UNIVERSITY OF THESSALY

DOCTORAL THESIS

Novel techniques for timing analysis of VLSI circuits in advanced technology nodes

Author:
Dimitrios GARYFALLOU

Supervisors:
Prof. Nestor EVMORFOPOULOS
Prof. Georgios STAMOULIS
Prof. Fotios PLESSAS

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Department of Electrical and Computer Engineering
Electronics Lab

October 19, 2021

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Declaration of Authorship

I, Dimitrios GARYFALLOU, declare that this thesis titled, "Novel techniques for timing analysis of VLSI circuits in advanced technology nodes" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: October 19, 2021

Copyright © 2021
Dimitrios GARYFALLOU
All rights reserved.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

To my family with eternal gratitude

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

UNIVERSITY OF THESSALY

Abstract

Department of Electrical and Computer Engineering

Doctor of Philosophy

Novel techniques for timing analysis of VLSI circuits in advanced technology nodes

by Dimitrios GARYFALLOU

Timing analysis is an essential and demanding verification method used during the initial design and iterative optimization of a Very Large Scale Integrated (VLSI) circuit, while it also constitutes the cornerstone of the final signoff that dictates whether the chip can be released to the semiconductor foundry for fabrication. Throughout the last few decades, the relentless demand for high-performance and energy-efficient circuits has been met by aggressive technology scaling, which enabled the integration of a vast number of devices into the same die but brought new problems and challenges to the surface. In nanometer technology nodes, on-chip VLSI interconnects are more resistive and have an ever-increasing impact on gate and interconnect delay, while nonlinear transistor and Miller capacitances imply that signals no longer resemble smooth and saturated ramps. At the same time, manufacturing process variations have become significantly more pronounced, which in turn calls for sophisticated timing analysis techniques to reduce the uncertainty in timing estimation. From another perspective, the timing guardbands enforced by the traditional design paradigm to protect circuits from variation-induced timing errors are overly pessimistic since they are estimated using Static Timing Analysis (STA) under rare worst-case timing conditions, ignoring the workload variability and leaving extensive dynamic timing margins unexploited. To this end, this dissertation presents novel techniques for accurate and efficient timing analysis of VLSI circuits in advanced technology nodes, which address different aspects of the problem, starting from gate and interconnect delay calculation and moving to timing analysis under process variation and Dynamic Timing Analysis (DTA).

In the first part of this thesis, we focus on gate and interconnect delay calculation, which is the heart of any timing analysis technique. On the gate side, we present an iterative algorithm that accurately approximates the nonlinear signal waveforms by piecewise linear ramps, using multiple effective capacitance values to take the resistive shielding effect into account. Contrary to prior works, our approach is compatible with industrial Current Source Models (CSMs), considers the Miller effect, and is computationally efficient since it relies on closed-form formulas and convergences in very few iterations. We demonstrate that our method achieves greater accuracy than related schemes that assume a single effective capacitance value or ignore the impact of Miller capacitance. On the interconnect side, we propose a sparsity-aware Model Order Reduction (MOR) technique for efficient signoff timing analysis of large interconnects with many ports. As opposed to well-established MOR techniques, our method produces sparse reduced-order models by applying key congruence transformations on the original interconnect model and then exploiting the correspondence between Laplacian matrices and circuit graphs. Moreover, the generated models can be straightforwardly realized into equivalent compact *RC* networks and utilized in several other analysis steps of the design flow. We show that a high sparsity ratio of the reduced system matrices can be achieved without significant accuracy loss, leading to enhanced simulation runtimes compared to a well-known MOR technique that produces dense matrices.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

In the second part, we introduce a novel statistical methodology based on Monte Carlo (MC) simulation and Extreme Value Theory (EVT) for timing analysis of VLSI circuits under process variations in gate and interconnect parameters. In contrast to corner-based or traditional statistical approaches, our method provides fast yet accurate results regardless of the underlying timing models and any assumption about the propagated distributions, thus being very suitable for both transistor-level and gate-level timing analysis. Experimental results indicate that our method requires only a few thousand MC trials to yield highly accurate worst-case delay estimates, providing a speedup of six orders of magnitude over exhaustive MC simulation.

Finally, the concept of gate-level event-driven simulation is leveraged to develop an accurate DTA framework that identifies the dynamic timing slacks existing during the operation of a VLSI circuit according to the processed data. Contrary to conventional graph-based DTA that inherently relies on worst-case assumptions, the proposed event-driven DTA considers the actual data-dependent timing properties of the activated paths. Thus, it reveals significantly more dynamic timing slack, especially for the most critical paths, enabling the opportunity for substantial dynamic frequency or voltage scaling and considerably more accurate estimation of timing failures.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Περίληψη

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Διδακτορικό Δίπλωμα

Καινοτόμες τεχνικές ανάλυσης χρονισμού κυκλωμάτων πολύ μεγάλης κλίμακας ολοκλήρωσης σε προηγμένες τεχνολογίες

από τον Δημήτριο ΓΑΡΥΦΑΛΛΟΥ

Η ανάλυση χρονισμού ανέκαθεν αποτελούσε το σημαντικότερο βήμα της διαδικασίας επαλήθευσης της λειτουργίας κυκλωμάτων πολύ μεγάλης κλίμακας ολοκλήρωσης (Very Large Scale Integration - VLSI). Πρόκειται για μια κρίσιμη και απαιτητική ανάλυση, η οποία χρησιμοποιείται τόσο κατά την αρχική σχεδίαση και την επανειλημμένη βελτιστοποίηση του κυκλώματος, όσο και στην τελική επαλήθευση που είναι καθοριστικής σημασίας για την ορθή κατασκευή και λειτουργία του ολοκληρωμένου κυκλώματος. Κατά τις τελευταίες δεκαετίες, η αμείλικτη ζήτηση για γρηγορότερα και χαμηλότερης ισχύος κυκλώματα VLSI ικανοποιείται με τη συνεχή κλιμάκωση της τεχνολογίας, η οποία έχει οδηγήσει σε ολοένα και πιο περίπλοκες σχεδιάσεις, φέρνοντας στην επιφάνεια νέα προβλήματα και προκλήσεις. Στις προηγμένες τεχνολογίες ολοκλήρωσης των μερικών νανομέτρων, οι αγωγοί διασύνδεσης έχουν ολοένα και αυξανόμενη επίδραση στην καθυστέρηση του κυκλώματος, καθώς εισάγουν μεγαλύτερη παρασιτική αντίσταση, ενώ παράλληλα τα λογικά σήματα αδυνατούν πλέον να προσεγγιστούν με ακρίβεια από απλές γραμμικές κυματομορφές λόγω των μη γραμμικών παρασιτικών χωρητικοτήτων των τρανζίστορ, συμπεριλαμβανομένων των χωρητικοτήτων Miller. Επιπρόσθετα, οι διακυμάνσεις των σχεδιαστικών παραμέτρων γίνονται ολοένα και πιο έντονες, δημιουργώντας την ανάγκη για εξελιγμένες στατιστικές τεχνικές ώστε να μειωθεί η αβεβαιότητα κατά την ανάλυση χρονισμού. Προκειμένου να προστατέψουν τα κυκλώματα από σφάλματα που οφείλονται στις συγκεκριμένες διακυμάνσεις, οι σχεδιαστές εισάγουν επιπλέον περιθώρια καθυστέρησης, τα οποία είναι άκρως πεσιμιστικά διότι παραδοσιακά υπολογίζονται μέσω στατικής ανάλυσης χρονισμού (Static Timing Analysis - STA) κάτω από παραδοχές χειρότερης περίπτωσης, αγνοώντας τις διαφοροποιήσεις των εισόδων, αφήνοντας έτσι ανεκμετάλλευτα εκτενή δυναμικά περιθώρια χρονισμού. Βάσει των παραπάνω, η παρούσα διδακτορική διατριβή προτείνει νέες τεχνικές για ακριβή και αποδοτική ανάλυση χρονισμού κυκλωμάτων VLSI, οι οποίες αντιμετωπίζουν διαφορετικές πτυχές του προβλήματος, από τον υπολογισμό της καθυστέρησης πυλών και διασυνδέσεων έως και την ανάλυση χρονισμού κάτω από διακυμάνσεις των σχεδιαστικών παραμέτρων και τη δυναμική ανάλυση χρονισμού (Dynamic Timing Analysis - DTA).

