.4 both transition types ('1'→'0' and '0'→'1') are considered and two D-type positive edge Flip-Flops with active low reset of the Nangate 45nm library are examined, i.e, one with driving strength 1 (DFFR_X1) and another with strength 2 (DFFR_X2). The increment of Clock-to-Q delay and setup time is reported as percentage increase with respect to their values measured at the nominal voltage value (1.10V). As expected, as the voltage value scales down, the flip flop becomes slower and the Clock-to-Q delay as well as the setup time increase significantly. The setup time increase depends on both the transition and the flip flop type. The DFFR_X2 flip flop features higher percentage setup time increase than the DFFR_X1 for both transition types. However, the DFFR_X1 features higher setup time increase for the '1'→'0' transition, whereas the DFFR_X2 exhibits higher



(a) Setup Time Increment w.r.t to Voltage Value



(b) Clock-to-Q delay Increment w.r.t to Voltage Value

Figure 3.4: The a) setup time increase and b) the clock-to-Q delay increase as the voltage value decreases.

setup time increase for the '0'→'1' transition type. As a result, regarding the transition type, the setup time of different types of flip flops exhibits different behavior when lowering the voltage value. On the other hand, the percentage Clock-to-Q delay increase is almost proportional to the voltage value and depends more on the transition type and less on the flip flop type.

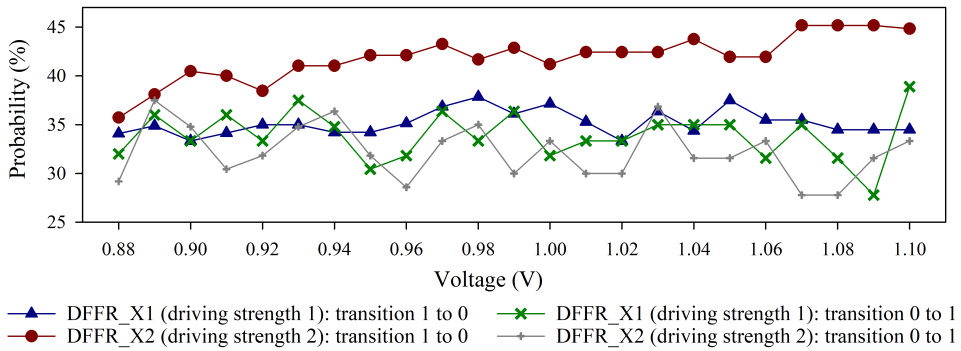
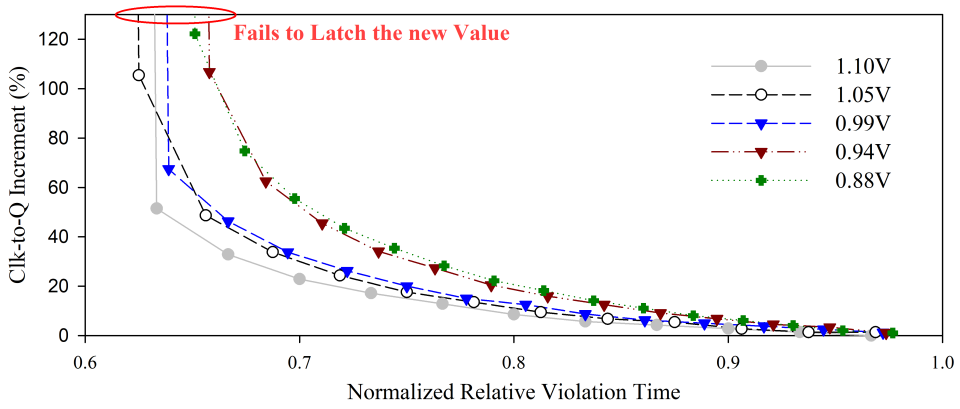


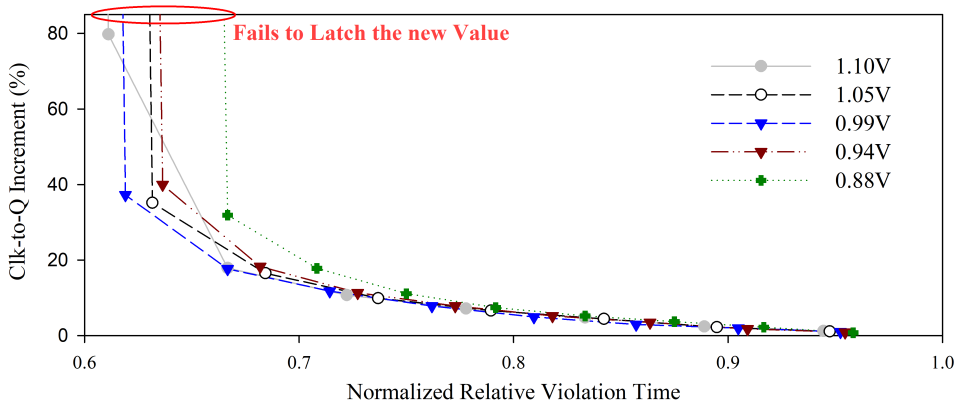
Figure 3.5: The probability that the flip flop latches the new value when a timing violation occurs.

Figure 3.5 presents, for varying voltage values, the probability that the flip flop latches the new value despite the occurrence of a timing violation. For every voltage value, we calculate this probability by producing timing violations every 1ps and measuring the number of violations where the flip flop latched the new value. On average, the '0'→'1' transition exhibits lower probability for both flip flop types. As shown in Figure 3.5, despite the occurrence of a timing violations, there is a 36% possibility, on average, that the flip flop latches the new value. These considerable latching probability values presented in Figure 3.5, highlight the importance of monitoring the flip flops for timing violations and estimating precisely their output value when VOS is applied. Moreover, Figure 3.4 and Figure 3.5 justify the approach followed in the proposed framework to consider the voltage value, the transition type, and the flip flop type when estimating the flip flop's output value.

Finally, targeting to examine the flip flop's behavior in case of a timing violation, Figure 3.6 and Figure 3.7 depict the variation of the Clock-to-Q delay with respect to the relative time of a setup violation. In Figure 3.6 the DFFR_X1 flip flop is considered and in Figure 3.6 the DFFR_X2 one. In both figures, five different voltage values are examined, corresponding to 0%, 5%, 10%, 15%, and 20% voltage decrease, respectively. For every voltage value, the Clock-to-Q delay increase is reported as a percentage increase with respect to its respective nominal value (measured at the nominal

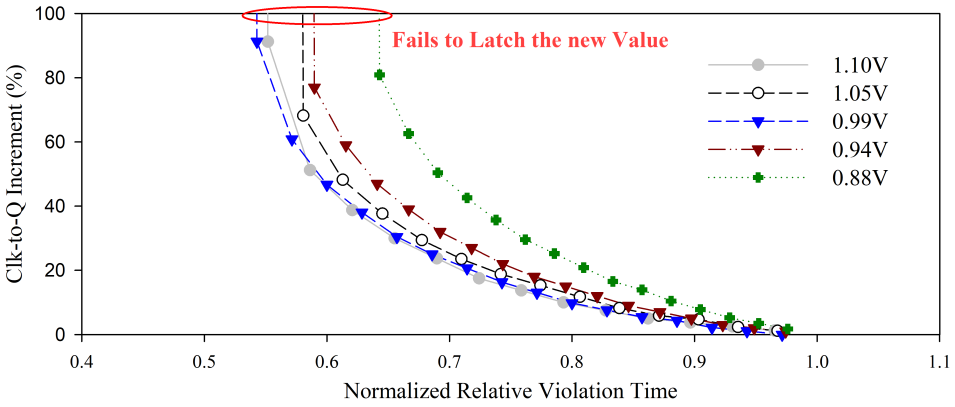


(a) DFFR_X1 '1'→'0' transition

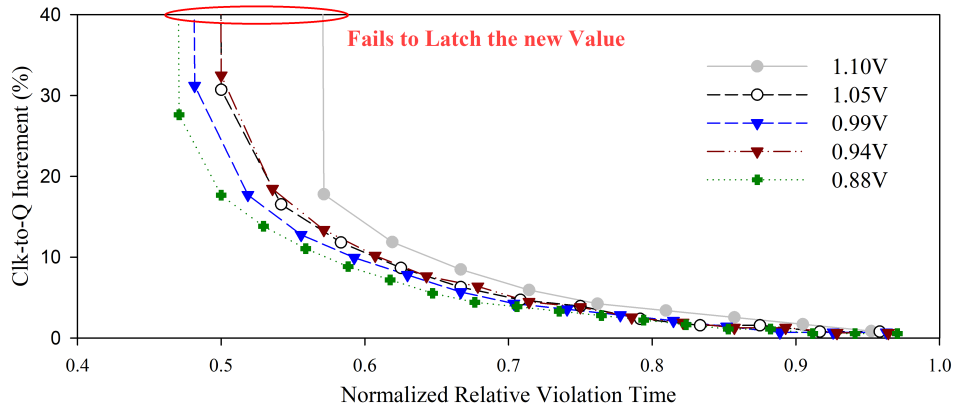


(b) DFFR_X1 '0'→'1' transition

Figure 3.6: The Clock-to-Q delay increase with respect to the normalized relative violation time for a) '1'→'0' and b) '0'→'1' transition. The DFFR_X1 flip flop is examined.



(a) DFFR_X2 '1'→'0' transition



(b) DFFR_X2 '0'→'1' transition

Figure 3.7: The Clock-to-Q delay increase with respect to the normalized relative violation time for a) '1'→'0' and b) '0'→'1' transition. The DFFR_X2 flip flop is examined.

voltage value and when no violations occur). As the setup time varies with respect to the voltage value and flip flop type, in Figures 3.6 and 3.7 the normalized relative time is reported, i.e., the ratio of the relative violation time over the respective setup time. Hence, the smaller the normalized relative time is, the closer to the clock's positive edge the violation occurs. The Clock-to-Q delay value defines if the flip flop latches the new input value in the case of a timing violation and becomes infinite when the flip flop malfunctions (due to the voltage decrease and/or the setup violation) and is not able to latch it. In both figures, for all the examined voltage values, as the normalized relative time decreases (the violation occurs closer to the clock edge) the Clock-to-Q delay increases significantly. For every voltage value and transition type, it exists a relative time threshold after which, if a setup violation occurs, the flip flop fails to latch the new value. As shown in Figures 3.6 and 3.7, this failure threshold depends the flip flop type, the transition type, and the voltage supply value. For both flip flop types, the Clock-to-Q delay of the '1'→'0' transition is more affected by the violation time, i.e., features higher increase compared to the respective increase of the '0'→'1' transition. Among all the examined flip flops and transition types, the Clock-to-Q delay value of the '1'→'0' transition of the DFFR_X2 flip flop is the most sensitive one, featuring the highest increase for every examined voltage value. On the contrary, the Clock-to-Q delay of the '0'→'1' transition of the same flip flop is the least sensitive one. This behavior is also confirmed by the latching probabilities presented in Figure 3.5. Concluding, Figures 3.6 and 3.7 justify the significance of considering the relative violation time when estimating the flip flop's output under VOS.

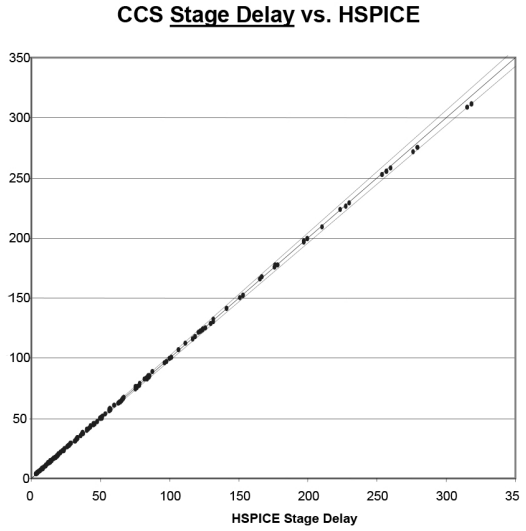
3.3.3 Brief Description of CCS Timing Model

In this Section a brief description of the CCS model (one of the core modules of the proposed VOSsim framework) is presented. For more detailed and technical information refer to [97] and [94]. CCS Timing consists of a driver model and a receiver model. The driver model describes how a timing arc propagates a transition from input to output, and how it can drive arbitrary RC networks. The receiver model describes the capacitance that

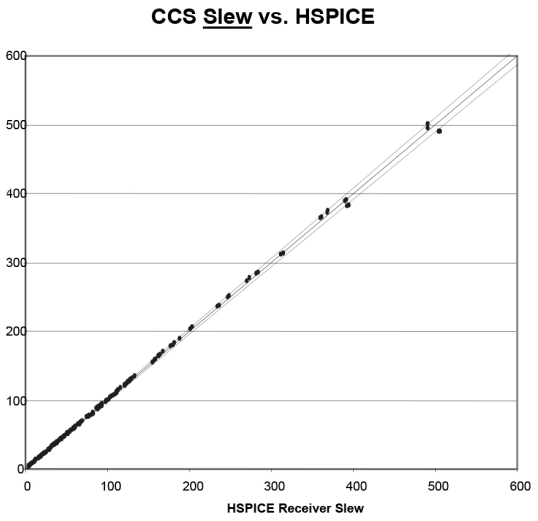
an input pin presents to driving cells. The CCS Timing driver model is a time and voltage dependent current source with an essentially infinite drive-resistance, which provides high accuracy even when the drive resistance R_d is much less than the network impedance Z_{net}. The model achieves this accuracy not by modeling the transistor behavior, but by mapping the arbitrary transistor behavior for lumped loads to the behavior for an arbitrary detailed parasitic network. CCS Timing delay calculation uses advanced interpolation technology to determine a current waveform when the input slew and/or output load values do not match those used during cell characterization. Additionally, interpolation is used for intermediate values of V_{DD} and temperature by using data from multiple libraries. CCS Timing delay calculation provides a high accuracy response for cell delay, interconnect delay, and pin slew. The CCS Timing receiver model produces excellent results on single-stage cells with large Miller effect. CCS Timing stage delay and slew results are typically within 2% of the golden circuit simulation values. Figure 3.8 shows a comparison of CCS Timing versus HSPICE for a large number of test cases, including highly-resistive nets. CCS Timing enables scaling for intermediate V_{DD} and temperature values. Library characterization is done for a small number of V_{DD} values, with advanced current waveform interpolation at runtime. Calculation can be done for any instance-specific value in a continuous range of V_{DD}. This is a key element of flows considering the timing effect of IR drop, and also supports multi-V_{DD} and DVFS (Dynamic Voltage and Frequency Scaling) designs. CCS Timing scaling also supports delay calculation for arbitrary temperature values between characterization points. Driver model and receiver model data are both scaled. In addition, timing check arcs such as setup, hold, recovery and removal are also scaled.

3.4 VOSsim vs SPICE complexity

In this Section a qualitative discussion regarding the complexity of using VOSsim as an alternative to SPICE-tools in performing voltage-aware simulations is presented. The proposed method to perform high level VOS-aware simulations comprises two parts, i.e., the offline and online ones.



(a) Stage Delay comparison



(b) Slew comparison

Figure 3.8: CCS timing results vs HSPICE. Source [97].

The first produces all the necessary libraries and models for the operation of the proposed framework and the latter performs the actual simulation. As aforementioned, the VOSsim's offline part, i.e., Library Voltage Re-characterization and Flip Flop VOS Characterization, needs to be executed only one time per technology library. Therefore, if the user initially executes the offline part for all the available libraries, he can seamlessly change the technology library during the design and simulation phase. The latter is easily performed by specifying the path to the folder containing the outputs of the framework's offline part (re-characterized library and flip flop model) for the desired library.

In our opinion and based on our experience, re-characterizing a library using SiliconSmart is not complex, or at least, it is less complex than performing SPICE-level simulations. Characterizing a new library can be a very complex task. However, if the user has an already characterized library at a specific voltage value (Liberty Timing File), re-characterizing it at a different voltage value using SiliconSmart is not a complex task. Using SiliconSmart, the re-characterization process is performed as follows: is i) create the new operating conditions (e.g., specify the voltage value) ii) read the existing library in liberty format and the spice netlists of the library's cells, iii) run the characterize command, and iv) write the new library specifying the CCS model. Note that all these steps are automated in the VOSsim framework and performed by executing a single script.

Furthermore, the complexity of the flip flop modeling component is insignificant. The flip flop characterization component executes a bash script that i) generates the timing violations input file, ii) runs the Spectre simulation using that file and the flip flop's SPICE netlist, and iii) gathers the results to create the lookup table. This script is executed for every voltage value and flip flop type. Regarding the time complexity of the offline part, (as mentioned in the previous section) we note that both components are scalable with respect to the number of executors (CPU threads). SiliconSmart re-characterizes the library's cells in parallel, leveraging the fact that this process is independent for every cell. Similarly, the flip flop characterization is also executed in parallel for every voltage value and/or flip flop type. Using 16 threads, the re-characterization of the Nangate 45nm library required

7 min and the characterization of all the Nangate's flip flops (16 in total) for 20 voltage values at a specific clock period required 8min. Therefore, considering that the offline part needs to be executed only once per library, its complexity (both from user and time perspective) can be considered insignificant.

The complexity of the VOSsim's online part is negligible. The circuit and voltage specific simulation script generated by VOSsim replaces Questasim's `run` command and therefore, it seamlessly extends the non-VOS gate-level simulations traditionally performed up to now. No testbench modifications are required and thus, it induces zero overhead to the user when switching from non-VOS to VOS-aware simulations. Regarding the timing complexity of VOSsim's online part, a detailed discussion is provided in Section 3.5 showing the high simulation time speedup attained by VOSsim, compared to SPICE-accurate simulations performed with the fast SPICE simulator CustomSim. Moreover, for complicity reasons, in Section 3.5, a discussion on VOSsim's simulation time overhead with respect to traditional non-VOS simulations Questasim simulations is also provided.

In addition, SPICE-level simulation also requires an offline part. Every time the technology library changes, the designer has to reproduce the SPICE-level netlist that will be simulated. Note that, the required designer effort to perform SPICE-level simulations increases significantly as the system complexity scales up [92]. Moreover, the offline part of the SPICE-level simulations needs to be performed for every new design, as well as every time a design is modified. *One of the major advantages of the proposed framework is that it extends typical digital design flows and it operates at gate-level, i.e., relieves the designer from the increased complexity of performing SPICE-level simulations.* This advantage of VOSsim (i.e., operating at gate-level) becomes even more significant during the design phase of hardware approximate accelerators, where several architectures with varying configurations are explored [41, 77].

Concluding, considering that the offline part of the proposed framework needs to be executed only once per library, while the offline part of SPICE-

Table 3.1: Benchmark Implementation Characteristics

Benchmark	Area (um^2)	Transistors Count	Flip Flop Monitored
MAC	829	5097	33
Matrix Multiplication	2279	13160	82
1-D DCT	5705	33200	128

level simulations must be executed for every new design, design configuration modification, and every time the library changes, we claim that its complexity (compared to the SPICE one) is negligible.

3.5 Experimental Evaluation

In this Section, we experimentally evaluate the accuracy of VOSsim in computing a circuit's output and power consumption when VOS is applied. Voltage reduction of 1% to 20% is considered, i.e., typical values in approximate computing [57, 92]. Examining near threshold operation is out of the scope of VOSsim and the proposed framework does not support considering external parameters (e.g., temperature) nor within-die variability in the performed simulations. Three different benchmarks are considered, i.e., a 2-stage pipeline Multiply-Accumulate (MAC), a 3-stage pipeline 3×3 matrix multiplication, and an 8-stage pipeline 1-D Discrete Cosine Transform (DCT). As shown in Table 3.1, these designs feature increased area complexity and transistor count as well as increased number of flip flops that have to be monitored by the proposed framework. *We examine stressed operating scenarios, i.e., all the designs are pipelined and synthesized-simulated at their critical path delay.* Pipelining a design may increase the number of critical paths [92] and operating at critical path delay makes the circuit more prone to small voltage decreases. Furthermore, increasing the number of flip flops increases the possibility of timing violations under VOS. The 45nm Nangate standard cell library is used in our evaluation. Nangate is

Table 3.2: The version of all the tools used in our evaluation

Tool	Version
Synopsys Design Compiler	M-2016.12
Synopsys SiliconSmart	L-2016.06
Synopsys Library Compiler	L-2016.06
Synopsys PrimeTime	M-2016.12
Cadence Spectre	14.1.0
Mentor Questasim	10.4c_5
Mentor Calibre	v2015.4 16.11
Synopsys CustomSim	M-2017.03

characterized at 1.1V and thus, the Library re-characterization component is used to characterize Nangate at 0.88V. Finally, the library's flip flops are modeled for 1%-20% voltage decrease. Using 16 threads, Library re-characterization requires 7 min, while the flip flop modeling for 20 voltage values at a specific period requires 8 min. In order to perform SPICE-level simulation we follow typical design flows as described in the Synopsys and Cadence manuals. We use i) Synopsys Design Compiler to synthesize the design and produce the gate-level netlist, ii) Mentor Calibre to produce the SPICE-level netlist from the respective gate-level one, and iii) Synopsys CustomSim to perform the SPICE-level simulation. In Spice simulations, the logical '1' threshold is set to 0.77V and the logical '0' to 0.33V. 50,000 random generated 8-bit inputs are considered for every benchmark and all experiments run on a Xeon E5-2650 server with 64GB RAM. The tools and their respective version that are used in our evaluation are summarized in Table 3.2

In order to highlight the importance of monitoring the flip flops for timing violations, we examine the possibility of their occurrences when applying VOS. In Figure 3.9 the error rate and the violation rate of the examined benchmarks are explored. As error rate, we denote the number of inputs that produce erroneous outputs over the total number of inputs when ap-

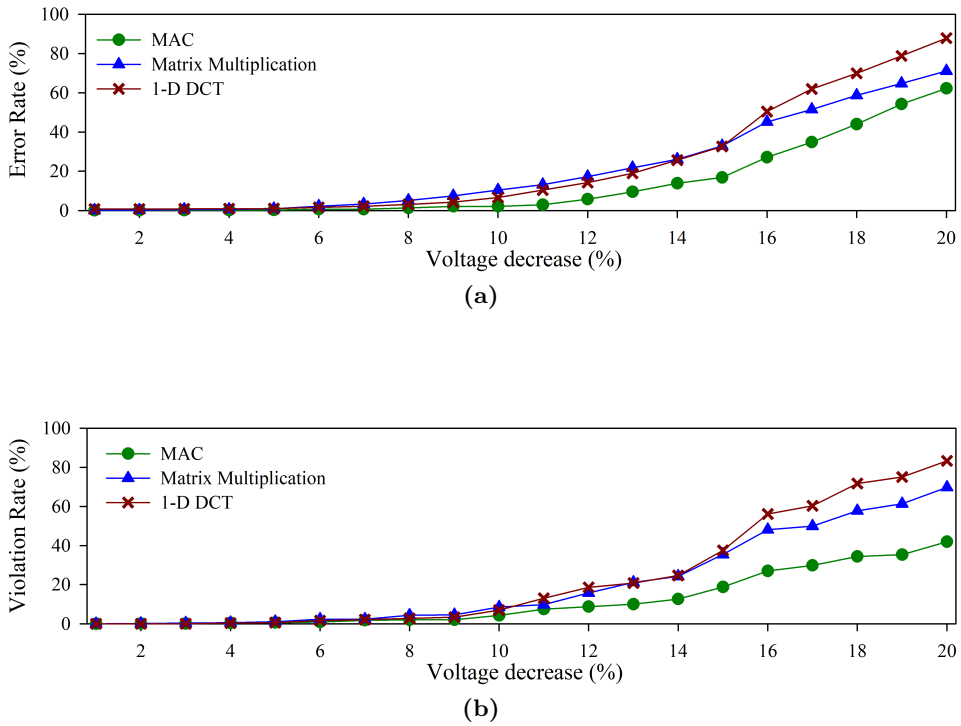
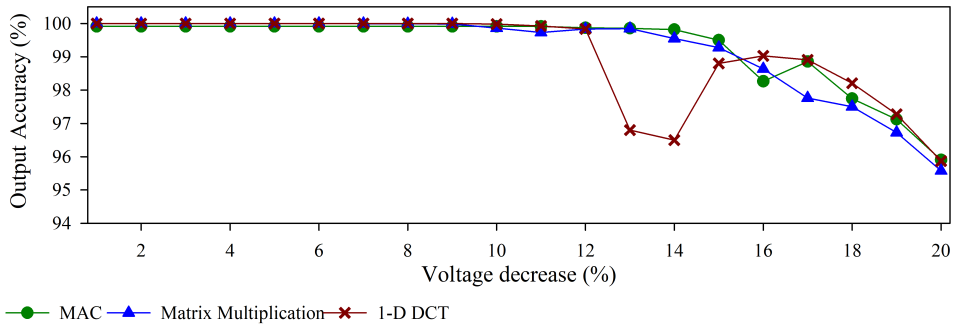


Figure 3.9: The a) error rate and b) violation rate of the examined benchmarks when applying VOS.

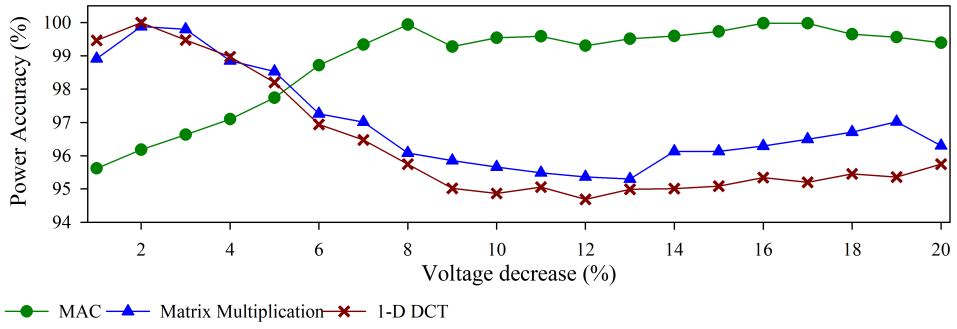
plying VOS. Similarly, as violation rate we denote the number of inputs where at least one flip flop timing violation occurs over the total number of inputs. As the voltage decreases the error and violation rate increases. Even for small voltage decrease, erroneous outputs are produced, e.g., at 1% voltage decrease the average error rate of the three benchmarks is 0.5%. At 20% voltage decrease, the average error rate is 74%. The violation rate behaves the same way as the error rate but it is possible to have erroneous outputs without having timing violations. However, it also possible to have violation rate higher than the error rate. In this case, although a timing violation occurs, an erroneous output is not necessarily produced. For example, at 13% voltage decrease the 1D-DCT benchmark features 21% violation rate, while the respective error rate is 19%, i.e., 2% smaller. Moreover, it

is also possible to have erroneous outputs without the occurrence of flip flop timing violations. For example, for all the examined benchmarks, when voltage is decreased by 2%, errors are produced but the violation rate is 0%. The high violation rate values in Figure 3.9b (e.g., 65% at 20% voltage decrease) show the significance of monitoring the circuit's flip flops for timing violations and justifies their special consideration implemented in our framework.

In Figure 3.10, we evaluate the efficacy of the proposed framework and examine its accuracy in estimating a circuit's output and power consumption when when performing VOS simulations. The output and power accuracy of VOSsim is reported with respect to the simulation outputs and power consumption obtained by performing SPICE-level simulations using CustomSim. CustomSim supports several simulation levels (described in Table 3.3) that feature varying performance as well as accuracy. To examine VOSsim's efficacy in producing SPICE accurate results, the CustomSim precision is set to its highest value (level 6). On average, for all the examined benchmarks and voltage values, the proposed framework attains 98.7% output and 98.4% power accuracy. As shown in Fig. 3.10a, for small voltage decrease the output accuracy is almost 100%, i.e., produces the same results compared to CustomSim. For voltage decrease up to 12%, where the violation rate is small, the output accuracy is more than 99%. For higher voltage decrease, the output accuracy decreases slightly, but even in the extreme case of 20% voltage decrease, where the average violation rate is 65%, the attained accuracy is more than 95%. On the other hand, the power accuracy is less affected by the voltage value and as the voltage decreases it tends to be constant for every benchmark, i.e, 99.6% for the MAC, 96.21% for the Matrix Multiplication, and 95.3% for the 1-D DCT. Therefore, the power accuracy depends more on the cells' power consumption calculation under VOS and less on the violation rate. The accuracy of the proposed framework depends on both the re-characterization of the circuit's cells (performed using the CCS model) when lowering the voltage value and on the flip flop's VOS modeling. However, although the examined benchmarks feature different number of monitored flip flops and violation rate, their accuracy values are very high and close enough. The latter shows with high confidence that the flip flop timing violations are modeled pre-



(a)



(b)

Figure 3.10: The a) output and b) power accuracy of VOSsim with respect to SPICE simulations with CustomSim (level 6).

cisely enough and that the output accuracy is more affected by the cells' delay calculation at low voltage values. Finally, Figure 3.11 presents the efficacy of VOSsim in estimating the error-power trade-off curve when applying VOS. As an error metric, we consider the Normalized Mean Error Distance (NMED) [98] and as power savings metric, the percentage power reduction. NMED is proven to be an effective metric for quantifying the accuracy of approximate circuits and also enables the evaluation of different bit-width

Table 3.3: CustomSim Simulation Levels

CustomSim Level	Description
3	Specifies a functional and timing verification of digital,memory, low-sensitivity analog, mixed-signal, and full-chip circuits.
4	Specifies a functional, timing and power verification of all circuits, especially for small current or low voltage applications.
5	Specifies an accurate timing and power simulation of all circuits, and block characterization.
6	Specifies a SPICE-like accuracy for timing and power simulation of all circuits, and cell characterization.

designs [98]. NMED is calculated as follows:

$$NMED = \frac{1}{max} \frac{\sum_{i=1}^{N} (\text{accurateOutput}_i - \text{approximateOutput}_i)}{N},$$

where N is the number of inputs (outputs) and max is the circuit's maximum possible output. accurateOutput is the circuit's output at nominal voltage and approximateOutput is the circuit's output when applying VOS. The Percentage power reduction is obtained from:

$$\text{Power Reduction}(\%) = \frac{\text{Power(accurate)-Power(approximate)}}{\text{Power(accurate)}} \times 100.$$

For every benchmark, we depict the respective error-power curves produced by simulations using CustomSim with high precision (simulation level 6) and VOSsim. As shown in Figure 3.11, for all the examined benchmarks, the curve produced by VOSsim closely follows the one produced by CustomSim. This is also confirmed by the low Mean Square Error (MSE) values delivered. The MSE (for the 20 examined voltage values) is calculated by:

$$MSE = \frac{1}{20} \sum_{\forall V} \Bigg(\left(NMED_{\text{VOSsim}}(V) - NMED_{\text{CustomSim}}(V)\right)^2 + \left(PowerReduction_{\text{VOSsim}}(V) - PowerReduction_{\text{CustomSim}}(V)\right)^2 \Bigg).$$

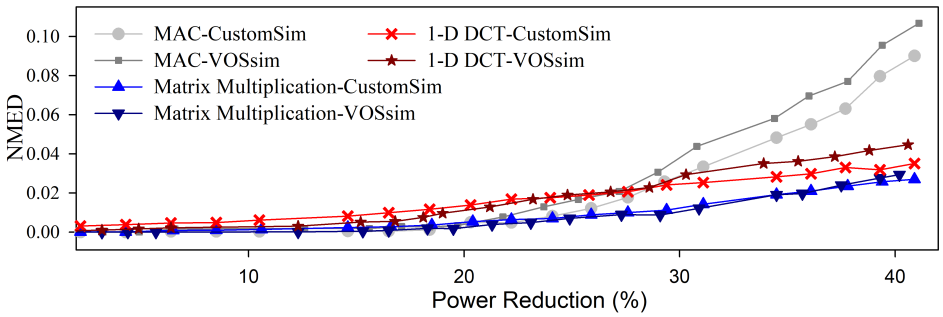


Figure 3.11: The error-power trade-off obtained by VOSsim and SPICE simulations with CustomSim (level 6).

The MSE for the MAC, Matrix Multiplication and 1-D DCT benchmarks is 0.00260, 0.00239, and 0.00245, respectively. Therefore, Figure 3.10 and Figure 3.11 show with high confidence that, for 1%-20% voltage reduction, VOSsim can be used to obtain a very accurate estimation of the error-power trade-off delivered by the VOS application on approximate computing circuits.

As shown in Figure 3.10a the 1-D DCT benchmark features the lowest average output accuracy among the examined benchmarks. Moreover, it is observed a small output accuracy decrease, i.e., 96% from 99.5%, at 13% and 14% voltage reduction. At these voltage values, for some input values, the proposed framework failed to precisely estimate circuit's VOS output, leading to lower output accuracy. As aforementioned, the proposed framework uses two models to perform the VOS-aware simulation. The first one is the CCS model and is used to re-calculate the cells' delays (and power) for the desired voltage value. The second one is the proposed flip flop model and is used to estimate a flip flop's output when a timing violation occurs. The accuracy of both models affects the final VOSsim accuracy. However, the accuracy of our framework is more affected by the cells' delays (and power) re-calculation. The flip flop model, to produce its output, considers the flip flop type, the transition type and the relative time of the timing violation. Therefore, errors at the CCS-based delay re-calculation may lead to erroneous estimations by the flip flop model. At these voltage values, the delay inaccuracies due to the scaling performed by the CCS models led to

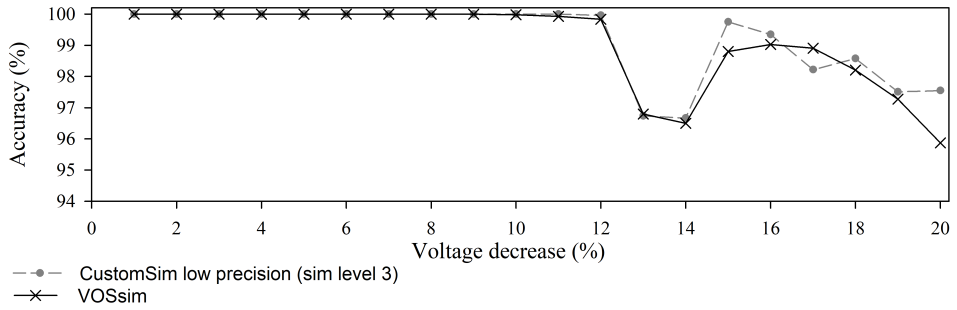


Figure 3.12: The output accuracy of VOSsim and CustomSim with low precision for the 1-D DCT benchmark.

some mispredictions by the flip flop model (with respect to high precision SPICE simulation with CustomSim). This accuracy loss is input (previous and current) and voltage depended. For this reason, we try to cover all the possible cases, by using large randomly generated input datasets and by evaluating the accuracy of VOSsim every 1% voltage decrease. Finally, it is noteworthy that identical results are obtained when performing low precision CustomSim simulations (simulation level 3). In Figure 3.12, the output accuracy of VOSsim and CustomSim with simulation level 3 is presented. As shown both curves are very close (almost identical) and exhibit the same accuracy drop at the same voltage values, i.e., the output accuracy of both drops to 96% and then increases again. At these voltage values, both gate-level simulation with VOSsim as well as low precision SPICE simulation with CustomSim fail to precisely estimate the increased accuracy of high precision (level 6) CustomSim simulations and deliver lower, but still very high, output accuracy. High accuracy SPICE simulations may increase the precision of the performed computations or use more precise models to reach convergence. Therefore, since both VOSsim and CustomSim with simulation level 3 feature lower accuracy at 13%/14% voltage decrease, it might be the case that CustomSim with level 6 used more precise models and/or computations at these voltage values to overcome convergence issues. Hence, the less accurate simulations (VOSsim and CustomSim with level 3) fail to precisely estimate this very high computations' precision, at some input values, resulting to reduced output accuracy. However, since in CustomSim's log files this information (regarding the computations' ac-

curacy and models used) is not provided, we cannot determine with high confidence that this is the case.

The efficacy of VOSsim for 21%-40% voltage decrease is also examined. Targeting to enable VOS-aware simulation at the RTL level of design abstraction, the proposed framework operates on digital standards cell libraries and uses high-level models such as the CCS and the flip flop ones. Hence, the accuracy of our framework highly depends on the accuracy of these models. As a result, since such high-level models are unable to capture the transistors' analog behavior at very low voltage values, the accuracy of the proposed framework drops significantly, especially as we approach the threshold voltage, i.e., approximating the near-threshold operating region. This is something already acknowledged by the industry, being already in a way to extend the Liberty modeling standard to support ultra-low voltage ICs. However, we must note that similar results are obtained when performing low precision simulations using CustomSim (simulation level 3). Specifically, for 21%-30% decrease, the average output accuracy of VOSsim drops to 81%, 83%, and 82% for the MAC, Matrix Multiplication, and 1-D DCT, respectively. For low precision CustomSim, the respective values are 85%, 91%, and 84%. For voltage decrease 31%-40%, the output accuracy drops even further to 53%, 34%, and 48% for VOSsim and 68%, 25%, and 46% for low precision CustomSim. As a result, in VOS simulations with voltage decrease higher than 20%, VOSsim and low precision CustomSim cannot be used as alternatives to high precision SPICE simulations (CustomSim with level 6).

Finally, we examine the efficiency of VOSsim in terms of simulation time. Figure 3.13 depicts i) the speedup of VOSsim with respect to SPICE-level simulation using CustomSim and ii) its slowdown with respect to typical gate-level simulation using Questasim. Gate-level simulation with Questasim does not support, by default, VOS and thus it is performed at the nominal voltage value. Therefore, the slowdown compared to Questasim is provided only to examine the overhead induced by monitoring the circuit's flip flops for timing violations. To evaluate the performance of VOSsim, we examine the attained speedup compared to both high as well as lower precision CustomSim simulations. The reported speedups refer to the aver-

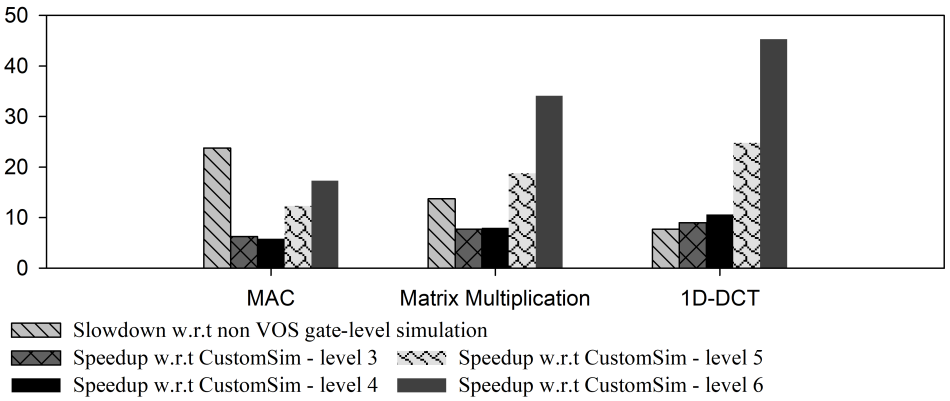


Figure 3.13: The average speedup/slowdown attained by the proposed framework compared to CustomSim Simulation/non-VOS Questasim simulation.

age speedup over all the examined voltage values. During the experimental evaluation, we observed a minor time variation for both the CustomSim and VOSsim simulation time. However, this time variation is not proportional to the voltage decrease. In other words, as the voltage value decreases, the simulation time does not necessarily increase. The simulation time coefficient of variation is 2.6% on average for the CustomSim simulations and 4.4% for the VOSsim ones. The observed time variations are attributed to the context switch of the running processes by the operating system and to compensate for this induced random delay, the simulations were repeatedly evaluated. As the simulation time of VOSsim is significantly smaller than the CustomSim's one and as VOSsim operates over several tools (that are executed at every simulation), its simulation time is slightly more affected. For every voltage value, the experiments are conducted 3 times and their average value is the speedup obtained for this voltage value. As shown, the proposed framework introduces an average overhead of 15× slower simulation compared to the non-VOS typical Questasim simulation. However, as the circuit complexity increases, this slowdown decreases (even though more flip flops are monitored). This is explained by the fact that for more complex circuits the time spent at monitoring for timing violations becomes less significant with respect to the total simulation time. Compared to high precision CustomSim simulations (level 6), VOSsim attains an average

32× speedup, while compared to low precision ones (level 3), the average speedup is 7.6×. As the circuit's transistor count increases, the CustomSim simulation becomes significantly slower and thus, VOSsim achieves higher speedups. The simulation of the MAC, Matrix Multiplication, and 1-D DCT with VOSsim requires 30 min, 46 min, and 76 min, respectively. Compared to high precision CustomSim simulations, the delivered speedup scales from 17× for a small circuit as the MAC, to 45× for a quite larger one as the 1-D DCT. Setting the CustomSim simulation level to 3 or 4, results in significantly faster execution compared to level 6 but reduced output and power accuracy. For the examined benchmarks, simulation level 3 and 4 produced similar results, regarding both the accuracy and simulation time. For 1%-20% voltage decrease, compared to CustomSim with level 3 or 4, VOSsim delivers more than 10% higher power accuracy (lower precision CustomSim considers less circuit detail) and similar output accuracy, i.e., within 2%. For higher voltage decrease (21%-40%), both VOSsim and CustomSim with level 3 or 4 deliver low output accuracy but still VOSsim attains significantly higher power accuracy. Finally, for 1%-40% voltage decrease, CustomSim with simulation level 5 delivers very high output and power accuracy (higher than VOSsim). The average speedup of VOSsim compared to CustomSim with level 3 is 6.0×, 7.7×, and 9.0× for the MAC, Matrix Multiplication, and 1-D DCT benchmarks, respectively. The respective values for VOSsim compared to CustomSim with level 4 are 5.8×, 8.0×, and 10.6×. Compared to CustomSim with simulation level 5, VOSsim achieves an average speedup of 12.3×, 18.8×, and 24.8×, respectively. Therefore, even when we lower the simulation precision of CustomSim, we still obtain significant simulation speedup. Moreover, in the case of simulation level 3 or 4 VOSsim attains higher power accuracy and comparable output accuracy for more than 5.8× speedup. We would like to underline that CustomSim is a Fast SPICE simulator and this is the reason why we compare VOSsim with CustomSim and not with Spectre or HSPICE. With respect to Spectre simulation, the delivered speedup by VOSsim is more than 100× on average.

Concluding, compared to SPICE-level simulation, VOSsim delivers very high simulation time speedup for very accurate output and power estimation. The accuracy of VOSsim does not depend on the circuit's complexity

and the speedup is expected to scale significantly for larger designs. However, its output accuracy depends on the voltage decrease, dropping from 99% to 95% for voltage decrease scaling from 12% to 20%. For very low voltage values, VOSsim fails to capture the transistors behavior, delivering poor output accuracy.

3.6 Conclusion

Approximate computing emerges as a promising paradigm for energy efficient design by aggressively decreasing power consumption of inherently error resilient applications. Recent research has pointed out the synergistic nature of Votage Over-Scaling (VOS) with other approximate techniques. Approximate techniques from the logic and algorithmic layers produce simpler circuits and can reduce the circuit's delay and decrease the number of critical paths. Hence, applying VOS, can further reduce the power consumption, at the cost of a small error increase, as fewer paths are affected by the voltage decrease. However, the lack of a methodology that rapidly quantifies the errors and power savings due to VOS (traditionally performed SPICE-level simulations are very time consuming), restricts it exploitation in the respective approximate computing works. In this chapter, we propose VOSsim, a framework that can be seamlessly integrated in typical hardware design flows and enables very fast VOS-aware simulation. One of the major advantages of the proposed framework is that it extends typical digital design flows and it operates at gate-level, i.e., relieves the designer from the increased complexity of performing SPICE-level simulations. Compared to SPICE simulations, for 1%-20% voltage decrease, VOSsim delivers very high output and power accuracy (more than 98% on average) and reduces the simulation time from 6× to 45×. All the reported experimental evaluations examine stressed operating conditions, i.e., pipelined designs and operation at critical path delay, in order to evaluate the accuracy of proposed framework at a worst-case scenario. At very high voltage decrease (more than 20%), the accuracy of VOSsim is very sensitive to small inaccuracies originated by the cell delay re-calculation under VOS; leading to a notable output accuracy drop at very low voltage values. However, we underline

that although VOSsim uses the CCS and the proposed flip flop models, it is not bound to them. The models used by the proposed framework can be easily changed without any particular modifications in the rest of the components. Therefore, if a more accurate power-delay model and/or a more sophisticated flip flop model is published, they can be straightforwardly used by VOSsim in order to improve its accuracy.

Chapter 4

Efficient Approximate Multiplication Circuits Through Partial Product Generation Approximation

Approximate computing has received significant attention as a promising strategy to decrease power consumption of inherently error tolerant applications. In this chapter, we focus on hardware level approximation by introducing the Partial Product Perforation technique for designing approximate multiplication circuits. We prove in a mathematically rigorous manner that in partial product perforation the imposed errors are bounded and predictable, depending only on the input distribution. Through extensive experimental evaluation, we apply the partial product perforation method on different multiplier architectures and expose the optimal architecture–perforation configuration pairs for different error constraints. We show that, compared with the respective exact design, the partial product perforation delivers reductions of up to 50% in power consumption, 45% in area and 35% in critical delay. Also, the product perforation method is compared with state-of-the-art approximation techniques, i.e. truncation, Voltage Over-Scaling and logic approximation, showing that it outperforms them in terms of power dissipation and error. This chapter is based on our publications in [99–101].

4.1 Introduction

In modern embedded electronic devices, power consumption is a first class design concern. Considering that a large number of application domains are inherently tolerant to imprecise calculations, e.g. Digital Signal Processing (DSP), data analytics and data mining [32] approximate computing appears as a promising solution to reduce their power dissipation. Such applications a) process large redundant data sets or noisy input data derived from the real world, b) do not have a “golden" result, c) perform statistical/probabilistic computations and/or d) demand human interaction, thus, their exactness is relaxed due to limited human perception [38, 67]. Approximate computing can be applied at both software and hardware level.