Το πρώτο μέρος της διατριβής επικεντρώνεται στον υπολογισμό της καθυστέρησης πυλών και διασυνδέσεων, ο οποίος αποτελεί τον πυρήνα οποιασδήποτε τεχνικής ανάλυσης χρονισμού. Σχετικά με την ανάλυση καθυστέρησης των πυλών, παρουσιάζεται ένας ακριβής επαναληπτικός αλγόριθμος, ο οποίος προσεγγίζει τα μη γραμμικά σήματα με τμηματικά γραμμικές κυματομορφές, υπολογίζοντας την ισοδύναμη χωρητικότητα των διασυνδέσεων σε πολλαπλές περιοχές, προκειμένου να λάβει υπόψη τη δυναμική της συμπεριφορά. Αντίθετα με προγενέστερες προσεγγίσεις, ο προτεινόμενος αλγόριθμος βασίζεται σε πρόσφατα βιομηχανικά μοντέλα πηγής ρεύματος (Current Source Models – CSMs), συνυπολογίζει το φαινόμενο Miller, ενώ παράλληλα είναι εξαιρετικά αποδοτικός μιας και αξιοποιεί απλές μαθηματικές εκφράσεις κλειστού τύπου για τους υπολογισμούς και επιτυγχάνει σύγκλιση εντός ελάχιστων επαναλήψεων. Η πειραματική αξιολόγηση του αλγορίθμου δείχνει πως

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

πετυχαίνει καλύτερη ακρίβεια σε σύγκριση με μεθόδους που είτε θεωρούν μοναδική ισοδύναμη χωρητικότητα είτε αγνοούν το φαινόμενο Miller. Όσον αφορά τους VLSI αγωγούς διασύνδεσης, προτείνουμε μια τεχνική μείωσης τάξης μοντέλου (Model Order Reduction - MOR) για ακριβή και γρήγορη ανάλυση χρονισμού μεγάλων παρασιτικών RC μοντέλων με πολλές θύρες εισόδου/εξόδου. Αντίθετα με καθιερωμένες τεχνικές MOR που οδηγούν σε πυκνούς πίνακες μειωμένης τάξης, η προτεινόμενη τεχνική προσεγγίζει τους πυκνούς πίνακες με τους κοντινότερους πίνακες που έχουν αντιστοιχία με γράφους και στη συνέχεια εφαρμόζει τεχνικές αραιοποίησης γράφων για να παράγει αραιά μειωμένα μοντέλα. Τα πλεονεκτήματα της μεθόδου είναι πως τα αραιά μοντέλα οδηγούν σε επιτάχυνση της προσομοίωσης με μικρή απώλεια στην ακρίβεια εκτίμησης της καθυστέρησης, ενώ επίσης μπορούν να μετατραπούν σε ισοδύναμα RC δίκτυα μεγέθους πολύ μικρότερου του αρχικού και να επαναχρησιμοποιηθούν κατά τη σχεδίαση.

Στη συνέχεια, εισάγουμε μια νέα στατιστική μεθοδολογία βασισμένη στην προσομοίωση Monte Carlo και στη θεωρία ακραίων τιμών, για την ανάλυση χρονισμού κυκλωμάτων VLSI υπό διακύμανση των φυσικών παραμέτρων των πυλών και των διασυνδέσεων. Συγκριτικά με τεχνικές που επικεντρώνονται στις ακραίες περιπτώσεις διακύμανσης και με παραδοσιακές στατιστικές τεχνικές, η μεθοδολογία μας δεν βασίζεται σε απλουστευμένες παραδοχές για τον τύπο της κατανομής καθυστέρησης σε κάθε κόμβο του κυκλώματος και είναι ανεξάρτητη των υποκείμενων μοντέλων καθυστέρησης, με αποτέλεσμα να είναι κατάλληλη για ανάλυση τόσο σε επίπεδο τρανζίστορ όσο και σε επίπεδο πυλών. Τα πειραματικά αποτελέσματα υποδεικνύουν ότι η συγκεκριμένη μέθοδος απαιτεί μόλις μερικές χιλιάδες δοκιμές Monte Carlo ώστε να παρέχει γρήγορη και ακριβή εκτίμηση της χειρότερης καθυστέρησης, επιτυγχάνοντας έως και έξι τάξεις μεγέθους επιτάχυνση συγκριτικά με μια πλήρη προσομοίωση Monte Carlo.

Τέλος, αναπτύσσεται ένα εργαλείο DTA βασισμένο σε προσομοίωση επιπέδου πύλης οδηγούμενη από γεγονότα (event-driven gate-level simulation), το οποίο υπολογίζει με ακρίβεια τα δυναμικά περιθώρια χρονισμού που υπάρχουν κατά τη λειτουργία του κυκλώματος σύμφωνα με τα επεξεργαζόμενα δεδομένα. Σε αντίθεση με συμβατικές graph-based μεθόδους, οι οποίες θεωρούν καθυστερήσεις χειρότερης περίπτωσης σε κάθε στοιχείο του κυκλώματος, η προτεινόμενη event-driven DTA προσέγγιση λαμβάνει υπόψη τα πραγματικά χαρακτηριστικά χρονισμού των ενεργοποιημένων μονοπατιών. Έτσι, αναδεικνύει σημαντικά περισσότερα δυναμικά περιθώρια χρονισμού, ειδικά για τα κρισιμότερα μονοπάτια, προσφέροντας τη δυνατότητα για αξιοσημείωτη δυναμική μεταβολή της συχνότητας λειτουργίας και της τάσης τροφοδοσίας του κυκλώματος, παρέχοντας παράλληλα ακριβέστερη εκτίμηση των σφαλμάτων χρονισμού.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Acknowledgements

As this long endeavor has reached its end, I would like to express my sincere gratitude to a group of people who provided me with guidance, assistance, support, and encouragement. It is certain that without their help, I would not have been able to complete my Ph.D. studies.