Hardware level approximation mainly targets arithmetic units, such as adders and multipliers widely used in portable devices to implement multimedia algorithms, e.g., image and video processing. Both general-purpose approximation techniques [56–58] applied to any arithmetic circuit, as well as circuit-specific approximation either to adder [59–61] or multiplier designs [62–66], have been presented. The most commonly used techniques for the generation of approximate arithmetic circuits are truncation [56,69], Voltage Over-Scaling (VOS) [57, 67] and simplification of logic complexity (i.e., alteration of the truth table) [61–63]. Regarding to the general approximation techniques, VOS [57, 67] and truncation [56, 68, 69] have been proposed. VOS is applied in any circuit by lowering the supply voltage below its nominal value. Decreasing the supply voltage reduces the circuit's power consumption, but produces errors caused by the number of paths that fail to meet the delay constraints [67]. In [69], a truncated multiplier with a constant correction term is proposed, significantly decreasing the error imposed by typical truncation. [56] proposed a truncated multiplier with variable correction that outperforms [69] in terms of error. Extensive research has been conducted on approximate adders [57, 59–61] providing significant gains in terms of area and power while exposing small error. Although the authors propose the use of such adders targeting to build approximate multipliers, it is not clear how they can be used in different tree architectures and how their error scales in the case of multi-operand addition. Despite the

extensive research on approximate adders, research activities on approximate multipliers are still very limited. Efficient approximate multipliers introduced in [62–64, 68] target the approximation of the partial product accumulation but do not examine approximations on the partial product generation. Targeting, approximate accumulation of the partial products, Momeni et al. [63] presented two approximate 4:2 compressors by modifying the respective accurate truth table. These compressors are used in a Dadda tree with 4:2 reduction and different multiplier architectures were not explored. In all the proposed approximate multipliers, the imposed error cannot be predicted as it depends on carry propagation and the circuits' implementation and requires simulations over all possible inputs in order to be calculated Targeting to tackle the limited research on approximate multipliers, in this chapter, we design of power–error efficient multiplication circuits. We differentiate from previous works by exploring approximation on the generation of the partial products. The proposed method can be easily applied in any multiplier architecture without the need of a special design, in contrast to related works. In addition, the error imposed by perforation depends only on the configuration parameters and, in contrast to existing work, can be analytically calculated without the need of exhaustive simulations. The latter is critical as, given the input distribution of the application, a precise estimation of the output quality can be extracted. Finally, the a priori knowledge of the induced error enables the selection of the optimal configuration in order maximize the power savings with respect to the error bound and input distribution.

Approximate hardware circuits, contrary to software approximations, offer transistors reduction, lower dynamic and leakage power, lower circuit delay and opportunity for down-sizing. Motivated by the limited research on approximate multipliers, compared to the extensive research on approximate adders, and explicitly the lack of approximate techniques targeting the partial product generation, we introduce the Partial Product Perforation method for creating approximate multipliers. Inspired from [102], we omit the generation of some partial products, thus, reducing the number of partial products that have to be accumulated, we decrease the area, power and depth of the accumulation tree. The major contributions of this work are summarized as follows:

- We adopt and apply, for the first time, the software based perforation technique [102], on the design of hardware circuits, obtaining optimized design solutions regarding the power–area–error trade-offs.

- We analyze in a mathematically rigorous manner the arithmetic accuracy of partial product perforation and prove that it delivers a bounded and predictable output error. Our error analysis is not bound to a specific multiplier architecture and can be applied with error guarantees to every multiplication circuit regardless of its architecture. Such a rigorous analysis enables precise error estimation over input data distributions.

- We explore and characterize the efficiency of the product perforation method on several multiplier schemes exposing its power–area impact on different architectures. This is the first time that such an exploratory analysis over different approximate multiplier architectures is offered to the designer, enabling also, the selection of the optimum architecture–perforation configuration for given error constraints.

- We show that partial product perforation outperforms related state-of-the-art works in terms of power consumption and error, as well as output quality, when applied to image processing and data analytics algorithms.

More specifically, we apply the partial product perforation on 16 different multiplier architectures, using industrial strength tools, i.e. Synopsys Design Compiler and PrimeTime. Through extensive experimental evaluation, we present the optimal approximate multiplier configurations for various error constraints. We show that, compared to the accurate multiplier, product perforation offers reductions of up to 50% in power consumption, 45% in area and 35% in critical delay for 0.1% normalized mean error distance [98]. Moreover, it is compared with state-of-the-art approximate computing works that use either VOS [57], logic approximation [63], or truncation [56], outperforming them significantly in terms of power dissipation and error. Finally, we examine the scalability of our technique by applying it on different bit-width multipliers and show that the delivered

savings increase with the width increase.

The rest of this chapter is organized as follows: Section 4.2 introduces the partial product perforation technique providing the corresponding error analysis error and error correction methods. In Section 4.3, we examine product perforation on different multiplier architectures, exposing the optimal architecture–perforation configuration pairs under differing error constraints. Finally, Section 4.4 evaluates the product perforation method by comparing it with related state-of-the-art works and Section 4.5 concludes this chapter.

4.2 Analyzing Partial Product Perforation

4.2.1 Method Analysis

In this section, the partial product perforation method for the design of approximate hardware multipliers is described. Consider two n-bit numbers A and B. The result of their multiplication $A \times B$ is obtained after summing all the partial products Ab_i, where b_i is the i^{th} bit of B. Thus,

$$A \times B = \sum_{i=0}^{n-1} Ab_i 2^i,\ b_i \in \{0,1\}. \tag{4.1}$$

The partial product perforation technique omits the generation of k successive partial products starting from the j^{th} one. A perforated partial product is not inserted in the accumulation tree and hence, n full adders can be eliminated. Applying product perforation with j and k configuration values on the multiplication $A \times B$ produces the approximate result

$$A \times B|_{j,k} = \sum_{\substack{i=0,\\ i \notin [j,j+k)}}^{n-1} Ab_i 2^i,\ b_i \in \{0,1\}. \tag{4.2}$$

Note that $j \in [0, n-1]$ and $k \in [1, \min(n-j, n-1)]$.

Similarly, when Modified Booth Encoding (MBE) [103] is used for generating the partial products, the result of the approximate multiplication is given by:

$$A \times B|_{j,k} = \sum_{\substack{i=0 \\ i \notin [j,j+k)}}^{n/2-1} A\mathbf{b_i^{MB}}4^i,\ \mathbf{b_i^{MB}} \in \{0, \pm 1, \pm 2\}. \tag{4.3}$$

Figure 4.1 depicts an example of applying the partial product perforation method on different 8-bit multipliers with j=2 and k=2 configuration values. For each architecture, the dot diagrams [103] of the accurate and the respective perforated tree are presented. The "dots" represent the bits of the partial products that have to be accumulated, while the "stages" the delay of the reduction process followed by each tree. The dashed boxes with four dots are 4:2 compressors, those with three are full adders and those with two are either full- or half-adders. Through the proposed approximation technique, the power, area and delay of the multiplication circuit are decreased, making though the computation imprecise. The higher the order of a perforated partial product, the greater the error imposed at the final result. Also, since the addition is an associative and commutative operation, when more than one partial products are perforated, the total error results from the addition of the errors produced from the perforation of each partial product separately.

We use **the notation D[*j*,*k*,*c*]** to label the different approximate multiplier architectural configurations. The parameter "D" refers to the tree architecture, j is the order of the first perforated partial product and k the number of the perforated partial products. If no j and k are specified, the respective notation refers to the exact design. Finally, c corresponds to the partial product generation technique and takes the values "s" for Simple Partial Products (SPP) or "m" for MBE. For example, Figure 4.1a depicts the array[s] configuration, while Figure 4.1b the array[2,2,s].

Partial product perforation should not be confused with the truncation technique. Truncation eliminates the circuit that produces specific least significant bits (LSB) of the accumulation tree, while perforation skips the

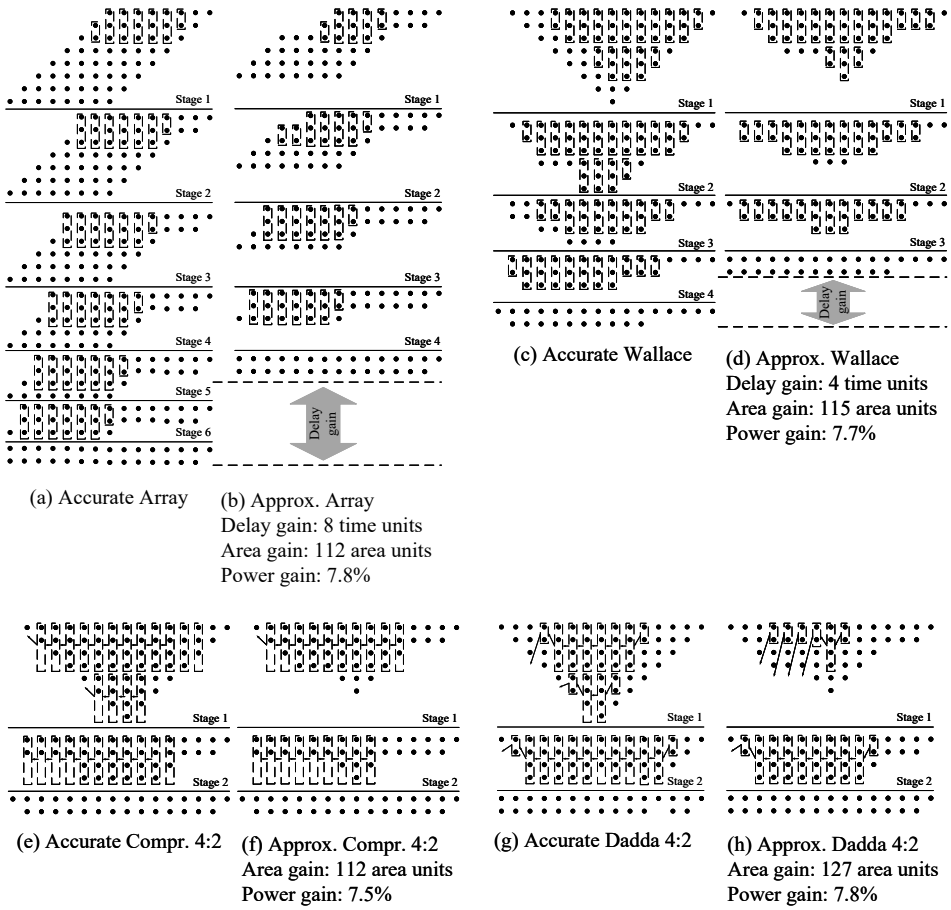


Figure 4.1: The partial product reduction process for 8×8 multiplication with a) Accurate Array, b) Approx. Array c) Accurate Wallace, d) Approx. Wallace, e) Accurate Compressor 4:2, f) Approx. Compressor 4:2, g) Accurate Dadda 4:2 h) Approx. Dadda 4:2. Approximation is performed by perforating the 3^{rd} and 4^{th} partial products. The boxes with 4 dots are 4:2 compressors, those with 3 are full adders and those with 2 are full or half adders.

generation of partial products and thus, decreases the number of operands to be accumulated. For example, in an 8-bit array multiplier, perforating a partial product removes 8 full adders from the accumulation tree and re-

duces its delay. In order to attain similar circuit reduction using truncation, the 6 LSB have to be truncated. However, truncating the 6 LSB does not offer any delay reduction. Moreover, in this example, truncation delivers in all cases incorrect results, whereas the outputs of perforation are 50% correct. Finally, perforating one partial product (out of eight) results in a 12.5% loss of information while truncating the 6 LSB (out of 16) results in a 37.5% information loss. In Section 4.4, the perforation and truncation techniques are quantitatively compared in greater detail regarding error and power metrics, in order to further expose their differences.

4.2.2 Error Analysis

A critical issue for the approximate computing is the error imposed during computations and how it affects the final result. In this section, an error evaluation analysis of the partial product perforation technique is presented. We evaluate error utilizing the error metrics proposed in [98], i.e., Error Distance (*ED*), Mean Error Distance (*MED*) and Normalized *MED* (*NMED*), as effective metrics for quantifying the accuracy of approximate arithmetic circuits. *ED* is defined as the absolute distance of the fully accurate product P and the approximate one P':

$$ED = |P - P'|.$$

The *MED* is the average of *EDs* for all inputs:

$$MED = \frac{1}{M} \sum_{i=1}^{M} |P_i - P'_i|,$$

where M is the number of inputs.

The Normalized Mean Error Distance ($NMED$) is defined as the ratio:

$$NMED = \frac{dm}{D} \quad [98],$$

where $dm = MED$ and D is the maximum possible error that an approximate circuit can produce. Therefore in the case of $n \times n$ multiplication, the

maximum possible error is $(2^n - 1) \times (2^n - 1)$. For example, consider an approximate multiplier that always returns zero. In this case multiplying $(2^n - 1)$ by $(2^n - 1)$ will return 0 and therefore $ED = (2^n - 1)^2$ and as a result, $D = (2^n - 1)^2$ Hence, in the case of n-bit multiplication $NMED$ is calculated by:

$$NMED = \frac{MRED}{(2^n - 1)^2},$$

Finally, the Relative Error Distance (RED) is defined as:

$$NMED = \frac{ED}{P},$$

and the Mean RED ($MRED$) is obtained by:

$$MRED = \frac{1}{M} \sum_{i=1}^{M} \frac{ED_i}{Pi} \quad [64].$$

In this chapter, we try to provide a broad analysis for approximate multiplication circuits focusing on more than one error metrics, both for completeness but also to enable direct comparison with previously published related research. Note that an error metric may be optimal for an application domain but suboptimal for another one. More specifically, in Section 4.2.2 we provide error analysis both on $NMED$ and $MRED$. In Section 4.3, where we explore the efficiency of partial product perforation technique (Figure 4.6), we use the $NMED$ metric in order to be in compliance with the error metrics used in [63] and [98]. However, the respective $MRED$ values of Figure 4.6 can be straightforwardly derive using the provided designs configuration. Finally, in Section 4.4, where we compare partial product perforation with state-of-the-art works, we provide an extensive discussion considering both the $NMED$ and $MRED$ values of all the examined approximate technique.

Error Evaluation

When applying the product perforation on a n-bit multiplier using SPP generation, the ED of multiplying two numbers A, B is calculated as follows:

$$\begin{aligned} ED(A,B) &= \left|P - P'\right| \\ &= A\sum_{i=0}^{n-1} b_i 2^i - A \sum_{\substack{i=0,\\ i\notin[j,j+k)}}^{n-1} b_i 2^i \\ &= A \sum_{i=j}^{j+k-1} 2^i b_i \\ &= A 2^j x_B, \end{aligned} \tag{4.4}$$

where $x_B \in [0, 2^k)$ and

$$x_B = \sum_{i=0}^{k-1} 2^i b_{j+i} = \lfloor B/2^j \rfloor \bmod 2^k. \tag{4.5}$$

If p_A and p_B are the probability density functions of A and B, respectively, then the MED is calculated from:

$$MED = \sum_{\forall A,B} p_A(A) p_B(B) ED(A,B). \tag{4.6}$$

Without loss of generality, the rest of our analysis considers a uniform distribution over the overall n-bit numbers, i.e., $(A,B) \in [0,2^n)^2$. Hence, $p_A(A) = 1/2^n \ \forall A$ and $p_B(B) = 1/2^n \ \forall B$. Therefore, MED is given from:

$$\begin{aligned} MED &= \sum_{\forall A,B} \frac{ED(A,B)}{2^n 2^n} \\ &= \frac{1}{2^{2n}} \sum_{\forall A} \sum_{\forall B} ED(A,B). \end{aligned} \tag{4.7}$$

Assuming that ED_A is the sum of $EDs\ \forall B$ for a given A, then:

$$\begin{aligned} ED_A &= \sum_{\forall B} ED(A,B) \\ &= 2^{n-k} \sum_{\forall x_B} x_B 2^j A \\ &= \frac{2^n 2^j (2^k-1)A}{2} \end{aligned} \tag{4.8}$$

and the sum of all EDs is:

$$\begin{aligned} \sum_{\forall A} ED_A &= \sum_{\forall A} \frac{2^n 2^j (2^k-1)A}{2} \\ &= \frac{2^n 2^j (2^k-1)}{2} \Big(\sum_{A=0}^{2^n-1} A \Big) \\ &= \frac{2^j 2^{2n} (2^k-1)(2^n-1)}{4}. \end{aligned} \tag{4.9}$$

Using (4.9), (4.7) equals:

$$\begin{aligned} MED &= \frac{2^j 2^{2n} (2^k-1)(2^n-1)}{2^{2n} 4} \\ &= \frac{2^j (2^k-1)(2^n-1)}{4}. \end{aligned} \tag{4.10}$$

Thus,

$$NMED = \frac{MED}{(2^n-1)^2} = \frac{2^j (2^k-1)}{4(2^n-1)}. \tag{4.11}$$

Similarly,

$$RED(A,B) = \frac{ED(A,B)}{A \times B} = \frac{x_B 2^j}{B} \tag{4.12}$$

and

$$MRED = \frac{2^n}{2^{2n}} \sum_{\forall B} \frac{x_B 2^j}{B} = \sum_{\forall B} \frac{x_B 2^j}{2^n B}. \tag{4.13}$$

Previous analysis provide rigorous expressions of error metrics, enabling fast error analysis of differing product perforation configurations. As shown later in Section 4.3, these analytical error expressions are used in an exploration loop for deriving optimized approximate design solutions. The analytical equations (4.11) and (4.13) consider uniform distribution, thus in case of differing distributions[1] they should be adjusted according to the new probability density functions (PDF), since the power–error efficiency of approximate designs highly depends on the multiplier's operands distribution. Using (4.6) and the input distribution we can calculate the configurations (j, k) that minimize the error for the respective distribution. In most applications, e.g. multimedia, the inputs are highly correlated [58]. In order to provide an initial insight on how the error metrics scale when the inputs are correlated (such as in multimedia applications [58]), we assume the case of differing approximate multipliers adopting correction Method 1 (see Section 4.2.2) with inputs A and B following the same distribution. In Table 4.1, we evaluate the $NMED$ and $MRED$ metrics when considering various Normal distributions ($N(\mu, \sigma)$) as inputs, usually characterizing the multimedia applications [58]. For comparison reasons, we also provide the respective error metrics when the inputs follow a uniform distribution (i) over all possible inputs and (ii) in the segment $[\mu - 3\sigma, \mu + 3\sigma]$. Table 4.1 shows that the $NMED$ and $MRED$ error metrics are highly affected by (i) the type of the distribution, i.e. the input normal distribution derives lower error values than the uniform, (ii) the distribution's mean, and (iii) in case of uniform distribution, the distribution's range. As an intuitive example, Figure 4.2a depicts the power–*NMED* Pareto graph for a 16-bit Dadda 4:2 multiplier when A, B follow the uniform distribution over the overall range of n-bit numbers, while Figure 4.2b presents the same graph with inputs derived from the GSM 06.10 audio benchmark [104]. As shown, increasing k values result to lower power consumption but increased error values, while the selection of the j value mostly depends on the input distribution. Intuitively, for a uniform distribution over all possible n-bit numbers (Figure 4.2a), where all the bits have equal probability of being one or zero, j should be kept small to minimize the error. This is also confirmed from Figure 4.2a where the 58% of the Pareto configurations feature $j = 0$ and

[1]In case of different input distributions, starting from equation (4.6) we apply the same steps given the respective PDFs of the input operands.

Table 4.1: Evaluation of the $NMED$ and $MRED$ for Correlated Input Distributions

j	k	Distribution	NMED	MRED
0	5	Uniform in [0, 65535]	7.762×10^{-5}	1.640×10^{-3}
0	5	N(4791, 30)	1.128×10^{-5}	2.110×10^{-3}
0	5	Uniform in [4701, 4881]	1.093×10^{-5}	2.045×10^{-3}
0	5	N(61234, 300)	1.443×10^{-4}	1.652×10^{-4}
0	5	Uniform in [60334, 62034]	1.455×10^{-4}	1.667×10^{-4}
1	3	Uniform in [0, 65535]	3.338×10^{-5}	7.228×10^{-4}
1	3	N(32767, 7000)	3.320×10^{-5}	1.396×10^{-4}
1	3	Uniform in [11767, 53767]	3.338×10^{-5}	1.583×10^{-4}
6	2	Uniform in [0, 65535]	4.281×10^{-4}	6.310×10^{-3}
6	2	N(19250,17)	1.230×10^{-5}	1.501×10^{-4}
6	2	Uniform in [19199, 19301]	4.205×10^{-5}	4.859×10^{-4}
6	4	Uniform in [0, 65535]	2.384×10^{-3}	3.161×10^{-2}
6	4	N(48383,255)	2.555×10^{-3}	4.682×10^{-3}
6	4	Uniform in [47618, 49148]	4.566×10^{-3}	8.370×10^{-3}

the 42% $j = 1$. However, as presented in Figure 4.2b, when the inputs are correlated without following a uniform distribution, we observe that the Pareto front is formed by configurations featuring many different j values, i.e., 0, 2, 6, and 15. Previous example shows that there is not a "golden" value for the j and k perforation variables but their selection highly depends on the error constraints and the inputs PDF. Thus, (4.6) should be used to extract the appropriate value for j and k.

Error Correction Methods

In this section, we introduce two methods to decrease the error induced from the application of partial product perforation. They are implemented

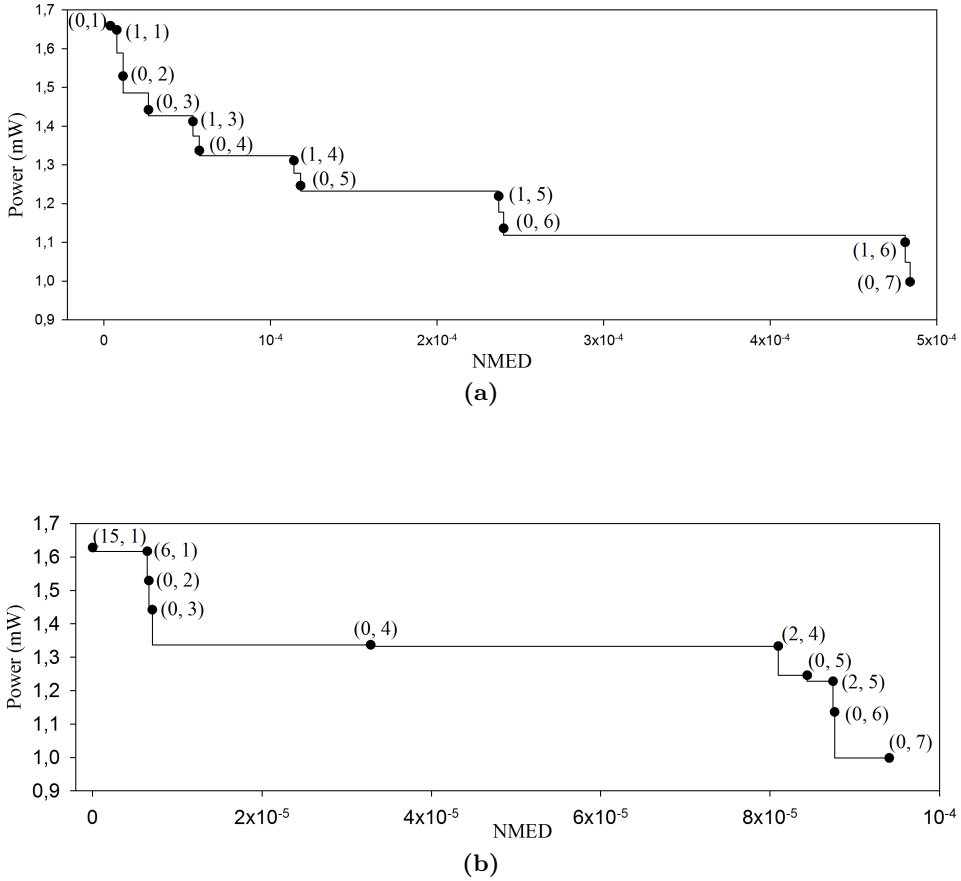


Figure 4.2: The Pareto power–*NMED* graph of a 16-bit Dadda 4:2 multiplier with a) uniform input distribution in $[0, 2^{16})$ and b) inputs obtained from audio benchmarks. All the configurations that feature $NMED < 5 \times 10^{-5}$ are presented. Next to each point is denoted the respective $(j,\ k)$ configuration.

as extra components complementing the multiplication circuit, thus their area, power and delay overheads as well as the error reduction they offer, do not depend on the architecture of the multiplier. Although multiplication is commutative, i.e. $A \times B = B \times A$, this does not apply in perforated multipliers. From (4.4), when multiplying $A \times B$, the imposed error is proportional to the multiplicand A and the term x_B and thus, decreasing one of these operands decreases the error delivered to the output. As a result, comparing A,B or x_A, x_B before the multiplication and swapping accordingly A,B can reduce the error.

- **Method 1**: Comparing x_A, x_B

In this method x_A, x_B are compared before the multiplication and, if $x_B > x_A$, A and B are swapped. Therefore, the imposed error is $ED(A,B) = A2^j x_B$, when $x_A \geq x_B$, and $ED(A,B) = B2^j x_A$, when $x_B > x_A$. Hence, MED equals:

$$\begin{aligned} MED &= \sum_{\forall A,B} p_A(A)p_B(B)ED(A,B) \\ &= 2^j\Big(\sum_{\substack{\forall A,B: \\ x_A \geq x_B}} p_A(A)p_B(B)x_B A + \sum_{\substack{\forall A,B: \\ x_A < x_B}} p_A(A)p_B(B)x_A B\Big). \end{aligned} \tag{4.14}$$

If A, B follow the uniform distribution in $[0, 2^n)$ (4.14) equals:

$$\begin{aligned} MED &= 2^j\Big(\sum_{\substack{\forall A,B: \\ x_A \geq x_B}} \frac{x_B A}{2^n 2^n} + \sum_{\substack{\forall A,B: \\ x_A < x_B}} \frac{x_A B}{2^n 2^n}\Big) \\ &= \frac{2^j}{2^{2n}}\Big(\sum_{\substack{\forall A,B: \\ x_A = x_B}} x_B A + 2 \sum_{\substack{\forall A,B: \\ x_A < x_B}} x_A B\Big). \end{aligned} \tag{4.15}$$

Every number A can be written in the form:

$$A = M_A 2^{j+k} + x_A 2^j + L_A,$$

where $M_A \in [0, 2^{n-(j+k)})$, $x_A \in [0, 2^k)$ and $L_A \in [0, 2^j)$.
M_A and L_A are computed similarly to x_A.

The sum ($S1(y)$) of all numbers A that have $x_A = y$, where y is a constant and $y \in [0, 2^k)$, is given by:

$$\begin{aligned} S1(y) &= \sum_{\substack{\forall A: \\ x_A = y}} A \\ &= \sum_{\substack{\forall A: \\ x_A = y}} \left(M_A 2^{j+k} + x_A 2^j + L_A\right) \\ &= \sum_{\forall M_A} \sum_{x_A = y} \sum_{\forall L_A} \left(M_A 2^{j+k} + x_A 2^j + L_A\right) \\ &= 2^j \frac{(2^{n-(j+k)} - 1)2^{n-(j+k)}}{2} 2^{j+k} + \\ &+ 2^{n-(j+k)} 2^j y 2^j + \\ &+ 2^{n-(j+k)} \frac{(2^j - 1)2^j}{2}. \end{aligned} \tag{4.16}$$

Supposing that B is fixed and $x_B = z$, we get that:

$$\begin{aligned} 2 \times \sum_{\substack{\forall A: \\ x_A < z}} x_A B &= 2^{n-k} 2B \sum_{x_A < z} x_A \\ &= 2^{n-k} z(z-1)B \end{aligned} \tag{4.17}$$

and

$$\begin{aligned} \sum_{\substack{\forall A: \\ x_A = z}} zA &= z \sum_{\substack{\forall A: \\ x_A = z}} A \\ &= zS1(z). \end{aligned} \tag{4.18}$$

By evaluating (4.17) for all B, we obtain:

$$\begin{aligned} 2 \sum_{\substack{\forall A,B: \\ x_A<x_B}} x_A B &= \sum_{\forall B} 2^{n-k} z(z-1)B \\ &= 2^{n-k} \sum_{z=0}^{2^j-1} z(z-1)S1(z). \end{aligned} \tag{4.19}$$

By evaluating (4.18) for all B, we obtain:

$$\begin{aligned} \sum_{\substack{\forall A,B: \\ x_A=x_B}} x_B A &= \sum_{\forall B} x_B S1(x_B) \\ &= 2^{n-k} \sum_{z=0}^{2^j-1} zS1(z). \end{aligned} \tag{4.20}$$

Using (4.19) and (4.20), (4.15) is equal to:

$$MED = \frac{2^j 2^{n-k}}{2^{2n}} \Big(\sum_{z=0}^{2^j-1} z^2 S1(z) \Big) \tag{4.21}$$

$$\text{and } NMED = \frac{2^j 2^{n-k}}{2^{2n}(2^n-1)^2} \Big(\sum_{z=0}^{2^j-1} z^2 S1(z) \Big). \tag{4.22}$$

The sum of all *REDs* is given by:

$$\begin{aligned} \sum_{\forall A,B} RED(A,B) &= 2^j \Big(\sum_{\substack{\forall A,B: \\ x_A \geq x_B}} \frac{x_B}{B} + \sum_{\substack{\forall A,B: \\ x_A<x_B}} \frac{x_A}{A} \Big) \\ &= 2^j \Big(\sum_{\substack{\forall A,B: \\ x_A=x_B}} \frac{x_B}{B} + 2 \sum_{\substack{\forall A,B: \\ x_A>x_B}} \frac{x_B}{B} \Big). \end{aligned} \tag{4.23}$$

Denoting $CI = 2^k - 1$ and using that

$$\begin{aligned}\sum_{\substack{\forall A,B:\\ x_A > x_B}} \frac{x_B}{B} &= \sum_{\forall B} \sum_{\substack{\forall A:\\ x_A > x_B}} \frac{x_B}{B} \\ &= \sum_{\forall B} \Big(\frac{x_B}{B} 2^{n-k} (CI - x_B) \Big)\end{aligned} \tag{4.24}$$

and

$$\begin{aligned}\sum_{\substack{\forall A,B:\\ x_A = x_B}} \frac{x_B}{B} &= \sum_{\forall B} \sum_{\substack{\forall A:\\ x_A = x_B}} \frac{x_B}{B} \\ &= \sum_{\forall B} \left(\frac{x_B}{B} 2^{n-k} \right),\end{aligned} \tag{4.25}$$

(4.23) is equal to:

$$\begin{aligned}\sum_{\forall A,B} RED(A,B) &= 2^j 2^{n-k} \sum_{\forall B} \left(\frac{x_B}{B} \Big(1 + 2(CI - x_B) \Big) \right) \\ &= 2^j 2^{n-k} \sum_{B=1}^{2^n - 1} \left(\frac{x_B}{B} \Big(1 + 2(CI - x_B) \Big) \right)\end{aligned} \tag{4.26}$$

and *MRED* is calculated as a relation of j and k from:

$$MRED = \frac{2^j}{2^{n+k}} \sum_{B=1}^{2^n - 1} \left(\frac{x_B}{B} \Big(1 + 2(CI - x_B) \Big) \right). \tag{4.27}$$

- **Method 2**: Comparing *A,B*

In this method A, B are compared before the multiplication and, if $A > B$, A and B are swapped. As a result the induced error $ED(A,B) = A2^j x_B$, when $A \leq B$ and $ED(A,B) = B2^j x_A$, when $A > B$.

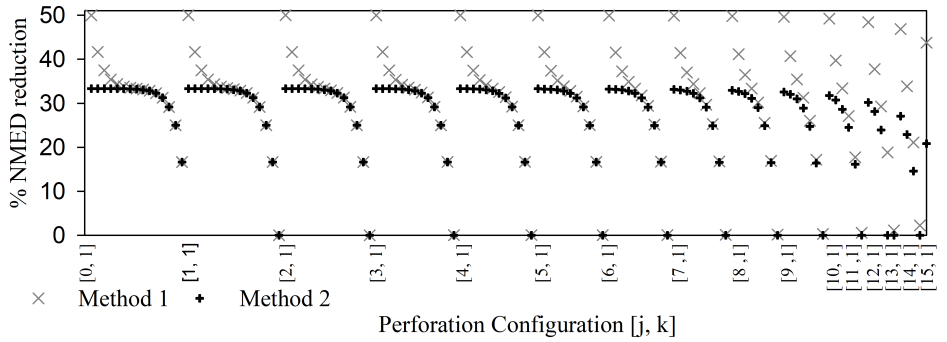
Similarly to Method 1:

$$\begin{aligned} MED &= \frac{2^j}{2^{2n}}\Big(\sum_{\substack{\forall A,B:\\ A\leq B}} x_B A + \sum_{\substack{\forall A,B:\\ A>B}} x_A B\Big) \\ &= \frac{2^j}{2^{2n}}\Big(\sum_{\substack{\forall A,B:\\ A=B}} x_A A + 2\sum_{\forall A}\sum_{\substack{\forall B:\\ B<A}} x_A B\Big) \\ &= \frac{2^j}{2^{2n}}\sum_{A=1}^{2^n-1} x_A A^2, \end{aligned} \tag{4.28}$$

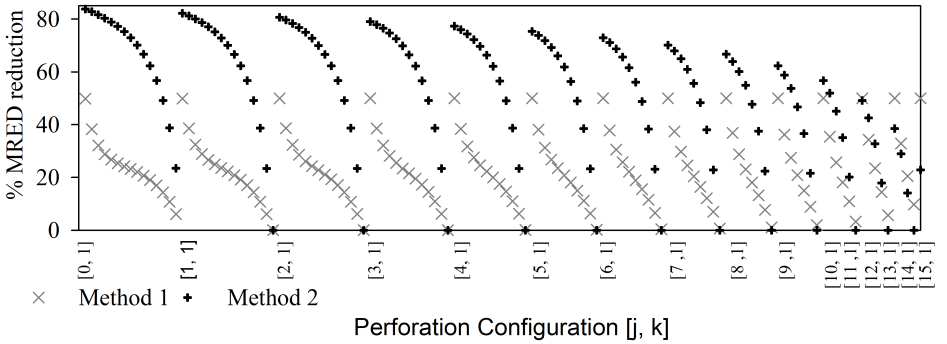
$$NMED = \frac{2^j \sum_{A=1}^{2^n-1} x_A A^2}{2^{2n}(2^n-1)^2}, \tag{4.29}$$

$$\text{and } MRED = \frac{2^j}{2^{2n}}\sum_{B=1}^{2^n-1}\Big(\frac{x_B}{B} + 2x_B\Big). \tag{4.30}$$

Figure 4.3 depicts the error improvement achieved by Methods 1 and 2, for a 16-bit (n=16) multiplier and all the product perforation configurations (j,k). Figure 4.3a presents the *NMED* reduction attained by the correction methods with respect to the *NMED* of product perforation without an error correction method. Figure 4.3b illustrates the respective graph for the *MRED* metric. The proposed corrective methods offer both *NMED* and *MRED* reduction. Method 1 offers higher *NMED* reduction, while Method 2 achieves higher *MRED* reduction. On average, Method 1 offers 30% *NMED* and 24% *MRED* reduction, while Method 2 offers 26% and 50% reduction, respectively. As a result, the selection of a corrective method depends on the application in which the perforated multiplier will be used. If the magnitude of the error is more important than its absolute distance from the accurate result, then Method 2 should be preferred; if not, then Method 1 should be selected. However, the implementation of Method 1 requires a k-bit comparator, while Method 2 requires a n-bit one and thus, Method 1 induces smaller area and power overheads. As a result, since both methods offer significant *NMED* and *MRED* reductions and Method 1 induces less power overhead, it should be preferred in the case the application is unknown.



(a)



(b)

Figure 4.3: The percentage reduction of a) *NMED* and b) *MRED* achieved by the correction Methods 1 and 2 with respect to the *NMED* and *MRED* values obtained by product perforation without correction. The x-axis contains all the $[j, k]$ configurations.

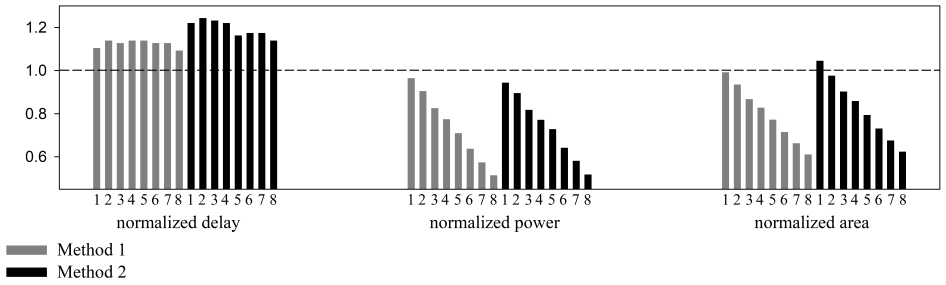


Figure 4.4: The normalized delay, power and area metrics achieved by applying product perforation with correction and with j=1 and k=1..8 on a Dadda 4:2 multiplier, with respect to those of the accurate design.

Methods 1 and 2 decrease the error metrics, but their implementation requires an additional comparator. Figure 4.4 presents the impact of correction Method 1 or 2 on the delay, power, and area on the Dadda 4:2 multiplier, in respect to the accurate design. Since the complexity of the comparator is mainly affected by the perforation variable k, Figure 4.4 depicts perforation configurations that feature j=1 and k= 1 to 8 (similar results are obtained for other j and for MBE designs). As expected, using Method 1 with perforation induces 13% overhead on critical delay, but also retains 26% and 20%, on average, power and area saving. The respective values for Method 2 are 20%, 26%, and 17%.

The *NMED* and *MRED* analytical relations show that *the error imposed by the product perforation method is bounded and predictable.* Therefore, when the application's input dataset is determined, it can be used to calculate the optimal combination of j and k that produce an error less than a desired upper bound.

4.3 Exploring the Efficiency of Partial Product Perforation

In this section, the partial product perforation method is applied to various multiplier architectures in order to explore how their power consumption, area, delay, and accuracy behave considering the perforation configuration variables j and k. This analysis targets to expose the optimal architecture–configuration pair for determined error values regarding both power dissipation and area complexity. This is critical, since different configurations may not have the same impact on a multiplier architecture, e.g. an architecture may be the power optimal one when accurate calculations are performed, but suboptimal when partial product perforation is applied.

Both SPP and MBE techniques are considered in our analysis. Regarding the accumulation tree, the most common architectures are used: 1) Array, 2) Balanced delay, 3) Compressor 4:2, 4) Counter 7:3, 5) Dadda, 6) Dadda with 4:2 compressors, 7) Redundant binary and 8) Wallace [103, 105, 106]. The Array is the simplest way to accumulate the partial products. It consists of successive Carry-Save Adders (CSA) and has the least complexity but the highest delay. The Wallace tree reduces to the least possible the number of partial products in each layer and is theoretically the fastest multi-operand adder. However, it has very complex interconnections that do not permit practical implementations. The Balanced delay tree provides a more regular routing and minimizes the number of wiring trucks. The Compressor 4:2 tree has also a regular structure and sums the partial products as a binary tree does, using 4:2 compressors instead of CSAs. Unlike the Wallace tree, Dadda makes the fewest reductions needed in each layer and can achieve similar overall delay, but requires less gates. The Dadda tree is based on 3:2 counters (full adders) but also 2:2 counters (half adders) to reduce the hardware complexity. The Dadda 4:2 and Counter 7:3 trees use the same reduction strategy with the Dadda tree using though 4:2 and 7:3 compressors, respectively. In the Redundant binary tree, the partial products are in a redundant representation and the addition is performed by redundant binary adders [107] in the form of a binary tree. A Carry

Look-Ahead adder is used as the final adder in all multipliers. Figure 4.1 depicts some typical reduction schemes of the aforementioned tree architectures and the respective perforated trees with configuration j=k=2. Using the unit gate model [108], where the area/delay of a full adder is 7 area units (au)/ 4 time units (tu), of a half adder 3au/2tu and of a 4:2 compressor 14au/6tu, the area of the Array is decreased by 112au and its delay by 8tu. The respective values for the Wallace tree are 115au and 4tu. The delay of the Dadda 4:2 and Compressor 4:2 is not decreased but their area decrease is 127au and 112au, respectively.

Exploration and analysis: The flow used for our evaluation is summarized in Figure 4.5. For our analysis, 16-bit unsigned multiplier architectures are considered. Note that applying product perforation to signed multiplication is performed similar to the unsigned one, except that we do not perforate the last partial product. Therefore, no extra circuit is needed and similar results are expected. They examined multipliers implemented in structural Verilog and synthesized using Synopsys Design Compiler and the TSMC 65nm standard cell library. We simulate the designs using Modelsim and calculate their power consumption with Synopsys PrimeTime triggering the average mode of calculation. All the possible combinations of j and k are explored and 1376 architectural configurations are examined in total. The metrics measured for each design are the *NMED*, *MRED*, minimum delay and, at the relaxed period of 2ns, its power consumption and area complexity. In [101], a detailed power, area and delay characterization and analysis of the examined perforated multiplier architectures has been performed showing that the aforementioned metrics are scaling gracefully, i.e. average slope -0.16%, -242% and -0.03% respectively, for increased values of k.

Since power, area, and delay metrics scale differently for each multiplier architecture when different error values are considered, we illustrate in Figure 4.6 the power–area Pareto curves for different *NMED* values in order to distinguish the optimal designs. Pareto optimality is used for multi-objective optimization of either synergetic or conflicting objectives. In this case, the power and the area objectives are synergetic. However, as different multiplier architectures exhibit different gains when partial prod-

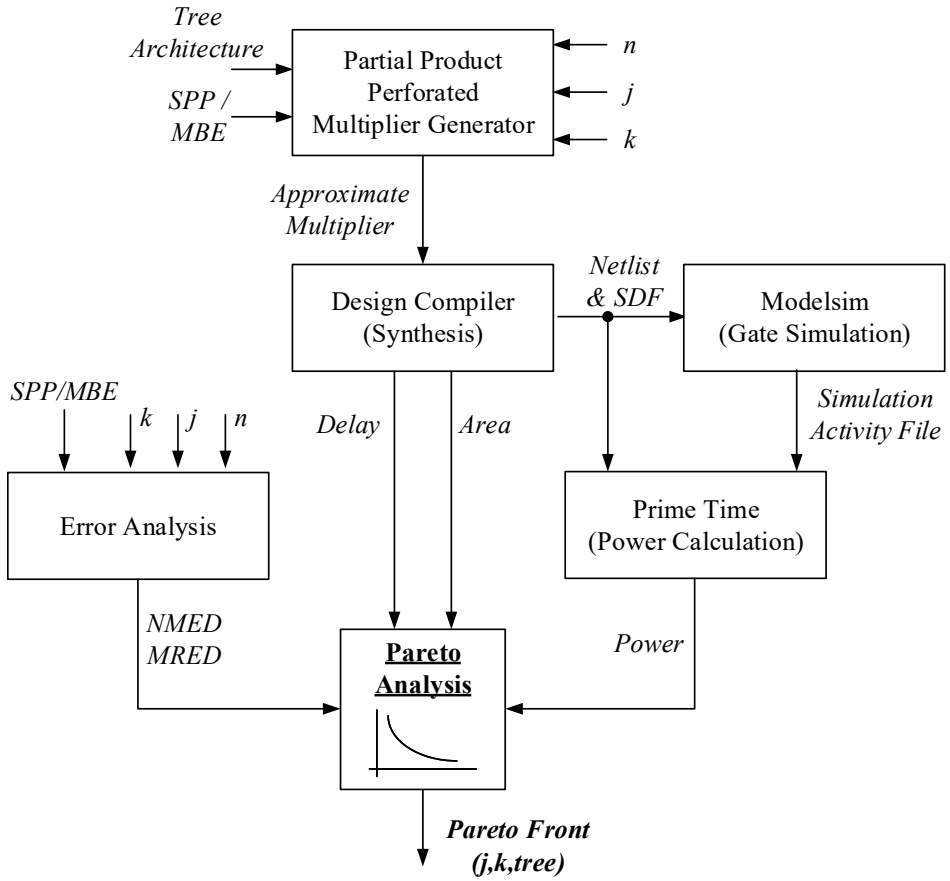


Figure 4.5: The flow used to evaluate the Partial Product Perforation method on different multiplier architectures.

uct perforation is applied to them, we plot Figure 4.6 that presents the power–area Pareto optimal points under differing error bounds, considering as parameters the applied perforation configuration and the multiplier architecture. As a result, for the presented error constraints, Figure 4.6 enables the selection of the optimal perforation configuration—multiplier architecture pair concerning the desired metric (power or area). We consider the *NMED* values of $10^{-4}, 5\times\ 10^{-4}$ and 10^{-3} which enclose a large set of different partial product perforation configurations while keeping the

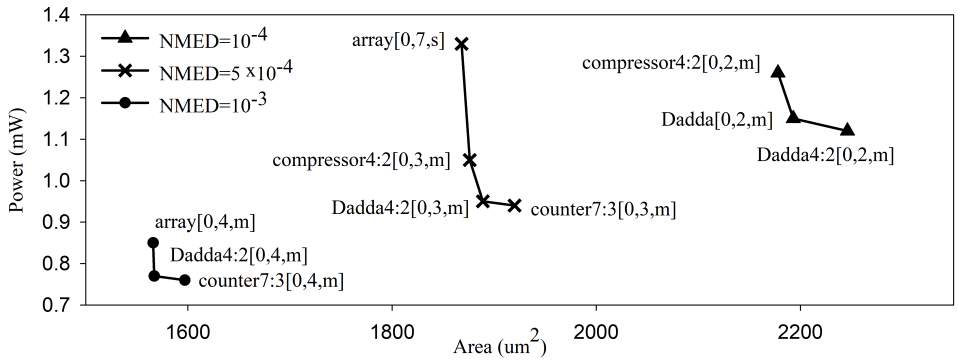


Figure 4.6: The Power–Area Pareto curves for different *NMED* values.

error small. The respective *MRED* values of the designs can be derived in a straightforward manner from the error equations presented in Section 4.2.2 utilizing the annotated j and k parameter values. The Dadda4:2[m] architecture appears in all curves but with different product perforation configuration (i.e., different j and k values), depending on the *NMED* bound. The Dadda[0,2,m], Dadda4:2[0,3,m] and Dadda4:2[0,4,m] designs offer the best power-area tradeoff when *NMED* takes the values $10^{-4}, 5{\times}10^{-4}$, and 10^{-3}, respectively.

The partial product perforation method offers significant power, area, and delay savings depending on the error bound and the multiplier architecture. It can achieve up to 50% power, 45% area and 35% delay reductions for only 0.1% error (i.e., $NMED < 10^{-3}$). For example, when concerning the Dadda4:2[m] architecture, the configurations appearing in the Pareto curves (Figure 4.6) have area and power savings of 20% and 26% for $NMED < 10^{-4}$, 32% and 37% for $NMED < 5{\times}10^{-4}$ and 44% and 49% for $NMED < 10^{-3}$, respectively.

Aiming to elucidate the impact of partial product perforation on each multiplier architecture, we examine their power variation (i.e., the range of power values) for a bounded error. Figure 4.7 presents the box plot diagram for all the architectures with regard to power, considering all the product perforation configurations that result to $NMED < 5{\times}10^{-4}$. We plot Figure 4.7 in

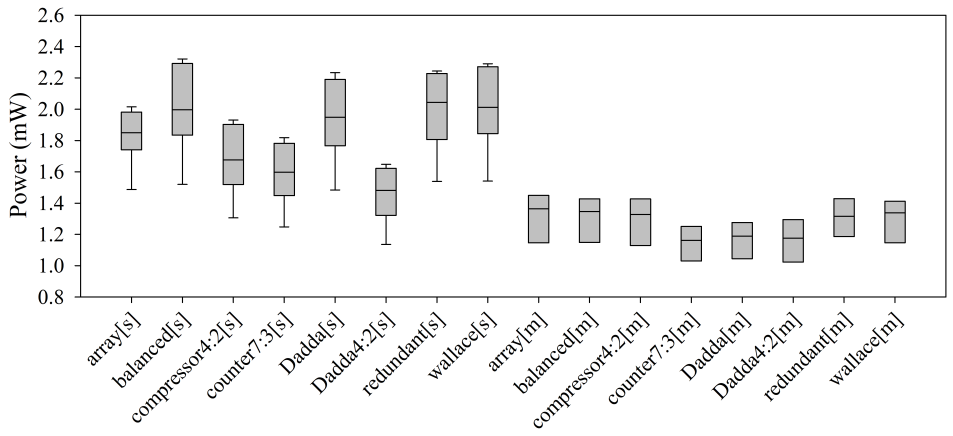


Figure 4.7: Box plots of power consumption for $NMED < 5 \times 10^{-4}$.

order to characterize the impact of product perforation, i.e., differing perforation levels, in each multiplier architecture through its power distribution. For presentation reasons, in the specific boxplot graph we excluded designs considered as outliers. However, the outliers that are located lower than the minimum whisker line, are forming the actual optimal design configurations. The MBE-based architectures exhibit smaller variation and lower median than the respective SPP-based ones. The lowest median and variation values are observed for the counter7:3[m] architecture. Thus, its power consumption for various perforation configurations is concentrated in a smaller range, making its power behavior more predictable. The same conclusion is confirmed in Figure 4.6 where the counter7:3[m] for *NMED* values $5{\times}10^{-4}$ and 10^{-3} is the Pareto optimal point with the lowest power. In Sections 4.4.1 and 4.4.1, the analysis considers the Dadda 4:2 architecture. The Dadda 4:2 appears in all the Pareto curves of Fig. 4.6 and achieves the lowest (or close to the lowest) power consumption, among all the tree architectures examined, for all perforation configurations.

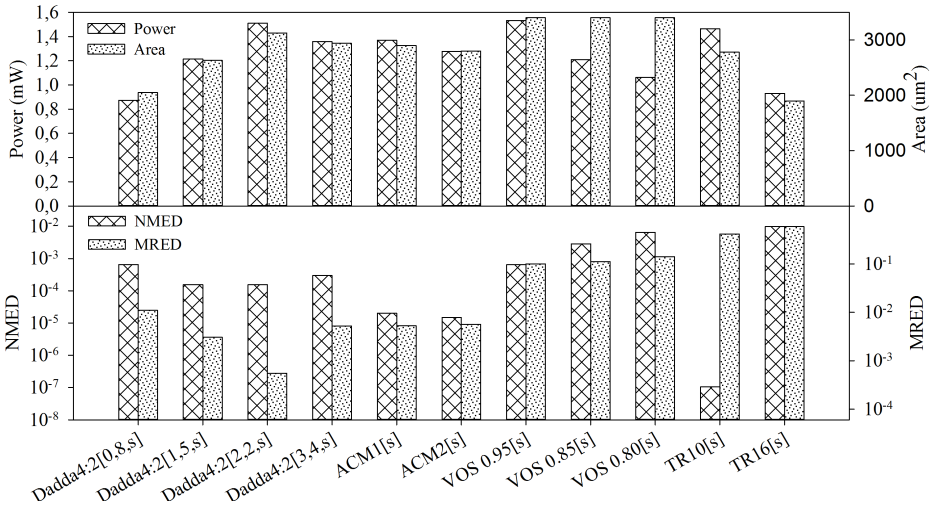
4.4 Experimental Evaluation

In this section, we extensively evaluate the efficacy of partial product perforation regarding both the circuit level implementation of approximate multipliers as well as its application on real-life use cases.

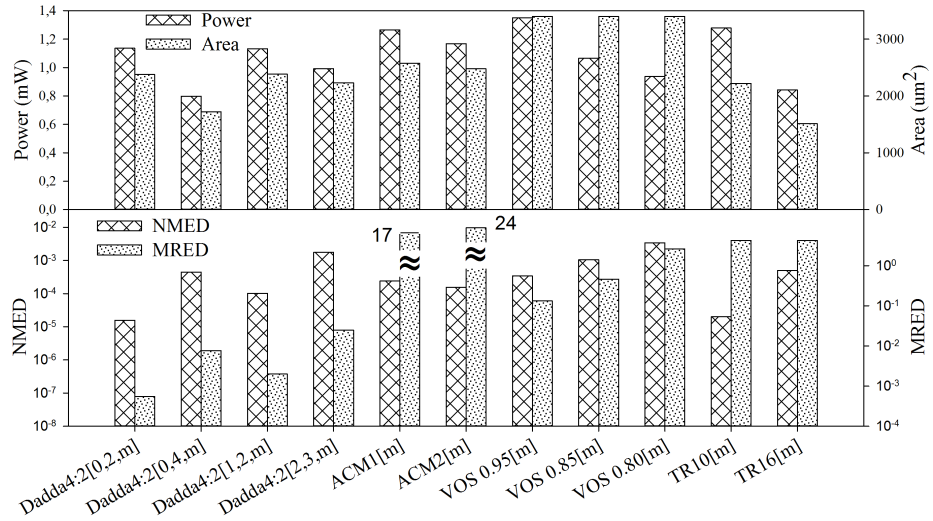
4.4.1 Comparative Study on Circuit Level

In this section, we evaluate the efficiency of partial product perforation in terms of power, area, and error, and we compare it with state-of-the-art approximation techniques, which apply either truncation [56], logic approximation [63] or the VOS technique [57]. Using the two inexact 4:2 compressors of [63] at the 16 LSB columns, two approximate 16-bit multipliers ACM1 and ACM2 are implemented in structural Verilog and synthesized at 2ns using Synopsys Design Complier and PrimeTime. Error metrics calculation is performed through exhaustive Matlab simulation. In order to compare the partial product perforation with the VOS technique, we use the Synopsys Composite Current Source model (CCS) [94]. CCS models are proven to deliver signoff-level accuracy to within 2% of HSPICE simulation, are designed to be scalable for voltage, temperature and process, and offer better accuracy than the Non-Linear Delay and Power Models [94]. For the exact multiplier architectures of Section 4.3, we scale the supply voltage from 1V (nominal) to 0.80V and measure their power consumption and error metrics using 10^5 randomly generated inputs. Regarding truncation, two truncated multipliers with variable correction [56] that use the Dadda 4:2 tree to accumulate the partial products are implemented. In the first one (TR10) the 10 LSBs are truncated while in the second (TR16) the 16 ones. For the perforated multipliers, the error correction Method 1 (Section 4.2.2) is used.

Figure 4.8 presents comparative results on the power, area, *NMED*, and *MRED* metrics after applying: i) the four different partial product perforation configurations, ii) the approximate compressors according to the technique presented in [63] (ACM1 and ACM2), iii) the VOS technique and iv)



(a)



(b)

Figure 4.8: Comparison of partial product perforation with ACM1, ACM2 [63], TR10, TR16 [56], and VOS [57] for (a) SPP and (b) MBE architectures.

the truncation (TR10 and TR16) on a 16-bit Dadda 4:2 multiplier using SPP (Figure 4.8a) and MBE (Figure 4.8b). The examined perforated designs exhibit different order of perforation (j variable) and they are on (designs Dadda4:2[0,8,s] and Dadda4:2[1,5,s]) or close to (designs Dadda4:2[2,2,s] and Dadda4:2[3,4,s]) the power-*NMED* Pareto optimal curve of the Dadda 4:2 architecture. Similar selection has been performed for the MBE-based designs.

The proposed Partial Product Perforation for the SPP-based designs, included in Figure 4.8a, delivers power savings of up to 49% and area reduction of up to 40% compared to the respective accurate design, while the *NMED* value is 6.5×10^{-4} at most and the *MRED* one goes up to 1.1×10^{-2}. The respective values for MBE-based configurations (Figure 4.8b) are 47% power savings, 38% area reduction, *NMED* 1.8×10^{-3}, and *MRED* 2.5×10^{-2}. **The approximate compressors multipliers** ACM1, ACM2 [63] with SPP (Figure 4.8a) have 15%, 20% power and 15%, 18% area savings, respectively, over the accurate Dadda 4:2 multiplier. Their *NMED* values are 2×10^{-5} and 1.5×10^{-5}, while their *MRED* ones are 5.3×10^{-3} and 5.6×10^{-3}, respectively. For the MBE (Figure 4.8b), ACM1, ACM2 have 16%, 23% power savings and 8%, 11% area reduction, respectively, over the accurate Dadda 4:2 multiplier. Their *NMED* values are 2.4×10^{-4} and 1.6×10^{-4} while their *MRED* ones are 17 and 24 respectively. Regarding the MBE-based designs, [63] is less efficient since less partial products compared to the SPP technique are accumulated in the tree and and an error occurring in one column has a greater impact on the output. **VOS** does not deliver any area reduction, offering though significant power savings compared to the accurate design. When decreasing the supply voltage of the SPP-based design to 0.80V (Figure 4.8a), the power consumption is 1.06mW (i.e., 37.9% less than the accurate one). Similarly, the power consumption of the MBE-based design (Figure 4.8b) is 0.94mW (i.e., 37.7% less than the precise design). However, even for small power savings (10% at 0.95V), the *NMED* and *MRED* values of VOS are too large, more than 0.65 and 10 respectively, as VOS errors are mainly impacting MSBs, resulting to large *ED*. **The truncated multipliers** TR10 and TR16 [56], when SPP is used, offer 14%, 46% power savings and 18%, 44% area reduction for 1.1×10^{-7}, 1.2×10^{-1} *NMED* and 0.4, 0.8 *MRED*, respectively.

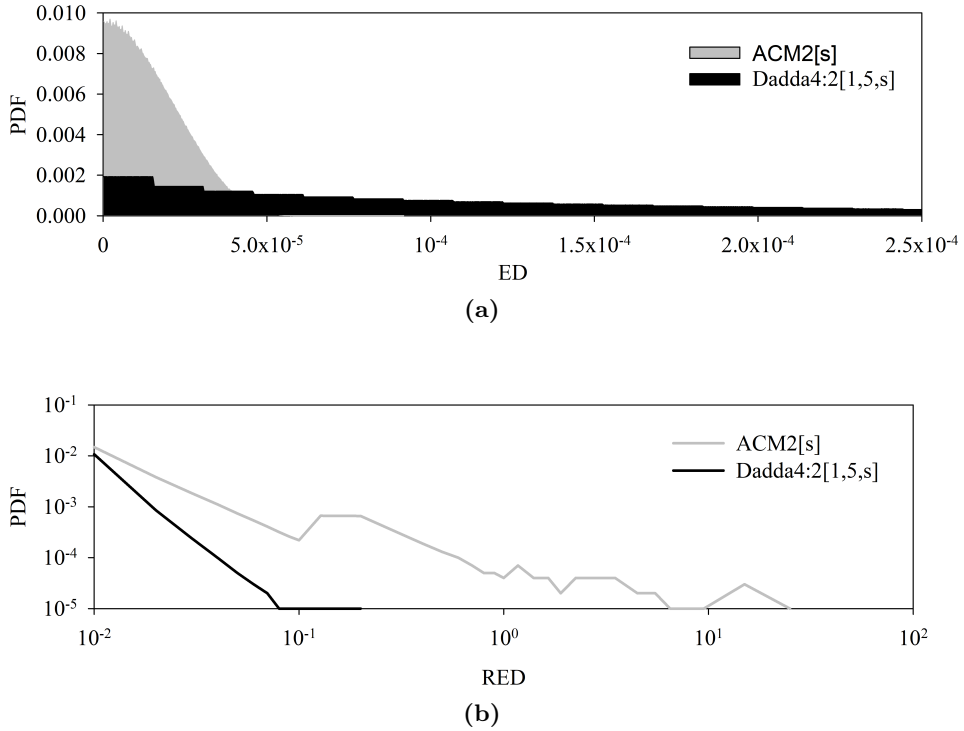


Figure 4.9: a)The Probability Density Function of the *ED* for the ACM2 [63] and the partial product perforation Dadda 4:2 multiplier with $j = 1$ and $k = 5$. *ED* is in the Q0.32 number format (fixed point representation of 32-bit integers in the range [0:1)). b) The respective Probability Density Function of the *RED*.

The respective values for the MBE-based designs are 15%, 44% power savings, 20%, 46% area reduction, 2×10^{-5}, $5.0{\times}10^{-4}$ *NMED* and 4.2, 4.3 *MRED*.

On average, the partial product perforation configurations, illustrated in Figure 4.8, exhibit lower *MRED* values than ACM2, but higher *NMED*. The large *NMED* value of partial product perforation implies that it may produce large *ED*. However, the small value of *MRED* shows that such large *ED* is insignificant compared to the accurate result. The aforemen-

Table 4.2: Ranking of the Savings and Errors of the Approximate Multipliers

Design		Power Gain		Area Gain		NMED		MRED	
SPP	MBE	SPP	MBE	SPP	MBE	SPP	MBE	SPP	MBE
[0,8,s]	[0,4,m]	1	1	2	2	8	7	6	3
TR16		2	2	1	1	11	8	11	9
VOS 0.80		3	3	11	11	10	11	9	7
VOS 0.85		4	5	11	11	9	9	8	6
[1,5,s]	[1,2,m]	5	6	3	6	5	3	2	2
ACM2		6	8	5	7	2	4	5	11
[3,4,s]	[0,2,m]	7	7	7	5	6	1	3	1
ACM1		8	9	6	8	3	5	4	10
TR10		9	10	4	3	1	2	10	8
[2,2,s]	[2,3,m]	10	4	8	4	4	10	1	4
VOS 0.95		11	11	11	11	7	6	7	5

tioned points can be further explained based on the error analysis of Section 4.2.2. As shown, the *ED* is proportional to the inputs and, thus, it can be as large as the input numbers. However, $RED = x_B 2^j / B$ and since few partial products are removed, the nominator is much smaller than B, resulting to small relative error values. On the other hand, [63] produces smaller *ED*, but its errors are of greater significance compared to the exact results. This behavior is also captured by Figure 4.9 where the Probability Density Function (PDF) of the *ED* and *RED* for ACM2 and Dadda4:2[1,5,s] is presented. ACM2 exhibits lower *NMED* but higher *MRED* compared with Dadda4:2[1,5,s]. Figure 4.9a depicts the PDF of the *ED* for the aforementioned multipliers. ACM2 has significantly greater error probability, but its probable error values are concentrated in a smaller range. In contrast, the Dadda4:2[1,5,s] errors are spread to a wider range and have almost equal, but very low, probability to appear. Figure 4.9b depicts the same graph for the *RED* metric. As presented in Figure 4.9b, ACM2[s] produces larger *RED* values than Dadda4:2[1,5,s] and with greater probability.

To summarize, the partial product perforation technique shows significant gains compared to the accurate design and state-of-the-art approximate techniques. On average, compared to VOS, partial product perforation configurations attain 3% lower power consumption and 96% lower *MRED*, when SPP is used, and 9% and 99%, respectively, when MBE is used. Compared to [63] for SPP schemes, their power consumption and their *MRED* are 6% and 9% lower, respectively. For MBE schemes, the respective values are 17% lower power and 3 orders of magnitude lower *MRED*. Compared with the SPP truncation [56], the perforated multipliers of Figure 4.8 deliver on average 3% higher power for 99% lower *MRED*, while for MBE the respective values are 4% lower power and 2 orders of magnitude lower *MRED*. Finally, Table 4.2 offers a more straightforward comparison among the examined approximation schemes, by ranking them according to their savings and error metrics. The examined designs have been grouped in four sub-groups each one with designs exposing similar power and/or error characteristics. In each sub-group, the perforated multipliers deliver the lowest power and *MRED* values and, in most cases, the lowest *NMED* and area as well.

4.4.2 Comparative Study on Real Life Applications

In this section, we evaluate the efficiency of the proposed technique on real-life use cases from the image processing and data analytics domains. For our analysis, we consider the Canny edge detection [109] and Geometric Mean filters from the image processing domain and the K-means clustering [110] from the data analytics domain, respectively. All the examined algorithms are implemented in C++, while for the image processing ones, OpenCV library is used.

Geometric mean filter removes noise from images, offering better results than the arithmetic mean filter for Gaussian type noise. The geometric mean filter with parameter r filters an image by replacing each pixel's value by the geometric mean of the values of all the neighboring pixels that are inside a $(2r+1)\times(2r+1)$ block centred on that pixel. For our evaluation, the r parameter is set to 3. We approximate the Geometric mean by replacing the

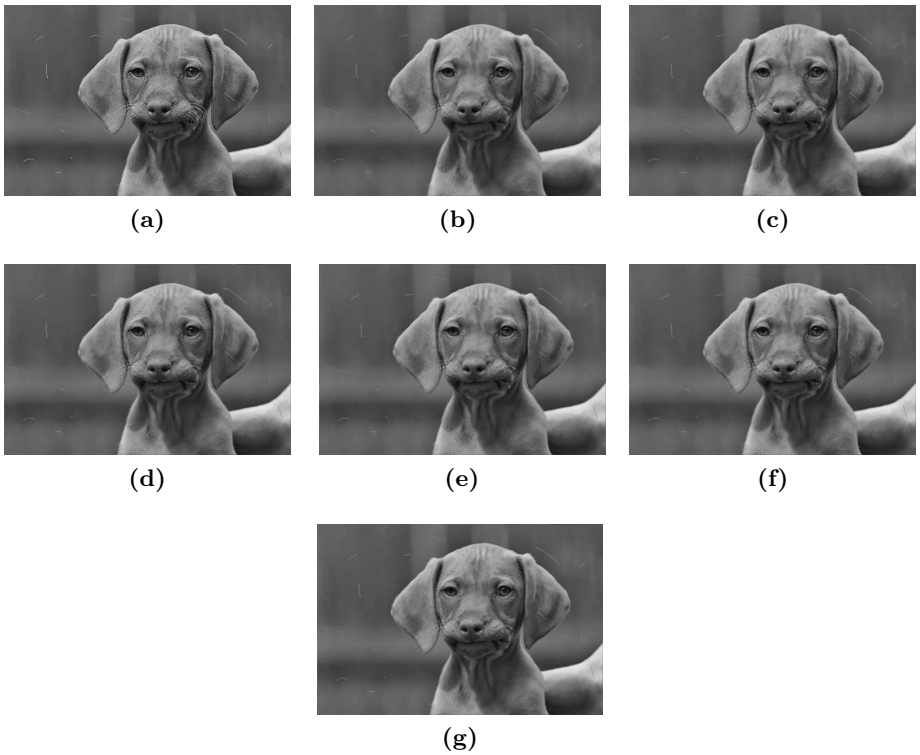


Figure 4.10: The a) 16-bit input image and the result of the geometric mean filter using the b) accurate multiplier Dadda4:2[s], c) Dadda4:2[1,5,s] w/o correction, d) Dadda4:2[1,5,s] w/ correction Method 1, e) Dadda4:2[3,4,s] without correction, f) Dadda4:2[3,4,s] with correction Method 1 and g) ACM2.

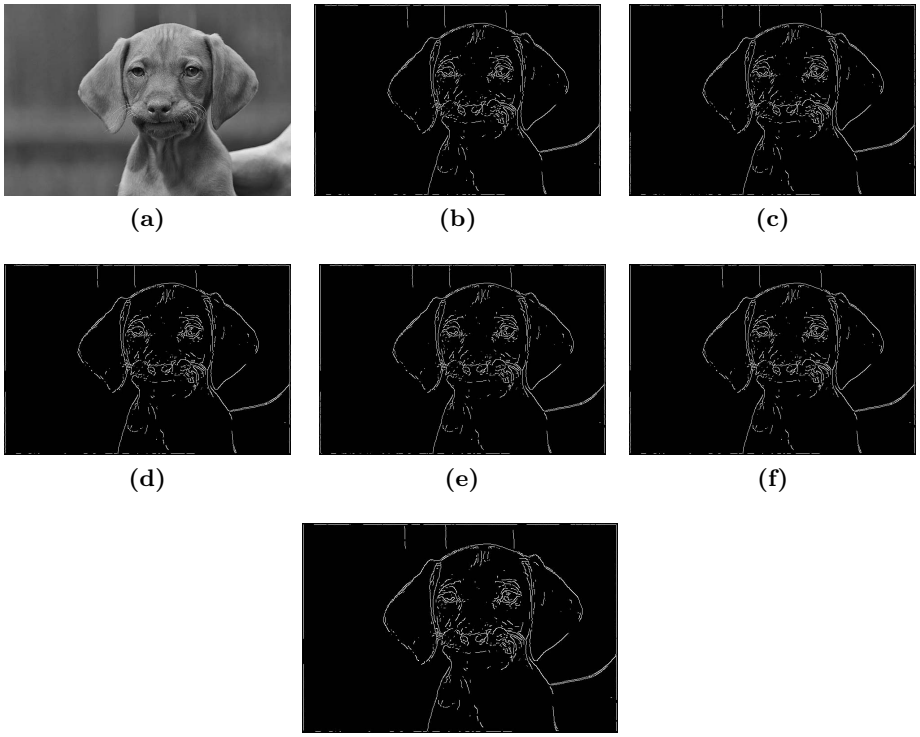


Figure 4.11: The a) 16-bit input image and the result of the Canny edge detection using the b) accurate multiplier Dadda4:2[s], c) Dadda4:2[1,5,s] w/o correction, d) Dadda4:2[1,5,s] w/ correction Method 1, e) Dadda4:2[3,4,s] without correction, f) Dadda4:2[3,4,s] with correction Method 1 and g) ACM2.

multiplication between the pixels with an approximate 16×16 multiplier. We used as input the 16 bits (16 bits/pixel) grayscale image depicted in Figure 4.10a. To evaluate the accuracy of the output images of the Geometric mean we use the Peak Signal Noise Ratio (*PSNR*).

Canny edge detection [109] filter, is considered to be an optimal edge detector. Specifically, i) it masks the image by applying a Gaussian filter to remove the noise, ii) it calculates the gradient of the image to find the edge strength, iii) it applies a non-maximum suppression to keep only the local maxima, iv) it determines the potential edges by thresholding, v) and, finally, it tracks edges by hysteresis, i.e, suppresses all the edges that are weak and not connected to strong edges. The size of the Gaussian kernel is 7×7 with 1.1 standard deviation value and uses 16-bit fixed point arithmetic. We approximate Canny edge by replacing the multiplication in the Gaussian filter with an approximate 16×16 multiplier. We used as input the 16 bits grayscale image depicted in Figure 4.11a. The percentage of the edges detected using the approximate multiplier over those detected using the accurate one is used as our quality metric.

K-means is a popular algorithm for clustering data points from a multi-dimensional space into k clusters. It uses a two phase iterative method and aims to partition the data points into sets, so as to minimize the within-cluster sum of distance functions of each point in the cluster to the center. We use the Euclidean distance as a distance function. We approximate the K-means algorithm by replacing the multiplications in the calculation of the Euclidean distance with an approximate 16×16 multiplier. We use a random generated input dataset of 100,000 4-dimensional points with 16 bits per dimension. The input dataset is clustered in 100 clusters. To evaluate the accuracy of the K-means algorithm we use the average relative L2-Norm, i.e., $\left\langle \frac{|\mathbf{x_{acc}} - \mathbf{x_{approx}}|_2}{|\mathbf{x_{acc}}|_2} \right\rangle$.

Similar to [60, 63], the approximate multiplier is considered as part of a general processing system that implements the aforementioned algorithms. The rest of hardware components (except the multiplier) are considered to deliver accurate results and thus, any applications inaccuracy and energy savings result from the usage of the approximate multiplier. The energy

values of each multiplication operation are delivered by post-synthesis simulations of the approximate multipliers on the input data traces extracted by the applications execution. Note that in the Canny edge detection and Geometric mean algorithms the number of the multiplications depends only on the image size and thus, it is the same for the accurate as well as the approximate version of the algorithm. On the other hand, the iterations performed by the K-means algorithm are not constant and as a result, the number of multiplications in the accurate may differ from the ones in the approximate version.

Figure 4.10 depicts both the input image and the output image of the geometric mean filter when using the accurate multiplier Dadda4:2[s], the perforated multipliers Dadda4:2[1,5,s] and Dadda4:2[3,4,s] with and without any correction method, and the approximate multiplier ACM2. Figure 4.11 shows the same images for the Canny edge detection. Table 4.3 summarizes the values of the energy savings and quality metrics of each application when using the aforementioned multipliers.

The use of the Dadda4:2[1,5,s] multiplier results in 85.95 dB PSNR for the geometric mean and 91.04% edges detected for the Canny edge detection. The application of the corrective Method 1 with the Dadda4:2[1,5,s] results in a small decrease of the energy savings (7.41%), but delivers better outputs as the PSNR increases by 2.9% and the edges detected by 7.6%. The Dadda4:2[3,4,s] multiplier detects the 84.79% of the edges and its PSNR is 89.93 dB. The use of correction Method 1 with the Dadda4:2[3,4,s] decreases the energy reduction by 10%, detects 16.6% more edges, and increases its PSNR by 3.1%. When ACM2[s] [63] is used, the output image has 86 dB PSNR and 97.85% edges detected. When we compare Dadda4:2[1,5,s] with ACM2, we observe that the former offers 25.6% higher energy reduction, detects 7% less edges, and has the same PSNR as the latter. When we compare ACM2[s] with Dadda4:2[3,4,s] using Method 1, we find that the latter delivers 18.6% lower energy savings, detects 1.8% more edges, and has 7.8% higher PSNR. Finally, when we compare Dadda4:2[1,5,s] using Method 1 with ACM2[s], the former achieves 16.3% higher energy reduction, detects 0.5% more edges, and has 2.8% higher PSNR. Regarding to the K-means algorithm, using a correction Method with product perfora-

Table 4.3: Evaluation of Partial Product Perforation in Image Processing and Data Analytics Algorithms

Multiplier	**Canny Edge**	
	PSNR (dB)	**Energy Gain (mJ)**
Accurate Dadda4:2[s][a]	Inf.	0
Dadda4:2[1,5,s]	85.95	1.18×10^{-3}
Dadda4:2[1,5,s] - Meth. 1	88.45	1.09×10^{-3}
Dadda4:2[3,4,s]	89.93	8.51×10^{-4}
Dadda4:2[3,4,s] - Meth. 1	92.75	7.63×10^{-4}
ACM2[s]	86.00	9.38×10^{-4}
	Geometric Mean	
	Edges Detected (%)	**Energy Gain (mJ)**
Accurate Dadda4:2[s]	100.00	0
Dadda4:2[1,5,s]	91.04	1.94×10^{-2}
Dadda4:2[1,5,s] - Meth. 1	98.33	1.79×10^{-2}
Dadda4:2[3,4,s]	84.79	1.40×10^{-2}
Dadda4:2[3,4,s] - Meth. 1	99.58	1.25×10^{-2}
ACM2[s]	97.85	1.54×10^{-2}
	K-Means	
	avg. Relative L2-Norm (%)	**Energy Gain (mJ)**
Accurate Dadda4:2[s]	0	0
Dadda4:2[1,5,s]	5.08	18.94
Dadda4:2[1,5,s] - Meth. 1	5.08	18.04
Dadda4:2[3,4,s]	7.18	9.13
Dadda4:2[3,4,s] - Meth. 1	7.18	8.04
ACM2[s]	8.97	-6.06

[a]The energy required for the accurate multiplication process in the Canny Edge, the Geometric mean, and the K-means algorithm is 3.73×10^{-3}mJ, 6.13×10^{-2}mJ, and 45.14mJ, respectively.

tion does not deliver any quality improvement. This is explained by the fact that in the Euclidean distance the multiplier is used as a squarer and as a result swapping the multiplicands does not decrease the multiplication's error. Moreover, we observe that using ACM2[s] in the K-means algorithm does not offer any energy reduction. The implementation of the K-means algorithm with ACM2[s] fails to converge and exits after reaching a maximum number of allowed iterations. As a result, although ACM2[s] has lower power consumption compared with the accurate multiplier, the increased number of multiplications results in an energy increase of the K-means algorithm.

4.4.3 Impact of Bit-width Scaling

In this section, we examine the scalability of the proposed technique in terms of increased multiplier's bit-width. More specifically, we study the impact of scaled bit-widths, i.e. 16- up to 128-bits, on the proposed perforation technique focusing on the delivered accuracy (*NMED, MRED*) and power and area gains. We consider the Dadda 4:2 as our driver architecture solution and $NMED \leq 10^{-4}$ as our quality constraint. Figure 4.12a depicts for each of the examined bit-widths the power and area reduction delivered by the perforated Dadda 4:2 solutions in respect to their accurate designs. In a complementary manner and for the same scaled bit-widths, Figure 4.12b presents the *NMED* and *MRED* values when targeting 50% power reduction. Specifically, for $NMED \leq 10^{-4}$, the power and area gains for 16-bit width is 21% and 31%, respectively. The respective gains in the case of 128-bit width design scales up to 74% and 91% regarding to power and area, respectively. Similarly, Figure 4.12b shows that for the same relative power gain, i.e. 50%, the 16-bit solution delivers an *NMED* and *MRED* value of 1.95×10^{-3} and 2.61×10^{-2}, respectively. For the 128-bit solution, *NMED* and *MRED* reduce to 1.73×10^{-18} and 2.05×10^{-16}, respectively. Thus, partial product perforation offers better results as the multiplier's bit-width increases, i.e., higher power and area reduction for the same error constraints or lower error values for the same power savings.

This good scaling behavior for increased multiplier's bit-widths can be also

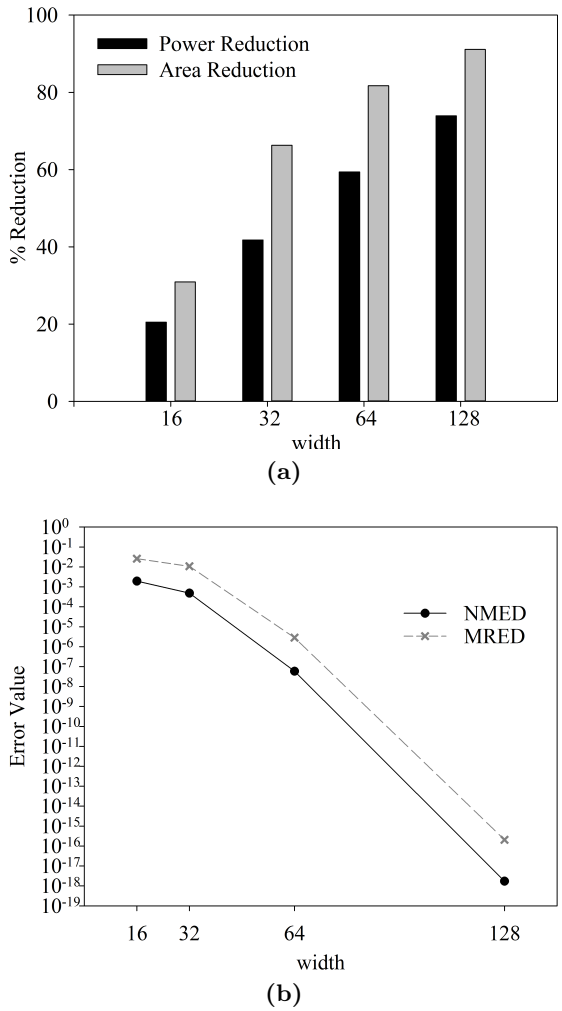


Figure 4.12: Impact of multiplier's bit-width scaling on partial product perforation. a) Power and area gains for $NMED \leq 10^{-4}$, and b) *NMED* and *MRED* values when targeting 50% power savings.

theoretically confirmed utilizing the error analysis of Section 4.2.2. Let us assume two multipliers M_1, M_2 with different bit-widths $n_1,\ n_2$ with $n_1 < n_2$ having the same j value for the partial product perforation. For both multipliers to achieve the same *NMED* the following relation should hold, according to Eq. (4.11):

$$\frac{2^j(2^{k_1}-1)}{4(2^{n_1}-1)} = \frac{2^j(2^{k_2}-1)}{4(2^{n_2}-1)} \implies \frac{(2^{n_2}-1)}{(2^{n_1}-1)} = \frac{(2^{k_2}-1)}{(2^{k_1}-1)}. \tag{4.31}$$

Given that $n_1 < n_2 \implies k_1 < k_2$. High k values imply the perforation of more partial products. Thus, for two approximate multipliers with the same *NMED* but different bit-widths, the higher the multiplier's bit-width the higher the the number of partial products that should be perforated, and thus the higher the power gains achieved in respect to their accurate counterparts.

4.4.4 Comparison with Approximate Partial Product Generation Multipliers

In our publication in [101] we introduced the generation of approximate multipliers by altering the multiplication algorithm and approximating the partial product generation. Beyond our publication in [101], several works were published that applied approximations on the partial products in order to produce power-efficient approximate multipliers. In this section we compare partial product perforation with the most notable and efficient (according to the respective scientific literature) among them, i.e., [74, 75, 100]. For complicity reasons, we also incorporate in our comparison [66] that similar to our technique modifies the multiplication algorithm to produce approximate designs. Jiang et al. [74] proposed the R8ABM imprecise multipliers. [74] presented an approximate radix-8 booth multiplier that uses an approximate adder for producing the $3 \times A$ partial product and uses the truncation technique in the accumulation tree. Liu et al. [75], in order to build the R4ABM approximate multiplier, designed approximate modified (radix-4) Booth encoders by altering accordingly the respective K-Map and used the approximate 4:2 compressors [63] in the accumulation tree. Since

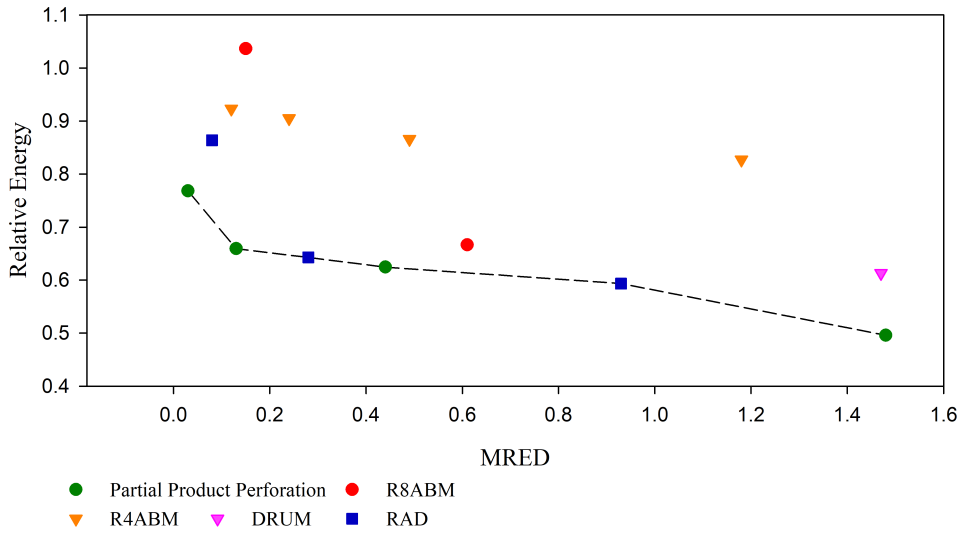


Figure 4.13: Comparison of 16-bit approximate multipliers that apply approximations on the partial product generation and/or the multiplication algorithm. Several configurations per approximate multiplier are considered. A color code is used to distinguish the configuration of the different architectures.

approximations in partial product generation and in the partial product accumulation are synergetic, in the following comparative evaluation we focus only on the approximations related to the generation of the partial products. In [66] the DRUM multiplier is presented extending the idea of [65]. DRUM modifies the multiplication algorithm and performs 6-bit multiplication instead of 16-bit one. The latter is achieved by multiplying the multiplicand's and multiplier's 6-bit segments that contain (and start with) the most significant '1' (leading one). In [100] we introduced a novel approximate hybrid high radix encoding and proposed the imprecise RAD multipliers. In the RAD multipliers the Most Significant Bits of the multiplicand are encoded using the radix-4 encoding, whereas the k Least Significant Bits are encoded using a radix-2^k (with $k > 4$) encoding and we approximate the latter by modifying accordingly its truth table. For more detailed information regarding these approximate multipliers [66, 74, 75, 100] and their evaluation refer to our publication in [100]. In Figure 4.13 the energy–error trade-off

of the examined multipliers is presented. As an error metric the *MRED* is used and as an energy savings metric the relative energy with respect to the accurate design. For every multiplier architecture several different configurations are considered with varying energy and error characteristics. All the different configurations of every approximate multiplier (that refer to the same technique) are color coded and depicted with the same color. For example, the partial product perforation multipliers are colored in green. The dashed grey line represents the Pareto front of the examined configurations, i.e., the configurations that feature the best energy savings-error tradeoff. As shown in Figure 4.13, the Pareto front comprises only the partial product perforation configurations and two of the RAD multipliers that we proposed in [100]. Therefore, even compared with multipliers published beyond our publication in [101] that also target the approximation of the partial product generation, the proposed product perforation multipliers constitute the most efficient approximate multiplier solutions in terms of energy and error.

4.5 Conclusion

In this chapter, we presented the partial product perforation technique for producing approximate hardware multiplier circuits. The proposed technique, by omitting the generation of a number of partial products, delivers high area and power savings, while retaining high accuracy. Although the code perforation is a well-known technique for software approximation, it is applied for the first time on the design of hardware circuits, delivering, as presented, very satisfactory and interesting results regarding to the power-area-error trade-offs. Moreover, to the best of our knowledge, targeting to create hardware approximate multipliers, our technique explored for the first time approximations at the partial product generation. Through an error analysis, we analytically characterised the induced error metrics proving that the error is bounded and predictable and we proposed two error correction methods that trade a small increase in power for high error reduction. Regarding to the introduced error analysis, we manage to rigorously model and analyze the induced perforation error for any input dataset-distribution,

perforation configuration, and multiplier size. To the best of our knowledge, such a rigorous analysis enabling error estimation over input data distributions, is presented for the first time. On top of that we can accurately predict, for the first time, the output error (i.e., maximum, average, and each multiplication's separately) in an approximate multiplier circuit without the need of simulation. In addition, the introduced error analysis is not bound to a specific multiplier architecture and thus, it can be transparently applied to every hardware multiplication scheme. This forms an extremely important feature of product perforation technique, which unlike other state-of-art approximation techniques, can be applied with error guarantees to every multiplication circuit regardless of its architecture. Furthermore, we provide an extensive experimental power-area analysis, on the impact of the proposed product perforation technique over the design space of the various multiplication architectures. To the best of our knowledge, his is the first time that such an exploratory analysis over different approximate multiplier architectures is offered to the reader/designer. Such an exploration campaign, allows the designer not only to evaluate the technique and explore the savings it offers in each multiplier, but also to select the optimum architecture – partial product perforation configuration combination – for given error constraints. Moreover, this chapter offers a direct comparison among a large set of the state-of-art different-level approximation techniques. We consider that through this extensive comparative study, we offer to the prospective reader/designer the possibility to cross-validate the efficiency of the currently available approximation techniques. Partial product perforation is applied at the algorithmic level of the multiplication and more specifically at the generation of the partial products. In our experimental evaluation we compare it with both techniques that target the accumulation of the partial products (logic level [63], [69]) and techniques that are applied at the circuit level (VOS [57]). We also compare partial product perforation with approximate computing works published beyond our publication in [101] that also target the approximation of the partial product generation (i.e., [74, 75, 100]). In our experimental evaluation, we showed that the proposed approach outperforms these state-of-the-art works, achieving significant gains in power, area, and quality metrics of image processing and data analytics algorithms. Finally, we showed that partial product perforation is a scalable approximation technique, delivering

better results as the multiplier's size increases.

Chapter 5

Multi-Level Approximate Arithmetic Library: Hybrid Approximate Multipliers

Approximate computing forms a promising design alternative for inherently error resilient applications, trading accuracy for power savings. In this chapter, we exploit multi-level approximation, i.e. at the algorithmic, the logic and the circuit levels, to design low power approximate arithmetic architectures for hardware multipliers. Motivated from the limited power savings that approximation techniques can achieve in isolation, we explore hybrid methods that apply simultaneously more than one techniques from different layers. We apply partial product perforation for approximations at the multiplier's algorithmic layer and we explore the newly defined design space of hybrid approximate multipliers (HAM) showing that it leads to lower power consumption at every examined error range. To address the increased complexity of the target design space, we introduce an heuristic optimization technique and the corresponding design HAM framework that automatically generates hybrid approximate multipliers requiring a small number of design evaluations, i.e. synthesis, simulation, power and timing analysis. Through extensive experimentation, we show that HAM converges towards optimal solutions and delivers approximate designs that are always more efficient with respect to state-of-art approaches. Power savings of more than 30% are reported moderate error bounds. This chapter is based on our publications in [41, 99, 101].

5.1 Introduction

Power consumption is considered as a first class design concern of modern embedded electronic devices. Digital Signal Processing (DSP) units are widely used in portable devices to implement multimedia algorithms, e.g. image and video processing. Given that such applications produce outputs for human consumption, their exactness is highly relaxed due to limited human perception [67]. This inherent error resilience of these applications allow approximate calculations to be performed by relaxing the numerical exactness of such applications, thus significantly reducing their power dissipation.

Arithmetic units such as adders and multipliers constitute the main units of DSPs. Up to now, extensive research has been conducted on approximate adders [57, 59–61]. In [59], the authors showed that the statistically longest carry chain in an n-bit adder is $\log_2 n$, and produced a fast approximate implementation limiting the carry propagation. An approximate adder design has been proposed in [60] comprising two partitions, an accurate and an inaccurate one. However, research activities on approximate multipliers are limited mainly due to their increased circuit complexity. Recently, [111] proposed a systematic method to compensate the error of approximate arithmetic circuits and applied it on a truncated multiplier, improving its error metrics compared with related truncation schemes. [62] proposed an imprecise 2x2 multiplier cell used as the basic block for constructing larger multiplier architectures. In [63], the authors presented approximate 4:2 compressors, by modifying the respective accurate truth table, which were then used to build two approximate multipliers outperforming [62]. Utilizing approximate adders that limit carry propagation, [64] proposed a fast low-power multiplier but with higher error than [63].

In this chapter, we target approximate multiplier designs that, given the error bounds, push power gains to the limits. We introduce the adoption of perforation technique [102], originally used in software, for approximate multiplier design and then, we present HAM, a novel design framework, that exploits multi-level approximation for designing power optimized hybrid approximate multiplier architectures. We show that even when ap-

plying state-of-the-art approximation techniques in isolation, limited power reductions are delivered. Motivated by this fact, we propose the exploration of hybrid approximation techniques that apply simultaneously more than one techniques. Through extensive experimentation, we prove that for any given error range, the usage of the proposed hybrid approximation techniques delivers, in all cases, approximate multiplier circuits with smaller power consumption than multiplier designs following the state-of-art approximation strategies. We define the design problem of finding the power-optimal hybrid approximate multiplier architecture subject to maximum error constraint and we show that its complexity is equivalent to the "nonlinear nonseparable bounded integer" KNAPSACK [112]. We devised an heuristic optimization procedure that exploits power-error analytical fitting models to produce optimized hybrid approximate multiplier designs without resorting to costly synthesis, simulation and power calculation steps. For small error values, the designs produced by HAM consume up to 11% less power than those with a single state-of-art approximation technique, while for relaxed error values they deliver more than 30% power savings.