First and foremost, I am deeply grateful to my advisors, Professor Nestor Evmorfopoulos and Professor Georgios Stamoulis, who not only gave me the opportunity to start this journey but also believed in my work and trusted me even during tough times. They have always been there to inspire me with new ideas and guide my research career. Being very different characters but at the same time outstanding personalities, they have managed to create a powerful team bringing extraordinary results. Their diligence, intelligence, and wisdom have had a great influence on my research development. I would also like to thank Professor Fotios Plessas, Professor Georgios Karakonstantis, and Professor Christos Sotiriou from the University of Thessaly, Professor Dimitrios Karampatzakis from the International Hellenic University, and Professor Labros Bisdounis from the University of Peloponnese, for providing insightful comments on my thesis.

My sincere appreciation to all my fellow labmates in the Electronics Lab of the University of Thessaly, for the fruitful discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had these years. Especially, I would like to thank Dr. Charalampos Antoniadis, Dr. George Floros, and Panagiotis Giannakou, with whom I have been collaborating since the beginning of my Ph.D. Moreover, I would like to extend my sincere thanks to Professor Georgios Karakonstantis and Dr. Ioannis Tsiokanos for our excellent collaboration and the great time we had during my stay at Queen's University Belfast. Immense thanks to Dr. Nikolaos Sketopoulos and (soon to be Dr.) Stavros Simoglou for contributing to quality publications. Also, I would like to wholeheartedly thank my closest friends, Achilleas, Christos, and George, for providing happy distractions to rest my mind outside of my research. Special gratitude is due to Faviola for her constant support and patience throughout my Ph.D. studies.

Last but not least, my deepest gratitude goes to my parents, Konstantinos and Styliani, for their unconditional support, love, and sacrifices for making this dissertation possible. Even if my research seems "Greek" to them, they were always there for me. The least I can do in recognition is to dedicate this dissertation to them.

This research was partially supported by the Center of Research, Innovation, and Excellence of the University of Thessaly, Greece and the European Union's Horizon 2020 Programme under grant agreement numbers 688540 (UniServer) and 687698 (HiPEAC).

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Contents

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

List of Figures

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

List of Tables

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

List of Algorithms

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

List of Abbreviations

Abbreviation	Meaning
AE	Absolute Error
AT, *at*	Arrival Time
C_{eff}	Effective Capacitance
C_{total}	Total Capacitance
CCS	Composite Current Source
cdf	cumulative density function
CPU	Central Processing Unit
CR	Clock Reduction
CSM	Current Source Model
DTA	Dynamic Timing Analysis
DTS	Dynamic Timing Slack
ECSM	Effective Current Source Model
EDA	Electronic Design Automation
ED-DTA	Event-Driven Dynamic Timing Analysis
EVT	Extreme Value Theory
FF	Flip-Flop
FinFET	Fin Field-Effect Transistor
GBA	Graph-Based Analysis
GB-DTA	Graph-Based Dynamic Timing Analysis
GB-STA	Graph-Based Static Timing Analysis
GEV	Generalized Extreme Value
GP	Generalized Pareto
GPU	Graphics Processing Unit
i.i.d.	independent, identically distributed
IC	Integrated Circuit
IRDS	International Roadmap for Devices and Systems
LEF	Library Exchange Format
LUT	Look-Up Table
MC	Monte Carlo
ML	Maximum Likelihood
MNA	Modified Nodal Analysis
MOR	Model Order Reduction
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
MRE	Mean Relative Error
MVTM	Multiple Voltage Threshold Models
NLDM	Non Linear Delay Model
pdf	probability density function
PDK	Process Design Kit
PI	Primary Input
PnR	Place and Route
PO	Primary Output
PoFF	Point of First Failure

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

PVTA	Process, Voltage, Temperature, and Aging
PWL	Piecewise Linear
rat	required arrival time
RE	Relative Error
RMSPE	Root Mean Square Percentage Error
ROM	Reduced-Order Model
RSM	Response Surface Methodology
RTL	Register-Transfer Level
SDC	Synopsys Design Constraints
SDF	Standard Delay Format
SPEF	Standard Parasitic Exchange Format
SPICE	Simulation Program with Integrated Circuit Emphasis
SSTA	Statistical Static Timing Analysis
STA	Static Timing Analysis
STS	Static Timing Slack
V_{dd}	Operating Supply Voltage
VCD	Value Change Dump
VLSI	Very Large Scale Integration
VRM	Voltage Response Model

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Chapter 1

Introduction

1.1 Motivation

Over the past six decades, Integrated Circuits (ICs) have revolutionized the world of electronics and have become pervasive in all aspects of our daily life, from low-cost personal computers and smart wearable devices to healthcare applications and autonomous driving. The sheer complexity of modern IC designs, as well as the strict performance requirements imposed by the consumers, have placed the task of timing analysis in a prominent position. The purpose of timing analysis is to analyze the circuit for timing issues and ensure that is able to operate reliably at the targeted clock frequency. Recently, it has been estimated that timing closure consumes up to 60% of the total design time [1]. In fact, timing analysis is an integral part of the design flow, which is applied after each main step to verify that all timing constraints (e.g., setup and hold) are met, while it also drives all optimization steps such as timing-driven placement and routing [2, 3]. Most importantly, it is used during the final signoff that dictates whether the chip can be released to the semiconductor foundry for fabrication. As a consequence, accurate and efficient Static Timing Analysis (STA) and Dynamic Timing Analysis (DTA) are essential for the successful design of contemporary ICs.

The relentless push for high-performance and low-power ICs has been met by aggressive technology scaling, which enabled the integration of a vast number of devices into the same die but brought new problems and challenges to the surface. In this dissertation, we focus on timing analysis of Very Large Scale Integrated (VLSI) circuits in the presence of highly resistive interconnects and process variations, while we also investigate DTA to reveal unexploited dynamic timing margins existing due to workload variability. In the following sections, we describe the motivation behind each problem tackled in this thesis.