The rest of this chapter is organized as follows: In Section 5.2, we introduce the adaptation of perforation technique for approximate multiplier design and we provide an overview of approximation techniques. In Section 5.3, the problem of designing power efficient approximate arithmetic

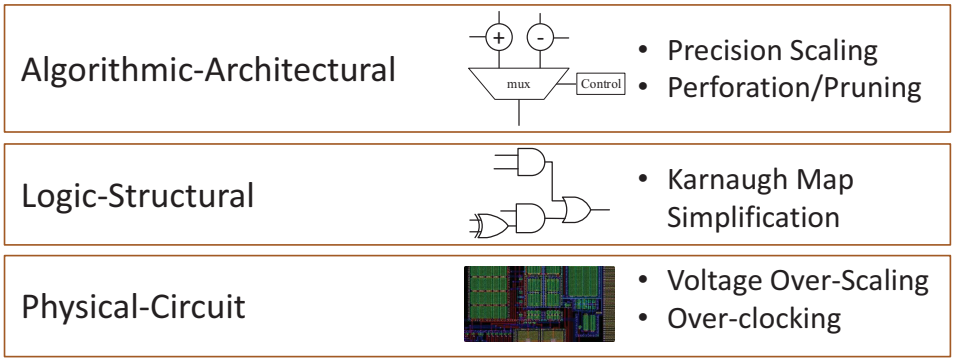


Figure 5.1: The approximation layers.

circuits subject to error constraints is defined and the proposed cross-layer design framework (HAM) for optimized hybrid multiplier architectures is analysed. In Section 5.4, we experimentally evaluate our techniques proving their significance on moving towards more optimized solutions, exhibiting better power-error trade-offs in respect to state-of-art approximation approaches available in the literature. Finally, Section 5 summarizes this chapter.

5.2 Analyzing the Design Space of Approximation Techniques for Arithmetic Circuits

Approximation techniques can be applied over a set of different design levels (Figure 5.1), i.e. (i) algorithmic-architectural, (ii) bit-structural and (iii) circuit. Lower layers are enclosed in the upper ones, since the application of approximate techniques in the former depends on decisions made in the latter. Algorithmic-architectural level approximation is achieved by modifying the algorithm accordingly, namely by substituting some instructions with simpler ones or even by avoiding to execute them [102]. In the bit-structural level, the correctness of some circuits' components is relaxed, thus, lowering their power consumption, but producing imprecise outputs [60, 61, 63]. Approximation techniques at circuit level can be applied once the circuit is implemented, i.e. the approximation is achieved by over-clocking or by voltage over-scaling [57, 67]. The rest of the section analyses the approximation techniques associated with each of the aforementioned design levels.

Driver circuit design: Without loss of generality, we considered the circuit of a Dadda multiplier implemented with 4:2 compressor tree [103] and a carry look-ahead adder as the final adder (Figure 5.2). Both Simple Partial Product (SPP) and Modified Booth Encoding (MBE) are considered in our analysis. All the designs discussed hereafter are synthesized using the Synopsys Design Compiler and the TSMC 65nm standard cell library at a 500Mhz frequency. The designs are simulated using Modelsim and their power consumption is calculated by Synopsys PrimeTime. For the

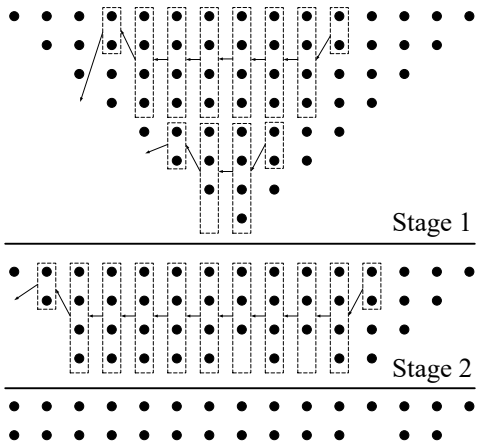


Figure 5.2: Reduction strategy of an accurate 8-bit Dadda tree using 4:2 compressors. The boxes with four dots are the 4:2 compressors, while those with two are either full or half adders. The arrows represent the output carries.

error evaluation, the metrics of Error Distance (ED)[1], Mean Error Distance (MED)[2] and Normalized MED $(NMED)$[3] are considered as effective metrics for quantifying the accuracy of approximate circuits [98]. Finally, for the error calculation, an exhaustive simulation of all possible inputs is performed.

5.2.1 Algorithmic Approximation

At the algorithmic level, approximation is performed through the omission of instructions, i.e. the execution of a subset of the original operations. Targeting this method of approximation, [102] proposed the loop perforation technique which skips a number of loop iterations and results in executing

[1]The ED metric is defined as the absolute distance of the accurate product P and the approximate one P', $ED = |P - P'|$.

[2]The MED is the average of all EDs.

[3]$NMED = MED/P_{max}$, where $P_{max} = (2^n - 1)^2$ in the case of an n-bit multiplier [64].

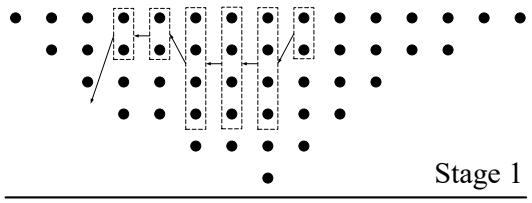
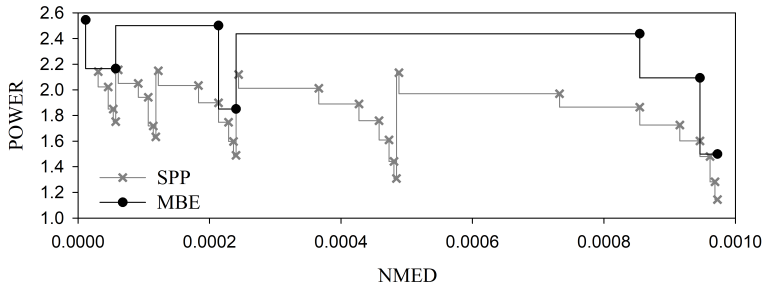
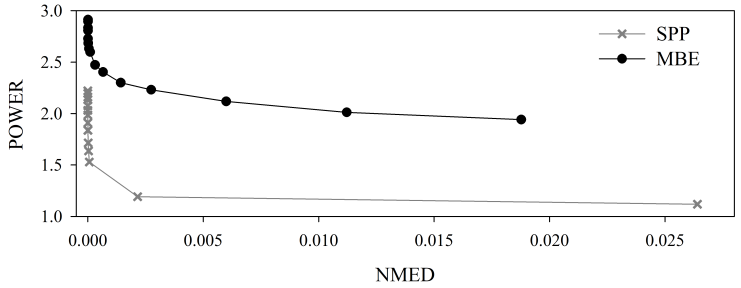


Figure 5.3: The Dadda 4:2 accumulation tree after perforating the 2^{nd} and 4^{th} partial products.

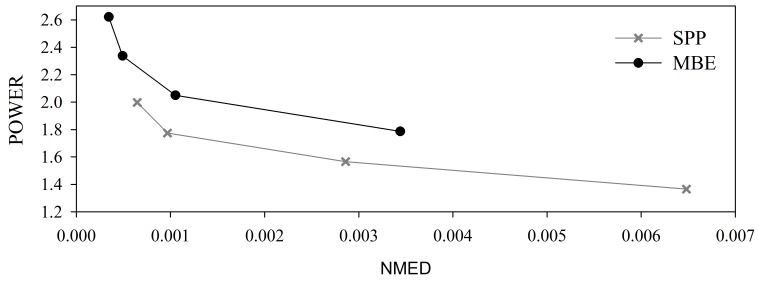
less computations, thus saving time and power, but producing inaccurate outputs. Driven by the fact that hardware multiplication algorithms are inherently iterative, we propose the adaptation of perforation techniques for approximate multiplier design [101]. The result $A \times B$ of multiplying two n-bit numbers A and B, is obtained after summing all the partial products Ab_i, where b_i is the i^{th} bit of B, thus $A \times B = \sum_{i=0}^{n-1} Ab_i 2^i$. Inspired from [102], we apply the perforation technique on a hardware multiplier. A first approach would be to examine the removal of some bits from the partial products (remove some dots from stage 1 in Figure 5.2) and to explore all the possible combinations. However, this approach leads to an explosion of the solution space. Therefore, we select and explore a more coarse grain method, the perforation on entire partial products, in which the generation of some products is omitted. If the j^{th}, k^{th} and m^{th} partial products are perforated, the approximate multiplication result is $A \times B = \sum_{\substack{i=0 \\ i \notin \{j,k,m\}}}^{n-1} Ab_i 2^i$. In Figure 5.3 the accumulation tree of a Dadda 4:2 multiplier after perforating the 2^{nd} and 4^{th} partial products is illustrated. The minimum order of the perforated partial products is the order of perforation and their cardinality is the length of perforation. A perforated partial product is not inserted in the accumulation tree and, as a result, n full adders can be removed from the tree, reducing its power consumption while producing approximate results. The greater the length of perforation, the greater the power decreases and the error increases. Moreover, as the order of a perforated partial product increases, so does the imposed error, as it affects more significant bits. Trying to minimize the induced error when perforating with length more than one, we perforate on successive partial products. There is no upper



(a) Algorithmic level (product perforation)



(b) Logic level (approximate 4:2 compressors)



(c) Circuit level (VOS)

Figure 5.4: The variation of the power consumption with respect to *NMED* of the approximate Dadda 4:2 multiplier when applying a) perforation, b) the approximate 4:2 compressors, and c) VOS. The power of the accurate multiplier with SPP and MBE is 2.24mW and 2.92mW, respectively.

bound on the power reduction that can be achieved using the perforation method (all the partial products can be perforated), however, perforating on many partial products makes the error enormous, rendering it infeasible. In Figure 5.4a the variation of the power consumption respectfully to *NMED* when applying the perforation technique on the Dadda 4:2 multiplier is depicted. All the possible combinations of perforation that lead to $NMED < 10^{-3}$ are presented. The spikes in the graph are caused by the increase in the order of perforation while its length is one and, at these points, the power and error are greater than those at the previous adjacent points. For this reason, for the rest of this chapter we consider an order of perforation as reference and vary its length in order to retain the monotony of the power-error curve. For more information regarding partial product perforation refer to Chapter 4

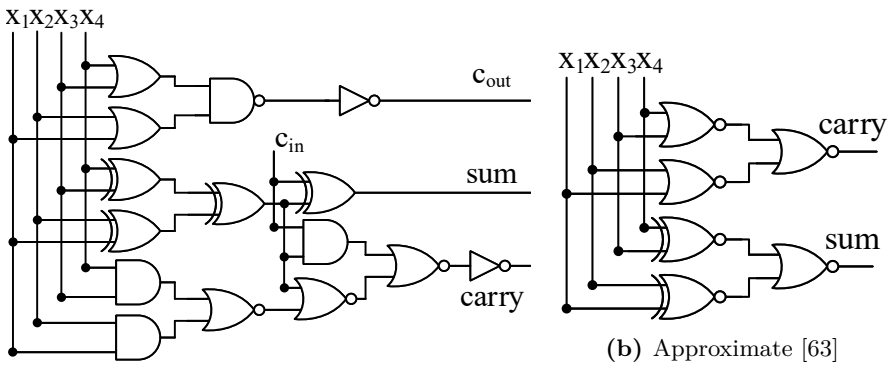
5.2.2 Logic Approximation

A widely used technique is the logic approximation applied to the structural level. This method targets mainly the simplification of a component's logic complexity (i.e. alteration of the truth table). The simplification of a unit's complexity reduces its power consumption, but induces computational errors. In order to apply the logic approximation on a multiplier, we use the approximate 4:2 compressor of [63]. Figure 5.5 depicts the gate level representation of both the accurate [113] as well as the approximate [63] 4:2 compressor. Note that in the approximate one the output carry is not calculated and is always '0'. Moreover, the accurate compressor features 52 transistors while the approximate one 26 [63], i.e., 50% transistor reduction. The carry and sum values of the approximate 4:2 compressor are calculated by:

$$sum = \overline{x1 \oplus x2} + \overline{x3 \oplus x4} \tag{5.1}$$

$$carry = \overline{\overline{x1x2} + \overline{x3x4}} \tag{5.2}$$

By replacing the accurate compressors (the boxes containing four dots in Figure 5.2) in a column of the Dadda 4:2 with imprecise ones, its power consumption decreases, however so does the accuracy of the multiplier; the more significant the column in which the compressors are replaced, the



(a) Accurate [113]

(b) Approximate [63]

Figure 5.5: The gate level implementation of the a) accurate [113] and b) approximate [63] 4:2 compressor

larger the error in the output. As a result, increasing the approximated columns decreases the power by increasing the error. Figure 5.4b illustrates the variation of the power consumption with respect to *NMED* and confirms that the *NMED* increases and the power decreases with the increase of the approximated columns. At lower error values of Figure 5.4b, the power decrease is faster than the error increase. This is explained by the architecture of the Dadda 4:2 tree. Moving from the right to the center of the tree, the number of the 4:2 compressors increases, while only the Least Significant Bits (LSB) are affected. As a result, the power reduction benefits more than the error increase. In contrast, moving from the center to the left, the number of the compressors in the columns decreases while the Most Significant Bits (MSB) are approximated. Thus, the error increase is more significant than the achieved power reduction. Finally, the power reduction attained by the application of approximate compressors is bounded. Even when all columns are approximated, the power consumption is 1.11 and 1.57 mW for the SPP and MBE designs respectively.

5.2.3 Voltage Over-Scaling

Voltage over-scaling (VOS) is applied at the circuit level. Unlike the other two techniques, VOS can be applied to every circuit without the need of special design or algorithm modification. VOS lowers the supply voltage below its nominal value. Decreasing the supply voltage reduces the circuit's power consumption, but produces errors caused by the number of paths that fail to meet the delay constraints [67], as qualitatively shown in Figure 5.6. The number of paths that violate these constraints varies and depends on the circuit's architecture. However, the paths that cannot reach the delay constraints are the longest ones and usually those that affect the MSBs of the output. As a result, when errors are produced by the application of VOS, they are of great significance compared to the accurate output. We implemented the VOS technique on the Dadda 4:2 multiplier, by using the Synopsys Composite Current Source model (CCS) [94]. CCS models have been proven to deliver signoff-level accuracy to within 2% of HSPICE simulation and are designed to be scalable for voltage, temperature and process parameters. In addition, they offer better accuracy than the Non-Linear Delay and Power models [94]. Retaining the original path timing constraints, we use CCS model to scale the supply voltage from 1V (nominal) to 0.80V and measure its power consumption and error metrics. Figure 5.4c shows the variation of the power consumption with respect to *NMED*. In average,

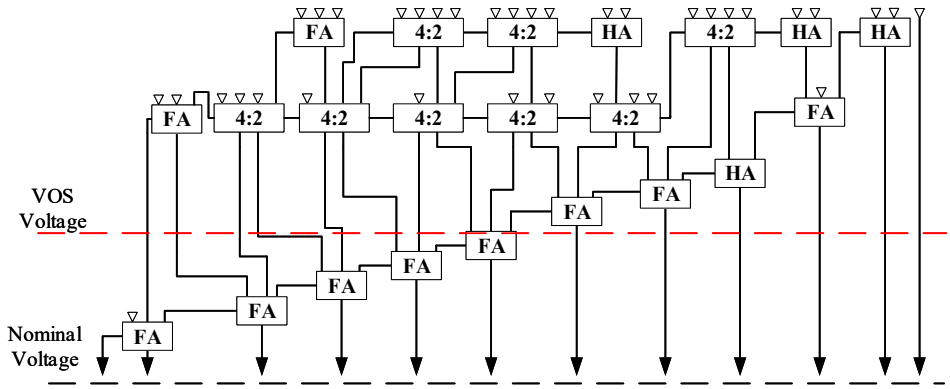


Figure 5.6: Example of path violation when applying VOS.

VOS delivers around 95% accurate results. However, whenever an error occurs its magnitude is high. Finally, the power reduction is bounded by the fact that the voltage supply cannot take lower values than the threshold voltage.

5.2.4 The need for hybrid approximation techniques

Previous analysis indicated that the approximation methods described above exhibit a variety of characteristics and power optimization potentialities. More specifically:

- The VOS technique, despite delivering more than 95% exact results, requires high decrease of the supply voltage in order to attain high power reduction, inducing high error in the output.

- Logic approximation offers moderate, though considerable, power reduction for small error values when some LSB columns are approximated. However, approximating more than the half columns is profitless, because it results in much faster error increase than the power decreases. As a result, despite the fact that the approximate compressors technique delivers small error, the achieved power reduction is limited. Furthermore, the logic approximation shows worse behavior when the MBE technique is preferred for the partial product generation.

- Finally, the introduced perforation method delivers significant power reduction in exchange for a considerable error. More specifically, for the MBE designs, it offers better results than the approximate 4:2 compressors. However, for the SPP designs and at smaller error bounds, the approximate 4:2 compressors achieve lower power values.

Taking into consideration the limited power reduction offered by the aforementioned approximation techniques, when they applied in isolation, we have been highly motivated to examine the power optimization potential

of hybrid designs generated by effective coordination of the aforementioned techniques.

5.3 HAM: Design Optimization Framework for Hybrid Approximate Multipliers

In this section we describe HAM, the proposed design framework that exploits multi-level approximation techniques to produce hybrid, low power approximate multipliers. The target design optimization problem can be formulated as follows:

$$\min_{\mathbf{x} \in \boldsymbol{D}} \left[\; Power(\mathbf{x}) \; \right] \tag{5.3}$$

subject to

$$\left[\; NMED(\mathbf{x}) \; \right] \leq \left[\; Max\_NMED \; \right], \tag{5.4}$$

where the optimization goal is to find the configuration $\mathbf{x}$, i.e. combination of approximation techniques that defines the overall design space $\mathbf{D}$, that minimizes power consumption for a given error constraint.

The computational complexity of our optimization problem increases significantly due to the need for an exhaustive search of all possible configurations of each technique. To assess its complexity, our problem can be viewed as a generalization of the well-known NP-complete KNAPSACK problem, i.e. the "nonlinear nonseparable bounded integer" KNAPSACK [112]. We can re-formulate the combinatorial problem of Eq. (1)-(2), by considering the *maximization of power reduction* rather than the minimization of power. The problem can be re-formulated as follows: given a knapsack (multiplier), select an integer number of objects (each object maps to a specific configuration of an approximation technique from each layer), each one with distinct cost (multiplier error) and gain (power savings), such that their collection will maximize the multiplier's power savings under a given constraint on its

output error. We consider that each object belongs to one of three categories (VOS, perforation and imprecise compressors) and that when applied to the multiplier (added to our knapsack), it has its own distinct effect on power and accuracy. In contrast to the conventional KNAPSACK, the cost/gain of each object is not constant and depends on the objects already placed in the knapsack. Thus, we use:

$$\mathbf{x} =< x_1, x_2, x_3 >\in \mathcal{N}^3$$

to denote the objects selected and we formulate:

$$\max_{\mathbf{x}\in \boldsymbol{D}} \left[\ Power\_Reduction(\mathbf{x}) \ \right] \tag{5.5}$$

subject to

$$\left[\ NMED(\mathbf{x}) \ \right] \leq \left[\ Max\_NMED \ \right], \tag{5.6}$$

where the nonlinear $Power\_Reduction(x) \in \mathcal{R}$ denotes the total power reduction and the nonlinear $NMED(x) \in \mathcal{R}$ the total error. This formulation corresponds to the definition of the nonlinear KNAPSACK problem. We note that the functions $Power\_Reduction(x)$ and $NMED(x)$ are not separable, as they depend on the combination of x_i's. Overall, the increased complexity of this little-studied version of KNAPSACK problem mandates the use of heuristic algorithms in practice [112].

In order to tackle the high computational cost, we developed an heuristic optimization algorithm that approximates the optimal solutions after a small number of synthesis/simulation runs. The proposed heuristic exploits/devises problem specific models to capture the behavior of the parameter space, i.e., configurations of approximation techniques, which are subsequently used during iterative optimization. The goal of the regressive models is to form an estimation proxy of the error and power to be further used during optimization search and not to be an accurate calculation of the error/power.

Error proxy of hybrid multipliers: Let us assume E_P, E_C, E_V and O_P, O_C, O_V to be the error functions and the multiplier output after applying the perforation, the approximate compressors, and the VOS techniques. O_A

refers to the output of an accurate design, while E_H refers to the error of the hybrid design when the three approximation techniques are applied. The perforation of some of the partial products results in skipping their generation and thus, they are not inserted in the accumulation tree. As a result a new, smaller tree that performs correct calculations is used to accumulate the remaining partial products and hence:

$$O_A = O_P + E_P. \tag{5.7}$$

Applying logic approximation to the new tree, i.e., replacing its exact 4:2 compressors with the approximate ones in some of its columns, induces a new error to the accurate output of the new tree and OA becomes:

$$O_A = (O_C + E_C) + E_P. \tag{5.8}$$

As mentioned in Section 5.2.3, VOS produces errors that mainly affect the critical paths, which in multipliers are formed at the MSBs, while logic approximation and perforation mainly affect the LSBs. Under this heuristic assumption that VOS generates errors in higher bit-rankings, we can write:

$$O_A = (O_V + E_V) + E_C + E_P. \tag{5.9}$$

Hence, the final error E_H of the hybrid design with the three techniques, can be estimated by:

$$\begin{aligned} E_H &= O_A - O_V \\ &= E_P + E_C + E_V. \end{aligned} \tag{5.10}$$

Power proxy of hybrid multipliers: Assume that RP_P, RP_C and RP_V are the respective Relative Power (RP) functions of applying the perforation, the approximate compressors, and VOS. For example the relative power when applying only perforation is:

$$RP_P = \frac{P_P}{P_A},$$

where P_P is the power of the design when applying perforation and P_A the power of the accurate one.

Similarly, for the RP, the perforation of some of the partial products reduces the power of the tree by a percentage that depends on the length of the perforation. The application of logic approximation on the new tree further reduces its power by a percentage that depends on the number of the compressors that are replaced, i.e., for λ approximated compressors over N compressors in total:

$$P_C = P_P \times \frac{\lambda}{N} \tag{5.11}$$

and thus:

$$\begin{aligned} RP_H &= \frac{P_C}{P_A} \\ &= RP_P \times \frac{\lambda}{N}. \end{aligned} \tag{5.12}$$

Finally, applying VOS on a circuit decreases its power consumption by a factor k:

$$k = (V_1/V_0)^2,$$

where V_0 is the nominal voltage and V_1 is the decreased voltage, that depends only on the percentage of its supply voltage decrease.

Therefore, applying VOS on the previous approximated tree with perforation and logic approximation will deliver a relative power for the hybrid design:

$$P_H = k \times P_C. \tag{5.13}$$

Hence,

$$\begin{aligned} RP_H &= k \times \frac{\lambda}{N} \times RP_P \\ &= RP_V \times RP_C \times RP_P. \end{aligned} \tag{5.14}$$

Validation of error and power proxies: In order to evaluate the accuracy of our regressive estimators, we calculate through synthesis and simulations the error and power of every hybrid model and compare it with the

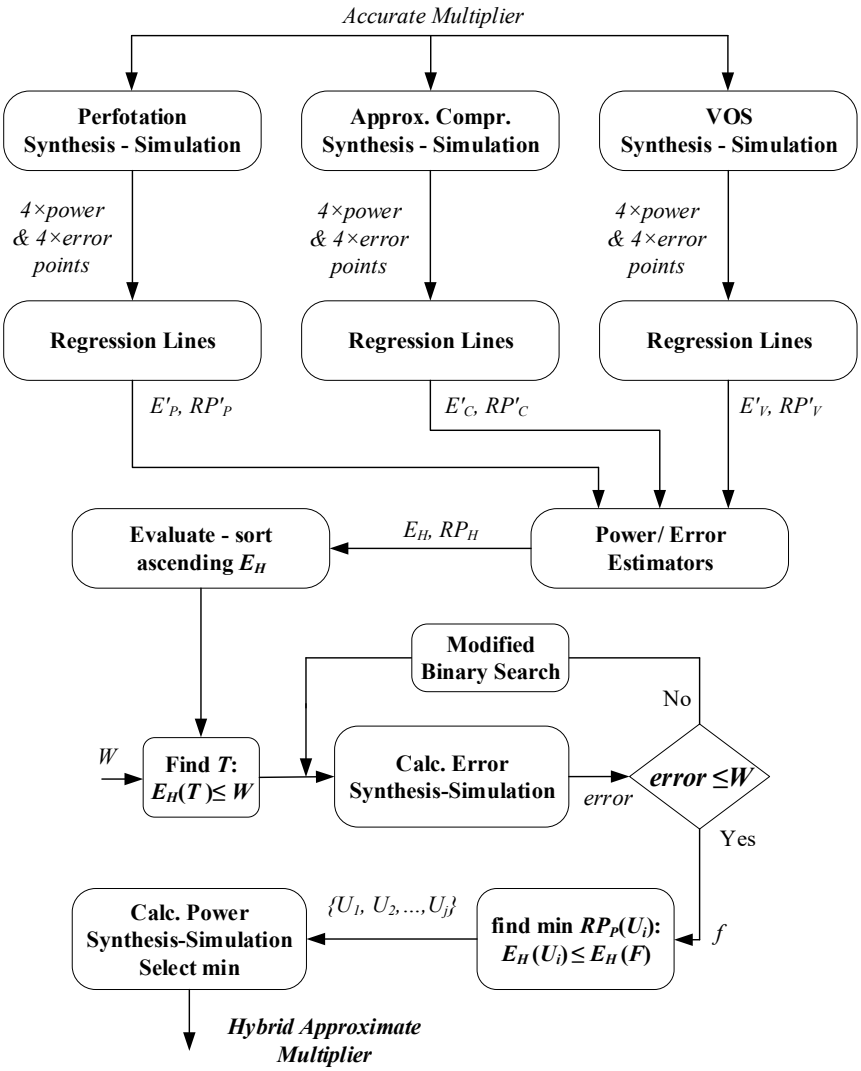


Figure 5.7: The proposed HAM framework. Its inputs are the multiplier architecture and the error bound W.

values obtained from the proposed estimators. The resulting mean square error (*MSE*) for the error estimator E_H is 1.2×10^{-9} and that of the *RP* estimator RP_H is 6.7×10^{-4}. Taking into account that the error of all hybrid designs is greater than 3.8×10^{-6} and their *RP* is greater than 2×10^{-1}, the proposed estimators are considered to offer satisfactory accuracy.

Design exploration and optimization: In Figure 5.7 the proposed framework is presented. The first step of HAM is to calculate the error and the *RP* functions of each approximation technique. We synthesize and simulate the accurate multiplier and four different configurations for each (single) technique and measure their power consumption and error. Using the points created, a regression curve is calculated to estimate each of the aforementioned functions. Since each of the examined approximation techniques preserves monotony and convexity regarding to the error and power values, the four points sampling is considered enough to produce accurate regression. For a 16-bit Dadda 4:2 multiplier with SPP, Figure 5.8 illustrates the measured points (black dots) and the regression lines (red ones) for each function . Table 5.1 presents the resulted regression lines and their respective *MSEs*. The *MSE* of each regression line is very low, verifying that the error and *RP* of each technique can be estimated precisely with only four points.

In Table 5.2 the accuracy of the proposed error/power estimators using as inputs the resulted regression lines from Table 5.1 are evaluated in comparison with: i) the same estimators trained with the overall design space, and ii) with well known regression techniques, i.e. Quadratic and Linear Regression, Regression Trees and a Neural Network with 10 neurons, trained with 13 points. As expected, the proposed estimators, have larger *MSE* when they use the regression lines than when they use the real measured power and error of the single technique approximate multipliers. Nevertheless, their *MSE* is still very small. Moreover, estimators using the regression lines require only 13 synthesis/simulation runs for training. In comparison with the rest estimators, trained with 13 hybrid designs, it is shown that these generalized regression models attain worse accuracy than the proposed estimators using the regression lines. Finally, we have to note that in order

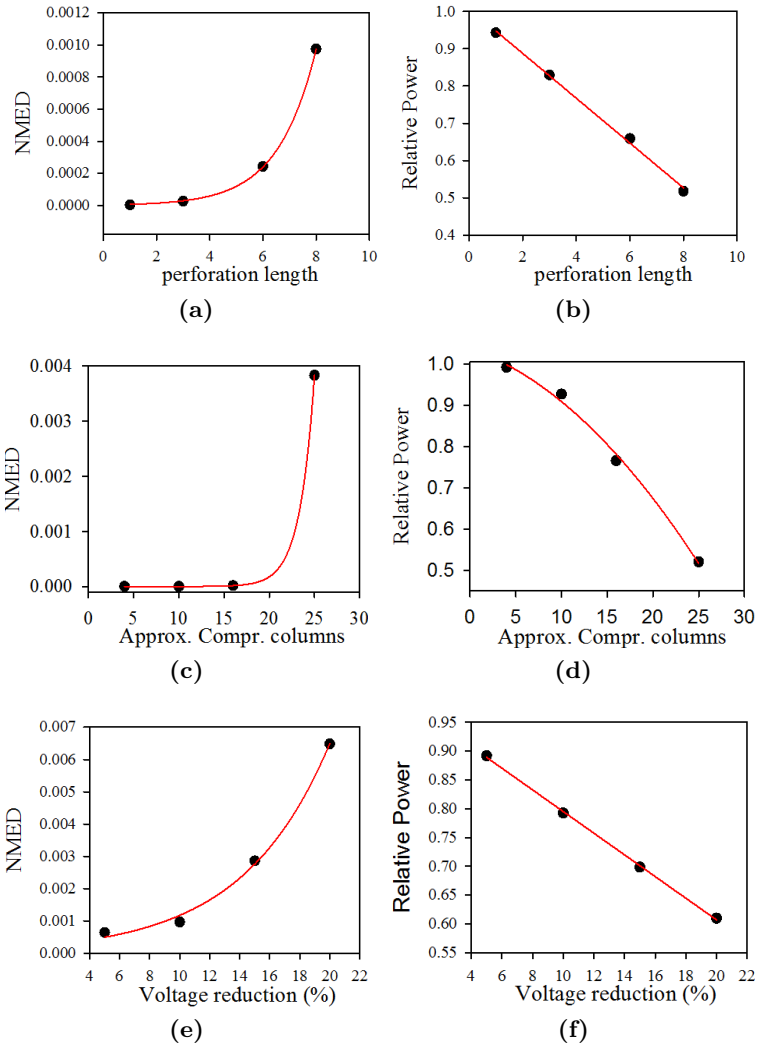


Figure 5.8: The computed regression lines of the perforation a) error b) Relative Power (RP), the approximate compressors c) error d) RP and VOS e) error f) RP. The black dots are the points resulted from the simulations performed by HAM in step 1.

to solve the hybrid approximate multipliers' optimization problem, retaining the relative order (in terms of power-error) of the hybrid designs is more important than precisely estimating their exact error and power values. The Pearson correlation coefficient of the proposed error and power estimators (using the regression lines as inputs) is 0.86 and 0.97, respectively. These very high correlation values show that our estimators exhibit a strong uphill linear relationship with the estimated values and that they efficiently capture the power-error ranking of the hybrid designs.

Table 5.1: Regression Lines of a 16-bit Dadda 4:2 Multiplier

Regression Lines	**MSE**
$E'_P(p)^a = 2 \times 10^{-6} e^{0.79p}$	1.3×10^{-10}
$RP'_P(p) = -0.057p + 1.003$	7.5×10^{-5}
$E'_V(v)^b = 2 \times 10^{-4} e^{0.16v}$	7.4×10^{-7}
$RP'_V(v) = -0.019v + 0.9826$	1.6×10^{-5}
$E'_C(c)^c = 6 \times 10^{-11} e^{0.74c}$	5.5×10^{-8}
$RP'_C(c) = -0.0015c^2 + 0.013c + 0.97$	1.4×10^{-3}

[a] p is the length of the perforation.
[b] v is the percentage decrease of the voltage vupply.
[c] c is the number of columns where logic approximation is applied.

Table 5.2: Comparison of the Error and *RP* Estimators

Estimator	**Error-MSE**	**RP-MSE**
Proposed Estimators using measured power/error	1.2×10^{-9}	6.7×10^{-4}
Proposed Estimators using regression lines	5.9×10^{-7}	1.0×10^{-3}
Quadratic regression	1.3	6.1
Linear regression	7.2×10^{-1}	1.0×10^{-1}
Regression Trees	2.4×10^{-1}	1.1
Neural Network	6.5×10^{-1}	5.6×10^{-1}

Algorithm 2 is an heuristic optimizer used to find the quasi-optimal configuration of approximation techniques for the specified error bound. This algorithm estimates the power and error of all the possible configurations of the hybrid designs using the proposed estimators and the previously calculated regression lines. The evaluation of their power and error characteristics is performed over the analytical RP_H and E_H proxies, and thus is performed very fast (i.e., a matter of seconds using Matlab). Then having estimated the error and power values of all the designs, Algorithm 2 sorts them in ascending error values. After the sort procedure, te first hybrid design with estimated error less than or equal to the error bound is extracted. This design is synthesized and simulated in order to evaluate its "real" error. In the case that its "real" error is not less than the upper bound, an iterative modified binary search procedure is invoked, in order to efficiently find the first hybrid design F that features "real" error (calculated through simulation) less than or equal to the error bound. Then, among the hybrid designs with estimated error less than or equal to the estimated error of F, those with the least estimated RP (if more than one) are synthesized and simulated. Finally, among all the synthesized and simulated hybrid designs (with "real" error less than the error bound) during this procedure, the one with the the minimum "real" power is the output of the algorithm/framework.

The proposed HAM framework does not depend on the multiplier's architecture nor the multiplier's width. The perforation and VOS techniques are directly applied on any architecture while the 4:2 compressors can replace any two successive full adders of the accumulation tree. The multiplier's width affects only the number of different configurations, and the only interaction between HAM and the input architecture is during the performed syntheses-simulations. In addition, the proposed optimization is scalable since searching is performed over analytical regression models. The sampling for training the models could be amenable to scalability issues. However, the error and power functions of each technique are strictly monotone retaining convexity independently from the bitwidth, thus, accurate regression can still be performed with a limited number of samples. Hence, the proposed HAM framework can be applied to any multiplier architecture and size without any efficiency loss.

Algorithm 2 Returns quasi-optimal hybrid designs for specified error bounds

INPUT: Estimator Functions: $E_H(p,c,v), RP_H(p,c,v)$
perforation values: $p \in [p_{min}, p_{max}]$
logic approximation values: $l \in [l_{min}, l_{max}]$
VOS values: $v \in [v_{min}, v_{max}]$
Error Bound: $MaxError$

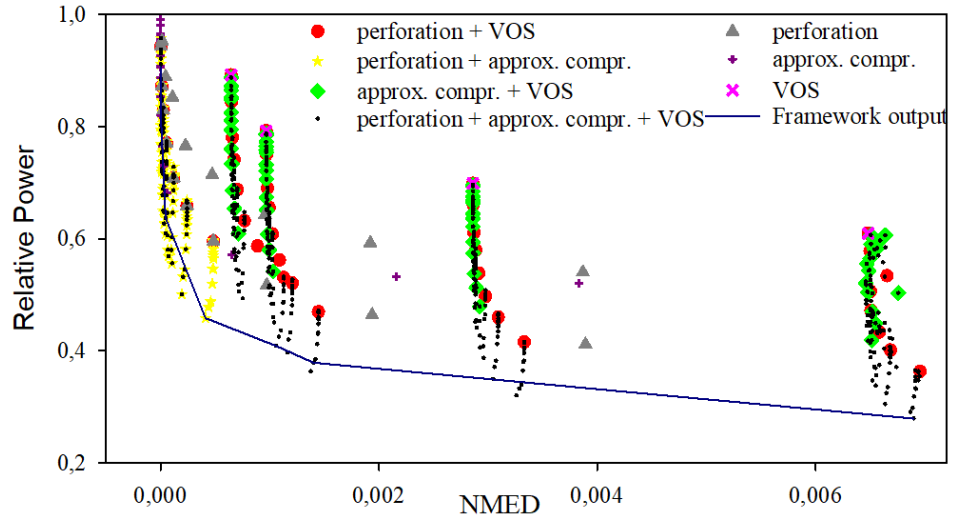
 # Estimate the error and RP of every hybrid design
for all p, c, v **do**
    AllHybrids←add(new Hybrid($p, c, v, E_H(p,c,v), RP_H(p,c,v)$))
end for
 # Sort the hybrid designs for ascending error
SortedError ← sort(AllHybrids, error)
 # Find the first hybrid design with estimated error ≤ MaxError
f ← search(SortedError, MaxError)
 # Among the hybrid designs with estimated error ≤ MaxError,
 # find the first design F with "real" error ≤ MaxError
SimT ← synth&sim(SortedError[f])
if SimT.Error > MaxError **then**
    imin ← 0, imax ← f
    while imin < imax **do**
        imid ← (imin+imax)/2
        SimR ← synth&sim(SortedError[imid])
        (SimR.Error ≤ MaxError)? imin ← imid + 1:imax ← imid - 1
    end while
    f ← imid
end if
 # Among the designs with estimated error ≤ estimated error of F,
 # find those with the minimum RP and synthesize-simulate them
SortedPower ← sort(SortedError[0:f], power)
for i←0 **to** f **do**
    if SortedPower[i].RP = SortedPower[0].RP **then**
        powerCandidates.add(synth&sim(SortedPower[i]))
    end if
end for
return min (powerCandidates, power)

5.4 Experimental evaluation

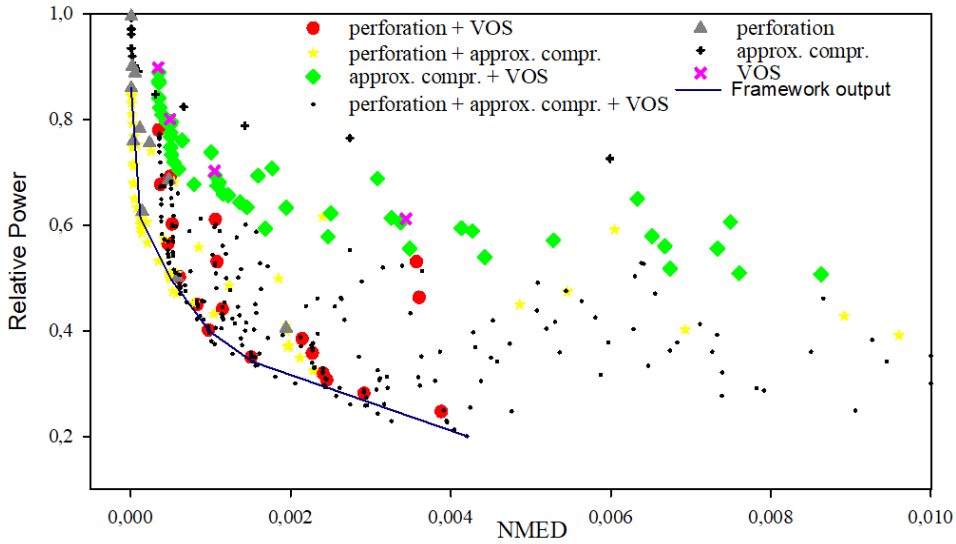
In this section we experimentally evaluate the efficiency of the proposed techniques, by providing comparative results against an exhaustive full-search exploration of all the possible hybrid designs, thus, proving the necessity and optimization potentials exposed by the adoption of hybrid approximation solutions. Moreover, we examine the efficacy of the proposed methodology to fastly extract (close to) Pareto optimal hybrid multipliers. Finally, we evaluate the efficacy of multi-level approximate multipliers by comparing them with the single-technique ones for specific error bounds. In the full-search exploration, more than 7000 hybrid designs in total (including both the SPP and MBE designs) have been evaluated, which translate to more than 3 weeks of experiments on an intel i7-2600k workstation with 16GB of RAM.

To create a hybrid approximate multiplier, first we select the perforation configuration, we perforate the selected partial products and generate the new accumulation tree. Then, we select the number of columns to be further approximated with logic approximation and in these columns we replace all the exact 4:2 compressors with the approximate ones. Finally, we synthesize the circuit that resulted from the previous two steps and decrease the supply voltage to the desired value. If only two approximate techniques of the possible three are applied, the respective step from the previous ones is omitted.

Figure 5.9 depicts the power - error (*NMED*) results of: i) all the explored hybrid designs, and ii) all the designs with only one approximate technique, that feature $NMED < 10^{-2}$. Each point in the graph represents a design with different configuration for each approximate method, grouped through the use of different colors. The points with the same color correspond to the application of the same approximate techniques, e.g. the yellow point constitute the application of both the perforation and the approximate 4:2 compressors on the multiplier. Different yellow points represent approximate hybrid designs with different configuration for either the perforation or the number of columns in which the imprecise compressors are used.



(a)



(b)

Figure 5.9: The relative power and error of all the single technique and hybrid approximate designs with $NMED < 0.01$ when a) SPP and b) MBE is preferred.

The blue line is the output of HAM. As shown in Figure 5.9, for both the SPP and the MBE designs, the application of hybrid approximate techniques exhibits higher quality design solutions. For the same error value, the lowest power is achieved by a hybrid design and there is no error region in which a single approximate technique achieves a (local) minimum. The hybrid designs produced by the framework are close to the optimal ones (Pareto points of Figure 5.9) that resulted from the conducted exploration, and are always below (or equal to) the approximate design with only one technique. For example, for the SPP designs and for error bound $NMED = 3 \times 10^{-5}$ the hybrid multiplier produced from HAM, consumes 11% less power than the single-technique design with the least power. For $NMED = 3 \times 10^{-4}$ and $NMED = 7 \times 10^{-3}$ the power savings are 30% and 48% respectively. When the MBE is preferred, for error bound $NMED \geq 5 \times 10^{-4}$ the power savings are more than 41%. For the error bounds examined in Figure 5.9 the proposed framework required 13 syntheses-simulations to calculate the regression lines and then an average of 4.71 syntheses-simulations to produce its output. The latter translates to an average speedup of 254× for the SPP hybrid designs and 140× for the MBE ones. The delivered speedup is defined by the different number of configurations that have to be explored in the exhaustive exploration, i.e., multiplier's width and the partial product generation method. For larger, multiplier sizes this speedup is expected to increase significantly. Finally, for low error values, hybrid models with two approximate techniques exhibit the lowest power values, while for larger ones, the hybrid designs that apply all three techniques attain the minimum power consumption. In conclusion, for small error values, the hybrid designs achieve lower power values than those with only one technique and for larger error values, they can offer further power reduction.

In Table 5.3, for two different error bounds, the hybrid designs produced by HAM are directly compared with the approximate designs that apply the state-of-art techniques of [63] and [57]. For maximum error 3.0×10^{-5}, among the single-technique designs, logic approximation on 17 columns offers lower area and power than perforation (with length 3), while VOS is not applicable as it cannot offer error less than this bound. The output of HAM is a hybrid design that applies perforation with length 3 and logic

Table 5.3: Evaluation of the Hybrid Designs at 500Mhz

Error Bound	Design	Area (um^2)	Power (mW)
3.0×10^{-5}	Hybrid (3,16,1.00)	2340	1.46
	Logic Approx. (17)	2737	1.64
	Product Perforation (3)	2842	1.86
	VOS	-	-
4.0×10^{-3}	Hybrid (7,17,0.85)	1615	0.72
	Logic Approx. (25)	2261	1.17
	Product Perforation (10)	1507	0.92
	VOS (0.82)	3401	1.12

approximation on 16 columns but does not apply VOS. This hybrid design attains 15% and 18% lower area than logic approximation and perforation and its power consumption is 11% and 22% lower, respectively. For higher error bound (4.0×10^{-3}), among the single-technique designs, VOS with 0.82V supply voltage achieves lower power than logic approximation on 25 columns, while perforation with length 10 delivers the lowest area and power. The hybrid design resulting from HAM applies perforation with length 7, logic approximation on 17 columns and VOS with 0.85V. Comparing this hybrid design with VOS and logic approximation, it achieves 53% and 29% lower area and 36% and 38% lower power respectively. Finally, compared with perforation, the hybrid design delivers 22% lower power but 7% more area.

Concluding, as shown in Figure 5.9 and Table 5.3 the hybrid multipliers constitute more efficient approximate design solutions compared to the single technique ones, delivering more energy and area efficient solutions for the same error bounds. The attained power and area savings depend on the error bound and as the quality requirements become more relaxed the hybrid multipliers are expected to deliver significantly higher gains. However, as shown, the induced complexity of generating optimal multi-level approximate multiplication circuits that satisfy an error bound is very high. The proposed methodology and the proposed power and error proxies tackle

this limitation and deliver very fast (more than 140× faster) close to optimal solutions. Finally, this delivered speedup is expected scale up as the multipliers size increases.

5.5 Conclusion

In this chapter, we address the problem of power-optimal approximate hardware multipliers that, for given error bounds, push power gains to the limits. We introduce multi-level approximation in the design of approximate computing arithmetic circuits and explore hybrid techniques that apply more than one approximate methods. The proposed hybrid approximate multipliers apply partial product perforation at the algorithmic level, approximate 4:2 compressors at the logic level, and voltage over-scaling at the circuit level. The design efficiency of the hybrid multipliers is evaluated by performing an exhaustive design exploration of all the hybrid and single-technique designs; showing that multi-level approximation (based on the error bound, applying two or three techniques simultaneously) produces more efficient approximate multiplier solutions. For the same error values, compared to the single-technique ones, the hybrid multipliers feature less power consumption as well as area complexity. Moreover, the delivered gains of the hybrid multipliers are expected to scale up as the multiplier's size and/or the error bound increase. Although the multi-level approximate multipliers are proven to constitute more efficient solutions, the design complexity of generating optimal approximate hybrid multipliers that satisfy an error constraint is very high. As shown, the size of the design space of hybrid multipliers (i.e., the number of the all the possible combinations of per technique configuration) is vast, even for a small 16-bit multiplier. Furthermore, the size of the design space increases significantly with respect to the circuit's (multiplier's) width. In this chapter, we presented the HAM framework, a design optimization framework that requires only a small number of synthesis and simulation runs to provide quasi-optimal solutions. We extensively evaluated the proposed heuristic framework and showed that it produces hybrid multiplier designs that closely follow the Pareto-front curve. HAM operates over analytical models using the proposed power-

error proxies and thus, all the evaluations are performed very fast. Compared to the exhaustive exploration, the HAM delivers very high speedup (more than 140×) in producing quasi-optimal hybrid designs. Our methodology can be applied to any multiplier architecture and, as presented in our experimental evaluation, produces hybrid designs that feature significantly less power consumption compared to existing state-of-the-art approximate multipliers. Finally, in a similar manner, HAM can be extended to support other arithmetic operators such as adders and subtractors (refer to Chapter 6) enabling the generation of a multi-level approximate arithmetic library.

Chapter 6

Multi-Level Approximate Hardware Accelerator Synthesis

Approximate computing forms a promising paradigm shift for energy efficient design by aggressively decreasing power consumption of inherently error tolerant applications. However, approximate computing architectures exacerbate the design complexity due to the diversity of inexact techniques and their impact on final circuit implementations. In this chapter, we introduce approximate accelerator synthesis which enables efficient inexact circuits implementations by leveraging the incorporation of diverse multi-level approximate techniques. We show the high efficiency of multi-level approximate accelerators and present METHADONE, a systematic approach for integrating multi-level approximation in accelerator synthesis under error and voltage island constraints. METHADONE applies multi-level approximation in very fine granularity in complex accelerator synthesis and, compared to the exhaustive exploration, produces, more than 589× faster, close to Pareto-optimal solutions. Additionally, METHADONE provides an efficient solution to the Voltage Islands Grouping problem originating from applying voltage over-scaling in such a fine granularity. METHADONE seamlessly extends typical behavioral/RTL synthesis tools and automatically selects the optimal multi-level configuration, relieving the programmer of this burden. The optimality and efficiency of METHADONE is evaluated against exhaustive design space exploration, showing, also, that it outperforms related state-of-the-art single-level techniques. This chapter is based on our publication in [88, 114, 115].

6.1 Introduction

Approximate computing has gained a lot of attention for energy efficient system design. Considering that a large number of application domains exhibit an intrinsic error resilience, approximate computing appears as a promising solution to improve their energy profile [86]. In such applications the 70%, on average, of their energy consumption is spent in amenable to approximation computations [116]. The key idea of approximate computing is to exploit this intrinsic error resilience and relax the accuracy of those computations to attain significant energy savings [117]. Recently, high interest is shown in the design of hardware approximate accelerators [41,58,76–79,86]. It is demonstrated that applying multiple approximation techniques in a coordinated manner leads to extremely improved circuit implementations in terms of energy efficiency [41, 86]. Notably, applying Voltage Over-Scaling (VOS) in cooperation to other approximate techniques leads to more efficient solutions [41]. The logic and algorithmic approximate techniques reduce the circuit's delay and decrease the number of critical paths [71]. Hence, applying VOS can further reduce the power consumption, at the cost of a small error increase, since fewer paths are affected by the voltage decrease [41,81]. Despite the high efficiency of multi-level approximation, its exploitation in the hardware approximate accelerator synthesis still remains an open issue due to the increased complexity and non-trivial interference effects among the differing approximations.

Hardware approximation mainly targets arithmetic circuits [41, 61, 63, 86, 117, 118], while research activities on more complex circuits, e.g., accelerators, are limited. Designing approximate hardware accelerators under error constraints heavily increases design time cycle due to the increased requirements for verifying both functionality as well as operating within the error bounds. To address this complexity, recent research focuses on automating the generation of approximate circuits. Given a set of feasible approximations, an approximate synthesis framework has to solve the optimization problem of assigning to the circuit components the respective approximate techniques and their configuration that minimize the energy, while satisfying the overall error bound. In example, when considering precision scaling, approximate synthesis has to annotate every arithmetic operation with the

respective numerical accuracy so that the energy is minimized and the error constraint is respected. Probabilistic pruning and logic minimization techniques are presented in [58], using a greedy algorithm to generate approximate circuits. These techniques eliminate circuit's components and simplify logic complexity according to the circuit's activity profile and output significance. Approximate adders are generated in [71] by reducing the carry chains and then decreasing accordingly the voltage value. In [99] the authors build architecture independent approximate multipliers by skipping the generation and addition of some partial products. In [76] and [77] the authors systematically synthesize logic approximate circuits by exploiting the "don't care" conditions. [78] applies several approximate transformation operators on the circuit's behavioral description and through a greedy approach identifies their optimal combination. The authors in [79] propose approximate accelerator synthesis through precision scaling using an integer linear programming problem formulation. [81] extends [79] and incorporates also voltage scaling to leverage the critical path delay reduction. However, in [71,81] voltage reduction is not used as an approximation method, mainly due to the increased complexity of modeling-quantifying errors due to VOS, limiting thus, the potential energy savings obtained by aggressively decreasing the voltage value. All the aforementioned state-of-the-art works exhibit several limitations since they are either very time consuming and/or do not leverage the full spectrum of approximate computing techniques. They mainly focus on the application of a single type of approximation to avoid design complexity, neglecting, thus, the potential benefits originated by the synergetic incorporation of multiple approximation techniques to structure the final accelerator circuit, e.g., none of these works incorporates logic and algorithmic approximations as well as VOS.

In this chapter, we leverage multi-level approximation to be effectively exploited in accelerator synthesis, thus drastically extending its scope. We present METHADONE, a framework for approximate accelerator synthesis, that given a behavioral specification and an error constraint produces the respective power-optimal multi-level approximate accelerator. The major contributions of this chapter are summarized as follows:

- We show that in the design of hardware approximate accelerators,

multi-level approximate designs constitute more efficient solutions compared to single-level ones in terms of both power and error solutions.

- We extend multi-level approximate arithmetic library of [41] to support addition and subtraction operators.

- We introduce fast and efficient techniques for fine-grained exploration and synthesis of multi-level approximate accelerators.

- We enable optimized accelerator implementations under voltage island constraints.

The proposed framework seamlessly extends typical behavioral and/or RTL synthesis tools by operating on the scheduled Data Flow Graph. We show that it provides significant speedup in synthesis time, more than 589×, delivering approximate accelerator solutions that follow closely the "exact" Power-Error Pareto-optimal design curve. Moreover, for the same error bounds, it delivers more power efficient solutions, 34% on average, compared to single level state-of-the-art approximate techniques.

The rest of this chapter is organized as follows: Section 6.2 describes the approximate arithmetic library that is used in the proposed METHADONE framework. Section 6.3 discusses the optimization problem of generating multi-level approximate accelerators, presents METHADONE and describes the approach we followed to address this problem. In Section 6.4 we experimentally evaluate the proposed framework in terms of optimality and performance in generating Pareto-optimal solutions as well as examining the efficiency of multi-level approximate solutions. Finally, Section 6.5 concludes this chapter, summarizing our findings and the major contributions of the proposed framework.

6.2 Multi-Level Approximate Arithmetic Library

At the core of the proposed approximate accelerator synthesis framework is a library of approximate arithmetic components. For each arithmetic module, the library contains a set of feasible implementations with differing power-error characteristics. To generate the library components, we extend the approach presented in [41] and given the accurate circuit description of an arithmetic operator and an error constraint, we return the respective power-optimal multi-level approximate design that satisfies the error bound. The approximate library supports also adders and subtractors, in addition to multipliers [41]. Approximation is applied in three distinct layers, i.e., the algorithmic, the logic, and the circuit. For the approximate multipliers, partial product perforation [99] is applied in the algorithmic level, while in logic level, inexact 4:2 compressor of [63] is used. For the approximate adders/subtractors, truncation is used in the algorithmic level and the approximate full adder of [61] is used in the logic level. In both cases, VOS is applied for approximations in the circuit level. VOS is applied to any circuit by keeping the operating frequency constant and decreasing the supply voltage below its nominal value [41]. The voltage decrease reduces the power consumption significantly, but erroneous outputs are generated due to the circuit paths that fail to meet the time requirements [92]. Figure 6.1a depicts the transistor implementation of the accurate full adder and Figure 6.1b the respective one of approximate full adder used in our library. As shown, the accurate full adder requires 24 transistors while the utilized approximate one only 11, i.e., 54% transistors reduction. The accurate outputs (Sum and C_{out}) of a full adder (Figure 6.1a) are given by:

$$Sum = A \oplus B \oplus C_{in} \tag{6.1}$$

and

$$C_{out} = AB + BC_{in} + AC_{in}. \tag{6.2}$$

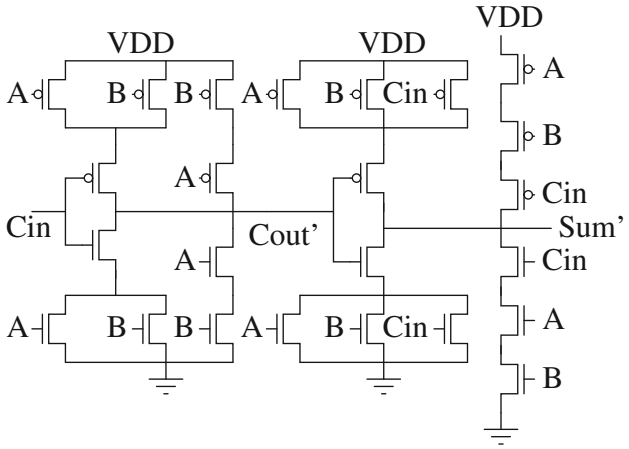
On the other hand, the Sum and C_{out} of the approximate full adder (Figure 6.1b) are calculated as follows:

$$Sum' = \bar{A}C_{in} + BC_{in} \tag{6.3}$$

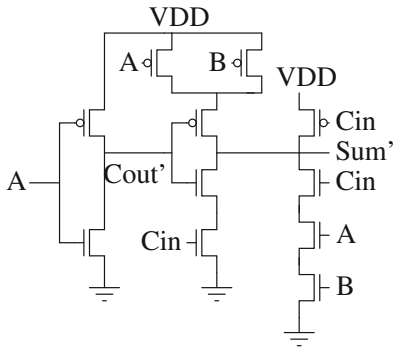
and

$$C'_{out} = A. \tag{6.4}$$

Equations 6.3 and 6.4 result to three errors in the calculation of the Sum output and two errors in the C_{out} computation.



(a) Accurate [61]



(b) Approximate [61]

Figure 6.1: The transistor level implementation of the a) accurate [61] and b) approximate [61] full adder.

In the proposed METHADONE framework, to avoid the design space explosion of the possible approximate configurations, only one state-of-the-art technique is used at every layer to build the final approximate accelerator. However, in a similar manner and without loss of optimality, METHADONE can be extended to support more than one approximate techniques at every layer. Note that the approximate techniques adopted in our library can be straightforwardly applied to any multiplier and adder/subtractor architectures. Partial product perforation [99] and truncation [56] are independent of the circuit's architecture and can be applied out-of-the-box to the respective circuit. The approximate full adders [61] can be seamlessly used instead of any accurate full adder and the approximate 4:2 compressors [63] can replace any accurate 4:2 compressors or two consecutive full adders. Finally, Voltage Over-Scaling [88] is a general approximate technique that can be transparently applied to any circuit.

Similar to the analysis presented in Chapter 5 for the hymbrid approximate multipliers (Figure 5.9), Figure 6.2 depicts, for a 16-bit ripple carry adder, the power - error (*NMED*) results of: i) all the hybrid adder designs, and ii) all the designs with only one approximation technique, that feature $NMED < 10^{-2}$. Each point in the graph represents a design with different configuration for each approximate method, grouped through the use of different colors. Furthermore, the blue line is the output of the proposed framework. As shown in Figure 6.2, similar to the hybrid approximate multipliers (Figure 5.9), the application of multi-level approximation techniques in the design of approximate adders produces higher quality design solutions. For the same error value, the lowest power is achieved by a hybrid approximate adder and there is no error region in which a single approximate technique achieves a (local) minimum. Moreover, as illustrated, the proposed framework (blue line) efficiently captures the power-error Pareto front derived by the exhaustive design space exploration. In the case of the approximate adders (Figure 6.2), the Mean Square Error value of the error (*NMED*) and power estimators is 2.95×10^{-5} and 2.19×10^{-3}, respectively. Finally, the respective Pearson correlation coefficients are 0.87 and 0.99, showing that the proposed estimators efficiently capture the power-error ranking of the hybrid approximate adders.

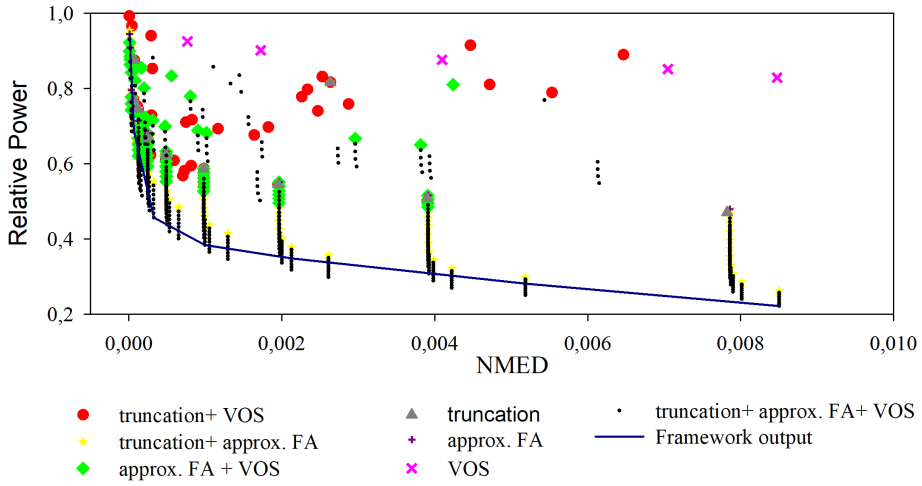


Figure 6.2: The power and error of all the single technique and hybrid approximate designs with $NMED < 0.01$ for a 16-bit ripple carry adder.

Figure 6.3 shows the derived Pareto optimal configurations with Mean Relative Error Distance [99] (MRED)≤10%, for an 8-bit Dada multiplier [63] and a 16-bit ripple carry adder. For readability reasons, in every segment (e.g. between 5% and 6%), if more than one configurations exist, only the one with the lowest relative power is depicted. The vector [L, A, V] declares the approximate configuration, i.e. L: number of columns (starting from the Least Significant Bit (LSB)) where logic approximation is applied, A: number of the perforated LSB partial products (truncated LSB) in multiplication (addition), and V: Voltage supply (Vdd) applied, with 1V being the nominal Vdd. As shown, the majority of the Pareto configurations apply approximations in every approximate computing layer, i.e., proving the importance of multi-level approximation in circuit efficiency.

The approximate arithmetic library of METHADONE supports adder, subtractor, and multiplier circuits. However, in similar manner, it can be extended to support also dividers. Adders, subtractors, multipliers, and dividers constitute the core and most complex components of all the Digital Signal Processing (DSP) applications [119] and, considering that DSPs are

perfect candidates for applying approximate computing [86], we cover a wide range of approximate computing applications.

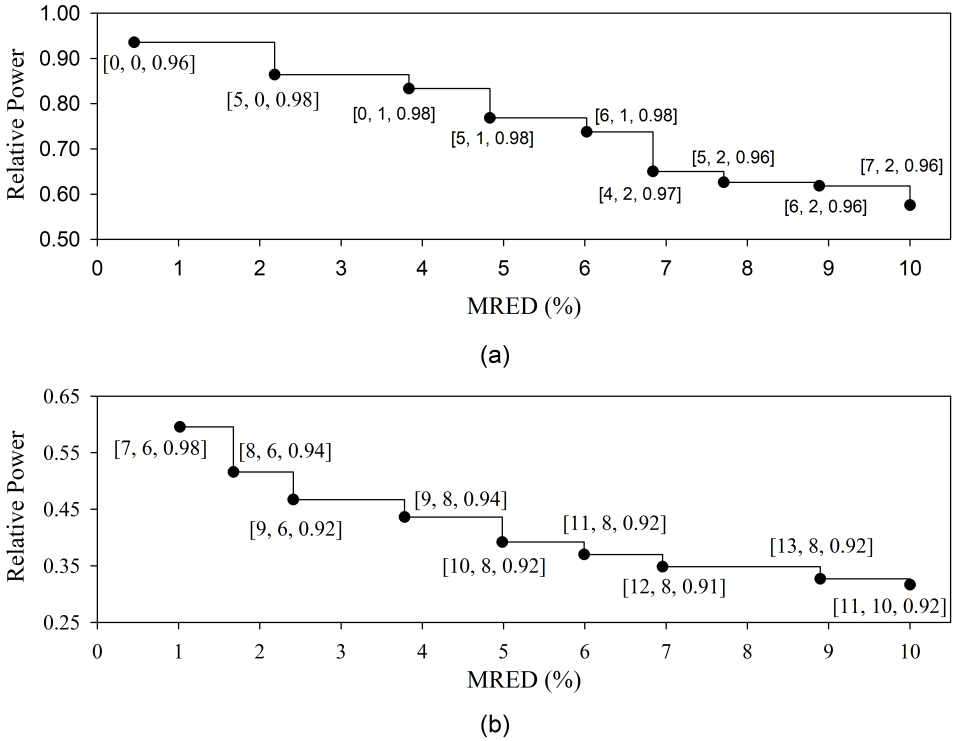


Figure 6.3: Pareto optimal configurations for MRED$\leq$ 10%, of a) an 8-bit Dada multiplier and b) a 16-bit ripple carry adder. The vector [L, A, V] denotes the multi-layer approximate configuration, i.e., level of logic, algorithmic approximation (L, A) and Voltage supply (V) (nominal Voltage: 1V).

6.3 METHADONE: Multi-lEvel approximaTe HArDware acceleratOr syNthEsis framework

The proposed framework, presented in Figure 6.4, accepts i) the behavioral specification of the kernel, ii) the global error bound of the accelerator, and iii) the number of the supported voltage islands and produces a power optimized approximate accelerator design operating within the specified error bounds.

Given the hardware description of the accelerator, METHADONE operates on its scheduled Data Flow Graph (DFG), that can be provided either manually by the designer or produced by HLS tools [120] after the scheduling and biding is performed. Each node of the DFG corresponds to a potential hardware operator. At first, the global error constraint of the the overall accelerator is distributed to the different nodes of the DFG. Targeting to exploit the efficiency of multi-level approximation, after the completion of the error distribution phase, a selection of the approximation techniques and their configuration is applied to every DFG node. Finally, a voltage island formation step is performed delivering a power optimized approximate accelerator design compliant with the respective error and voltage island constraints. The proposed framework is described more comprehensively as follows:

Error Distribution and Multi-Level Approximation Selection: In METHADONE, we combine the tasks of error distribution and selection of the multi-level approximation. Algorithm 3 assigns an approximate multi-level configuration to every node so that the DFG error bound is satisfied and its power dissipation is minimized. Each node's approximate configuration defines the node's unbiased error value, i.e., the node's error value when its inputs are accurate. Therefore, having assigned an error value to every node, we estimate how the error is propagated in the DFG by generating a DFG specific analytical error model, $Error_{DFG}$. The output error of a node is related to its own unbiased error (the error produced by the node when its inputs are accurate, i.e., unbiased from the error propagated to the node by its inputs) and to the error value of its inputs (error gener-

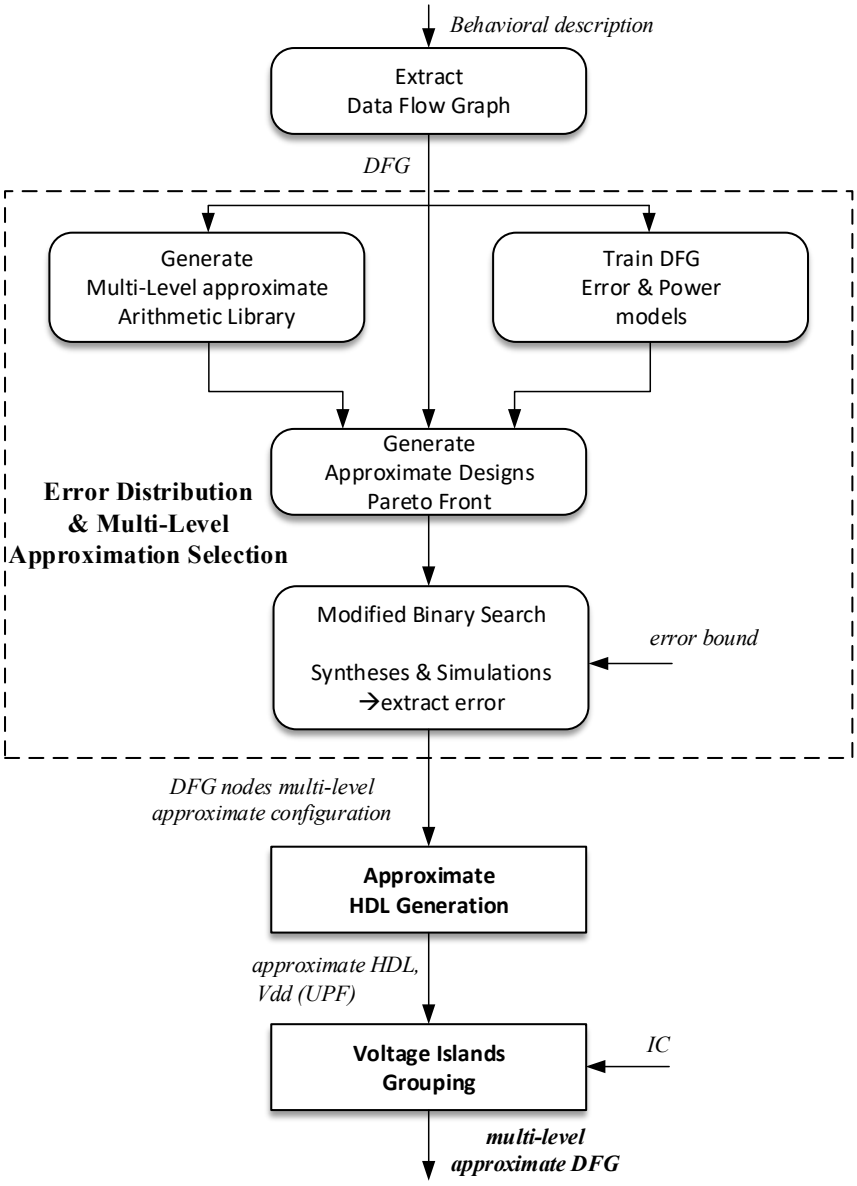


Figure 6.4: The proposed framework for multi-level approximate accelerators.

ated by previous approximate nodes). For this reason, to model the error propagation among the DFG nodes, we train an Artificial Neural Network (ANN), exploiting its ability to capture efficiently both linear and non-linear error distributions. In order to accomplish this, we leverage the fact that, after assigning a multi-level approximate configuration to a DFG node, we know its error when its inputs are accurate (form the approximate library). Therefore, to estimate the error propagation in the DFG, we generate several (but a limited number) different DFG configurations (aka nodes with differing multi-level approximation configurations) and calculate the overall DFG output error. The ANN model is trained on realistic DFG output errors, produced by first synthesizing the approximate (algorithmic, logic) DFG configuration and then applying a VOS-aware simulation on the post-synthesis gate-level netlist. The obtained results in coordination with the unbiased error value of every node are used to train the ANN to predict the DFG output error with respect to the unbiased error of every node. In order to fast and accurately estimate the power and error characteristics of circuits under VOS, we use the Synopsys Composite Current Source model (that delivers within 2% HSPICE accuracy) and our proposed simulator in [88] and extensively described in Chapter 3. The latter enables VOS-aware circuit simulations at gate-level and compared to SPICE accurate simulations delivers very high power/error accuracy (more than 98% on average) while attaining significant simulation time speedup (32× on average). Considering the vast variety of the different multi-level approximate configurations and high latency SPICE simulations [92], using [88] is mandatory in the implementation of METHADONE.

Generalized error distribution techniques [44] can be utilized straightforwardly. However, in this work, we leverage the basic principle of Pareto optimality found in multi-level approximate arithmetic library[1] to further reduce the execution runtime by pruning (without loss of optimality) the design space of approximate DFG configurations. The latter is achieved by limiting each DFG node to be configured only with an error-power Pareto configuration of the corresponding multi-level approximate arithmetic library generated using [41].

[1]Pareto optimality ensures no other configuration to exhibit lower power consumption for the same error constraint

In more detail, Algorithm 3 generates the multi-level approximate arithmetic library [41] of every DFG arithmetic node. It then produces K differing DFG instances by randomly sampling separately the configuration of each DFG node over the Pareto frontier associated with the respective DFG node. Each of the K DFG instances is synthesized and simulated using VOSsim [88] to obtain its 'power and error values. Then, the aforementioned ANN is trained to model the overall DFG's output error as a function of the DFG's nodes unbiased error values (which are obtained from the arithmetic library). For estimating the overall DFG's power consumption, $P_p = \sum_i (P_i)$ is used as a power proxy, where P_i is the power dissipation of DFG node i, taken from the arithmetic library. The approximate power-error Pareto set of the overall DFG is derived by iteratively searching over the $Error_{DFG}$ and P_p analytical models. Specifically, a binary search is employed on this power-error Pareto front to identify the configuration with error less or

Algorithm 3 Power optimized DFG nodes error annotation and multi-level approximation assignment

Input: 1. DFG,
2. Error Bound e_B,
3. Power model P_p
4. Training size K

1: ArithmeticLibrary ← For every DFG node generate the power-error Pareto-optimal multi-level approximate configurations according to [41]
2: TrainConfigurations ← generate K random DFG configurations by assigning to every node a Pareto configuration from the ArithmeticLibrary
3: DFGsimulation ← Synthesize & Simulate TrainConfigurations to compute the DFG error and power.
4: $Error_{DFG}$ ← train ANN over [TrainConfigurations, DFGsimulation]
5: Estimate power & error of all DFG configurations using ($Error_{DFG}$,P_p)
6: DFGParetoCurve ← extract the Pareto front using the estimated values
7: DFG_{opt} ← Binary search over the DFGParetoCurve. Find the DFG configuration with highest error e_o s.t. $e_o \leq e_B$. In each step of the binary search, the DFG is synthesized and simulated to compute its "exact" error.
return DFG_{opt}

equal to the error bound. Convergence towards optimality of the proposed algorithm is guaranteed through the inherent monotonicity of Pareto spaces and the high accuracy of the utilized models. During the binary search, a SPICE-accurate simulation of the DFG configuration under investigation is performed to accurately compute its error value.

Finally, we note that syntheses and simulations are invoked only for generating the approximate library, for training the ANN, and during the binary search. All the rest of the evaluations are performed over the analytical models, being thus, very fast.

Voltage island formation: Supporting operators with VOS approximation introduces voltage heterogeneity, since different inexact components may feature different V_{dd} values. However, the maximum number of allocated voltage islands is usually constrained by either technological or area/power efficiency constraints. Voltage Island Grouping (VIG) forms the final step of METHADONE aiming to limit the number of accelerator's voltage islands in a power efficient manner. Considering that decreasing the V_{dd} of a component increases its assigned error (i.e., violating the error constraint), VIG is only allowed to increase a component's V_{dd}. Thus, VIG solves the optimization problem of grouping and assigning the V_{dd} of the different approximate arithmetic units in voltage islands, while minimizing the power increase due to their V_{dd} increase. The VIG optimization problem is formulated as follows:

$$\min \Big(P_{DFG}(V'_1, ..., V'_n) - P_{DFG}(V_1, ..., V_n) \Big) \tag{6.5}$$

subject to:

$$|\{V'_1, ..., V'_n\}| \leq IC \text{ and } V'_i \geq V_i, \tag{6.6}$$

where n is the number of the DFG nodes, IC is the voltage island constraint, P_{DFG} is the power consumption of the DFG, V_i is the known V_{dd} of node i given from Algorithm 3, and V'_i is the decision variable, i.e., the new V_{dd} assigned to the node i for the grouping process. In order to reduce the time-complexity of this optimization problem, we reformulate it as follows: the power proxy of Algorithm 3 $P_p = \sum_i (P_i)$ is reused in order to prohibit the optimizer to resort to costly syntheses, simulations, and power analyses. However, although $P_i(V_i)$ has been computed in the first step of Algorithm 3

Algorithm 4 Power optimized VIG under error constraints

Input: IC: Voltage island constraint,
2. $V_1, ..., V_n$: V_{dd} assigned to the DFG nodes from Algorithm 3,
n: number of the DFG nodes

/*I is the set of all the possible V_{dd} values*/
1: $I \leftarrow \{\text{distinct}(V_1, ..., V_n)\}$
/*S is the set of the V_{dd} values after VIG. $S \subseteq I$.
2: $S \leftarrow \{\max(\text{I})\}$
3: **for** r=1; $r < IC$; r++ **do**
/*IC V_{dd} values can be selected in total*/
4: **for all** $v \in I$ **do**
/*examine if v will be included in the solution*/
5: **for** i=1; $i \leq n$; i++ **do**
6: $V_i' = min(t \in S \cup \{v\}$ s.t. $t \geq V_i)$
/*since $max(V_1, ..., V_n) \in S$ there is always a valid value for V_i'*/
7: **end for**
8: evaluate $P_{DFG}^{est}(V_1', ..., V_n')$
9: **end for**
/*select the v that minimizes P_{DFG}^{est}*/
10: $k \leftarrow v$ that minimizes P_{DFG}^{est}
/*add it to the solution set*/
11: $S \leftarrow \text{S} \cup \{k\}$
/*remove it from the next iteration's possible values*/
12: $I \leftarrow I/\{k\}$
13: **end for**
return S, $V_i' = \min(t \in S$ s.t. $t \geq V_i)$

(arithmetic libraries generation), $P_i(V_i')$ is unknown. Therefore, $P_i(V_i')$ is approximated by the model used in [41] for estimating a circuit's power consumption under VOS, i.e., $P_i(V) = c_i \times V^2$. Hence, $P_{DFG}(V_1', ..., V_n')$ is estimated by:

$$P_{DFG}^{est}(V_1', ..., V_n') = \sum_i (c_i \times V_i'^2). \tag{6.7}$$

Finally, c_i is replaced by $P_i(V_i)/V_i^2$ and (6.5) is written as:

$$\min\left(\sum_i P_i(V_i) \times \left(\frac{V_i'}{V_i}\right)^2 - \sum_i P_i(V_i)\right). \tag{6.8}$$

Since $V_i' \geq V_i$, (6.5)-(6.6) are reformulated as follows:

$$\min \sum_i P_i(V_i) \times \left(\frac{V_i'}{V_i}\right)^2 \tag{6.9}$$

subject to:

$$|\{V_1', ..., V_n'\}| \leq IC \text{ and } V_i' \geq V_i. \tag{6.10}$$

We solve this optimization problem using the greedy Algorithm 4 where we have to select IC different V_{dd} values and the cost of the selection is given by P_{DFG}^{est}. The output of Algorithm 4 is a set S of IC distinct values and the voltage value of every DFG node i (V_i') after VIG is given by $V_i' = \min(t \in S \text{ s.t. } t \geq V_i)$. Briefly, Algorithm 4 iterates IC times (line 3) and in each iteration evaluates the selection cost (6.9) for all the possible voltage values, i.e., $\forall v \in I$ (line 4). In every iteration, it selects the element k that minimizes the selection cost (line 10), adds k to the solution set and removes it from the possible voltage values of the next iteration (lines 11-12). Solving the VIG optimization problem using exhaustive exploration, in the worst case where all the nodes feature a different V_{dd} value, requires $n!$ circuit simulations. On the other hand, Algorithm 4 requires, in the worst case, $(n-1)\times...\times(n-IC+1)$ evaluations of P_{DFG}^{est}.

6.4 Experimental Evaluation

In this section, we experimentally evaluate the efficiency and optimality of METHADONE by providing comparative results with respect to both i) conventional approximate accelerator design considering only single-level approximation, as well as ii) multi-level approximate accelerators derived by exhaustive full-search exploration strategy.

The following benchmarks are considered: i) a 3 stage pipelined 3×3 matrix multiplication [63], ii) a 5 stage pipelined sobel filter [116], and iii) a 8 stage pipelined 1-D Discrete Cosine Transform (DCT) [61]. These benchmarks are representative approximate computing use cases, widely used in state-of-the-art works [86,116]. Matrix multiplication is a core kernel of machine learning and digital signal processing [63, 116], while sobel and DCT are typical image processing benchmarks [61, 116]. These domains exhibit high error resilience [86], being thus, perfect candidates for approximate computing. For every benchmark, two random generated input datasets of 50,000 inputs are used (one for the training and one for the evaluation of our framework). All the experiments are conducted on a cluster of 3 dual Intel Xeon E5-2658A servers with 128GB RAM each. The logical '1' is set to 0.7V, the logical '0' to 0.3V and voltage related information (e.g., voltage islands and value) is described using the Unified Power Format (UPF). Although UPF supports level shifters, they are not used in our implementations in order to avoid the induced overhead. Approximate configurations that result to unknown states due to VOS, are considered to have infinite error value. Industrial strength tools are used for the syntheses and simulations, i.e., Synopsys Design Compiler and Prime Time, Mentor Questasim, and the TSMC 65nm standard cell library. For every design, its relative power is reported with reference to the respective accurate one and its error value is calculated through exhaustive VOS-aware simulation. All the designs are synthesized and simulated at the critical path delay of the respective accurate one and the compile ultra command is used for the synthesis. When applying VOS the voltage value (V_{dd}) is decreased by 1%-20%, i.e, typical values in approximate computing [92].

In order to evaluate the optimality of the proposed framework, we compare the designs produced by METHADONE with the "exact" Pareto-optimal designs derived through exhaustive full-search design space exploration, i.e., solving the optimization problem of finding the power optimal multi-level approximate configuration that satisfies the error bound with brute force. For error bounds ranging from 1% up to 10%, Figure 6.5 compares in a graphical manner the DFG configurations produced by METHADONE with the Pareto-optimal ones provided by exhaustive design space exploration for the sobel filter and matrix multiplication benchmarks. In the exhaus-

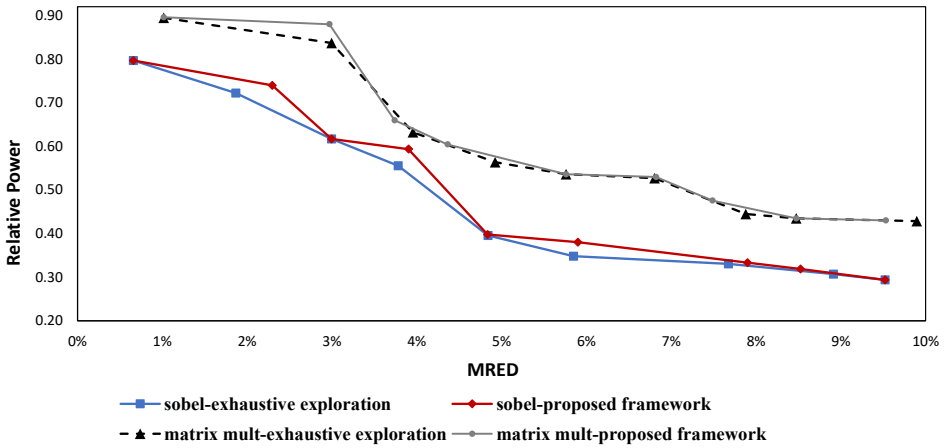
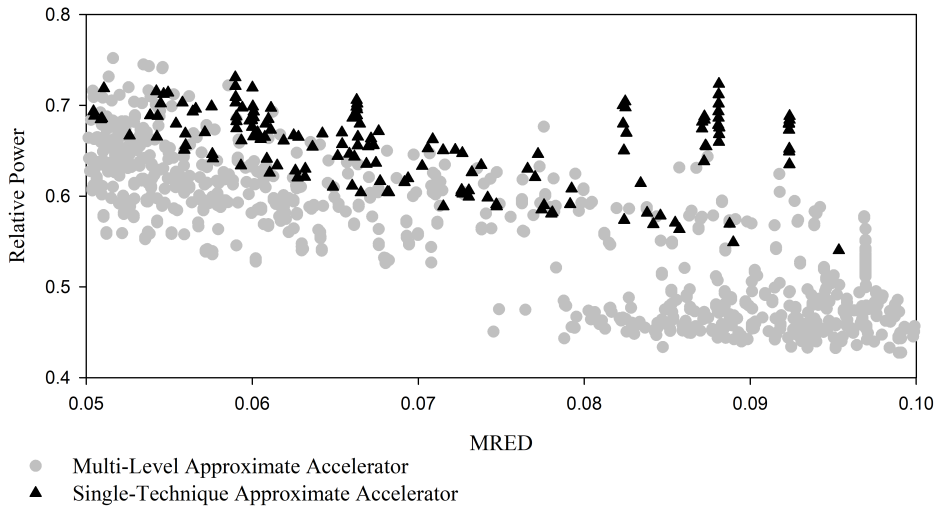


Figure 6.5: The power-error Pareto optimal DFG configurations, for error bounds from 1% to 10%, produced by the proposed framework and by exhaustive design space exploration for the Sobel and matrix multiplication benchmarks.

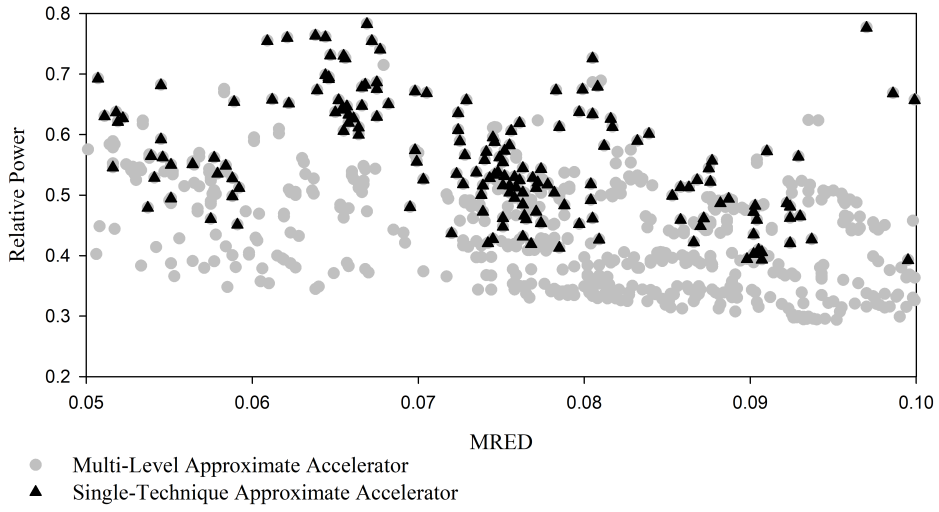
tive exploration, the designs' power and error characteristics are obtained by performing SPICE-accurate simulations (i.e., golden result) [71, 81, 92] using Synopsys CustomSim. In our optimization framework, we precisely estimate the power and error values of the examined designs by performing post-synthesis VOS-aware timing simulations using [88]. Then the designs extracted by METHADONE are simulated using CustomSim to obtain their "exact" power and error values. In order to perform SPICE simulations the gate-level netlist produced by Design Compiler is fed to Mentor Calibre to generate the SPICE netlist and the latter is simulated using CustomSim. To focus on the effectiveness of the optimization Algorithm 3, no voltage island constraints are considered in Figure 6.5. It is observed that for both benchmarks the designs produced by the proposed METHADONE framework are very close to the ones derived by exhaustive exploration. This is also confirmed by the low Mean Square Error (MSE) values of the utilized error and the power analytical models, i.e., 9×10^{-5} and 6×10^{-7} and the respective Pearson correlation coefficients are 0.90 and 0.98. The MSE and Pearson values of the error and power models of the proposed framework are evaluated over the overall design space of the exhaustive exploration. Com-

pared to the exhaustive exploration, for the matrix multiplication benchmark, METHADONE trained with 120 configurations attained a speedup of 589×. The respective speedup for the the Sobel filter, trained with 200 configurations, is 1827×. The speedup gain mainly depends on the DFG size (i.e., DFG configurations that must be synthesized and simulated in the exhaustive exploration), the number of the configurations used to train the ANN, and the simulation time speedup delivered by [88]. As a result, for accelerators with larger DFGs, higher speedups are expected. Selecting the size of the ANN training dataset is subject to machine learning theory, but we show that even for small training sizes (120 configurations in matrix multiplication and 200 configurations in the Sobel filter) the MSE values of the error estimator are very small. Note that without using [88] to perform VOS-aware simulations, the delivered speedups would drop to 34× and 63×, respectively. Considering the increased time needed to perform SPICE-accurate simulations for the DCT benchmark (58h per simulation) and the large number of its possible multi-level approximate configurations, we haven't performed an exhaustive design space exploration for this benchmark.

In Figures 6.6 and 6.7, the efficiency of applying multi-level approximation for inexact accelerator design is evaluated by comparing it with existing state-of-art single level techniques, i.e., logic approximation (approximate full adders [61], approximate 4:2 compressors [63]), algorithmic approximation (perforation [99], truncation), and VOS, when applied in isolation. We note that among these techniques, only the algorithmic approximation can be applied directly on the circuit's DFG without the need of our framework. In Figure 6.6, for the matrix multiplication and sobel benchmarks, we depict all the multi-level configurations as well as the single-level ones that resulted from the exhaustive exploration and feature an error value of $5\% \leq MRED \leq 10\%$. As shown, for both benchmarks, the multi-level approximate designs (coloured in grey) constitute far more efficient solutions compared to the single-technique ones (coloured in black). In Figure 6.6, for every error value it always exists a multi-level approximate design that features at least 10% less power consumption compared to the most efficient single-technique one. In Figure 6.6, similar to Figure 6.5, no voltage islands constraint is considered. On the other hand, in Figure 6.7, different



(a) Matrix Multiplication



(b) Sobel

Figure 6.6: The approximate multi-level (in grey) and single-technique (in black) configurations of the a) matrix multiplication and b) sobel benchmarks that exhibit error value (MRED) between 5% and 10%.

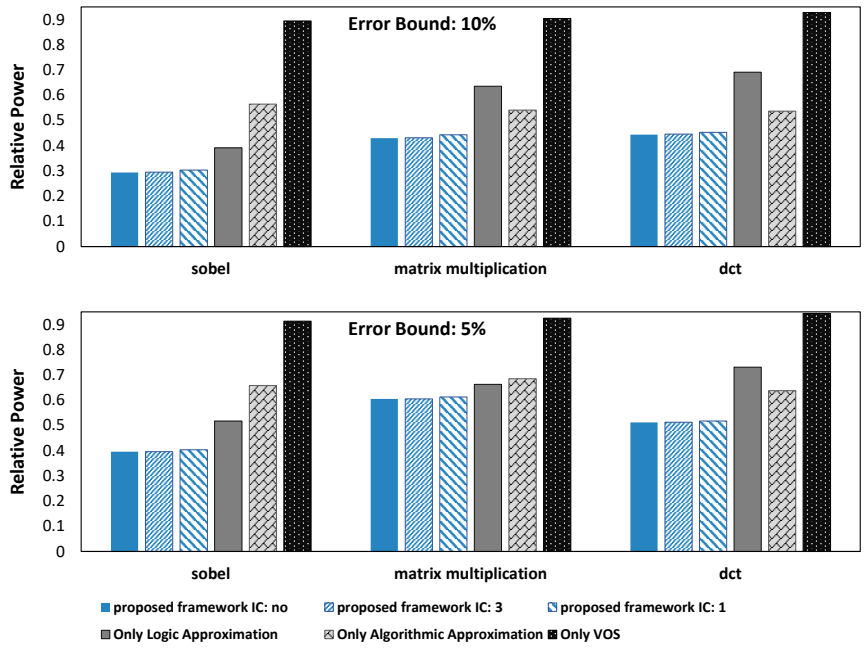


Figure 6.7: Comparison of METHADONE (with varying *IC* values) with state-of-art single layer techniques for error bounds 5% and 10%. The proposed solutions apply inexact techniques from every approximation layer.

voltage island constraints are examined. For error bounds 5% and 10%, the relative power (power of approximate/power of accurate) of the designs provided by METHADONE is compared against the single level approximation techniques. In single level approximation, an exhaustive exploration is performed in order to find the best single-level DFG configuration, i.e., satisfies the error bound and delivers the highest power reduction. As shown, for every benchmark and error bound, the solutions provided by the framework outperform the single-level techniques even in the worst case, where only one voltage island is permitted (IC=1). For example, for error bound 5% and when no island constraint is applied, the designs produced by METHADONE exhibit 61%, 40%, and 49% less power consumption compared to the accurate design for the Sobel, matrix multiplication and DCT benchmarks,

respectively. Compared to the most efficient single-level approximate technique the respective values are 24%, 9%, and 20% less power consumption. When IC=1 the respective power gains slightly decrease, attaining the values of 60%, 39%, 48.5% and 22%, 8%, 19%. For error bound 10% and for no IC contain, the proposed framework produces solutions with 71%, 57%, and 57% lower power consumption than the accurate design and 43%, 21%, and 30% less power consumption compared with the most efficient single-level approximation. Finally, note that even for very small DFG graphs, e.g., graphs with only one node, the proposed METHADONE framework produces more efficient solutions, e.g., as shown in Figure 6.3 the Pareto front of both the approximate adders as well as multipliers consists of multi-level approximate configurations.

The Sobel filter, that mainly uses adders and subtractors, achieves higher power reduction compared to the other two benchmarks. This is because addition attains higher power savings compared to multiplication when multi-level approximation is applied, as presented in Figure 6.3. Finally, Sobel and DCT that feature larger DFGs than matrix multiplication, exhibit higher power reduction compared to the single-level approximation techniques. As shown in Figure 6.3, multi-level approximation of arithmetic components is more efficient than the single-level one and for the same error bounds achieves higher power reduction. Therefore, the more arithmetic nodes in a DFG, the more efficient solutions METHADONE can produce compared to single-level approximation.

Finally, for the matrix multiplication benchmark, Table 6.1 presents an example of the V_{dd} values assigned from VIG (Algorithm 4) to every DFG node. Compared with the exhaustive exploration for solving the voltage islands grouping problem, the MSE value of P^{est}_{DFG} used in VIG is 7.81×10^{-7}. Furthermore, for the specific benchmark and the error bounds, Algorithm 4 returned exactly the same solutions with the exhaustive exploration. For error bound 10%, Algorithm 3 produced a solution with 4 different V_{dd} values (in 5 nodes). Applying VIG with IC 3 down to 1 increases the power consumption by 0.2%, 1%, and 3%, respectively, compared to the design with no voltage islands constraint. Similarly, for error bound 5%, the initial solution comprises 3 voltage islands and when VIG is applied

Table 6.1: V_{dd} assigned from VIG to the Matrix Multiplication DFG.