1.1.1 Gate delay estimation

As process geometries shrink below 45 nm, gate delay estimation becomes even more challenging. Modern VLSI on-chip interconnects are becoming more resistive, while nonlinear transistor and Miller capacitances imply that signals no longer resemble smooth, saturated ramps [4]. As a result, this renders traditional Voltage Response Models (VRMs), such as the Non Linear Delay Model (NLDM), inadequate to capture the nonlinear driver waveforms, leading to significant errors in delay and slew computations. Over recent years, the semiconductor industry has adopted Current Source Models (CSMs) [5, 6] for accurate gate modeling, which capture more detail compared to VRMs. Industrial gate models, however, are precharacterized assuming capacitive loads, which poses significant challenges to the approximation of the highly resistive load interconnect with an effective capacitance (C_{eff}) due to the significant resistive shielding effect. In fact, most related works are either computationally expensive or unable to approximate the driver output slew, which is essential in timing

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

analysis since it also impacts interconnect delay and slew estimation. Furthermore, they require additional non-standard precharacterization and ignore the impact of Miller effect on C_{eff} and driver output waveform [7–15].

1.1.2 Interconnect delay estimation

On the other hand, interconnect delay has become the main performance limiter in nanometer-scale VLSI circuits, as it represents an increasingly dominant portion of the total path delay. Recent projections made by the International Roadmap for Devices and Systems (IRDS) indicate that the clock frequency at nominal supply voltage is forecasted to mildly improve from 3.1 GHz in 2020 to 3.5 GHz in 2025, while it is predicted to be decreased to 2.9 GHz in 2034 [16]. This limited scaling is due to the increased interconnect average length and routing density, which has led to increased parasitics. Although gate delay and slew may be calculated by interpolating on waveforms captured in standard cell libraries, interconnect timing analysis has remained a mystery. Traditional SPICE transient simulation [17] offers golden accuracy results but fails to meet the performance and memory requirements for full-chip analysis, while fast closed-form delay and slew metrics [18–22] may be quite inaccurate, especially for large *RC* networks with many branches, as they rely on assumptions which are invalid in recent technology nodes. In practice, Model Order Reduction (MOR) techniques are typically employed to reduce the large interconnect model and provide a good compromise between accuracy and performance in timing analysis since we only need to compute the voltage response at the interconnect output ports. However, all the established MOR techniques [23–25] result in dense system matrices that render their simulation impractical.

1.1.3 Timing analysis under process variation

In an ideal world, the fabrication of ICs would be a perfectly predictable process and all chips would meet their timing specifications. However, in advanced technology nodes, the manufacturing process is getting more complex due to shrinking physical dimensions, which are now approximating what is considered the fundamental limit of device operation [26, 27] (e.g., the oxide in 22 nm process is only five atomic layers thick). Equipment imprecision and process limitations lead to extensive variations of the physical parameters and electrical characteristics of transistors and interconnects, which critically affect the circuit timing behavior and may result in up to 30% variation in operating frequency, causing timing failures [26]. Conventionally, chip designers apply corner-based analysis [28] to consider the impact of process variation on timing. However, this technique is too slow as the number of variability sources proliferates with process scaling and integration of more components into a chip or inaccurate due to the assumption that the worst-case delay resides at the corners of the design parameters. Another approach that has gained excessive research interest is Statistical Static Timing Analysis (SSTA) [29], which may be performed either by probability distribution propagation [30] or by Monte Carlo (MC) simulation [31]. The former method is significantly faster but leads to less accurate results than the latter one, as it relies on simplistic assumptions about the underlying delay models and propagated distributions.

1.1.4 Dynamic timing analysis

To ensure reliable operation, designers adopt timing guardbands that force circuits to operate at a lower frequency, thus providing sufficient margins to mitigate timing errors induced by Process, Voltage, Temperature, and Aging (PVTA) variations [32]. However, such timing margins are considered to be overly pessimistic since they are estimated according to the most critical paths identified through multi-corner STA [3], ignoring the dynamically activated critical paths existing due to workload variability. SSTA may have helped to trim down

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

these guardbands but still assumes worst-case operating conditions, resulting in unnecessary performance loss. It has been recently demonstrated that roughly 99% of the statically estimated critical paths are triggered by less than 10% of all possible input vectors [33], while there is a very low possibility to experience the worst-case input conditions [34]. These findings have turned the attention of many studies into the exploitation of the so-called Dynamic Timing Slack (DTS) that may exist within any path depending on the dynamically changing processed data [33, 35–39]. Such studies may have revealed extensive DTS but rely on Graph-Based DTA (GB-DTA) [38, 40, 41], which inherently makes worst-case assumptions and still ignores some data-dependent timing properties. This may cause significant DTS underestimation, leading to unexploited frequency scaling margins and incorrect timing failure estimation.

1.2 Contributions and Outline

In this thesis, we introduce several novel methodologies for accurate and efficient timing analysis of VLSI circuits in advanced process technology nodes, which effectively overcome the limitations of existing methods. In particular, we first address the problem of gate and interconnect delay estimation, which is the heart of any timing analysis methodology. Then, we deal with worst-case delay estimation under process variation, and finally, we tackle the challenge of accurate DTA. Below, we briefly discuss the contributions of our work.

Chapter 2 - Gate Delay Estimation. In Chapter 2, we present an iterative algorithm for fast and accurate gate delay estimation. The proposed methodology accurately approximates the nonlinear driver output waveform and C_{eff} in multiple waveform regions while considering their interdependence. Therefore, it allows for variable analysis resolution exploiting an accuracy/runtime trade-off, enabling applicability to both optimization steps and signoff timing analysis. In contrast to prior works [7–15], our approach is compatible with industrial CSMs and considers the impact of Miller capacitance. Experimental results on representative stages implemented in 7 nm Fin Field-Effect Transistor (FinFET) technology indicate that our method achieves 1.3% and 2.5% delay and slew error against SPICE, respectively. In addition, it provides high efficiency, as it relies on closed-form formulas and achieves convergence in 2.3 iterations on average. Moreover, we investigate the impact of the resistive shielding and the Miller effect on gate delay and slew estimation, demonstrating that the proposed method achieves greater accuracy than related schemes that assume a single C_{eff} value [22] or ignore the impact of Miller capacitance [15]. This work has been published in [42], while it also won first place in the ACM TAU 2020 timing analysis contest [43].

Chapter 3 - Interconnect Delay Estimation. In Chapter 3, we propose a sparsity-aware MOR methodology for efficient timing analysis of VLSI interconnects with many ports. As opposed to well-established MOR techniques that result in dense system matrices [23–25], our proposed approach produces sparse reduced-order models, exploiting the correspondence between Laplacian matrices and graphs. The sparsified reduced-order models have a straightforward realization to equivalent *RC* interconnect networks which may be dumped into a more compact Standard Parasitic Exchange Format (SPEF) file to be used in industrial design flows. Although the proposed method may be more suitable for the final timing signoff, it also can be incorporated in iterative optimization flows to improve the convergence rate of the optimization by providing a fast and highly accurate estimation of the critical paths. Experimental results on large interconnects of widely used circuits demonstrate that the proposed method achieves up to 30× simulation time speedups over SPICE transient simulation of the original interconnect model, maintaining a reasonable delay accuracy of 4%, while the Elmore delay may deviate up to 288%. This work has been published in [44].