Error Bound	*IC*	M1[a]	M2	M3	A1	A2	Power Overhead
5%	no	0.98	0.98	0.98	0.92	0.94	-
	3	0.98	0.98	0.98	0.92	0.94	0%
	2	0.98	0.98	0.98	0.94	0.94	0.5%
	1	0.98	0.98	0.98	0.98	0.98	2%
10%	no	0.97	0.97	0.95	0.91	0.93	-
	3	0.97	0.97	0.95	0.93	0.93	0.2%
	2	0.97	0.97	0.97	0.93	0.93	1%
	1	0.97	0.97	0.97	0.97	0.97	3%

[a]M1-M3 are multipliers and A1-A2 adders. Power Overhead is reported with respect to the configuration with no island constraint.

with *IC* 2 and 1, the power consumption is increased by 0.5% and 2%, respectively.

6.5 Conclusion

In this chapter we propose METHADONE, an approximate synthesis framework that leverages the efficiency of multi-level approximation for producing energy optimal hardware accelerators. The proposed framework takes into consideration both error and voltage island constraints, exploits analytical power and error models to distribute the error budget to the data flow graph (DFG) nodes, and produces their approximate counterpart. METHADONE extends the approximate framework HAM, presented in [41], to support also multi-level approximate addition and subtraction besides the originally targeted multiplication. However, and most importantly, this does not form the main contribution of this chapter. Specifically, even though multi-level approximation is proposed in [41] for a specific family arithmetic circuits, i.e. multiplier, its possible benefits haven't been yet explored in larger and more complex hardware designs like accelerators. The application of multi-level

approximation in the hardware accelerator design is not trivial and hasn't been comprehensively analyzed in the related literature. Although someone could make assumptions regarding the possible benefits of multi-level approximate accelerators, the production of multi-level approximate accelerators that satisfy a given error bound has not been efficiently addressed. The latter is verified by the limited research in multi-level approximate accelerator synthesis (especially the lack of VOS application) and by the size of the design space of multi-level approximate accelerator synthesis. We further note that the design space size can explode according to the accelerator's DFG size, thus we strongly believe that the introduction of automated exploration, optimization and synthesis techniques for efficient approximate accelerator architectures are considered of great importance in the design and approximate computing community. The main novelties and contributions of this work can be summarized as follows: We apply for the first time multi-level approximation in very fine granularity in complex hardware accelerator synthesis and demonstrate its efficiency. Furthermore, we provide a synthesis framework (METHADONE) to support efficient multi-level approximation in accelerator design, producing very fast (more than 589×) close to optimal solutions (with respect to the exhaustive design space exploration). Additionally, we propose an efficient solution to the Voltage Islands Grouping problem originating from the VOS application in such a fine granularity. Finally, the proposed framework can seamlessly extend typical behavioral and/or RTL synthesis tools (it operates on the circuit's DFG) and automatically selects the optimal approximate configuration, relieving the programmer of this burden. The optimality and efficiency of the proposed framework is compared with respect to exhaustive design space exploration, showing, also, that it outperforms single-level state-of-art approximate techniques and proving the efficacy of multi-level approximation in the design of approximate hardware accelerator circuits.

Chapter 7

Conclusions

This chapter discusses the conclusions derived from this Ph.D. Thesis. It also summarizes the novelties and presents the future extensions of the dissertation.

7.1 Summary of Ph.D. Thesis

We are at the threshold of an explosion in new data, produced not only by large, powerful scientific and commercial computers, but also by the billions of low-power devices of internet connected devices. Energy efficiency is now a first-class design constraint in the ICT domain and radical departures from conventional approaches are needed to sustain and further improve the performance and efficiency of the computing systems. Approximate computing forms a radical paradigm shift in systems design and operation, based on the idea that we are hindering computer systems' efficiency by demanding too much accuracy from them. Considering that a large number of application domains are inherently tolerant to imprecise calculations, approximate computing appears as a promising solution to reduce their power dissipation. This inherent error resilience of these applications allows approximate calculations to be performed by relaxing the numerical exactness of such applications, thus significantly reducing their energy profile. However, resilient applications are not a license for computing systems to

abandon predictability in favour of arbitrary errors. Approximate computing should be carefully applied, in a disciplined manner, delivering accuracy guarantees. In this thesis, we focus on the design of hardware approximate accelerators and our goal is to efficiently address some of the major problems and limitations of the respective field:

- Limited research on energy efficient approximate multiplication circuits.
- Provide architecture/circuit independent approximation techniques.
- Maximize the energy savings of an approximate accelerator with respect to satisfying the error constraints.
- The increased requirements for verifying functionality, as well as operating within the error bounds greatly increase the design time cycle of approximate hardware accelerators.
- Design automation of power-optimal approximate hardware accelerators under error constraints.

In this thesis, motivated by these limitations and having as our main objective to provide automated solutions that enable efficient and straightforward application of approximate computing in the design of imprecise hardware accelerators; we presented a general approximation technique for generating approximate multipliers, i.e., Partial Product Perforation, and four automated frameworks, i.e., VOSsim, Partial Product Perforation framework, HAM, and METHADONE.

Hereafter, we present a brief summary of our work, the introduced novelties and how through the proposed frameworks we addressed the aforementioned problems. VOSsim operates over widely used, industry strength tools and can seamlessly extend typical hardware design flows. VOSsim can be used as an alternative to SPICE-level simulations, and by performing voltage-aware simulations at gate-level, it delivers very fast (34× faster on average) and accurate quantification of the power-error characteristics of

voltage over-scaled approximate circuits. Applying logic and algorithmic approximations leads to simpler circuits with fewer critical paths and decreased critical path delay, hence, applying also voltage over-scaling, can further reduce the power consumption, at the cost of a small error increase, as fewer paths are affected by this voltage decrease. However, the increased complexity of performing a SPICE simulation and the high simulation time it requires, restrict the voltage over-scaling application as an approximation method. VOSsim enables the approximate computing research community to incorporate voltage over-scaling in their works and exploit its synergetic cooperation with approximation techniques from the logic and algorithmic layers to produce more efficient approximate circuits. In order to address the limited research activity on approximate multipliers we proposed Partial Product Perforation and HAM. Partial Product Perforation is a general approximation technique for generating imprecise multipliers and can be applied with error guarantees to any multiplier architecture. Product Perforation is applied to any multiplication circuit in exactly the same way, without depending on its architecture, and the imposed error depends only on the perforation configuration. A significant advantage of this technique is that the produced error is analyzed in a mathematically rigorous manner, being predictable and bounded. However, although the output error is independent of the multiplier architecture, the delivered power-area savings vary. For this reason, we proposed the Partial Product Perforation framework that, given an error bound, extracts the power-area Pareto optimal multiplier architecture-perforation configurations that satisfy this error bound. Similar to the Partial Product Perforation, HAM is an automated framework that generates approximate multipliers that satisfy user provided error constraints and can be applied to any multiplier circuit. HAM aims to maximize the power gains of the approximate hardware multipliers by examining and applying multi-level approximation. HAM introduces the Hybrid Approximate Multipliers that apply Partial Product Perforation at the algorithmic level, approximate compressors at the logic level, and voltage over-scaling at the circuit level. HAM demonstrates that, for the same error bounds, multi-level approximate multipliers are more power efficient solutions compared to the single-level ones, demonstrating, thus, the efficiency of multi-level approximation. Considering the vast number of different multi-level approximate configurations, the design complexity of power

optimal hybrid multipliers under error constraints increases significantly. The HAM framework exploits power-error proxies and uses an heuristic algorithm to produce close to optimal (compared to an exhaustive design space exploration) hybrid multipliers, requiring only a small number of syntheses and simulations. Following the impressive results of multi-level approximation for generating approximate multipliers (demonstrated in HAM), we extended the proposed framework to also support the addition and subtraction operations and we produce a multi-level approximate arithmetic library. The METHADONE synthesis framework uses this library and VOSsim to enable the application of multi-level approximation in more complex circuits, i.e., hardware accelerators. METHADONE operates on the data flow graph of the accelerator and thus, it can be applied to any accelerator circuit. The proposed framework seamlessly extends typical behavioral and/or RTL synthesis tools and given the behavioral specification of an accelerator, it applies multi-level approximation in a very fine granularity. METHADODE uses power-error proxies and trains an error propagation model in order to distribute the overall error budget to the data flow graph arithmetic nodes. Then, using the proposed approximate multi-level library, it assigns an approximate multi-level configuration to every node. METHADONE produces close to power-optimal multi-level approximate accelerators, achieving more than 589× speedup compared to the exhaustive design space exploration. As shown, the multi-level accelerators produced by METHADONE outperform the state-of-the-art single-level ones, regarding both output error as well as power consumption. Moreover, the METHADONE's attained speedup as well as the power gains of multi-level accelerators (compared to the single-level ones) are expected to significantly increase as the size of the data flow graph (i.e., the accelerator's complexity) increases. Concluding, all the proposed frameworks operate in an automated manner, extending typical hardware design tools and/or flows, and require minimum user effort for their application. Furthermore, in order to maximize the benefits of the approximate computing application, we introduce and enable the application of multi-level approximation in the design of approximate arithmetic circuits and accelerators. Additionally, all the proposed design frameworks are compared against the respective exhaustive design space exploration to demonstrate their efficacy and it is shown that they produce, very fast, close to Pareto optimal solutions. Finally, all the approximate designs produced

by the proposed frameworks are extensively evaluated against related state-of-the-art works demonstrating, their efficiency in terms of induced error as well as power reduction.

7.2 Future Extensions

The increased benefits of approximate computing in energy efficient systems design have attracted significant research interest in almost all the computer science domains. Approximate computing research targets programming languages, compilers, runtime systems, software applications, hardware circuits/accelerators, and processor micro-architectures. In this thesis, through the proposed frameworks and especially METHADONE that incorporates all the other ones, we have efficiently addressed the design of energy-optimal approximate hardware accelerators that satisfy provided error constraints. However, in order to exploit the full potential of approximate computing and maximize its delivered energy reduction, a system-level holistic approach is required. In order to address this problem, an extension of this work would be to adopt and apply the methodology followed in METHADONE at application level. Given a software application and an error bound, distribute the error budget to the application's data flow graph nodes and apply state-of-the-art software approximations at the nodes that are executed at software-level and use METHADONE to produce the respective approximate accelerators of the hardware accelerated nodes. Moreover, we propose to leverage the fact that, in approximate computing, the imposed error highly depends on the input characteristics. Although, some inputs distributions may tolerate higher degrees of approximation, approximate computing is usually applied based on mean or worst–case scenarios, limiting the potential energy saving and performance speedup. In approximate computing research area, there is a significant lack of runtime systems that apply dynamic and input-driven approximations. Such a runtime system could exploit the ability of METHADONE to produce very fast energy-optimal approximate accelerators that satisfy error bounds and thus, produce hardware accelerators with varying error characteristics based on the requirements of the input distribution.

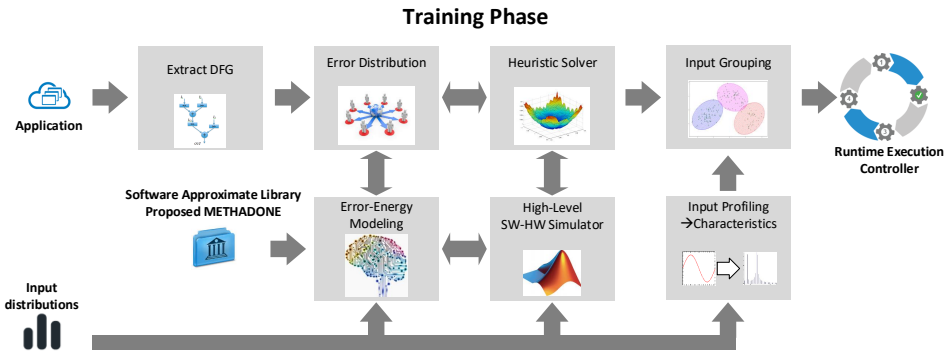


Figure 7.1: An abstract overview of the proposed extension of this thesis. Research Project "A Framework for Input-Specific Multi-level Approximate Computing on Heterogeneous Platforms", MIS: 5005377.

A holistic approach to address energy-optimal approximate application execution under error guarantees is a framework that would extend the METHADONE's methodology and comprise the following components:

- Extract the application Data Flow Graph (DFG).
- Application profiling and software-hardware partitioning of the DFG nodes.
- Application error-power modeling and then error distribution to the DFG nodes.
- Generate varying software approximate knobs of the software nodes based on the assigned error values.
- Generate varying approximate accelerator kernels, using the proposed METHADONE framework, of the hardware nodes based on the assigned error values.
- A runtime system that based on the input distribution will select for execution the appropriate software knobs and hardware kernels.

An abstract overview of such a framework is presented in Figure 7.1. The proposed extension of this thesis, as described in this section, has been accepted for granting by the Operational Program "Human Resources Development, Education and Lifelong Learning" and is co-financed by the European Union (European Social Fund) and Greek national funds under the name "A Framework for Input-Specific Multi-level Approximate Computing on Heterogeneous Platforms" and MIS: 5005377.

Chapter 8

Συνοπτική Περιγραφή των Προτεινόμενων Μεθοδολογιών στα Ελληνικά

8.1 Εισαγωγή

Από τα μέσα της δεκαετίας του 2000 όταν σταμάτησε να ισχύει η κλιμάκωση του Dennard, η ενεργειακή απόδοση των υπολογιστικών συστημάτων αποτέλεσε πρώτης τάξης σχεδιαστικό περιορισμό. Λαμβάνοντας επίσης υπόψη το χαρακτηριστικό παράδειγμα του Dark Silicon («σκοτεινό πυρίτιο») που εισήχθηκε στα σύγχρονα υπολογιστικά συστήματα, ριζικές αλλαγές (σε σχέση με τις παραδοσιακές τεχνικές που ακολουθούνται σήμερα) είναι πλέον αναγκαίες για να διατηρηθεί και να βελτιωθεί η απόδοση των υπολογιστικών συστημάτων. Για το λόγο αυτό, η βιομηχανική και επιστημονική κοινότητα έχει στραφεί στη μελέτη και χρησιμοποίηση εναλλακτικών αρχιτεκτονικών και τεχνικών υπολογισμού. Πρόσφατα, μεγάλο ενδιαφέρον εμφανίζει η έρευνα που διεξάγεται στον τομέα των τεχνικών του προσεγγιστικού υπολογισμού. Ο προσεγγιστικός υπολογισμός μπορεί να εφαρμοστεί τόσο στο λογισμικό όσο και στο υλικό, και χαλαρώνοντας την ακρίβεια των υπολογισμών (σε αποδεκτά ανά εφαρμογή πλαίσια), προσφέρει εξαιρετικά αποτελέσματα (μεγαλύτερα του 30%) τόσο στην επιτάχυνση όσο και στην μείωση της κατανάλωσης των εφαρμογών και των κυκλωμάτων.

Η έρευνα που εκπονήθηκε σε αυτή τη διατριβή, επικεντρώνεται στη μελέτη και σχεδίαση προσεγγιστικών επιταχυντών υλικού (Approximate Hardware Accelerators). Τα προσεγγιστικά κυκλώματα προσφέρουν χαμηλότερη δυναμική αλλά και στατική κατανάλωση ισχύος, μικρότερη κυκλωματική καθυστέρηση, καθώς επίσης και δυνατότητα σμίκρυνσης. Οι προσεγγίσεις, στη σχεδίαση προσεγγιστικών κυκλωμάτων, μπορούν να εφαρμοστούν σε τρία ανεξάρτητα επίπεδα:

- Αλγοριθμικό επίπεδο: τροποποίηση του αλγορίθμου που υλοποιεί ο επιταχυντής υλικού.
- Λογικό επίπεδο: τροποποίηση του πίνακα αλήθειας του κυκλώματος.
- Κυκλωματικό επίπεδο: υπερκλιμάκωση της τάσης και υπερκλιμάκωση της συχνότητας λειτουργίας

Οι υπάρχουσες μεθοδολογίες σχεδίασης προσεγγιστικών κυκλωμάτων εφαρμόζουν κατά κύριο λόγο μονο-επίπεδες προσεγγιστικές τεχνικές, περιορίζοντας έτσι σημαντικά τα πιθανά ενεργειακά οφέλη από την εφαρμογή του προσεγγιστικού υπολογισμού. Επιπροσθέτως, οι υπάρχουσες μεθοδολογίες εφαρμόζουν, ως επί το πλείστον, προσεγγιστικές τεχνικές ειδικευμένες για την εκάστοτε εφαρμογή και με μη αυτοματοποιημένο τρόπο. Η σχεδίαση προσεγγιστικών κυκλωμάτων και οι εφαρμογο-κεντρικές μεθοδολογίες που ακολουθούνται, δυσχεραίνουν ακόμα περισσότερο το ήδη εξαιρετικά πολύπλοκο έργο της σχεδίασης ψηφιακών συστημάτων, καθώς ο σχεδιαστής πρέπει να ελέγχει τόσο τη λειτουργία και τη βελτιστοποίηση του κυκλώματος όσο και την τήρηση των ορίων του σφάλματος. Σε αντίθεση με τις υπάρχουσες μεθοδολογίες στην επιστημονική βιβλιογραφία, η προτεινόμενη μεθοδολογία εφαρμόζει πολύ-επίπεδες, γενικευμένες, και αυτοματοποιημένες προσεγγιστικές τεχνικές. Ο στόχος αυτής της διατριβής είναι να μεγιστοποιήσει αφενός τα ενεργειακά οφέλη των προσεγγιστικών κυκλωμάτων και αφετέρου να προτείνει συστηματικές μεθοδολογίες για την εφαρμογή του προσεγγιστικού υπολογισμού, ώστε να επιτρέψει την αξιοποίηση αλλά και εγκαθίδρυσή του στον τομέα της σχεδίασης ψηφιακών συστημάτων. Για να μεγιστοποιήσουμε την απόδοση των προσεγγιστικών κυκλωμάτων μελετάμε, προτείνουμε, και καταστούμε δυνατή την εφαρμογή πολυ-επίπεδων προσεγγιστικών τεχνικών για την παραγωγή τόσο

προσεγγιστικών αριθμητικών κυκλωμάτων όσο και επιταχυντών υλικού. Με τον όρο πολυ-επίπεδες προσεγγιστικές τεχνικές αναφερόμαστε στην εφαρμογή μιας προσεγγιστικής τεχνικής σε κάθε ένα από τα τρία επίπεδα σχεδιασμού, δηλαδή το αλγοριθμικό, το λογικό, και το κυκλωματικό. Για να αντιμετωπίσουμε την αυξημένη σχεδιαστική δυσκολία των προσεγγιστικών κυκλωμάτων, που δυσχεραίνεται ακόμα περισσότερο στον χώρο των πολυ-επίπεδων προσεγγιστικών επιταχυντών υλικού, προτείνουμε γενικευμένες και αυτοματοποιημένες μεθοδολογίες για τη σχεδίαση και σύνθεση αυτών. Τα ανωτέρω επιτυγχάνονται μέσα από τα τέσσερα αυτοματοποιημένα πλαίσια σχεδίασης και σύνθεσης που προτάθηκαν και υλοποιήθηκαν κατά την εκπόνηση της διατριβής: VOSsim, Partial Product Perforation, HAM, και METHADONE. Σε αυτό το κεφάλαιο παρουσιάζεται μια συνοπτική περιγραφή της κάθε προτεινόμενης μεθοδολογίας, του προβλήματος που καλείται να αντιμετωπίσει καθώς επίσης, και τα αποτελέσματα της εκτενούς πειραματικής αξιολόγησης που διεξήχθη για κάθε ένα από τα προτεινόμενα αυτά πλαίσια.

8.2 Πλαίσιο VOSsim

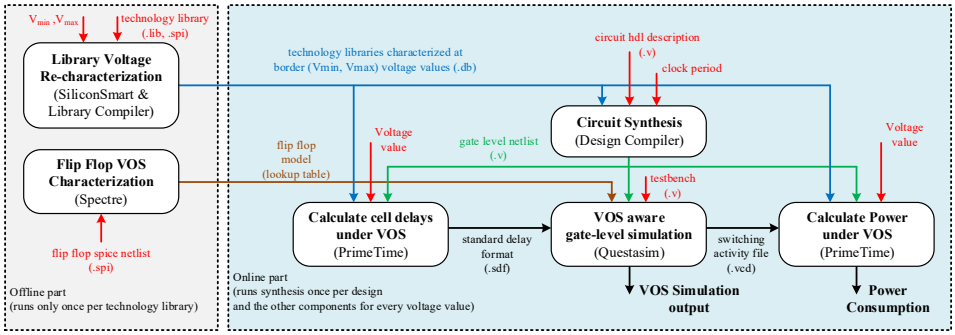
Η ενότητα αυτή βασίζεται στη δημοσίευση μας [88] στο περιοδικό Trans. on VLSI 2018.

Η υπερκλιμάκωση της τάσης (Voltage Over-Scaling ή VOS) είναι μια γενικού σκοπού κυκλωματικού επιπέδου προσεγγιστική τεχνική και μπορεί να εφαρμοστεί σε κάθε κύκλωμα. Η υπερκλιμάκωση της τάσης (εφεξής θα αναφέρεται ως VOS) εφαρμόζεται χωρίς να τροποποιηθεί το κύκλωμα, διατηρώντας την συχνότητα λειτουργίας σταθερή και μειώνοντας την τάση κάτω απ' την ονομαστική της τιμή. Μειώνοντας την τιμή της τάσης επιτυγχάνεται χαμηλότερη κατανάλωση ισχύος αλλά το κύκλωμα γίνεται πιο αργό και έτσι παράγονται λάθη στην έξοδο λόγω των μονοπατιών του κυκλώματος που παραβιάζουν τις χρονικές απαιτήσεις. Αξίζει να σημειωθεί ότι το VOS επιδεικνύει μια συνεργατική σχέση με τις προσεγγιστικές τεχνικές των άλλων επιπέδων. Οι προσεγγίσεις αλγοριθμικού και λογικού επιπέδου μειώνουν την καθυστέρηση του κρίσιμου μονοπατιού του κυκλώματος καθώς επίσης και τον αριθμό των κρίσιμων μονοπατιών. Ως εκ τούτου, εφαρμόζοντας επιπλέον αυτών και VOS,

μπορούμε να μειώσουμε περαιτέρω την κατανάλωση ισχύος αυξάνοντας ελάχιστα το παραγόμενο σφάλμα καθώς λιγότερα μονοπάτια επηρεάζονται από την μείωση της τάσης (σε σχέση με το ακριβές κύκλωμα). Για παράδειγμα εφαρμόζοντας VOS σε έναν ήδη προσεγγιστικό αθροιστή 16-bit μπορεί να μειώσει την κατανάλωση ισχύος κατά 15% αυξάνοντας το παραγόμενο σφάλμα κατά μόλις 0.1% [115].

Μολονότι το VOS είναι μια από τις πιο αποτελεσματικές προσεγγιστικές τεχνικές στην μείωση της κατανάλωσης ισχύος ενός κυκλώματος, η εφαρμογή του VOS στον προσεγγιστικό υπολογισμό είναι πολύ περιορισμένη. Ο λόγος που συμβαίνει αυτό αφορά ως επί τω πλείστον την πολύ αυξημένη δυσκολία να προσδιοριστούν και να ποσοτικοποιηθούν τα λάθη που παράγονται από την εφαρμογή του VOS. Η μόνη διαθέσιμη λύση που υπάρχει σήμερα για να μετρήσουμε την κατανάλωση και το σφάλμα των κυκλωμάτων στα οποία έχει εφαρμοστεί VOS, είναι να εκτελέσουμε προσομοιώσεις SPICE. Όμως, η εκτέλεση SPICE προσομοιώσεων είναι ένα πολύ πολύπλοκο και πολύ χρονοβόρο έργο που περιορίζει και δυσχεραίνει σημαντικά την εφαρμογή και αξιοποίηση του VOS στον σχεδιασμό προσεγγιστικών κυκλωμάτων.

Το προτεινόμενο πλαίσιο VOSsim αντιμετωπίζει αυτά τα μειονεκτήματα καθιστώντας δυνατή την εκτέλεση προσομοιώσεων σε επίπεδο πυλών (gate-level) που λαμβάνουν υπόψιν την τάση. Η εκτέλεση προσομοιώσεων σε επίπεδο πυλών είναι πολύ πιο γρήγορη και πολύ πιο εύκολη από τη διεξαγωγή SPICE προσομοιώσεων. Μολαταύτα τα υπάρχοντα εργαλεία σχεδιασμού κυκλωμάτων διέπονται από κάποιους περιορισμούς που δεν επιτρέπουν την απευθείας εκτέλεση προσομοιώσεων σε επίπεδο πυλών που λαμβάνουν υπόψιν την τάση. Οι κυριότεροι περιορισμοί είναι ότι εξ΄ ορισμού η μείωση της τάσης πρέπει να εφαρμοστεί μετά τη σύνθεση του κυκλώματος και επίσης ιδιαίτερη μεταχείριση χρειάζονται οι παραβάσεις χρονισμού των καταχωρητών (Flip-flop timing violations). Όταν μειώνεται η τάση λειτουργίας τα κυκλώματα γίνονται πιο αργά κι έτσι, κατά τη διάρκεια της προσομοίωσης, θα συμβούν Flip-flop timing violations. Όμως οι προσομοιωτές σε επίπεδο πυλών (π.χ. Questasim) εξ΄ ορισμού δεν διαχειρίζονται τέτοιες περιπτώσεις και αντ΄ αυτού τυπώνουν ένα μήνυμα λάθος και θέτουν την τιμή του σήματος σε μια άγνωστη κατάσταση "x". Αυτή η άγνωστη κατάσταση διαδίδεται σε όλους του υπόλοιπους υπολογισμούς και έχει ως αποτέλεσμα να μην είναι δυνατό να υπολογιστεί η έξοδος

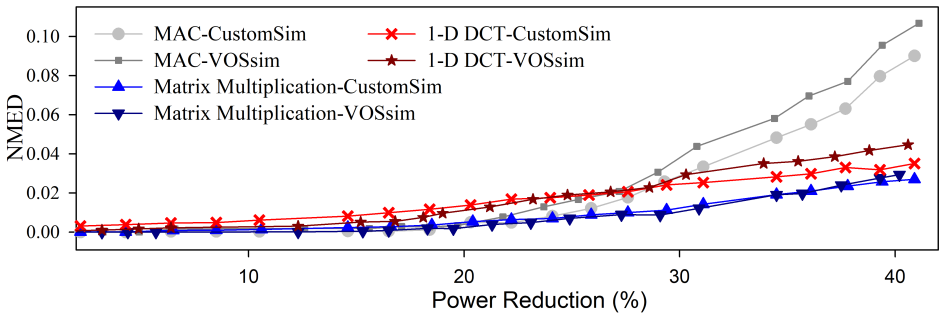


Σχήμα 8.1: Το προτεινόμενο πλαίσιο VOSsim. Με κόκκινο χρώμα φαίνονται οι είσοδοι του πλαισίου.

και η κατανάλωσης ισχύος του κυκλώματος.

Το προτεινόμενο πλαίσιο VOSsim αντιπαρέρχεται αυτούς τους περιορισμούς χρησιμοποιώντας δύο μοντέλα υψηλού επιπέδου. Το πρώτο είναι το Synopsys CCS μοντέλο [94] που χρησιμοποιείται για να χαρακτηρίσει το κύκλωμα σε μια επιθυμητή τιμή τάσης. Το δεύτερο είναι το προτεινόμενο Flip-flop μοντέλο το οποίο χρησιμοποιείται για να αντικαταστήσει τις άγνωστες καταστάσεις "x" που παράγονται κατά την προσομοίωση. Το Flip-flop μοντέλο είναι ένας πίνακας αναζήτησης που δέχεται σαν είσοδο: i) την τιμή της Τάσης, ii) τον τύπο του Flip-flop, iii) τη σχετική χρονική στιγμή ως προς την θετική ακμή του ρολογιού που συνέβη η παράβαση, και iv) τη μετάβαση 0 σε 1 ή 1 σε 0 και επιστρέφει μια κατάλληλη δυαδική τιμή 0 ή 1 για να αντικαταστήσει την τιμή "x". Για να παραχθεί το Flip-flop μοντέλο, εκτελούμε SPICE προσομοιώσεις για κάθε τύπο Flip-flop στη βιβλιοθήκη και για κάθε εξεταζόμενη τιμή τάσης.

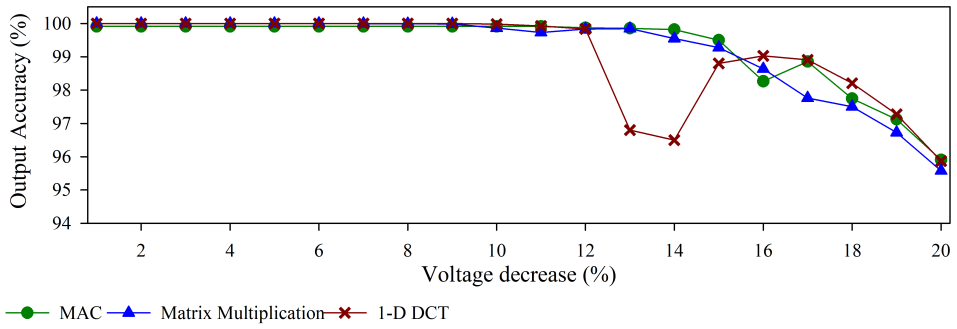
Στο Σχήμα 8.1 απεικονίζεται το προτεινόμενο πλαίσιο VOSsim. Το VOSsim χωρίζεται σε δύο μέρη. Το πρώτο εκτελείται μόνο μια φορά για κάθε βιβλιοθήκη και είναι υπεύθυνο για να προετοιμάσει τα μοντέλα που χρησιμοποιεί το VOSsim. Το δεύτερο μέρος είναι αυτό που επιτελεί την προσομοίωση και εκτελείται για κάθε κύκλωμα και τιμή τάσης. Εν συντομία το δεύτερο μέρος συνθέτει το κύκλωμα στην επιθυμητή συχνότητα λειτουργίας και στην συνέχεια επιτελεί ανάλυση χρονισμού του συνθεμένου κυκλώματος (Static Tim-



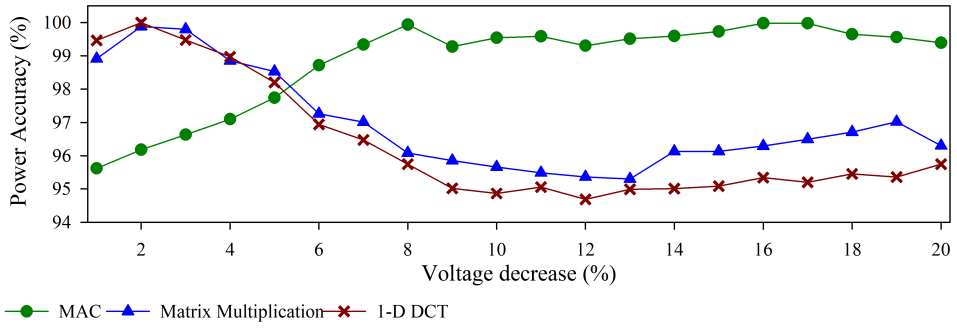
Σχήμα 8.2: Η καμπύλη σφάλματος-κατανάλωσης ισχύος των κυκλωμάτων υπερκλιμακούμενης τάσης όπως παρήχθησαν από το VOSsim και το CustomSim.

ing Analysis) χρησιμοποιώντας το μοντέλο CCS και ξανα-χαρακτηρίζει την καθυστέρηση των κελιών του κυκλώματος στην επιθυμητή τάση. Έπειτα επιτελεί προσομοίωση σε επίπεδο πυλών χρησιμοποιώντας τις νέες καθυστερήσεις και σε κάθε κύκλο ελέγχει για Flip-flop timing violations. Σε περίπτωση που μια παράβαση συμβεί χρησιμοποιεί το μοντέλο Flip-flop και αντικαθιστά την άγνωστη τιμή "x" με αυτή που επιστρέφει το μοντέλο. Μετά από αύτο το στάδιο έχει παραχθεί η έξοδος του κυκλώματος στην επιθυμητή τάση. Τέλος, εκτελεί μια ανάλυση ισχύος του συνθεμένου κυκλώματος (Power Analysis) χρησιμοποιώντας το μοντέλο CCS και ξανα-χαρακτηρίζει την κατανάλωση των κελιών του κυκλώματος στην επιθυμητή τάση.

Αξιολογήσαμε το πλαίσιο VOSsim χρησιμοποιώντας τρία κυκλώματα σαν σημεία αναφοράς τα οποία χαρακτηρίζονται από αύξον αριθμό τρανζίστορ και Flip-flop. Τα κυκλώματα που χρησιμοποιήσαμε στην ανάλυσή μας είναι τα MAC, Πολλαπλασιασμός Πινάκων, και 1-D DCT. Τα κυκλώματα αυτά είναι χαρακτηριστικά παραδείγματα προσεγγιστικού υπολογισμού και χρησιμοποιούνται ευρέως στην αξιολόγηση προσεγγιστικών τεχνικών. Κατά την ανάλυση του VOSsim μελετήσαμε τόσο την ακρίβειά του στον υπολογισμό της εξόδου και της κατανάλωσης ισχύος κυκλωμάτων με υπερκλιμακούμενη τάση όσο και την επιτάχυνση του χρόνου προσομοίωσης που επιτυγχάνει. Για τον λόγο αυτό το συγκρίνουμε με τα αποτελέσματα του γρήγορου SPICE προσομοιωτή CustomSim. Στο Σχήμα 8.2 απεικονίζεται, για όλες τις εξεταζόμενες εφαρμογές, η καμπύλη σφάλματος-κατανάλωσης ισχύος των κυκλωμάτων υπερκλιμακούμε-



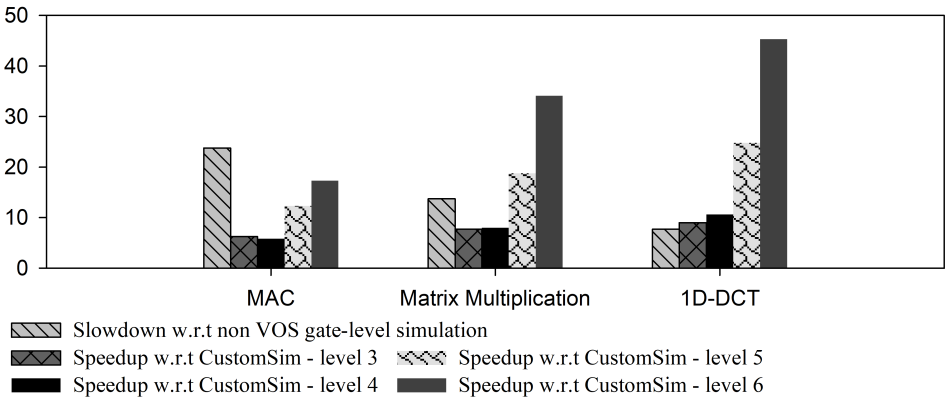
(α′)



(β′)

Σχήμα 8.3: Η ακρίβεια του VOSsim στο να υπολογίζει α′) την έξοδο και β′) την κατανάλωση κυκλωμάτων υπερκλιμακούμενης τάσης σε σύγκριση με την εκτέλεση υψηλής ακρίβειας SPICE προσομοιώσεων με το CustomSim (level 6).

νης τάσης όπως παρήχθησαν από το VOSsim και το CustomSim. Αντίστοιχα, στο Σχήμα 8.3 απεικονίζεται η ακρίβεια του VOSsim στο να υπολογίζει την έξοδο και κατανάλωση ισχύος των κυκλωμάτων υπερκλιμακούμενης τάσης. Όπως βλέπουμε η καμπύλη που παράγει το VOSsim είναι πολύ κοντά σε αυτή που παράγει το CustomSim δείχνοντας την ικανότητα του VOSsim να υπολογίζει με μεγάλη ακρίβεια τα χαρακτηριστικά σφάλματος και κατανάλωσης των κυκλωμάτων υπερκλιμακούμενης τάσης. Πιο συγκεκριμένα (Σχήμα 8.3) το VOSsim υπολογίζει το σφάλμα με ακρίβεια περίπου 99% και την κατανάλω-



Σχήμα 8.4: Η επιτάχυνση του χρόνου προσομοίωσης που επιτυγχάνεται από το VOSsim σε σύγκριση με το CustomSim. Επίσης, για λόγους πληρότητας, απεικονίζεται και η επιβράδυνση σε σχέση με τις προσομοιώσεις με το Questasim που δεν υποστηρίζουν όμως υπερκλιμάκωση της τάσης (gate-level non-VOS simulations).

ση ισχύος με ακρίβεια 98%. Τέλος αξιολογήσαμε την επιτάχυνση του χρόνου προσομοίωσης που επιτυγχάνει το VOSsim σε σχέση με την εκτέλεση υψηλής ακρίβειας SPICE προσομοιώσεων με το CustomSim. Όπως απεικονίζεται στο Σχήμα 8.4 η προσομοίωση με το VOSsim είναι 35 φορές γρηγορότερη, κατά μέσο όρο, σε σχέση με το CustomSim. Η επιτάχυνση αυτή κυμαίνεται από 17× για το πολύ μικρό MAC κύκλωμα μέχρι 45× για το λίγο μεγαλύτερο 1-D DCT και αναμένεται να είναι πολύ μεγαλύτερη όσο η πολυπλοκότητα του κυκλώματος αυξάνει.

8.3 Πλαίσιο Partial Product Perforation

Η ενότητα αυτή βασίζεται στη δημοσίευση μας [99] στο περιοδικό Trans. on VLSI 2016.

Οι αθροιστές και πολλαπλασιαστές αποτελούν τις βασικές μονάδες των επιταχυντών υλικού και καθορίζουν επί το πλείστον την κατανάλωση και την κυκλωματική τους καθυστέρηση. Μολονότι υπάρχει εκτενής έρευνα σε προσεγγιστικούς αθροιστές, η αντίστοιχη έρευνα σε προσεγγιστικούς πολλαπλασιαστές είναι πολύ περιορισμένη, κατά βάση λόγω της αυξημένης κυκλωματικής πολυπλοκότητας του πολλαπλασιαστή. Ένας πολλαπλασιαστής αποτελείται από τρία στάδια: την παραγωγή των μερικών γινομένων (partial product generation), τη συσσώρευση των μερικών γινομένων (partial product accumulation), και την τελική άθροιση (final addition). Οι υπάρχουσες προσεγγιστικές τεχνικές εφάρμοζαν προσεγγίσεις στο στάδιο της συσσώρευσης των μερικών γινομένων και κατά πλειονότητα ήταν άρρηκτα συνδεδεμένες με την αρχιτεκτονική του πολλαπλασιαστή τον οποίο εξέταζαν. Επίσης, όλες αυτές οι τεχνικές χρειάζονταν χρονοβόρες συνθέσεις και προσομοιώσεις ώστε να χαρακτηρίσουν στατιστικά το σφάλμα του προσεγγιστικού πολλαπλασιαστή. Σε αυτή την ενότητα παρουσιάζουμε την τεχνική Partial Product Perforation. Με την τεχνική Partial Product Perforation προτείναμε και εφαρμόσαμε για πρώτη φορά προσεγγίσεις στο στάδιο της παραγωγής των μερικών γινομένων. Το Partial Product Perforation είναι μια αλγοριθμική προσεγγιστική τεχνική καθώς τροποποιεί τον αλγόριθμο του πολλαπλασιασμού παραλείποντας την παραγωγή ορισμένου αριθμού συνεχόμενων μερικών γινομένων. Ως αποτέλεσμα το Partial Product Perforation μπορεί να εφαρμοστεί σε κάθε πολλαπλασιαστή ανεξάρτητα από την αρχιτεκτονική. Επιπροσθέτως, το σφάλμα που παράγεται λόγω της εφαρμογής του Partial Product Perforation μπορεί να χαρακτηριστεί αυστηρά μαθηματικά. Για παράδειγμα, ενώ ο ακριβής πολλαπλασιασμός δίνεται από την σχέση:

$$A \times B = \sum_{i=0}^{n-1} Ab_i 2^i,\ b_i \in \{0,1\}. \tag{8.1}$$

όταν εφαρμόζουμε Partial Product Perforation με μεταβλητές j και k το

γινόμενο $A \times B$ δίνεται από:

$$A \times B|_{j,k} = \sum_{\substack{i=0, \\ i \notin [j, j+k)}}^{n-1} A b_i 2^i, \; b_i \in \{0, 1\}. \tag{8.2}$$

Όπου j είναι το πρώτο μερικό γινόμενο που θα απαλείψουμε και k είναι το πλήθος των μερικών γινομένων που δεν παράγονται. Το απόλυτο σφάλμα, λοιπόν, του Partial Product Perforation ισούται με:

$$\begin{aligned} ED(A, B) &= \left| P - P' \right| \\ &= A \sum_{i=0}^{n-1} b_i 2^i - A \sum_{\substack{i=0, \\ i \notin [j, j+k)}}^{n-1} b_i 2^i \\ &= A \sum_{i=j}^{j+k-1} 2^i b_i \\ &= A 2^j x_B, \end{aligned} \tag{8.3}$$

όπου $x_B \in [0, 2^k)$ και

$$x_B = \sum_{i=0}^{k-1} 2^i b_{j+i} = \lfloor B / 2^j \rfloor \bmod 2^k. \tag{8.4}$$

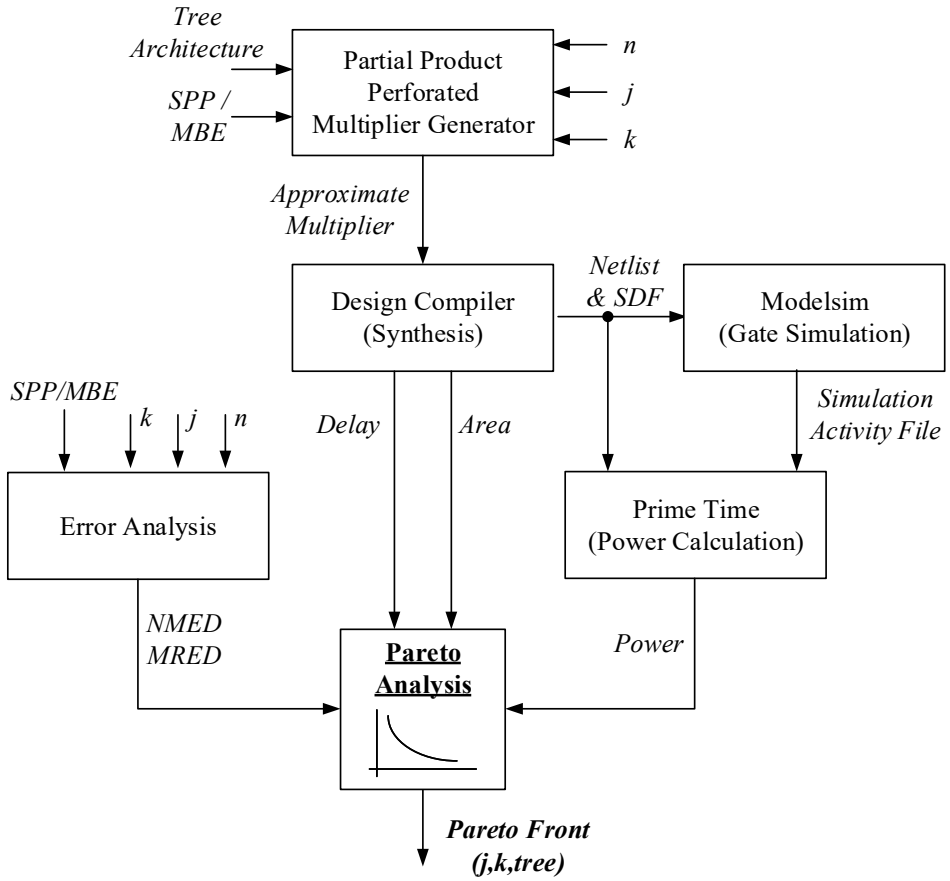
Έστω λοιπόν ότι A και B ακολουθούν την ομοιόμορφη κατανομή, το μέσο απόλυτο σφάλμα δίνεται από:

$$\begin{aligned} MED &= \frac{2^j 2^{2n} (2^k - 1)(2^n - 1)}{2^{2n} 4} \\ &= \frac{2^j (2^k - 1)(2^n - 1)}{4}. \end{aligned} \tag{8.5}$$

και το μέσο σχετικό σφάλμα από:

$$MRED = \frac{2^n}{2^{2n}} \sum_{\forall B} \frac{x_B 2^j}{B} = \sum_{\forall B} \frac{x_B 2^j}{2^n B}. \tag{8.6}$$

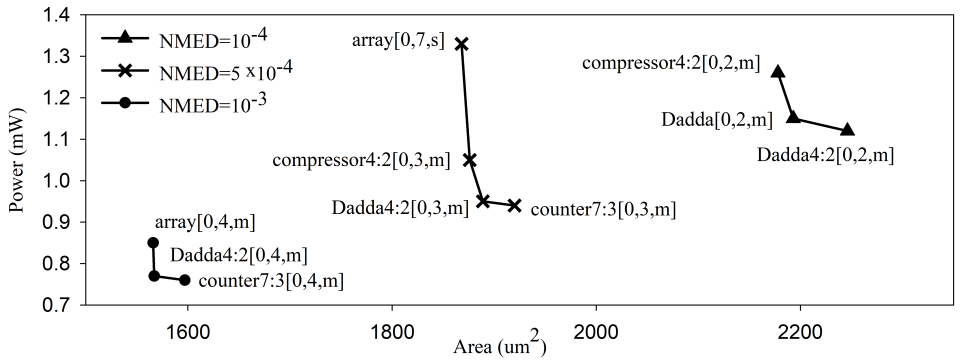
Όπως φαίνεται στις ανωτέρω εξισώσεις, το σφάλμα από την εφαρμογή του Partial Product Perforation εξαρτάται αποκλειστικά και μόνο από τις μεταβλητές



Σχήμα 8.5: Το πλαίσιο Partial Product Perforation.

j και k. Ένας τέτοιος απόλυτος μαθηματικός χαρακτηρισμός επιτρέπει εκ των προτέρων γνώση του σφάλματος της τεχνικής του Partial Product Perforation αλλά και επιλογή της βέλτιστης διαμόρφωσης (configuration) του Partial Product Perforation δεδομένης της κατανομής εισόδου.

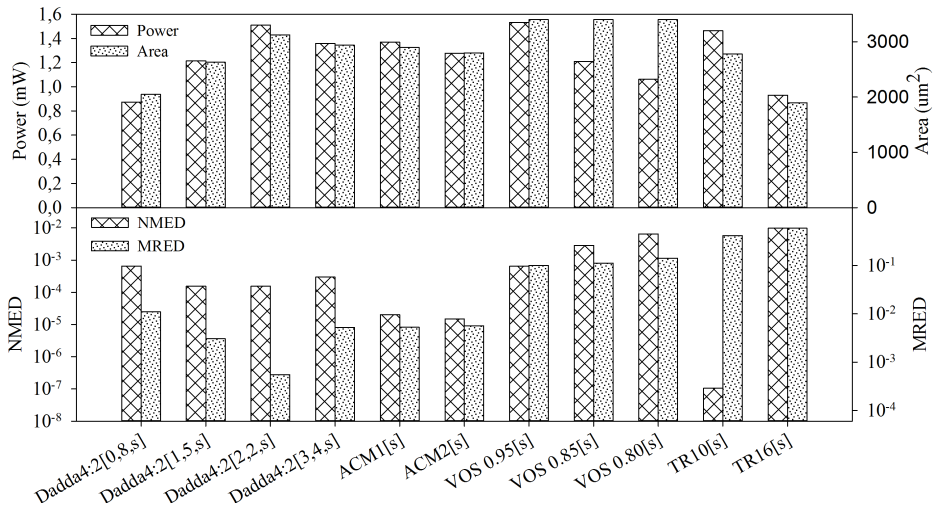
Μολονότι το σφάλμα που προκύπτει από την εφαρμογή του Partial Product Perforation εξαρτάται μόνο από τις μεταβλητές j και k, τα παραγόμενα κέρδη (κατανάλωση, καθυστέρηση, χωρική πολυπλοκότητα) εξαρτώνται από την αρχιτεκτονική του πολλαπλασιαστή. Για τον λόγο αυτό προτείνουμε το πλαίσιο



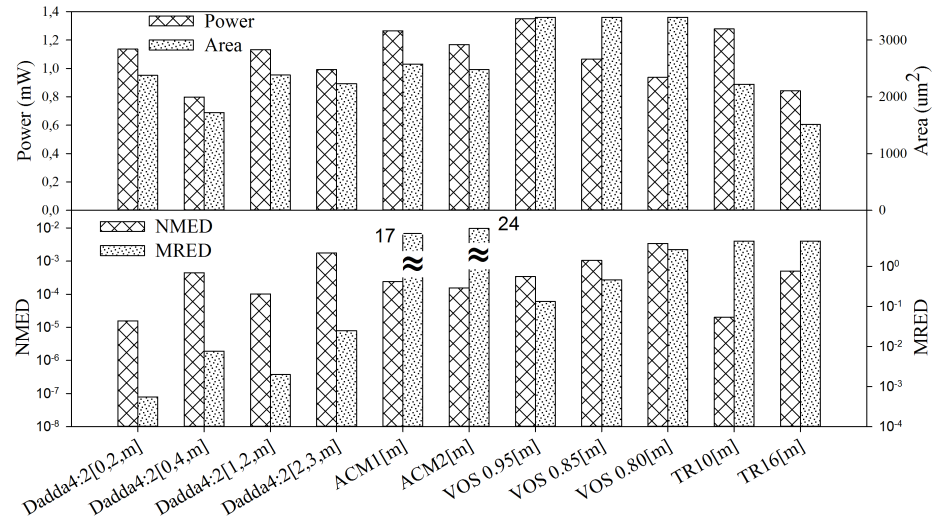
Σχήμα 8.6: Το Pareto μέτωπο για διάφορες τιμές κανονικοποιημένου απόλυτου σχετικού σφάλματος *NMED*. Στον άξονα y απεικονίζεται η κατανάλωση ισχύος και στον άξονα x η χωρική πολυπλοκότητα. Δίπλα σε κάθε Pareto σημείο αναγράφεται ο συνδυασμός αρχιτεκτονικής πολλαπλασιαστή-διαμόρφωσης της τεχνικής Partial Product Perforation.

Partial Product Perforation που δεδομένου ενός ορίου σφάλματος παράγει το βέλτιστο συνδυασμό διαμόρφωσης του Partial Product Perforation - αρχιτεκτονικής πολλαπλασιαστή. Το προτεινόμενο πλαίσιο απεικονίζεται στο Σχήμα 8.5 και εν συντομία λειτουργεί ως εξής: Επιλύει τις εξισώσεις σφάλματος του Partial Product Perforation και υπολογίζει τους συνδυασμούς των μεταβλητών j και k που ικανοποιούν το όριο σφάλματος. Στη συνέχεια, δοσμένης μιας βιβλιοθήκης αρχιτεκτονικών πολλαπλασιαστών, συνθέτει και προσομοιώνει τους συνδυασμούς που υπολογίστηκαν προηγουμένως, για να μετρήσει την κατανάλωση, την καθυστέρηση, και την χωρική τους πολυπλοκότητα. Τέλος, επιτελεί μια Pareto ανάλυση για να υπολογίσει το αντίστοιχο Pareto μέτωπο. Για παράδειγμα, στο Σχήμα 8.6 φαίνονται τα Pareto μέτωπα που παρήχθησαν από το πλαίσιο Partial Product Perforation για τρία διαφορετικά όρια σφάλματος. Όπως φαίνεται στο Σχήμα 8.6, ανάλογα με το όριο σφάλματος διαφορετικοί πολλαπλασιαστές με διαφορετική Partial Product Perforation διαμόρφωση εμφανίζονται σε κάθε Pareto μέτωπο.

Αξιολογήσαμε πειραματικά την τεχνική Partial Product Perforation χρησιμοποιώντας έναν 16-bit DAdda 4:2 πολλαπλασιαστή (Σχημα 8.7) και την συγκρίναμε με i) τον ορθό πολλαπλασιαστή, ii) με δύο υπάρχουσες προσεγγι-

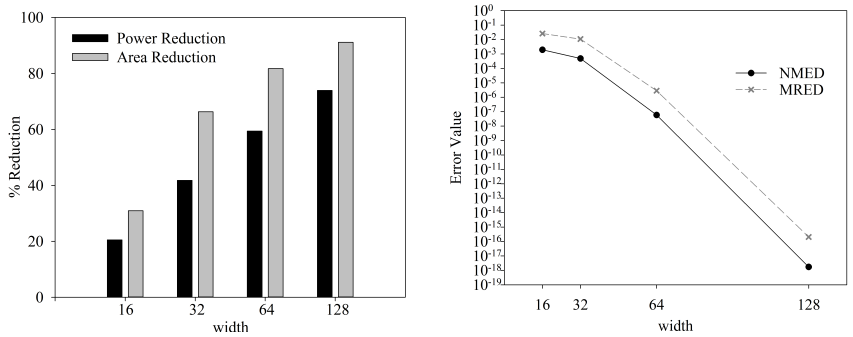


(α′)



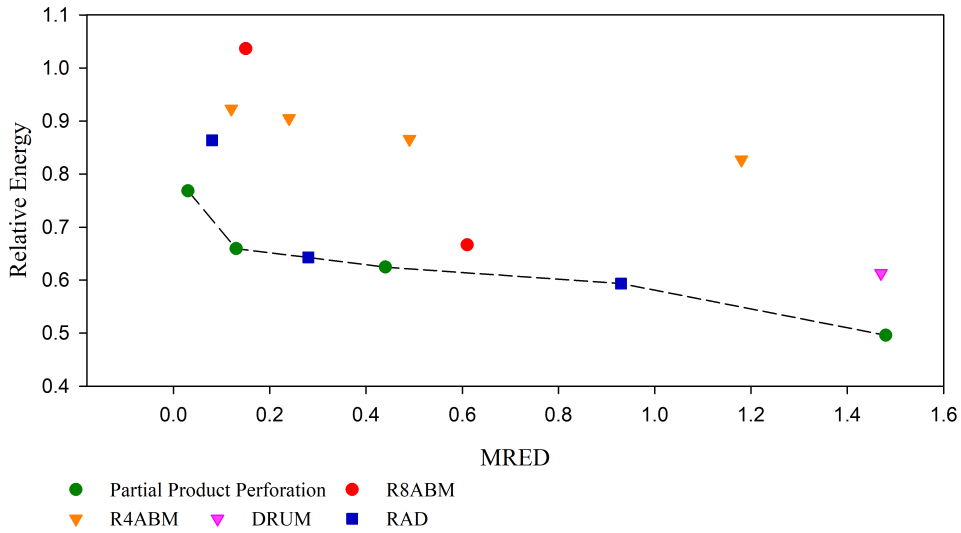
(β′)

Σχήμα 8.7: Σύγκριση της τεχνικής Partial Product Perforation με ACM1, ACM2 [63], TR10, TR16 [56] και VOS [57]



Σχήμα 8.8: Εφαρμογή της τεχνικής Partial Product Perforation σε κλιμακούμενου μεγέθους πολλαπλασιαστές. α′) Η μείωση της κατανάλωσης ισχύος και επιφάνεια για σφάλμα $NMED \leq 10^{-4}$, και β′) οι τιμές σφάλματος $NMED$ και $MRED$ για 50% μείωση της κατανάλωσης.

στικές τεχνικές που εφαρμόζουν προσεγγίσεις στο στάδιο της συσσώρευσης των μερικών γινομένων: την τεχνική της αποκοπής [56] (truncation) και την τεχνική των προσεγγιστικών 4:2 συμπιεστών [63] (approximate 4:2 compressors), και iii) με την υπάρχουσα κυκλωματική προσεγγιστική τεχνική VOS. Επίσης, μελετήσαμε την επεκτασιμότητα (Σχήμα 8.8) αλλά και διατηρησιμότητα (Σχήμα 8.9) της προτεινόμενης τεχνικής. Συγκριτικά με τον ορθό πολλαπλασιαστή η τεχνική Partial Product Perforation επιτυγχάνει 50% μείωση της κατανάλωσης ισχύος και 40% μείωση της χωρικής επιφάνειας για μόλις 1% μεσο σχετικό σφάλμα ($MRED \leq 1\%$). Όπως φαίνεται στο Σχήμα 8.7, σε σύγκριση με τις υπάρχουσες τεχνικές truncation και VOS, η προτεινόμενη τεχνική Partial Product Perforation επιτυγχάνει ίδιες τιμές κατανάλωσης ισχύος και επιφάνειας αλλά το παραγόμενο σφάλμα του ($NMED$ και $MRED$) είναι τάξεις μεγέθους μικρότερο από αυτό των truncation και VOS. Σε σύγκριση με την υπάρχουσα τεχνική ACM [63], το Partial Product Perforation είναι λιγότερο αποδοτικό για πολύ μικρές τιμές σφάλματος, αλλά μέσο σχετικό σφάλμα ($MRED = 1\%$) επιτυγχάνει 30% χαμηλότερη κατανάλωση. Στη συνέχεια, μελετήσαμε την επεκτασιμότητα της προτεινόμενης τεχνικής Partial Product Perforation εξετάζοντας την εφαρμογή της σε διαφορετικού μεγέθους πολλαπλασιαστές. Όπως φαίνεται στο Σχήμα 8.8 η τεχνική Partial Product Perforation είναι επεκτάσιμη καθώς α) για σταθερή τιμή σφάλματος



Σχήμα 8.9: Σύγκριση 16-bit προσεγγιστικών πολλαπλασιαστών που, όπως και το Partial Product Perforation, εφαρμόζουν προσεγγίσεις στο στάδιο παραγωγής των μερικών γινομένων και/ή στον αλγόριθμο του πολλαπλασιασμού. Πολλαπλασιαστές με ίδιο χρώμα εφαρμόζουν την ίδια τεχνική με διαφορετική διαμόρφωση.

όσο αυξάνεται το μέγεθος του πολλαπλασιαστή αυξάνουν και τα παραγόμενα κέρδη (μείωση κατανάλωσης κι επιφάνειας) και β) για σταθερή μείωση κατανάλωσης όσο αυξάνεται το μέγεθος του πολλαπλασιαστή μειώνεται η τιμή του σφάλματος ($NMED$ και $MRED$). Τέλος, συγκρίναμε την τεχνική Partial Product Perforation με προσεγγιστικές τεχνικές που δημοσιεύτηκαν μετά το Partial Product Perforation [66, 74, 75, 100] και επίσης εφαρμόζουν προσεγγίσεις στο στάδιο της παραγωγής των μερικών γινομένων ή/και τροποποιούν τον αλγόριθμο του πολλαπλασιασμού. Όπως φαίνεται στο Σχήμα 8.9 όλοι οι προσεγγιστικοί πολλαπλασιαστές που ανήκουν στο Pareto μέτωπο είναι πράσινα και μπλε σημεία. Τα πράσινα σημεία εφαρμόζουν την προτεινόμενη Partial Product Perforation τεχνική και τα μπλε σημεία είναι οι RAD πολλαπλασιαστές που προτείναμε στο [100]. Συνεπώς, το Σχήμα 8.9 δείχνει την αποδοτικότητα αλλά και διατηρησιμότητα του Partial Product Perforation σχετικά με τη δημιουργία ενεργειακά αποδοτικών προσεγγιστικών πολλαπλασιαστών.

8.4 Πλαίσιο Hybrid Approximate Multipliers (HAM)

Η ενότητα αυτή βασίζεται στη δημοσίευση μας [41] στο συνέδριο ISLPED.

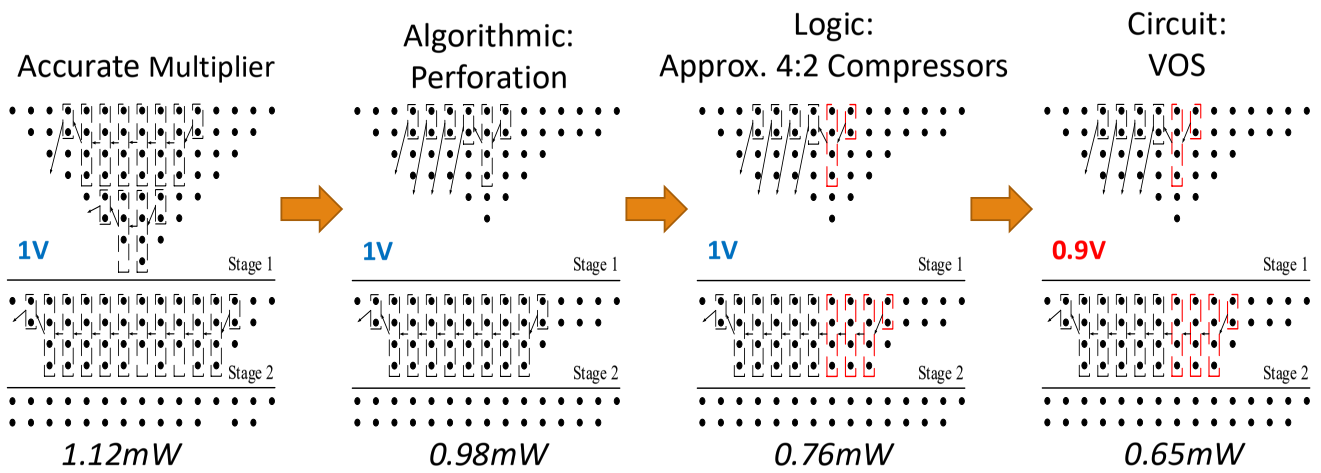
Όπως είδαμε στην πειραματική αξιολόγηση της τεχνικής Partial Product Perforation, στον σχεδιασμό προσεγγιστικών πολλαπλασιαστών μπορούμε να εφαρμόσουμε α) Partial Product Perforation στο αλγοριθμικό επίπεδο, β) να χρησιμοποιήσουμε του προσεγγιστικούς 4:2 συμπιεστές (4:2 compressors) [63] στο λογικό επίπεδο, και γ) να εφαρμόσουμε VOS στο κυκλωματικό επίπεδο. Οι τεχνικές αυτές εφαρμόζονται σε διαφορετικά προσεγγιστικά επίπεδα και μοιάζουν να είναι «ορθογώνιες» αλλά και να μπορούν να εφαρμοστούν συνεργατικά. Μολαταύτα, οι υπάρχουσες προσεγγιστικές τεχνικές εφαρμόζουν κατά κύριο λόγο προσεγγίσεις σε ένα προσεγγιστικό επίπεδο οδηγώντας σε περιορισμένη μείωση της κατανάλωσης ισχύος. Με στόχο να αυξήσουμε την ενεργειακή απόδοση του προσεγγιστικού υπολογισμού εξετάζουμε την εφαρμογή προσεγγιστικών τεχνικών σε κάθε επίπεδο και προτείνουμε τους υβριδικούς προσεγγιστικούς πολλαπλασιαστές (Hybrid Approximate Multipliers or HAM). Όπως απεικονίζεται ενδεικτικά στο Σχήμα 8.10, δεδομένου ενός ορθού πολλαπλασιαστή, εφαρμόζουμε Partial Product Perforation στο αλγοριθμικό επίπεδο. Στο δέντρο που προκύπτει, αντικαθιστούμε τους ορθούς 4:2 compressors ή δύο συνεχόμενους πλήρεις αθροιστές (full adders) με τους προσεγγιστικούς 4:2 compressors. Τέλος, στο προκύπτον κύκλωμα από αυτό το στάδιο, εφαρμόζουμε VOS σαν προσεγγιστική τεχνική στο κυλωματικό επίπεδο.

Το πρόβλημα που καλούμαστε να λύσουμε είναι να βρούμε το βέλτιστο συνδυασμό αναφορικά με τη διαμόρφωση του Partial Product Perforation, το πλήθος των στηλών που θα χρησιμοποιήσουμε τους προσεγγιστικούς 4:2 compressors, και την τιμή της μείωσης τάσης. Αυτό το πρόβλημα βελτιστοποίησης μπορεί να μοντελοποιηθεί ως εξής: Βρες τον συνδυασμό $\mathbf{x}$ ο οποίος ελαχιστοποιεί την κατανάλωση ισχύος και σέβεται το όριο σφάλματος.

$$\min_{\mathbf{x}\in \boldsymbol{D}} \left[\; Power(\mathbf{x}) \; \right]$$

subject to:

$$\left[\; NMED(\mathbf{x}) \; \right] \leq \left[\; Max\_NMED \; \right],$$



Σχήμα 8.10: Εφαρμογή πολυ-επίπεδων προσεγγίσεων και παραγωγή των προτεινόμενων υβριδικών προσεγγιστικών πολλαπλασιαστών (HAM).

Ισοδύναμα μπορεί να γραφτεί ως εξής:

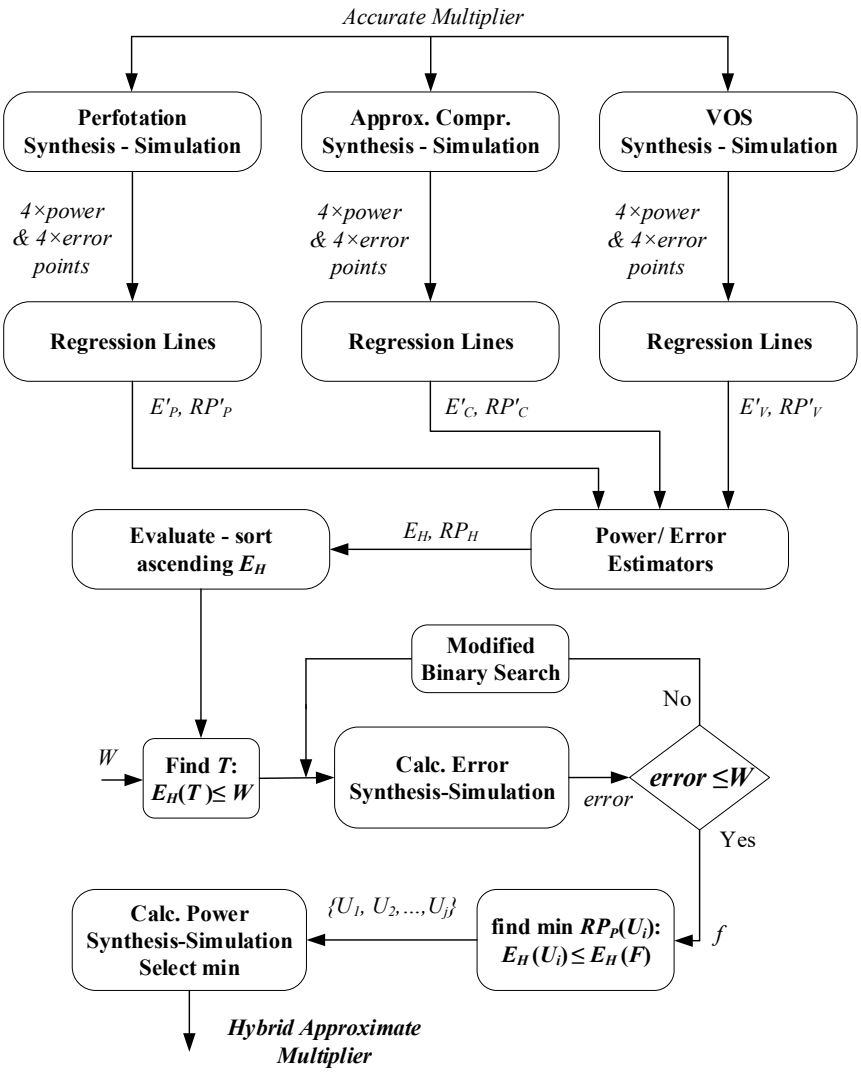
$$\mathbf{x} =< x_1, x_2, x_3 >\in \mathcal{N}^3$$

$$\max_{\mathbf{x}\in \boldsymbol{D}} \left[\ Power\_Reduction(\mathbf{x}) \ \right]$$

subject to:

$$\left[\ NMED(\mathbf{x}) \ \right] \leq \left[\ Max\_NMED \ \right],$$

όπου η μη γραμμική συνάρτηση $Power\_Reduction(x) \in \mathcal{R}$ υποδηλώνει την ολική μείωση της κατανάλωσης του υβριδικού πολλαπλασιαστή και η μη γραμμική συνάρτηση $NMED(x) \in \mathcal{R}$ το ολικό του σφάλμα. Το πρόβλημα αυτό είναι ένα συνδυαστικό πρόβλημα και είναι ισοδύναμο με το NP-πλήρες πρόβλημα του ακέραιου μη γραμμικού σακιδίου (integer non-linear KNAPSACK problem) [112]. Η αυξημένη πολυπλοκότητα αυτής της έκδοσης του προβλήματος του σακιδίου καθιστά απαραίτητη τη χρησιμοποίηση ευριστικών αλγορίθμων. Επίσης, πρέπει να αναφερθεί ότι η αξιολόγηση κάθε σημείου του χώρου σχεδιασμού είναι πολύ χρονοβόρα διαδικασία καθώς απαιτεί σύνθεση του κυκλώματος και προσομοίωση με υπερκλιμακούμενη τάση. Για την επίλυσή του χρησιμοποιήσαμε δύο μοντέλα για να εκτιμήσουμε το σφάλμα και την σχετική κατανάλωση (λόγος της κατανάλωσης του προσεγγιστικού πολλαπλασιαστή ως προς την κατανάλωση του ορθού πολλαπλασιαστή) των προτεινόμενων υβριδικών πολλαπλασιαστών. Το μοντέλο $E_H = E_P + E_C + E_V$ χρησιμοποιείται για την εκτίμηση του σφάλματος όπου E_P, E_C, και E_V είναι οι τιμές σφάλματος όταν εφαρμόζουμε ξεχωριστά την κάθε μια προσεγγιστική τεχνική με ακριβώς την ίδια διαμόρφωση. Το μοντέλο $RP_H = RP_P \times RP_C \times RP_V$ χρησιμοποιείται για την εκτίμηση της σχετικής κατανάλωσης του υβριδικού πολλαπλασιαστή όπου RP_P, RP_C και RP_V είναι η σχετική κατανάλωση όταν εφαρμόζεται ξεχωριστά η κάθε μια προσεγγιστική τεχνική με ακριβώς την ίδια διαμόρφωση. Τέλος χρησιμοποιούμε παλινδρόμηση (regression) για να εκτιμήσουμε τις τιμές των E_P, E_C, E_V, RP_P, RP_C και RP_V. Δεδομένου ότι οι συναρτήσεις του σφάλματος και της σχετικής κατανάλωσης είναι μονότονες και διατηρούν κυρτότητα, μόνο λίγα σημεία χρειάζονται για να εκπαιδευτούν τα προτεινόμενα μοντέλα. Για παράδειγμα στην περίπτωση ενός 16-bit πολλαπλασιαστή χρησιμοποιήσαμε μόνο 4 σημεία ανά τεχνική. Πρέπει να σημειωθεί ότι στο πρόβλημα βελτιστοποίησης που καλούμαστε να λύσουμε, η εκτίμηση της κατάταξης των υβριδικών πολλαπλασιαστών είναι πιο σημαντική από την ακριβή εκτίμηση των τιμών σφάλματος και κατανάλωσής τους. Σε σύγκριση

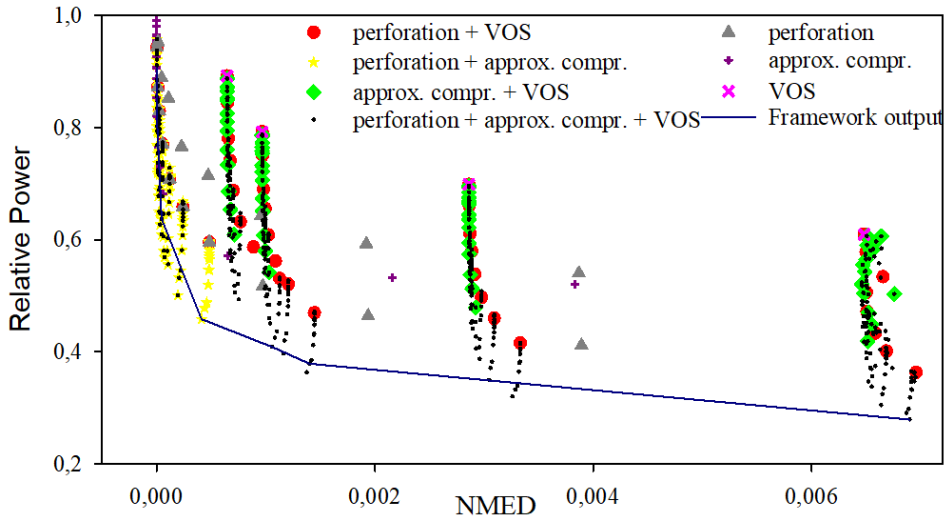


Σχήμα 8.11: Το προτεινόμενο πλαίσιο HAM.

με την πλήρη εξερεύνηση του χώρου των υβριδικών πολλαπλασιαστών, τα προτεινόμενα μοντέλα χαρακτηρίζονται από πολύ υψηλό δείκτη Pearson(Pearson correlation coefficient). Ο δείκτης Pearson του εκτιμητή E_H είναι 0.86 και ο αντίστοιχος του RP_H είναι 0.97. Οι πολύ υψηλές αυτές τιμές (μέγιστο 1) δείχνουν ότι τα προτεινόμενα μοντέλα μπορούν να εκτιμήσουν με πολύ μεγάλη ακρίβεια τη διάταξη των υβριδικών πολλαπλασιαστών. Επίσης, η ακρίβεια των E_H και RP_H στο να εκτιμούν την τιμή του σφάλματος και της σχετικής κατανάλωσης των υβριδικών πολλαπλασιαστών είναι εξίσου υψηλή, καθώς το μέσο τετραγωνικό σφάλμα (MSE) του E_H είναι 5.9×10^{-7} και του RP_H είναι 1.0×10^{-3}.

Το προτεινόμενο πλαίσιο HAM για την παραγωγή ενεργειακά-βέλτιστων υβριδικών προσεγγιστικών πολλαπλασιαστών απεικονίζεται στο Σχήμα 8.11 και λειτουργεί ως ακολούθως. Δεδομένου ενός ορθού πολλαπλασιαστή εφαρμόζουμε κάθε προσεγγιστική τεχνική ξεχωριστά και πραγματοποιούμε μια δειγματοληψία για να εκπαιδεύσουμε τα προτεινόμενα μοντέλα σφάλματος και σχετικής κατανάλωσης. Στη συνέχεια εκτελούμε μια ευριστική διαδικασία για να εξάγουμε τον υβριδικό πολλαπλασιαστή. Κατά τη διάρκεια της ευριστικής διαδικασίας χρησιμοποιείται μια τροποποιημένη δυαδική αναζήτηση στον χώρο λύσεων που καθορίζεται από τα προτεινόμενα αναλυτικά μοντέλα E_H και RP_H έτσι ώστε να εξαχθεί γρήγορα η διαμόρφωση του υβριδικού πολλαπλασιαστή. Έπειτα, επαληθεύεται η λύση αυτή συνθέτοντας και προσομοιώνοντας τον αντίστοιχο υβριδικό πολλαπλασιαστή που εξήχθη έτσι ώστε να μετρηθεί το πραγματικό του σφάλμα. Σε περίπτωση που παραβιάζεται το όριο του σφάλματος, ξανα-εκτελείται η ευριστική διαδικασία.

Το προτεινόμενο πλαίσιο αξιολογήθηκε πειραματικά εκτενώς μελετώντας τόσο την αποδοτικότητα των πολυ-επίπεδων προσεγγίσεων όσο και την ικανότητα του HAM να βρίσκει βέλτιστες λύσεις. Στην πειραματική διαδικασία που εκτελέστηκε, το πλαίσιο HAM συγκρίθηκε τόσο με υπάρχουσες μονο-επίπεδες τεχνικές όσο και με την εξαντλητική εξερεύνηση του χώρου σχεδιασμού μέσω της τεχνικής της πλήρους αναζήτησης (full search design space exploration). Για την πειραματική αξιολόγηση χρησιμοποιήθηκε ένας 16-bit Dadda 4:2 πολλαπλασιαστής. Στο Σχήμα 8.12 απεικονίζονται όλοι οι μονο-επίπεδοι και πολυ-επίπεδοι υβριδικοί πολλαπλασιαστές που εξήχθησαν από την πλήρη εξερεύνηση του χώρου σχεδιασμού και έχουν κανονικοποιημένο μέσο απόλυτο σφάλμα



Σχήμα 8.12: Το σφάλμα και η σχετική κατανάλωση ισχύος όλων των μονο-επίπεδων και πολυ-επίπεδων υβριδικών πολλαπλασιαστών με τιμή σφάλματος $NMED < 0.01$.

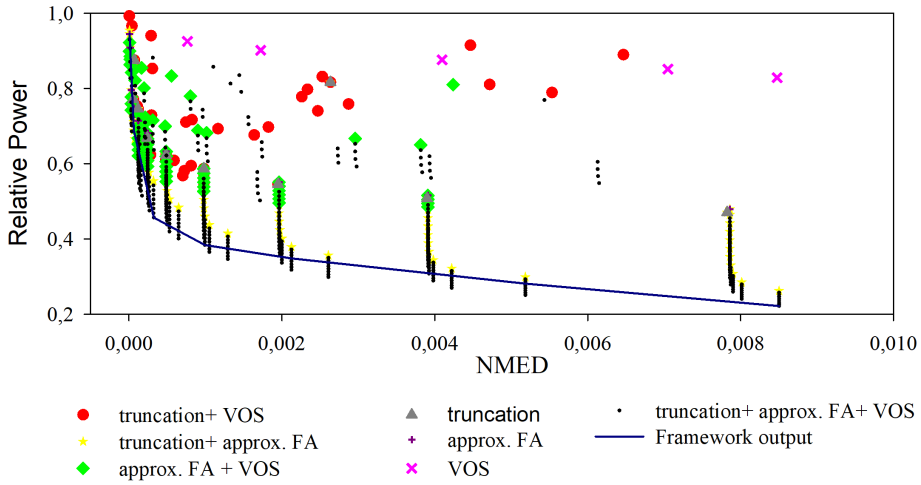
$NMED < 0.01$. Όλα τα σημεία του Σχήματος 8.12 είναι ομαδοποιημένα με χρώμα με βάση τις προσεγγιστικές τεχνικές που εφαρμόζουν. Για παράδειγμα οι μαύρες κουκίδες είναι οι υβριδικοί πολλαπλασιαστές που εφαρμόζουν και τις τρεις τεχνικές ενώ τα γκρι, μωβ, και ροζ σημεία είναι οι υπάρχοντες μονο-επίπεδοι προσεγγιστικοί πολλαπλασιαστές. Επίσης, η μπλε γραμμή είναι η έξοδος του προτεινόμενου πλαισίου HAM. Όπως απεικονίζεται στο Σχήμα 8.12 οι υβριδικοί πολυ-επίπεδοι πολλαπλασιαστές σχηματίζουν το Pareto μέτωπο αποτελώντας τις βέλτιστες λύσεις. Επίσης, παρατηρείται ότι η έξοδος του προτεινόμενου πλαισίου συγκλίνει στο ακριβές Pareto μέτωπο που παράχθηκε από την πλήρη εξερεύνηση του χώρου σχεδιασμού. Συγκρινόμενο με την πλήρη εξερεύνηση το προτεινόμενο πλαίσιο πετυχαίνει 254 φορές γρηγορότερη αναζήτηση και οι λύσεις που εξάγει είναι στο 97% των βέλτιστων λύσεων. Σε σύγκριση με τον ορθό πολλαπλασιαστή οι πολλαπλασιαστές που παράγονται από το HAM έχουν από 35% μέχρι και 72% χαμηλότερη κατανάλωση. Σε σύγκριση με τους υπάρχοντες μονο-επίπεδους προσεγγιστικούς πολλαπλασιαστές, οι αντίστοιχες τιμές είναι 11%-48%.

8.5 Πλαίσιο METHADONE

Η ενότητα αυτή βασίζεται στη δημοσίευση μας [115] στο περιοδικό Trans. on Circuits and Systems II 2018.

Όπως προαναφέραμε ο προσεγγιστικός υπολογισμός εισάγει τη διάσταση του σφάλματος και δυσχεραίνει το ήδη δύσκολο έργο του σχεδιασμού κυκλωμάτων. Η πολυπλοκότητα αυτή αυξάνει ακόμα περισσότερο όταν εφαρμόζουμε πολυ-επίπεδες προσεγγίσεις καθώς το μέγεθος του χώρου σχεδίασης γιγαντώνεται. Μολονότι ο σχεδιασμός προσεγγιστικών αριθμητικών μονάδων έχει κεντρίσει το ερευνητικό ενδιαφέρον, ερευνητικές δραστηριότητες που στοχεύουν στη δημιουργία προσεγγιστικών επιταχυντών υλικού είναι πολύ περιορισμένες λόγω της αυξημένης πολυπλοκότητας. Επιπροσθέτως, καμία από τις υπάρχουσες μεθοδολογίες και τεχνικές δεν εφαρμόζει πολυ-επίπεδες προσεγγίσεις σε επιταχυντές υλικού. Όμως, όπως είδαμε στην πειραματική αξιολόγηση του HAM, οι βέλτιστες λύσεις εξασφαλίζονται όταν χρησιμοποιούνται πολυ-επίπεδες προσεγγίσεις. Με στόχο να καλύψουμε αυτές τις ελλείψεις, προτείνουμε το πλαίσιο METHADONE το οποίο, δεδομένης της συμπεριφορικής περιγραφής ενός επιταχυντή υλικού και ενός ορίου σφάλματος, παράγει με ένα αυτοματοποιημένο τρόπο τον αντίστοιχο προσεγγιστικό επιταχυντή που τηρεί το δοσμένο σφάλμα και εφαρμόζει πολυ-επίπεδες προσεγγίσεις για να μεγιστοποιήσει τα ενεργειακά οφέλη. Το προτεινόμενο πλαίσιο λειτουργεί πάνω στο γράφο ροής δεδομένων του επιταχυντή και εφαρμόζει πολυ-επίπεδες προσεγγίσεις στις αριθμητικές του μονάδες. Με αυτό τον τρόπο, το προτεινόμενο πλαίσιο μπορεί να εφαρμοστεί σε κάθε κύκλωμα.

Για την υλοποίηση του METHADONE το πρώτο πράγμα που κάναμε ήταν να επεκτείνουμε το προτεινόμενο πλαίσιο HAM και να δημιουργήσουμε το HAMA (Hybrid Approximate Multipliers & Adders) το οποίο εκτός από πολλαπλασιαστές υποστηρίζει αθροιστές και αφαιρέτες. Παρόμοια με το HAM, στο σχεδιασμό πολυ-επίπεδων αθροιστών εφαρμόζουμε αποκοπή στο αλγοριθμικό επίπεδο, χρησιμοποιούμε τον προσεγγιστικό πλήρη αθροιστή [61] στο λογικό επίπεδο, και εφαρμόζουμε VOS στο κυκλωματικό επίπεδο. Όπως φαίνεται στο Σχήμα 8.13, όπως και στους υβριδικούς πολλαπλασιαστές, πήραμε τα ίδια αποτελέσματα. Οι υβριδικοί αθροιστές σχηματίζουν το Pareto μέτωπο και αποτελούν τις βέλτιστες λύσεις. Το πλαίσιο HAMA συγκλίνει στο Pareto

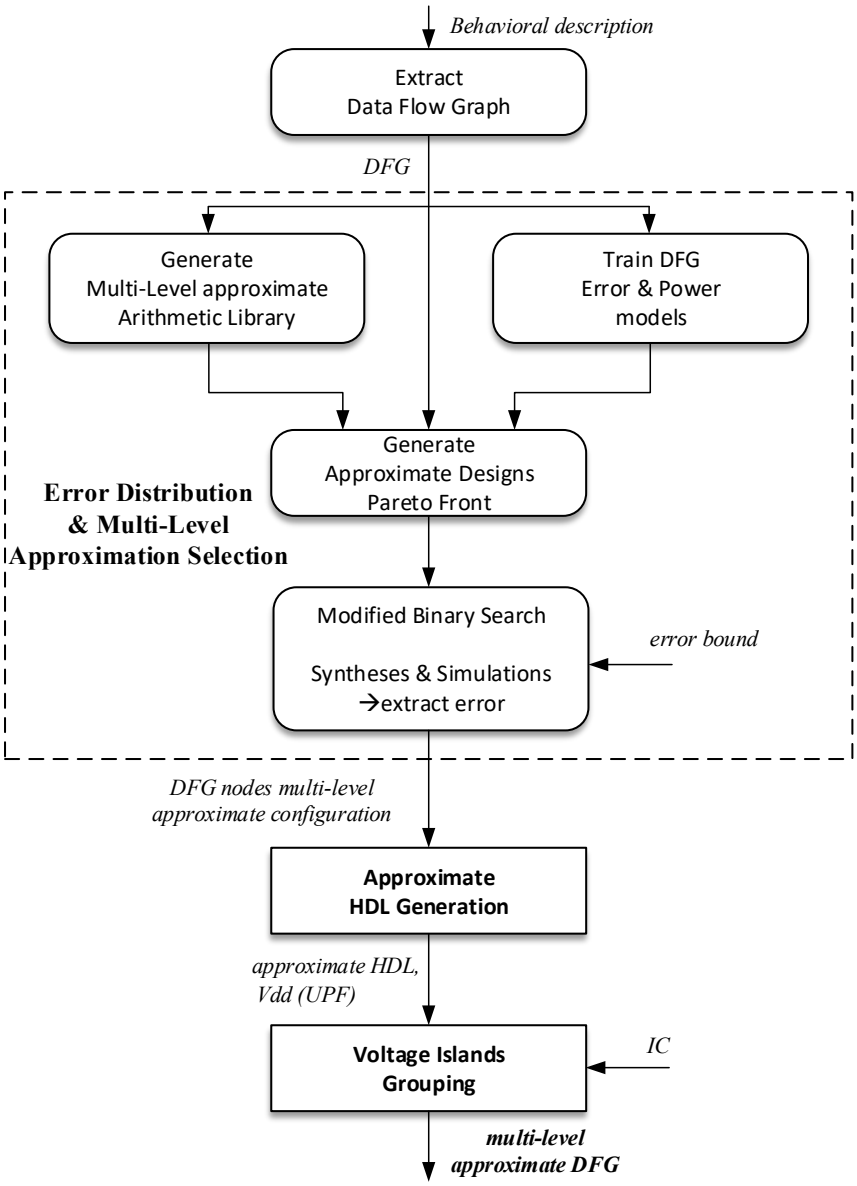


Σχήμα 8.13: Το σφάλμα και η σχετική κατανάλωση ισχύος όλων των μονο-επίπεδων και πολυ-επίπεδων υβριδικών αθροιστών (16-bit) με τιμή σφάλματος $NMED < 0.01$.

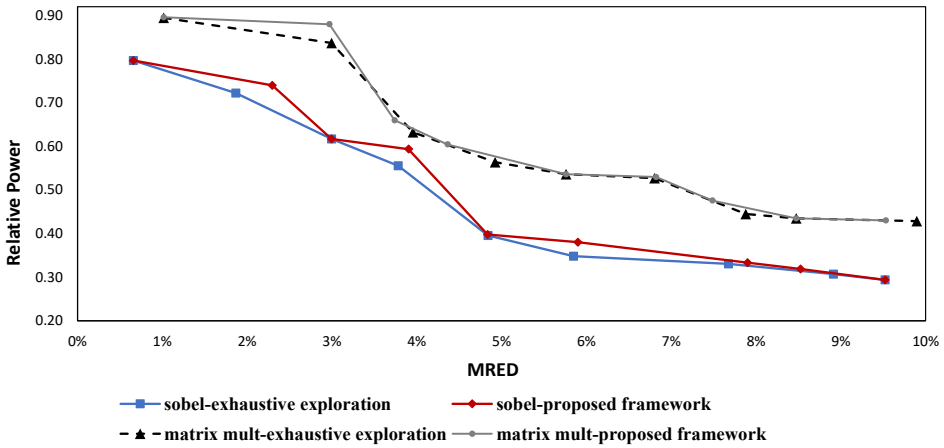
μέτωπο της πλήρους εξερεύνησης του χώρου σχεδιασμού και προσφέρει επιτάχυνση στο χρόνο αναζήτησης κατά 250 φορές. Αντίστοιχα με το HAM ο εκτιμητής σφάλματος του υβριδικού αθροιστή έχει δείκτη Pearson 0.87 και μέσο τετραγωνικό σφάλμα 2.95×10^{-5}. Οι αντίστοιχες τιμές για τον εκτιμητή της σχετικής κατανάλωσης είναι 0.99 και 2.19×10^{-5}. Σε σύγκριση με τον ορθό αθροιστή οι αθροιστές που παράγονται από το HAMA έχουν από 36% μέχρι και 80% χαμηλότερη κατανάλωση. Σε σύγκριση με τους υπάρχοντες μονο-επίπεδους προσεγγιστικούς αθροιστές, οι αντίστοιχες τιμές είναι 6%-40%.

Εκμεταλλευόμενοι το πλαίσιο HAMA, μπορούμε, αφου ανιχνεύσουμε όλους τους πολλαπλασιαστές και αθροιστές/αφαιρέτες στο γράφο ροής δεδομένων του επιταχυντή, να το χρησιμοποιήσουμε για να παράξουμε μια πολυ-επίπεδη προσεγγιστική βιβλιοθήκη για κάθε αριθμητικό κόμβο του γράφου. Το πρόβλημα βελτιστοποίησης που καλούμαστε να λύσουμε, είναι να βρούμε τη διαμόρφωση του γράφου (πολυ-επίπεδη προσεγγιστική διαμόρφωση του κάθε κόμβου) η οποία ελαχιστοποιεί την κατανάλωση ισχύος όλου του γράφου ενώ παράλληλα τηρεί και το όριο του σφάλματος. Χρησιμοποιώντας το HAMA ξέρουμε

για κάθε κόμβο το σφάλμα και την κατανάλωση ισχύος του, αλλά δεν μας παρέχει καμία πληροφορία για το ολικό σφάλμα και την ολική κατανάλωση του γράφου. Για το λόγο αυτό προτείνουμε να εκτιμήσουμε το σφάλμα και την κατανάλωση του γράφου με βάση τα αντίστοιχα των κόμβων. Η κατανάλωση ισχύος του γράφου μπορεί να εκτιμηθεί από ένα γραμμικό συνδυασμό της κατανάλωσης των κόμβων καθώς δεν παράγεται ισχύς μέσα στο κύκλωμα και η ολική κατανάλωση προσεγγίζεται από το άθροισμα των καταναλώσεων των υποκυκλωμάτων. Ως εκ τούτου χρησιμοποιούμε γραμμική παλινδρόμηση για να εκτιμήσουμε την ολική κατανάλωση του γράφου με βάση την κατανάλωση του κάθε κόμβου. Για το σφάλμα του γράφου όμως δεν έχουμε καμία εκτίμηση για το πώς διαδίδεται στο γράφο και πώς αντισταθμίζεται στους διάφορους προσεγγιστικούς κόμβους και για αυτό χρησιμοποιούμε ένα νευρωνικό δίκτυο για να εκτιμήσουμε το ολικό σφάλμα με βάση το σφάλμα του κάθε κόμβου. Εκτελέσαμε μια τυχαία δειγματοληψία διαμορφώσεων του γράφου τις οποίες συνθέσαμε και προσομοιώσαμε για να μετρήσουμε το σφάλμα και την κατανάλωση του γράφου και εκπαιδεύσαμε τους προτεινόμενους εκτιμητές. Ο εκτιμητής του σφάλματος του γράφου έχει Pearson δείκτη 0.90 και μέσο τετραγωνικό σφάλμα 9×10^{-5}. Οι αντίστοιχες τιμές για τον εκτιμητή της κατανάλωσης ισχύος του γράφου είναι 0.98 και 6×10^{-7}. Οι πολύ υψηλές τιμές δείχνουν ότι τα προτεινόμενα μοντέλα μπορούν να εκτιμήσουν με μεγάλη ακρίβεια τόσο το σφάλμα όσο και την κατανάλωση καθώς και τη διάταξη των πολυ-επίπεδων προσεγγιστικών γράφων. Το προτεινόμενο πλαίσιο απεικονίζεται στο Σχήμα 8.14. Δεδομένου του γράφου ροής δεδομένων, εξάγει τους αριθμητικούς κόμβους και χρησιμοποιεί το HAMA για να παράξει μια πολυ-επίπεδη προσεγγιστική βιβλιοθήκη για τον καθένα. Στην συνέχεια εκτελεί μια τυχαία δειγματοληψία και εκπαιδεύει τους εκτιμητές σφάλματος και κατανάλωσης. Κατά τη διάρκεια της εκπαίδευσης, το πλαίσιο VOSsim χρησιμοποιείται για την εκτέλεση όλων των προσομοιώσεων υπερκλιμακούμενης τάσης. Στη συνέχεια εκτελείται μια ευριστική διαδικασία (τροποποιημένη δυαδική αναζήτηση και Pareto ανάλυση) στον χώρο σχεδιασμού που ορίζεται από τα προτεινόμενα αναλυτικά μοντέλα και εξάγεται η λύση (πολυ-επίπεδη προσεγγιστική διαμόρφωση του γράφου). Η λύση αυτή επαληθεύεται συνθέτοντας και προσομοιώνοντας τη διαμόρφωση του γράφου που εξήχθη έτσι ώστε να μετρηθεί το πραγματικό του σφάλμα. Κατά την επαλήθευση, οι προσομοιώσεις υπερκλιμακούμενης τάσης γίνονται με τον SPICE προσομοιωτή CustomSim. Σε περίπτωση που παραβιάζεται το όριο του σφάλματος, ξανα-εκτελείται η προηγούμενη ευριστική διαδικασία.