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Chapter 4 - Timing Analysis Under Process Variation. In Chapter 4, we introduce a novel statistical methodology based on MC simulation and Extreme Value Theory (EVT) [45] to estimate the worst-case delay of VLSI circuits under variations in gate and interconnect parameters. In contrast to corner-based or traditional SSTA approaches toward maximum delay estimation [28–31, 46–48], our methodology can be applied regardless of the underlying timing models or any assumption about the distribution of the Arrival Time (AT) at every circuit node and provide fast yet accurate results, thus being very appealing for integration into any level of timing analysis abstraction (from transistor-level to gate-level). Experimental results on widely used circuits show that the estimated maximum AT on path endpoints can be within 5% of the actual value, at a cost of a few thousand MC trials, providing a runtime speedup of 6 orders of magnitude over an exhaustive MC simulation. Furthermore, we describe possible machine learning enhancements to the proposed method that could lead to even greater accuracy and performance. This work has been published in [49].

Chapter 5 - Dynamic Timing Analysis. In Chapter 5, we present an accurate framework that reveals available DTS underestimated by previous works. Contrary to GB-DTA which relies on worst-case assumptions [38, 40, 41], our approach exploits gate-level Event-Driven DTA (ED-DTA) to consider the actual data-dependent timing properties of the activated paths. Experimental results on various post-place-and-route designs show that ED-DTA achieves an average of 2.35% and up to 194.51% DTS improvement over a conventional GB-DTA method based on delay-annotated gate-level simulation [40]. Compared to existing frequency scaling schemes [35, 36], the proposed approach enables us to further increase the clock frequency by up to 10.42%. Finally, we also demonstrate that the use of ED-DTA can reveal that timing errors may be up to 2.94× less than the ones estimated by existing failure estimation techniques [33, 37], under potential variation-induced delay increase. This work has been published in [50].

The rest of the dissertation is organized as follows. The main research contributions, as highlighted above, are presented in Chapters 2 through 5. For clarity, and owing to the range of topics considered, all chapters are self-contained; each chapter includes necessary specific background material, a description of the proposed technique, and the corresponding experimental results.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Chapter 2

Gate Delay Estimation with Current Source Models & Effective Capacitance

2.1 Introduction

With continuous technology scaling, accurate and efficient timing analysis plays an ever-increasing role in the successful design of complex ICs. Transistor-level electrical simulators [17] may offer golden accuracy results, however, they fail to meet the performance and memory requirements for full-scale analysis of modern IC designs. Thus, timing analysis is typically abstracted at the gate-level, where circuit delay is analyzed in stages [3]. Each stage consists of a driver gate, one or multiple receiver gate(s), and an interconnect. The objective of this chapter is the fast and accurate gate delay and slew estimation, which is essential for timing analysis. Interconnect delay plays an important role in contemporary nanometer-scale technologies, but it also depends on gate slew estimation. At the same time, the accuracy and performance of gate delay and slew estimation depend not only on the driver and receiver gate models but also on the interconnect load model.

Gate models are generated by performing transistor-level simulations, per library standard cell, for a set of input signal slews and output loads. This standard cell characterization information is stored in Look-Up Tables (LUTs) of technology libraries and is used during timing analysis to compute driver gate delay and output slew, given the input slew and output load. For simplicity and speed, a lumped capacitive load is assumed for LUT characterization. Thus, this single capacitance value must be used to represent both the interconnect load, as well as the nonlinear receiver input pin capacitance. However, at 45 nm and below, interconnects are becoming increasingly resistive, while nonlinear transistor and Miller capacitances imply that signals no longer resemble smooth, saturated ramps [4]. As a consequence, conventional Voltage Response Models (VRMs), such as the Non Linear Delay Model (NLDM), are inadequate to accurately capture the nonlinear driver waveform, thus leading to significant errors in delay and slew computations. To address this key challenge, Current Source Models (CSMs) [5, 6, 51–53] have been proposed, which capture more detail compared to VRMs. Hence, the classical NLDM, used in Electronic Design Automation (EDA) for decades, has now been replaced by the Synopsys Composite Current Source (CCS) model [5] and the Cadence Effective Current Source Model (ECSM) [6].

The interconnect load model is itself an issue, as modeling the driving point admittance of highly resistive on-chip interconnects is challenging. It is worth noting that due to the high resistance of on-chip interconnects, inductive effects are not significant in timing analysis, and thus only RC interconnect load models are typically considered [3]. A reduced-order π-model [54] of the distributed RC interconnect may provide sufficient accuracy, however, it is not part of the technology library characterization process. On the other hand, modeling

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

the complex RC network using total interconnect capacitance (C_{total}) is overly pessimistic, due to the resistive shielding effect [7]. For accurate gate delay estimation, previous approaches [7–15] compute an effective capacitance (C_{eff}) to account for resistive shielding, while maintaining compatibility with precharacterized gate models. However, most of these approaches, whether iterative [7,9–12,14,15] or non-iterative [8,13], are either computationally expensive or inadequate to approximate the output slew. Moreover, they require explicit instantiation of a Thevenin equivalent gate model, as well as precharacterization of information that is not part of standard cell libraries. Few works propose library compatible methods for gate delay estimation using C_{eff} [9,15], however [9] uses NLDM and only [15] exploits CSMs. A common shortcoming of all the aforementioned methods is that they do not consider the Miller effect, thus ignoring the impact of receiver input pin capacitance on C_{eff}, delay and slew estimation.

In this chapter, we focus on improving gate delay estimation by considering the receiver Miller capacitance, as well as the behavior of C_{eff} in multiple regions, while exploiting library compatible CSMs and being very computationally efficient. The contributions of this chapter can be summarized as follows:

- We propose a methodology to estimate the driver output voltage waveform and C_{eff} in multiple waveform regions. To achieve this, we implemented an iterative algorithm that considers their interdependence, while taking into account the impact of Miller effect. The proposed approach is compatible with CSMs widely adopted by industry [5,6].
- Our approach is computationally efficient, relying on closed-form formulas, while achieving convergence in very few iterations. At the same time, accuracy is not compromised. Experimental results on stages implemented in 7 nm Fin Field-Effect Transistor (FinFET) technology show that our method results in 1.3% and 2.5% delay and slew Root Mean Square Percentage Error (RMSPE) over SPICE, respectively, while it achieves convergence in 2.3 iterations on average.
- We investigate the impact of resistive shielding and Miller effect on gate delay estimation, by comparing our method with six methods that adopt different gate and load models. Our results indicate that the proposed method achieves greater accuracy, especially for output slew, compared to single C_{eff} methods [22], while C_{total} is extremely inaccurate for highly resistive loads. For stages with low impedance interconnects and significant receiver Miller capacitance, our method further improves delay and slew estimation.