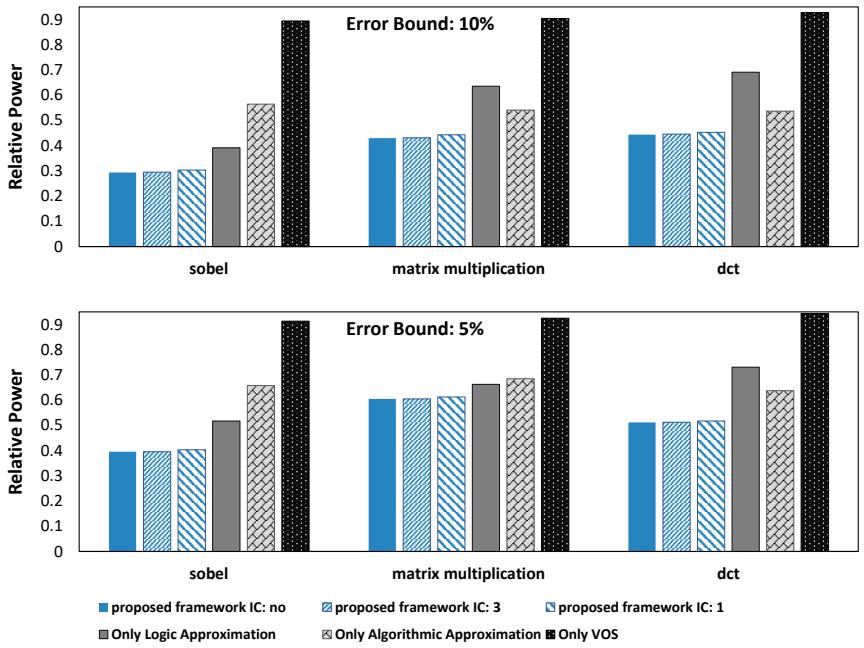
Σχήμα 8.14: Το προτεινόμενο πλαίσιο METHADONE για σύνθεση πολυ-επίπεδων επιταχυντών υλικού.



Σχήμα 8.15: Το Pareto μέτωπο των πολυ-επίπεδων προσεγγιστικών επιταχυντών, για όρια σφάλματος από 1% μέχρι 10%, όπως εξήχθησαν από το προτεινόμενο πλαίσιο METHADONE και την πλήρη αναζήτηση του χώρου σχεδιασμού για το φίλτρο Sobel και τον πολλαπλασιασμό πινάκων.

Τέλος, συμπεριλάβαμε κι ένα στάδιο μείωσης των τιμών τάσεις που απαιτούνται. Η τόσο λεπτομερής εφαρμογή της υπερκλιμάκωσης της τάσης (ανά αριθμητικό κόμβο) μπορεί να οδηγήσει σε μεγάλη ποικιλία τιμών τάσεων. Η απαίτηση για τόσο πολλές διαφορετικές τιμές τάσης μπορεί είτε να μην είναι εφικτή, λόγω περιορισμών της τεχνολογίας, είτε να μην είναι αποδοτική για σχεδιαστικούς λόγους. Ο ρόλος λοιπόν αυτού του τελευταίου σταδίου είναι να ομαδοποιήσει με τέτοιο τρόπο τις απαιτούμενες νησίδες τάσης (voltage islands) ώστε να μειώσει αποδοτικά τον αριθμό των αναγκαίων διαφορετικών τιμών των τάσεων.

Το προτεινόμενο πλαίσιο αξιολογήθηκε πειραματικά εκτενώς μελετώντας τόσο την αποδοτικότητα των πολυ-επίπεδων επιταχυντών υλικού που εξάγει όσο και την ικανότητά του να εξάγει Pareto βέλτιστες λύσεις. Στην πειραματική διαδικασία που εκτελέστηκε, το πλαίσιο METHADONE συγκρίθηκε τόσο με υπάρχουσες μονο-επίπεδες τεχνικές όσο και με την εξαντλητική εξερεύνηση του χώρου σχεδιασμού μέσω της τεχνικής της πλήρους αναζήτησης (full search design space exploration). Για την πειραματική αξιολόγηση χρησι-



Σχήμα 8.16: Σύγκριση των πολυ-επίπεδων προσεγγιστών υλικού που παράγονται από το προτεινόμενο πλαίσιο METHADONE με υπάρχουσες μονο-επίπεδες προσεγγιστικές τεχνικές. Όρια σφάλματος 5% και 10% εξετάζονται.

μοποιήθηκαν τρεις τυπικές εφαρμογές του προσεγγιστικού υπολογισμού: το φίλτρο Sobel, ο πολλαπλασιασμός πινάκων, και το 1-D DCT. Στο Σχήμα 8.15, για το φίλτρο Sobel και τον πολλαπλασιασμό πινάκων, απεικονίζεται Pareto μέτωπο των πολυ-επίπεδων προσεγγιστικών επιταχυντών, για όρια σφάλματος από 1% μέχρι 10%, που παρήχθησαν από α) το προτεινόμενο πλαίσιο METHADONE και β) την πλήρη αναζήτηση του χώρου σχεδιασμού. Όπως βλέπουμε, και για τις δύο εφαρμογές οι δύο καμπύλες είναι πολύ κοντά. Σε σύγκριση με την πλήρη αναζήτηση του χώρου σχεδιασμού, το METHADONE μπορεί να παράξει με μεγάλη ακρίβεια πολυ-επίπεδους επιταχυντές υλικού κοντά στους Pareto-βέλτιστους προσφέροντας επιτάχυνση 1208 φορές (κατά μέσο όρο) στο χρόνο αναζήτησης. Στο Σχήμα 8.16 εξετάζουμε την αποδοτικότητα των πολυ-επίπεδων προσεγγιστικών επιταχυντών υλικού και συγκρίνουμε τα κυκλώματα

που παράγονται από το METHADONE με υπάρχουσες μονο-επίπεδες προσεγγιστικές τεχνικές. Σε σύγκριση με το ορθό κύκλωμα το METHADONE επιτυγχάνει 50%-62% χαμηλότερη κατανάλωση ενώ σε σύγκριση με τις υπάρχουσες τεχνικές τα κέρδη είναι 18%-31% χαμηλότερη κατανάλωση. Συμπερασματικά, το METHADONE μπορεί μέσω μιας αυτοματοποιημένης διαδικασίας να παράξει πάρα πολύ γρήγορα κοντά στους Pareto-βέλτιστους πολυ-επίπεδους προσεγγιστικούς επιταχυντές υλικού οι οποίοι επιτυγχάνουν πολύ μεγάλα ενεργειακά οφέλη σε σχέση με το ορθό κύκλωμα και σε σχέση με υπάρχουσες μονο-επίπεδες προσεγγιστικές τεχνικές.

Γλωσσάρι

Accurate Circuit	Ορθό κύκλωμα. Το κύκλωμα αναφοράς, χωρίς την εφαρμογή προσεγγιστικών τεχνικών.
Approximate Circuit	Το προσεγγιστικό κύκλωμα. Το κύκλωμα που προκύπτει από την εφαρμογή προσεγγίσεων στο ορθό κύκλωμα.
Approximate Computing	Προσεγγιστικός Υπολογισμός. Είναι μια τεχνική σχεδιασμού υπολογιστικών συστημάτων και εφαρμογών. Ο Προσεγγιστικός Υπολογισμός χαλαρώνει την ακρίβεια των εκτελούμενων υπολογισμών για να επιφέρει κέρδη σε άλλες μετρικές όπως κατανάλωση ισχύος και/ή απόδοση.
Behavioral Description	Περιγραφή κυκλώματος σε επίπεδο συμπεριφοράς.
Behavioral Synthesis	Σύνθεση κυκλώματος περιγραμμένο σε επίπεδο συμπεριφοράς.
Circuit Synthesis	Η διαδικασία σύνθεσης του κυκλώματος. Για παράδειγμα η παραγωγή της περιγραφής του κυκλώματος σε επίπεδο λογικών πυλών δεδομένης της περιγραφής του κυκλώματος σε επίπεδο μεταφοράς καταχωρητών.

Critical Path Delay	Καθυστέρηση κρίσιμου μονοπατιού του κυκλώματος.
Data Flow Graph (DFG)	Γράφος Ροής Δεδομένων.
Design Space Exploration (DSE)	Εξερεύνηση του χώρου σχεδίασης.
Error Distance (ED)	Απόλυτη Απόσταση. Είναι μετρική σφάλματος και ορίζεται ως η απόλυτη τιμή της διαφοράς του αποτελέσματος (έξοδος) του ορθού κυκλώματος μείον αυτής του αντίστοιχου προσεγγιστικού. $ED = \|P - P'\|$, όπου P είναι το ορθό αποτέλεσμα και P' το προσεγγιστικό.
Error value	Το σφάλμα του προσεγγιστικού κυκλώματος. Υπολογίζεται συγκρίνοντας την έξοδο του ορθού κυκλώματος και του αντίστοιχου προσεγγιστικού, χρησιμοποιώντας μια κατάλληλη μετρική ποιότητας/ακρίβειας.
Flip Flop	Καταχωρητής. Σύγχρονα ακολουθιακά κυκλώματα.
Flip Flop Timing Violation	Σφάλματα χρονισμού των καταχωρητών.
Gate-level Description	Περιγραφή κυκλώματος σε Επίπεδο Λογικών Πυλών. Η περιγραφή αυτή μπορεί να παραχθεί με την σύνθεση της RTL περιγραφής ενός κυκλώματος για μια δεδομένη τεχνολογική βιβλιοθήκη.

Gate-level Timing Simulation	Προσομοίωση κυκλώματος περιγραμμένου σε επίπεδο λογικών πυλών που λαμβάνει υπόψη τον χρονισμό των πυλών. Συνήθως εφαρμόζεται μετά τη σύνθεση του κυκλώματος.
Hardware Accelerator	Κύκλωμα ειδικού σκοπού/συνεπεξεργαστής υλικού που σχεδιάζεται/χρησιμοποιείται για την επιτάχυνση μιας εφαρμογής/διαδικασίας.
High-level Synthesis	Σύνθεση κυκλώματος από υψηλά επίπεδα σχεδιαστικής αφαίρεσης.
Linear Regression	Απλή γραμμική παλινδρόμηση.
Mean Error Distance (MED)	Μέση Απόλυτη Απόσταση. Είναι η μέση τιμή της απόλυτης απόστασης (ED) για την εξεταζόμενη κατανομή εισόδου και ορίζεται ως $MED = \frac{1}{M}\sum_{i=1}^{M} \lvert P_i - P_i' \rvert$, όπου P_i είναι το ορθό αποτέλεσμα, P_i' το προσεγγιστικό αποτέλεσμα, και M το πλήθος των αποτελεσμάτων.
Mean Relative Error Distance (MRED)	Μέση Σχετική Απόσταση. Είναι η μέση τιμή της σχετικής απόστασης (RED) για την εξεταζόμενη κατανομή εισόδου και ορίζεται ως $MRED = \frac{1}{M}\sum_{i=1}^{M} \lvert \frac{P_i - P_i'}{P_i} \rvert$, όπου P_i είναι το ορθό αποτέλεσμα, P_i' το προσεγγιστικό αποτέλεσμα, και M το πλήθος των αποτελεσμάτων.
Multi-level Approximation	Η τεχνική της εφαρμογής πολλαπλών και από διαφορετικά επίπεδα προσεγγιστικών τεχνικών. Στον σχεδιασμό προσεγγιστικών κυκλωμάτων, οι προσεγγίσεις μπορούν να εφαρμοστούν στο αλγοριθμικό, το λογικό, και το κυκλωματικό επίπεδο.

Normalized Mean Error Distance (RED)	Κανονικοποιημένη Μέση Απόλυτη Απόσταση. Είναι μετρική σφάλματος και ορίζεται ως η μέση τιμή της απόλυτης απόστασης (MED) ως προς τη μέγιστη πιθανή τιμή του σφάλματος. Για παράδειγμα σε ένα n-bit η μέγιστη πιθανή τιμή του σφάλματος είναι 2^n. $NMED = MED/D$ όπου D είναι το μέγιστο σφάλμα.
Pareto front	Μέτωπο Pareto. Σε προβλήματα βελτιστοποίησης με πολλαπλά κριτήρια, οι Pareto βέλτιστες λύσεις είναι αυτές για τις όποιες η βελτιστοποίηση του ενός κριτηρίου μπορεί να επιτευχθεί μόνο αν χειροτερεύσει τουλάχιστον ένα άλλο κριτήριο.
Partial Product Generation with Modified Booth Encoding (MBE)	Παραγωγή των μερικών γινομένων του πολλαπλασιασμού με την τροποποιημένη τεχνική του Booth.
Register Transfer Level (RTL) Description	Περιγραφή κυκλώματος σε Επίπεδο Μεταφοράς Καταχωρητών.
Relative Error Distance (RED)	Σχετική Απόσταση. Είναι μετρική σφάλματος και ορίζεται ως η απόλυτη τιμή του λόγου διαφοράς του αποτελέσματος (έξοδος) του ορθού κυκλώματος μείον αυτής του αντίστοιχου προσεγγιστικού προς το αποτέλεσμα του ορθού κυκλώματος. $RED = \|(P - P')/P\|$, όπου P είναι το ορθό αποτέλεσμα και P' το προσεγγιστικό.
Simple Partial Product (SPP) Generation	Παραγωγή των μερικών γινομένων του πολλαπλασιασμού με την παραδοσιακή τεχνική που διδάσκεται στο δημοτικό σχολείο.

SPICE	Διεθνές πρότυπο αλγορίθμων για προσομοίωση κυκλωμάτων σε επίπεδο τρανζίστορ.
Volage Over-Scaling (VOS)	Υπερκλιμάκωση της τάσης. Προσεγγιστική τεχνική που εφαρμόζεται στο κυκλωματικό επίπεδο. Η τεχνική αυτή εφαρμόζεται διατηρώντας το κύκλωμα και τη συχνότητα λειτουργίας του σταθερά και μειώνοντας την τάση λειτουργίας χαμηλότερα από την ονομαστική της τιμή.
Volage Islands	Νησίδες Τάσης.

Publications

Journals

1. G. Zervakis, S. Xydis, D. Soudris and K. Pekmestzi, *"Multi-Level Approximate Accelerator Synthesis Under Voltage Island Constraints,"* in IEEE Transactions on Circuits and Systems II: Express Briefs. 2018. doi: 10.1109/TCSII.2018.2869025

2. V. Leon, G. Zervakis, S. Xydis, D. Soudris and K. Pekmestzi, *"Walking through the Energy-Error Pareto Frontier of Approximate Multipliers,"* in IEEE Micro, vol. 38, no. 4, pp. 40-49, Jul./Aug. 2018. doi: 10.1109/MM.2018.043191124

3. G. Zervakis, F. Ntouskas, S. Xydis, D. Soudris and K. Pekmestzi, *"VOSsim: A Framework for Enabling Fast Voltage Overscaling Simulation for Approximate Computing Circuits,"* in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 6, pp. 1204-1208, June 2018. doi: 10.1109/TVLSI.2018.2803202

4. V. Leon, G. Zervakis, D. Soudris and K. Pekmestzi, *"Approximate Hybrid High Radix Encoding for Energy Efficient Inexact Multipliers,"* in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 26, no. 3, pp. 421-430, March 2018. doi: 10.1109/TVLSI.2017.2767858

5. G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris and K. Pekmestzi, *"Design Efficient Approximate Multiplication Circuits Through Par-*

tial Product Perforation," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 24, no. 10, pp. 3105-3117, Oct. 2016.
doi: 10.1109/TVLSI.2016.2535398

6. K. Tsoumanis, N. Axelos, N. Moschopoulos, G. Zervakis and K. Pekmestzi, *"Pre-Encoded Multipliers Based on Non-Redundant Radix-4 Signed-Digit Encoding,"* in IEEE Transactions on Computers, vol.65, no.2, pp.670-676, Feb. 1 2016.
doi: 10.1109/TC.2015.2428691

7. K. Tsoumanis, S. Xydis, G. Zervakis and K. Pekmestzi, *"Flexible DSP Accelerator Architecture Exploiting Carry-Save Arithmetic,"* in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.24, no.1, pp.368-372, Jan. 2016.
doi: 10.1109/TVLSI.2015.2390974

Book chapters

1. G. Zervakis. Chapter 9: On Accelerating Data Analytics: An Introduction to the Approximate Computing Technique. In *IoT for Smart Grids: Design Challenges and Paradigms*. 2019. 163-180. Springer, Cham.

2. C. Kachris, E. Koromilas, I. Stamelos, G. Zervakis, S. Xydis and D. Soudris. Chapter 5: Energy-Efficient Acceleration of Spark Machine Learning Applications on FPGAs. In *Hardware Accelerators in Data Centers*. 2019. 87-107. Springer, Cham.

Conferences

1. D. Masouros, K. Koliogeorgi, G. Zervakis, A. Kosvyra, A. Chytas, S. Xydis, I. Chouvarda, and D. Soudris, *"Co-design Implications of On-Demand-Acceleration for Cloud Healthcare Analytics: The AEGLE approach"*, Design Automation Conference (DATE 2019), accepted for publication, 2019

2. K. Koliogeorgi, D. Massouros, G. Zervakis, S. Xydis, T. Becker, G. Gaydadjiev and D. Soudris, *"AEGLE's Cloud Infrastructure for Resource Monitoring and Containerized Accelerated Analytics,"* in 2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Bochum, 2017, pp. 362-367.

3. G. Zervakis, S. Xydis and D. Soudris, *"Performance-Power Exploration of Software-Defined Big Data Analytics: The AEGLE Cloud Backend,"* in International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XVI), 2016

4. G. Zervakis, S. Xydis, K. Tsoumanis, D. Soudris and K. Pekmestzi, *"Hybrid Approximate Multiplier Architectures for Improved Power-Accuracy Trade-offs,"* in International Symposium on Low Power Electronics and Design (ISLPED), 2015, pp. 79-84

5. N. Eftaxiopoulos, N. Axelos, G. Zervakis, K. Tsoumanis and K. Pekmestzi, *"Delta DICE: A Double Node Upset Resilient Latch,"* in Circuits and Systems (MWSCAS), 2015 IEEE 58th International Midwest Symposium on, vol., no., pp.1-4, 2-5 Aug. 2015

6. G. Zervakis, K. Tsoumanis, S. Xydis, N. Axelos and K. Pekmestzi, *"Approximate Multiplier Architectures Through Partial Product Perforation: Power-Area Tradeoffs Analysis,"* in Proc. of the 25th ACM Great Lakes VLSI Symposium (GLSVLSI), 2015, pp. 229-232.

7. N. Axelos, N. Eftaxiopoulos, G. Zervakis, K. Tsoumanis and K. Pekmestzi, *"FF-DICE: An 8T soft-error tolerant cell using Independent*

Dual Gate SOI FinFETs," in International On-Line Testing Symposium (IOLTS), 2014

8. G. Zervakis, N. Eftaxiopoulos, K. Tsoumanis, N. Axelos and K. Pekmestzi, *"A Segmentation-Based BISR Scheme,"* in Asia and South Pacific Design Automation Conference (ASP-DAC), 2014

9. N. Eftaxiopoulos, N. Axelos, G. Zervakis, K. Tsoumanis, and K. Pekmestzi, *"An independent dual gate SOI FinFET soft-error resilient memory cell,"* in 9th International Design & Test Symposium (IDT), 2014

10. G. Zervakis, N. Eftaxiopoulos, K. Tsoumanis, N. Axelos and K. Pekmestzi, *"A high radix Montgomery multiplier with concurrent error detection,"* in 9th International Design & Test Symposium (IDT), 2014

11. N. Eftaxiopoulos, G. Zervakis, K. Pekmestzi and C. Efstathiou, *"High performance MAC designs,"* in 9th International Design & Test Symposium (IDT), 2014

12. N. Eftaxiopoulos, G. Zervakis, K. Tsoumanis and K. Pekmestzi, *"A radiation tolerant and self-repair memory cell,"* in International On-Line Testing Symposium (IOLTS), 2013

Workshops

1. G. Zervakis, S. Xydis, V. Tsoutsouras, D. Soudris and K. Pekmestzi, *"Multi-Level Approximation for Inexact Accelerator Synthesis Under Voltage Island Constraints,"* in Workshop on Approximate Computing AC16, part of ESWEEK 2016, 2016

Bibliography

[1] H. Rong, H. Zhang, S. Xiao, C. Li, and C. Hu, "Optimizing energy consumption for data centers," *Renewable and Sustainable Energy Reviews*, vol. 58, pp. 674 – 691, 2016.

[2] K. Kant, "Data center evolution: A tutorial on state of the art, issues, and challenges," *Computer Networks*, vol. 53, no. 17, pp. 2939 – 2965, 2009.

[3] A. Shehabi, S. J. Smith, D. A. Sartor, R. E. Brown, M. Herrlin, J. G. Koomey, E. R. Masanet, H. Nathaniel, I. L. Azevedoan, and L. William, "United states data center energy usage report," tech. rep., Lawrence Berkley National Laboratory, 2016.

[4] E. Oro, V. Depoorter, A. Garcia, and J. Salom, "Energy efficiency and renewable energy integration in data centres. strategies and modelling review," vol. 42, 02 2015.

[5] D. Beaty, "Internal it load profile variability," vol. 55, pp. 72–74, 02 2013.

[6] L. Castellazzi, A. Maria, and P. Bertoldi, "Trends in data centre energy consumption under the european code of conduct for data centre energy efficiency," tech. rep., European Commission, Joint Research Centre (JRC), 2017.

[7] J. Ni and X. Bai, "A review of air conditioning energy performance in data centers," *Renewable and Sustainable Energy Reviews*, vol. 67, pp. 625 – 640, 2017.

[8] T. A. Henzinger and J. Sifakis, "The embedded systems design challenge," in *Proceedings of the 14th International Conference on Formal Methods*, FM'06, (Berlin, Heidelberg), pp. 1–15, Springer-Verlag, 2006.

[9] H. Ghasemzadeh and R. Jafari, "Ultra low-power signal processing in wearable monitoring systems: A tiered screening architecture

with optimal bit resolution," *ACM Trans. Embed. Comput. Syst.*, vol. 13, pp. 9:1–9:23, Sept. 2013.

[10] "Energy efficiency." https://ec.europa.eu/energy/en/topics/energy-efficiency.

[11] G. E. Moore, "Cramming more components onto integrated circuits, reprinted from electronics, volume 38, number 8, april 19, 1965, pp.114 ff.," *IEEE Solid-State Circuits Society Newsletter*, vol. 11, pp. 33–35, Sept 2006.

[12] H. Esmaeilzadeh, *Approximate Acceleration for a Post Multicore Era.* PhD thesis, University of Washington, 2013.

[13] K. J. Kuhn, "Moore's law past 32nm: Future challenges in device scaling," in *2009 13th International Workshop on Computational Electronics*, pp. 1–6, May 2009.

[14] K. J. Kuhn, "Cmos transistor scaling past 32nm and implications on variation," in *2010 IEEE/SEMI Advanced Semiconductor Manufacturing Conference (ASMC)*, pp. 241–246, July 2010.

[15] M. Bohr, "Continuing moore's law." https://newsroom.intel.com/newsroom/wp-content/uploads/sites/11/2017/09/mark-bohr-on-continuing-moores-law.pdf, 2017.

[16] R. H. Dennard, F. H. Gaensslen, V. L. Rideout, E. Bassous, and A. R. LeBlanc, "Design of ion-implanted mosfet's with very small physical dimensions," *IEEE Journal of Solid-State Circuits*, vol. 9, pp. 256–268, Oct 1974.

[17] M. Bohr, "A 30 year retrospective on dennard's mosfet scaling paper," *IEEE Solid-State Circuits Society Newsletter*, vol. 12, pp. 11–13, Winter 2007.

[18] G. G. Shahidi, J. D. Warnock, J. Comfort, S. Fischer, P. A. McFarland, A. Acovic, T. I. Chappell, B. A. Chappell, T. H. Ning, C. J. Anderson, R. H. Dennard, J. Y. C. Sun, M. R. Polcari, and B. Davari, "Cmos scaling in the 0.1-um, 1.x-volt regime for high-performance applications," *IBM Journal of Research and Development*, vol. 39, pp. 229–244, Jan 1995.

[19] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pp. 10–14, Feb 2014.

[20] C. Hu, "Gate oxide scaling limits and projection," in *International*

Electron Devices Meeting. Technical Digest, pp. 319–322, Dec 1996.

[21] M. Koh, W. Mizubayashi, K. Iwamoto, H. Murakami, T. Ono, M. Tsuno, T. Mihara, K. Shibahara, S. Miyazaki, and M. Hirose, "Limit of gate oxide thickness scaling in mosfets due to apparent threshold voltage fluctuation induced by tunnel leakage current," *IEEE Transactions on Electron Devices*, vol. 48, pp. 259–264, Feb 2001.

[22] R. H. Dennard, J. Cai, and A. Kumar, "A perspective on today's scaling challenges and possible future directions," *Solid-State Electronics*, vol. 51, no. 4, pp. 518 – 525, 2007.

[23] H. Esmaeilzadeh, E. Blem, R. St. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *International Symposium on Computer Architecture (ISCA)*, pp. 365–376, 2011.

[24] A. Agrawal, J. Choi, K. Gopalakrishnan, S. Gupta, R. Nair, J. Oh, D. A. Prener, S. Shukla, V. Srinivasan, and Z. Sura, "Approximate computing: Challenges and opportunities," in *2016 IEEE International Conference on Rebooting Computing (ICRC)*, pp. 1–8, Oct 2016.

[25] H. Esmaeilzadeh, E. Blem, R. S. Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," *IEEE Micro*, vol. 32, pp. 122–134, May 2012.

[26] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *2013 18th IEEE European Test Symposium (ETS)*, pp. 1–6, May 2013.

[27] A. K. Mishra, R. Barik, and S. Paul, "iact: A software-hardware framework for understanding the scope of approximate computing," in *Workshop on Approximate Computing Across the System Stack (WACAS)*, 2014.

[28] R. Nair, "Big data needs approximate computing: Technical perspective," *Commun. ACM*, vol. 58, pp. 104–104, Dec. 2014.

[29] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Architecture support for disciplined approximate programming," in *Int. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pp. 301–312, Mar. 2012.

[30] J. Bornholt, T. Mytkowicz, and K. S. McKinley, "Uncertain<t>: Abstractions for uncertain hardware and software," *IEEE Micro*, vol. 35, pp. 132–143, May 2015.

[31] D. L. Quoc, *Approximate Data Analytics Systems*. PhD thesis, Technische Universitat Dresden, 2017.

[32] V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Analysis and characterization of inherent application resilience for approximate computing," in *Design Automation Conference*, pp. 1–9, May 2013.

[33] S. Venkataramani, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Approximate computing and the quest for computing efficiency," in *Proceedings of the 52Nd Annual Design Automation Conference*, DAC '15, (New York, NY, USA), pp. 120:1–120:6, ACM, 2015.

[34] S. Misailovic, S. Sidiroglou, H. Hoffmann, and M. Rinard, "Quality of service profiling," in *Proceedings of the 32Nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ICSE '10, (New York, NY, USA), pp. 25–34, ACM, 2010.

[35] J. Park, E. Amaro, D. Mahajan, B. Thwaites, and H. Esmaeilzadeh, "Axgames: Towards crowdsourcing quality target determination in approximate computing," in *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '16, (New York, NY, USA), pp. 623–636, ACM, 2016.

[36] A. Sampson, *Approximate Acceleration for a Post Multicore Era*. PhD thesis, University of Washington, 2015.

[37] S. Mitra, M. K. Gupta, S. Misailovic, and S. Bagchi, "Phase-aware optimization in approximate computing," in *Proceedings of the 2017 International Symposium on Code Generation and Optimization*, CGO '17, (Piscataway, NJ, USA), pp. 185–196, IEEE Press, 2017.

[38] S. T. Chakradhar and A. Raghunathan, "Best-effort computing: Rethinking parallel software and hardware," in *Design Automation Conference (DAC)*, pp. 865–870, June 2010.

[39] W. J. Poppelbaum, C. Afuso, and J. W. Esch, "Stochastic computing elements and systems," in *Proceedings of the November 14-16, 1967, Fall Joint Computer Conference*, AFIPS '67 (Fall), (New York, NY, USA), pp. 635–644, ACM, 1967.

[40] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," *Automata Studies*, pp. 43–98, 1956.

[41] G. Zervakis, S. Xydis, K. Tsoumanis, D. Soudris, and K. Pekmestzi, "Hybrid approximate multiplier architectures for improved power-accuracy trade-offs," in *International Symposium on Low Power Electronics and Design (ISLPED)*, pp. 79–84, July 2015.

[42] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman, "Enerj: Approximate data types for safe and general low-power computation," in *Proceedings of the 32Nd ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '11, pp. 164–174, 2011.

[43] M. Carbin, S. Misailovic, and M. C. Rinard, "Verifying quantitative reliability for programs that execute on unreliable hardware," in *Proceedings of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages & Applications*, OOPSLA '13, (New York, NY, USA), pp. 33–52, ACM, 2013.

[44] S. Misailovic, M. Carbin, S. Achour, Z. Qi, and M. C. Rinard, "Chisel: Reliability- and accuracy-aware optimization of approximate computational kernels," in *Proceedings of the 2014 ACM International Conference on Object Oriented Programming Systems Languages Applications*, pp. 309–328, 2014.

[45] C. Rubio-González, C. Nguyen, H. D. Nguyen, J. Demmel, W. Kahan, K. Sen, D. H. Bailey, C. Iancu, and D. Hough, "Precimonious: Tuning assistant for floating-point precision," in *2013 SC - International Conference for High Performance Computing, Networking, Storage and Analysis (SC)*, pp. 1–12, Nov 2013.

[46] E. Schkufza, R. Sharma, and A. Aiken, "Stochastic optimization of floating-point programs with tunable precision," in *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '14, (New York, NY, USA), pp. 53–64, ACM, 2014.

[47] A. Sampson, P. Panchekha, T. Mytkowicz, K. S. McKinley, D. Grossman, and L. Ceze, "Expressing and verifying probabilistic assertions," in *Proceedings of the 35th ACM SIGPLAN Conference on*

Programming Language Design and Implementation, PLDI '14, (New York, NY, USA), pp. 112–122, ACM, 2014.

[48] H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Neural acceleration for general-purpose approximate programs," in *Proceedings of the 2012 45th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO-45, pp. 449–460, 2012.

[49] D. R. Krishnan, D. L. Quoc, P. Bhatotia, C. Fetzer, and R. Rodrigues, "Incapprox: A data analytics system for incremental approximate computing," in *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, (Republic and Canton of Geneva, Switzerland), pp. 1133–1144, International World Wide Web Conferences Steering Committee, 2016.

[50] I. Goiri, R. Bianchini, S. Nagarakatte, and T. D. Nguyen, "Approxhadoop: Bringing approximations to mapreduce frameworks," in *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '15, (New York, NY, USA), pp. 383–397, ACM, 2015.

[51] W. Baek and T. M. Chilimbi, "Green: A framework for supporting energy-conscious programming using controlled approximation," in *Proceedings of the 31st ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '10, (New York, NY, USA), pp. 198–209, ACM, 2010.

[52] M. Samadi, J. Lee, D. A. Jamshidi, A. Hormati, and S. Mahlke, "Sage: Self-tuning approximation for graphics engines," in *2013 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 13–24, Dec 2013.

[53] M. Samadi, D. A. Jamshidi, J. Lee, and S. Mahlke, "Paraprox: Pattern-based approximation for data parallel applications," in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '14, (New York, NY, USA), pp. 35–50, ACM, 2014.

[54] M. Ringenburg, A. Sampson, I. Ackerman, L. Ceze, and D. Grossman, "Monitoring and debugging the quality of results in approximate programs," in *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and*

Operating Systems, ASPLOS '15, (New York, NY, USA), pp. 399–411, ACM, 2015.

[55] X. Sui, A. Lenharth, D. S. Fussell, and K. Pingali, "Proactive control of approximate programs," in *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '16, (New York, NY, USA), pp. 607–621, ACM, 2016.

[56] E. King and E. Swartzlander, "Data dependent truncated scheme for parallel multiplication," in *Proc. of the Thirty First Asilomar Conference on Signals, Circuits and Systems*, pp. 1178–1182, 1998.

[57] Y. Liu, T. Zhang, and K. K. Parhi, "Computation error analysis in digital signal processing systems with overscaled supply voltage," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, pp. 517–526, Apr. 2010.

[58] A. Lingamneni, C. Enz, K. Palem, and C. Piguet, "Synthesizing parsimonious inexact circuits through probabilistic design techniques," *ACM Trans. Embed. Comput. Syst.*, vol. 12, pp. 93:1–93:26, May 2013.

[59] A. K. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in *Design, Automation and Test in Europe*, pp. 1250–1255, Mar. 2008.

[60] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong, "Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, pp. 1225–1229, Aug. 2010.

[61] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, "Impact: Imprecise adders for low-power approximate computing," in *IEEE/ACM International Symposium on Low Power Electronics and Design*, pp. 409–414, Aug. 2011.

[62] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in *24th Int. Conf. on VLSI Design*, pp. 346–351, Jan. 2011.

[63] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication," *IEEE Transactions on Computers*, vol. 64, pp. 984–994, Apr. 2015.

[64] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in *Design, Automation and Test in Europe*, pp. 1–4, Mar. 2014.

[65] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 23, pp. 1180–1184, June 2015.

[66] S. Hashemi, R. I. Bahar, and S. Reda, "Drum: A dynamic range unbiased multiplier for approximate applications," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, 2015*, pp. 418–425, November 2015.

[67] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "Macaco: Modeling and analysis of circuits for approximate computing," in *Int. Conf. on Computer-Aided Design*, pp. 667–673, Nov. 2011.

[68] S. Banescu, F. de Dinechin, B. Pasca, and R. Tudoran, "Multipliers for floating-point double precision and beyond on fpgas," *SIGARCH Comput. Archit. News*, vol. 38, pp. 73–79, Jan. 2011.

[69] M. Schulte and E. Swartzlander, "Truncated multiplication with correction constant," in *VLSI Signal Processing VI*, pp. 388–396, Oct 1993.

[70] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient vlsi implementation of soft-computing applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, pp. 850–862, April 2010.

[71] J. Miao, K. He, A. Gerstlauer, and M. Orshansky, "Modeling and synthesis of quality-energy optimal approximate adders," in *International Conference on Computer-Aided Design*, pp. 728–735, Nov 2012.

[72] R. Zendegani, M. Kamal, M. Bahadori, A. Afzali-Kusha, and M. Pedram, "RoBa multiplier: A rounding-based approximate multiplier for high-speed yet energy-efficient digital signal processing," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 25, pp. 393–401, Feb 2017.

[73] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep

learning with limited numerical precision," in *International Conference on Machine Learning*, pp. 1737–1746, July 2015.

[74] H. Jiang, J. Han, F. Qiao, and F. Lombardi, "Approximate radix-8 booth multipliers for low-power and high-performance operation," *IEEE Transactions on Computers*, vol. 65, pp. 2638–2644, Aug 2016.

[75] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, and F. Lombardi, "Design of approximate radix-4 booth multipliers for error-tolerant computing," *IEEE Transactions on Computers*, vol. 66, pp. 1435–1441, Aug 2017.

[76] S. Venkataramani, A. Sabne, V. Kozhikkottu, K. Roy, and A. Raghunathan, "Salsa: Systematic logic synthesis of approximate circuits," in *Design Automation Conference (DAC)*, pp. 796–801, June 2012.

[77] J. Miao, A. Gerstlauer, and M. Orshansky, "Multi-level approximate logic synthesis under general error constraints," in *International Conference on Computer-Aided Design (ICCAD)*, pp. 504–510, 2014.

[78] K. Nepal, Y. Li, R. I. Bahar, and S. Reda, "Abacus: A technique for automated behavioral synthesis of approximate computing circuits," in *Design, Automation & Test in Europe (DATE)*, pp. 361:1–361:6, 2014.

[79] C. Li, W. Luo, S. S. Sapatnekar, and J. Hu, "Joint precision optimization and high level synthesis for approximate computing," in *Design Automation Conference (DAC)*, pp. 104:1–104:6, 2015.

[80] P. Duben, Parishkrati, S. Yenugula, J. Augustine, K. Palem, J. Schlachter, C. Enz, and T. N. Palmer, "Opportunities for energy efficient computing: A study of inexact general purpose processors for high-performance and big-data applications," in *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 764–769, March 2015.

[81] S. Lee, L. K. John, and A. Gerstlauer, "High-level synthesis of approximate hardware under joint precision and voltage scaling," in *Design, Automation Test in Europe*, pp. 187–192, March 2017.

[82] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, "A low latency generic accuracy configurable adder," in *2015 52nd*

ACM/EDAC/IEEE Design Automation Conference (DAC), June 2015.

[83] H. A. F. Almurib, T. N. Kumar, and F. Lombardi, "Inexact designs for approximate low power addition by cell replacement," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016.

[84] B. S. Prabakaran, S. Rehman, M. A. Hanif, S. Ullah, G. Mazaheri, A. Kumar, and M. Shafique, "Demas: An efficient design methodology for building approximate adders for fpga-based systems," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 917–920, March 2018.

[85] F. J. Kurdahi, A. Eltawil, K. Yi, S. Cheng, and A. Khajeh, "Low-power multimedia system design by aggressive voltage scaling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, pp. 852–856, May 2010.

[86] M. Shafique, R. Hafiz, S. Rehman, W. El-Harouni, and J. Henkel, "Invited: Cross-layer approximate computing: From logic to architectures," in *Design Automation Conference (DAC)*, pp. 1–6, June 2016.

[87] S. Rehman, W. El-Harouni, M. Shafique, A. Kumar, J. Henkel, and J. Henkel, "Architectural-space exploration of approximate multipliers," in *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2016.

[88] G. Zervakis, F. Ntouskas, S. Xydis, D. Soudris, and K. Pekmestzi, "Vossim: A framework for enabling fast voltage overscaling simulation for approximate computing circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. PP, no. 99, pp. 1–5, 2018.

[89] A. Lingamneni, C. Enz, K. Palem, and C. Piguet, "Highly energy-efficient and quality-tunable inexact fft accelerators," in *Custom Integrated Circuits Conference (CICC)*, pp. 1–4, Sept 2014.

[90] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Transactions on Computers*, vol. 32, pp. 124–137, Jan 2013.

[91] A. Cavelan, Y. Robert, H. Sun, and F. Vivien, "Voltage overscaling algorithms for energy-efficient workflow computations with timing

errors," in *Fault Tolerance for HPC at eXtreme Scale (FTXS)*, pp. 27–34, 2015.

[92] D. Jeon, M. Seok, Z. Zhang, D. Blaauw, and D. Sylvester, "Design methodology for voltage-overscaled ultra-low-power systems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 59, pp. 952–956, Dec 2012.

[93] T. Okumura and M. Hashimoto, "Setup time, hold time and clock-to-q delay computation under dynamic supply noise," in *IEEE Custom Integrated Circuits Conference 2010*, pp. 1–4, Sept 2010.

[94] G. Mekhtarian, *Composite Current Source (CCS) Modeling Technology Backgrounder*.
Synopsys, Inc., Nov. 2005.

[95] H. Abrishami, S. Hatami, and M. Pedram, "Multi-corner, energy-delay optimized, nbti-aware flip-flop design," in *2010 11th International Symposium on Quality Electronic Design (ISQED)*, pp. 652–659, March 2010.

[96] Synopsys, *HSPICE Applications Manual*.

[97] *CCS Timing Technical White Paper*.
Synopsys, Inc., June 2006.

[98] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Transactions on Computers*, vol. 62, pp. 1760–1771, June 2012.

[99] G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, "Design-efficient approximate multiplication circuits through partial product perforation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, pp. 3105–3117, Oct 2016.

[100] V. Leon, G. Zervakis, D. Soudris, and K. Pekmestzi, "Approximate hybrid high radix encoding for energy-efficient inexact multipliers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, pp. 421–430, March 2018.

[101] G. Zervakis, K. Tsoumanis, S. Xydis, N. Axelos, and K. Pekmestzi, "Approximate multiplier architectures through partial product perforation: Power-area tradeoffs analysis," in *Proc. of the 25th Great Lakes Symposium on VLSI*, pp. 229–232, 2015.

[102] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard, "Managing performance vs. accuracy trade-offs with loop per-

foration," in *Foundations of software engineering (ESEC/FSE)*, pp. 124–134, Sept. 2011.

[103] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*.
NY: Oxford University Press, 2000.

[104] C. Lee, M. Potkonjak, and W. H. Mangione-Smith, "Mediabench: a tool for evaluating and synthesizing multimedia and communications systems," in *Microarchitecture, 1997. Proceedings., Thirtieth Annual IEEE/ACM International Symposium on*, pp. 330–335, Dec 1997.

[105] D. Zuras and W. McAllister, "Balanced delay trees and combinatorial division in vlsi," *Solid-State Circuits, IEEE Journal of*, vol. 21, pp. 814–819, Oct 1986.

[106] L. Dadda, "Some schemes for parallel multipliers," *Alta Frequenza*, vol. 34, pp. 349–356, Mar 1965.

[107] B. Jose and D. Radhakrishnan, "Delay optimized redundant binary adders," in *Electronics, Circuits and Systems, 2006. ICECS '06. 13th IEEE International Conference on*, pp. 514–517, Dec 2006.

[108] N. Weste and D. M. Harris, *Datapath Subsystems, CMOS VLSI Design: A Circuits and Systems Perspective*.
Addison-Wesley, 2010.

[109] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-8, pp. 679–698, Nov 1986.

[110] C. Ranger, R. Raghuraman, A. Penmetsa, G. Bradski, and C. Kozyrakis, "Evaluating mapreduce for multi-core and multiprocessor systems," in *High Performance Computer Architecture, 2007. HPCA 2007. IEEE 13th International Symposium on*, pp. 13–24, Feb 2007.

[111] B. Shao and P. Li, "A model for array-based approximate arithmetic computing with application to multiplier and squarer design," in *ISLPED*, pp. 9–14, 2014.

[112] K. Bretthauer and B. Shetty, "The nonlinear knapsack problem - algorithms and applications," *European Journal of Operational Research*, vol. 138, pp. 459–472, May 2002.

[113] C.-H. Chang, J. Gu, and M. Zhang, "Ultra low-voltage low-power

cmos 4-2 and 5-2 compressors for fast arithmetic circuits," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 51, pp. 1985–1997, Oct 2004.

[114] G. Zervakis, S. Xydis, V. Tsoutsouras, D. Soudris, and K. Pekmestzi, "Multi-level approximation for inexact accelerator synthesis under voltage island constraints," in *Workshop on Approximate Computing AC16, part of ESWEEK 2016*, 2016.

[115] G. Zervakis, S. Xydis, D. Soudris, and K. Pekmestzi, "Multi-level approximate accelerator synthesis under voltage island constraints," *IEEE Transactions on Circuits and Systems II: Express Briefs*, pp. 1–1, 2018.

[116] A. Yazdanbakhsh, D. Mahajan, P. Lotfi-Kamran, and H. Esmaeilzadeh, "Axbench : A benchmark suite for approximate computing across the system stack," technical report, Georgia Institute of Technology, 2016.

[117] O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, "Rap-cla: A reconfigurable approximate carry look-ahead adder," *IEEE Trans. Circuits Syst. II*, 2017.

[118] Z. Zhang and Y. He, "A low-error energy-efficient fixed-width booth multiplier with sign-digit-based conditional probability estimation," *IEEE Trans. Circuits Syst. II*, vol. 65, pp. 236–240, Feb 2018.

[119] G. A. Constantinides, P. Y. K. Cheung, and W. Luk, *Synthesis And Optimization Of DSP Algorithms*.
Norwell, MA, USA: Kluwer Academic Publishers, 2004.

[120] "Legup." http://legup.eecg.utoronto.ca.