The rest of this chapter is organized as follows. We present other research approaches for gate delay estimation in Section 2.2. In Section 2.3, we describe the fundamentals of library compatible CSMs, C_{eff} computation, as well as the challenges of exploiting them. Section 2.4 presents our methodology for accurate and efficient gate delay estimation using CSMs and C_{eff}. In Section 2.5, we evaluate the accuracy and performance of our approach. Finally, conclusions are drawn in Section 2.6.

2.2 Related Work

During the past two decades, various works have focused on improving gate and load models to enable accurate driver output waveform estimation in the presence of RC interconnects.

The simplest approximation of the driving point admittance of an RC interconnect is C_{total}, which is computed by summing all interconnect capacitance values. However, this results in pessimistic gate delay estimation, as it totally ignores interconnect resistance which shields a part of total capacitance. A more accurate approximation is a reduced-order model. Authors in [54] propose a π-model, which may be computed by matching the first three

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

moments of the driving point admittance using a moment-matching technique [55]. It follows that for accurate gate delay estimation, a four-dimensional LUT indexed by input slew and the π-model parameters (C_{near}, R, C_{far}) may be used. However, this is costly in terms of storage and computational requirements. Additionally, it is incompatible with gate models precharacterized in standard cell libraries assuming lumped capacitive loads (*e.g.,* NLDM, CCS, ECSM) [5, 6]. To address these limitations, the concept of C_{eff} is introduced [7].

Early research on gate delay estimation using C_{eff} focuses on computing a single capacitance value to approximate the output waveform [7, 8, 12]. Authors in [7] use a two-piece output waveform and propose a C_{eff} calculation method for single stage gates. C_{eff} is calculated by equating the average current at the gate output, (i) when using the driving point admittance as a load, and (ii) when using the estimated C_{eff} as a load. Average output currents are equated until the output voltage reaches the 50% threshold. However, this approach involves an expensive iterative algorithm that requires 5 to 10 iterations to converge, and uses empirical equations which assume fast input transitions. Aiming at modeling complex gates, the approach in [12] introduces an empirical time-varying Thevenin equivalent gate model, independent of the input signal thresholds. Similarly to [7], they equate the average currents for a specific output voltage waveform region, from 20% to 50%, denoted as the "active region". A disadvantage of this method is that it is computationally expensive, as it uses Newton-Raphson iteration to calculate C_{eff} and the Thevenin voltage source parameters. To sidestep the performance limitations, authors in [8] propose a non-iterative method for C_{eff} estimation. Regarding the gate model, the main difference is that they use a Thevenin model composed of a fixed linear ramp and a fixed resistance, which yields a delay error up to 15%. The main drawback of the above methods [7, 8, 12] is that they are inadequate to accurately match the output waveform as they focus on gate delay estimation, thus leading to slew errors up to 50% [10].

To overcome this limitation, many works propose the use of a single or two C_{eff} values for matching the output slew directly [10, 11, 14] or matching specific output voltage thresholds (*e.g.,* 10% and 50%) separately [13]. In [10], authors compute a single C_{eff} by tuning an osculating Thevenin model, until the delay when the model is loaded by the original RC interconnect and the delay when the model is loaded by C_{eff} are approximately equal. Authors in [11] use a Thevenin model and present an iterative method to estimate a single C_{eff}. In contrast to previous approaches, this method does not equate the average currents or gate delays, but minimizes the error between the output voltage waveforms from $0.2V_{dd}$ to $0.8V_{dd}$, where V_{dd} is the operating supply voltage. The approach in [14] adopts a multi-ramp driver model and uses two C_{eff} values to model different slew rates of the nonlinear output waveform in the presence of process variations. However, this method requires complicated statistical precharacterization. In a different approach, authors in [13] use two C_{eff} values to match the lower (*e.g.,* 10%) and upper (*e.g.,* 50%) output voltage thresholds instead of matching slew directly. Although this method is non-iterative and may be reasonably fast, it induces inaccuracy in slew estimation since it is based on empirical gate modeling and assumes a fixed switching resistance for complex gates.

Note that most of the aforementioned schemes [7, 8, 10–14] apply moment-matching techniques and approximate the interconnect load admittance with poles and residues, either to reduce the load to a π-model (*e.g.,* using [54]) or to compute C_{eff} and the Thevenin model parameters. However, our proposed methodology does not include expensive techniques and can be extended to handle distributed interconnects without requiring computation of moments and reduction to a π-model.

Common to all these approaches [7, 8, 10–14] is that they require explicit instantiation of a Thevenin equivalent model and precharacterization of non-standard information such as the Thevenin model parameters and the delays to arbitrary output voltage thresholds. Besides being inefficient and unable to accurately capture the output voltage waveform, these

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

approaches are also incompatible with standard cell libraries.

A library compatible load model is presented in [9]. However, this method cannot approximate the output slew, as it uses a single C_{eff} and assumes that driver output voltage is a ramp with fixed slew estimated using NLDM. Instead, our method uses library compatible CSMs [5, 6], to accurately estimate a Piecewise Linear (PWL) driver voltage waveform, by computing a different C_{eff} value for each linear segment. The approach in [15] is the one closest to ours, in the sense that it uses Multiple Voltage Threshold Models (MVTM), a variant of CSM, and multiple C_{eff}. However, the computation of the driver output waveform involves moment-matching techniques and is not straightforward like our proposed approach.

Additionally, none of the previous works [7–15] accounts for the Miller effect, ignoring the impact of receiver input pin capacitance on C_{eff} and driver output waveform. On the contrary, our proposed method takes this effect into account, thus leading to better delay and output slew accuracy results.

2.3 Background

In this section, we present library compatible CSMs and provide an example of the CCS model. Moreover, we describe the traditional way to compute C_{eff} and the challenges arising when exploiting CSMs and C_{eff} for gate delay calculation.

2.3.1 Gate modeling and library compatible CSMs

In a gate-level design, almost every gate acts as a driver in one stage and a receiver in an other one. As a result, both driver and receiver models are essential to accurately capture the electrical behavior of a gate. More specifically, the driver model should capture the timing characteristics of a gate (*e.g.,* gate delay and output slew), while the receiver model should capture the capacitive load that is presented to the driver gate.

Traditional gate modeling includes a VRM (which defines characteristics of the driver output voltage response) as a driver model and a single capacitance value (or two values, for rise and fall) as a receiver model [56]. In nanometer technologies, however, this modeling is highly inaccurate, as signal waveforms and input capacitances are nonlinear. On the driver side, the main issue is that interconnect resistance can become several kOhms, resulting in a much higher interconnect impedance compared to the resistance of the driver gate. In such cases, VRMs, such as Thevenin models and NLDM, tend to produce an output waveform that is the same as the input waveform, dwarfing the effect of the output load. On the receiver side, input pin capacitance actually depends both on the input signal slew and the receiver output load, while it also varies considerably during a transition, due to the Miller effect. This effect describes the increase in input pin capacitance caused by the presence of input-to-output coupling capacitance (known as Miller capacitance). As gate-to-drain capacitance increases, Miller effect becomes even more pronounced [4]. Thus, since the load seen by the driver depends on the receiver input pin capacitance, it is evident that constant capacitance models are insufficient for accurate gate delay estimation.

In contrast to the traditional gate models, CSMs use a time-varying voltage-controlled current source as a driver model, and a complex voltage-controlled capacitor as a receiver model. As a consequence, they are able to approximate the nonlinear waveforms and yield SPICE-accurate results within reasonable time. Although several CSMs have been proposed in the literature [51–53], our methodology focuses on CSMs adopted by industry, such as ECSM and CCS, which are precharacterized in standard cell libraries. In this work, we refer to such models as *library compatible CSMs*. These CSMs store the driver output voltage or current waveforms into two-dimensional LUTs, indexed by input signal slew (tr^{in}) and output load (c^{out}), for each timing arc (*i.e.,* input-to-output connection). More specifically,

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

ECSM represents the voltage waveform in the form of $V(t) = F_v(tr^{in}, c^{out})$, while CCS represents the current waveform as $I(t) = F_i(tr^{in}, c^{out})$. ECSM and CCS precharacterized waveforms are equivalent, as the voltage waveform can be derived by integrating the corresponding current waveform. Further on, both models are able to account for the Miller effect by providing multiple capacitance values in similar LUTs. To better explain library compatible CSMs, we provide a brief demonstration of the CCS model and the differences compared to NLDM.

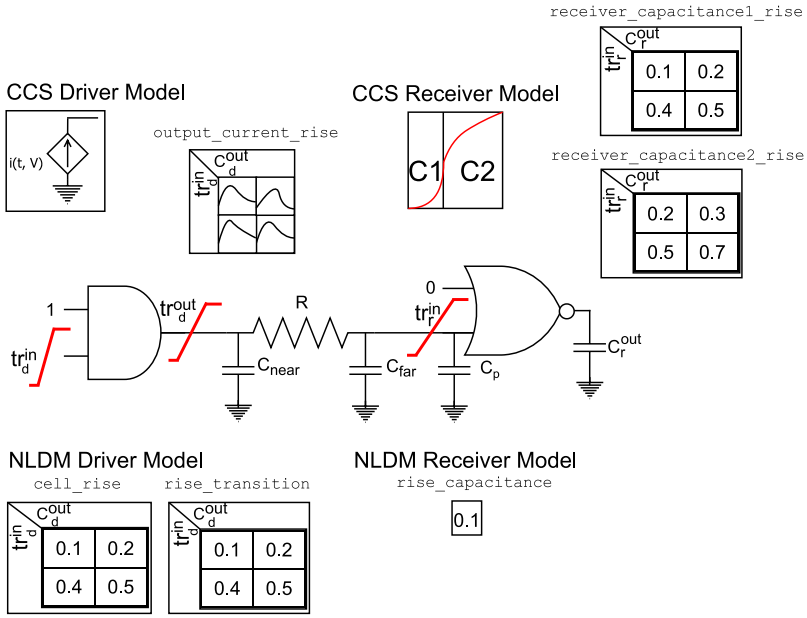


FIGURE 2.1: NLDM vs CCS timing model.

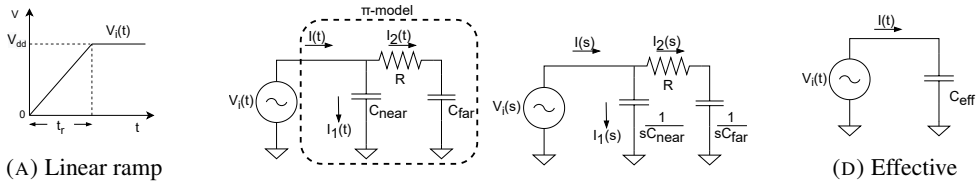
CCS driver model captures the nonlinear current waveforms in `output_current_rise/fall` LUTs, as demonstrated in Fig. 2.1, which are used during timing analysis to estimate driver delay and output slew. Also, the time instant when the corresponding driver input waveform crosses the delay threshold (usually $0.5V_{dd}$), which is necessary to calculate gate delay, is stored as `reference_time`. On the contrary, NLDM captures fixed delay and output slew values (stored in `cell_rise`, `rise_transition` LUTs), and thus provides inferior accuracy in delay estimation compared to CCS.

CCS receiver model typically uses two different input pin capacitance (C_p) values, $C1$ and $C2$, to model the nonlinear receiver input transistor capacitance and the Miller effect. $C1$ is considered up to the delay threshold of the driver output waveform, while $C2$ is considered past this point. More than two regions and corresponding capacitance values can be used to improve the accuracy. As shown in Fig. 2.1, CCS receiver capacitance values are stored in `receiver_capacitance_1/2_rise/fall` LUTs. Contrary to CCS, NLDM provides only a single capacitance value (stored in `rise_capacitance` attribute) and ignores the Miller effect.

2.3.2 Modeling the RC interconnect with a single C_{eff}

It has been shown that the input admittance of a distributed RC interconnect may be approximated by an equivalent π-model, without a significant loss of accuracy [54]. Traditionally, interconnect loads had been highly capacitive and less resistive. Hence, resistance R had negligible impact on delay calculation, and the use of C_{total} (*i.e.,* the sum of near capacitance C_{near} and far capacitance C_{far}) was sufficient to achieve accurate results. With technology

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131



(A) Linear ramp input voltage (B) Time domain (C) Frequency domain (D) Effective capacitance

FIGURE 2.2: Effective capacitance calculation for a π-model with linear ramp input voltage waveform.

scaling, however, interconnect loads are becoming more and more resistive, which complicates the approximation of the RC interconnect with a single capacitance. Considering the π-model of Fig. 2.2b, as R tends to zero, the π-model capacitors are effectively connected in parallel and can be summed together without loss of accuracy. However, when R possesses a significant value, it acts as an open-circuit, shielding C_{far}, and thus only C_{near} is "seen" by the driver. Therefore, we cannot neglect the shielding effect when substituting the π-model with an equivalent C_{eff} (shown in Fig. 2.2d).

Let us now describe how C_{eff} is calculated. The following are modifications of the approach proposed in [22]. Consider the example of Fig. 2.2b, where the π-model is driven by a voltage source $V_i(t)$ which represents the driver output waveform. Assuming that $V_i(t)$ is a linear ramp with rise transition time t_r, as shown in Fig. 2.2a, the waveform equation is:

$$V_i(t) = \begin{cases} \frac{V_{dd}}{t_r} t, & t < t_r \\ V_{dd}, & t \geq t_r \end{cases}$$

The circuit representation in the frequency domain is depicted in Fig. 2.2c. Using Kirchhoff's current law and Ohm's law, the supplied current can be expressed as a function of the input voltage and the π-model RC parameters, as follows:

$$\begin{aligned} I(s) = I_1(s) + I_2(s) &= \frac{V_i(s)}{1/(sC_{near})} + \frac{V_i(s)}{R + 1/(sC_{far})} \\ &= V_i(s)\left(sC_{near} + \frac{sC_{far}}{1 + sRC_{far}}\right) \end{aligned} \tag{2.1}$$

Additionally, if we transform $V_i(t)$ into the frequency domain, we obtain:

$$V_i(s) = \frac{V_{dd}}{s^2 t_r}\left(1 - e^{-st_r}\right) \tag{2.2}$$

Now, by substituting (2.2) in (2.1), when $t < t_r$, we get:

$$\begin{aligned} I(s) &= \frac{V_{dd}}{t_r}\left(1 - e^{-st_r}\right)\left[\frac{C_{near}}{s} + \frac{C_{far}}{s(1 + sRC_{far})}\right] \\ &= \frac{V_{dd}}{t_r}\left(1 - e^{-st_r}\right)\left[\frac{C_{near}}{s} + \frac{C_{far}(1 + sRC_{far}) - sRC_{far}^2}{s(1 + sRC_{far})}\right] \\ &= \frac{V_{dd}}{t_r}\left(1 - e^{-st_r}\right)\left[\frac{C_{near}}{s} + \frac{C_{far}}{s} - \frac{C_{far}}{s + 1/(RC_{far})}\right] \end{aligned}$$

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Finally, after transforming the current equation back to the time domain, the resulting equation is given by:

$$I(t) = \underbrace{\frac{V_{dd}}{t_r} C_{near}}_{I_1(t)} + \underbrace{\frac{V_{dd}}{t_r} C_{far}\left(1 - e^{\frac{-t}{RC_{far}}}\right)}_{I_2(t)} \tag{2.3}$$

At this point, C_{eff} can be defined as a capacitance seen by the driver voltage source $V_i(t)$, which requires the same charge transfer as that required by the π-model. Typically, C_{eff} is calculated up to a specific voltage threshold $V_\ell = \ell V_{dd}$, with factor ℓ representing a percentage of V_{dd}. For this reason, we denote this effective capacitance by C_ℓ. Note that V_ℓ corresponds to a time instant T_ℓ, where $V_i(T_\ell) = V_\ell$, which also represents the driver output slew from $0V_{dd}$ to ℓV_{dd}, assuming that the output transition begins at $t = 0$ (*i.e.,* $T_{0V_{dd}} = 0$).

The equation for the charging of the π-model, up to T_ℓ, can be derived, using (2.3), as follows:

$$Q_\ell = \int_0^{T_\ell} I(t)\,\mathrm{d}t = \int_0^{T_\ell} \frac{V_\ell}{T_\ell}\left[C_{near} + C_{far}\left(1 - e^{-\frac{t}{RC_{far}}}\right)\right]\mathrm{d}t$$

Since the charge on C_ℓ is given by $Q_\ell = C_\ell V_\ell$, equating the two charge transfer equations yields:

$$V_\ell C_{near} + V_\ell C_{far}\left[1 - \frac{RC_{far}}{T_\ell}\left(1 - e^{-\frac{T_\ell}{RC_{far}}}\right)\right] = C_\ell V_\ell$$

By solving for C_ℓ, we obtain:

$$C_\ell = C_{near} + C_{far}\left[1 - \frac{RC_{far}}{T_\ell}\left(1 - e^{-\frac{T_\ell}{RC_{far}}}\right)\right] \tag{2.4}$$

or in a more compact form: $C_\ell = C_{near} + K_\ell C_{far}$, where

$$K_\ell = 1 - \frac{RC_{far}}{T_\ell}\left(1 - e^{-\frac{T_\ell}{RC_{far}}}\right)$$

As shown in the above formula, K_ℓ factor, which is the capacitance shielding factor, depends on the time constant RC_{far} and the input slew T_ℓ of the π-model interconnect. It is evident, from (2.4), that when the π-model interconnect is highly resistive, K_ℓ tends to zero. On the contrary, when R is close to zero, K_ℓ approaches 1. This confirms the intuition that C_ℓ is equivalent to the parallel connection of C_{near} and C_{far}, when R is negligible, while it effectively accounts for a virtually disconnected C_{far}, when R is very large.

The main advantage of the described method is that it provides a closed-form formula for C_{eff} estimation, rather than applying expensive moment-matching techniques [55]. It is important to note that although C_ℓ of (2.4) does not include the receiver input pin capacitance C_p, it may be easily extended to account for it, by adding a constant C_p (obtained by NLDM) together with C_{far}, as they are in parallel. However, even including a constant C_p, it still ignores the Miller effect.

2.3.3 Challenges in gate delay estimation using CSMs and C_{eff}

Two main challenges arise when attempting to estimate gate delay and output slew using CSMs and C_{eff}. First, there is an interdependence between driver output slew and receiver input pin capacitance. As described in Section 2.3.1, driver output current and voltage waveforms depend on the load seen by the driver, which in turn depends on input pin capacitance of receiver gate(s). On the other hand, receiver input capacitance is a function of receiver input slew, which depends both on interconnect RC parasitics and on driver output slew.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Second, the modeling of C_{eff} is essential for accurate delay calculations. However, it is infeasible to obtain a single C_{eff} value, that is suitable for both delay and slew calculations, as it cannot exactly match the actual load in terms of driver output current at any time instant. In addition, as shown in (2.4), C_ℓ is a function of driver output slew, and thus there is an interdependence between them as well.

The above challenges dictate the use of an iterative approach, which can handle both interdependencies simultaneously. To the best of our knowledge, such an approach has not yet been proposed in the literature.

2.4 Proposed Approach

In this section, we present our approach for accurate and efficient gate delay estimation. To this end, we propose an iterative algorithm that effectively addresses the aforementioned challenges, exploiting library compatible CSMs and the dynamic behavior of C_{eff}, while considering the Miller effect.

2.4.1 Computation of a single C_{eff} considering the Miller effect

As discussed in Section 2.3.1, the Miller effect may be very strong, especially at technology nodes below 45 nm [4]. In contrast to the resistive shielding effect, the Miller effect impact on gate delay and slew is higher in stages with small impedance interconnects, where the receiver input pin capacitance dominates C_{eff}. As a result, for accurate C_{eff} estimation, we must consider the entire driver RC load (π-model and receiver input pin capacitance C_p), while taking into account the Miller effect. In our proposed approach, we exploit the dynamic behavior of C_p, instead of using a constant value, using library compatible CSMs which model the Miller effect.

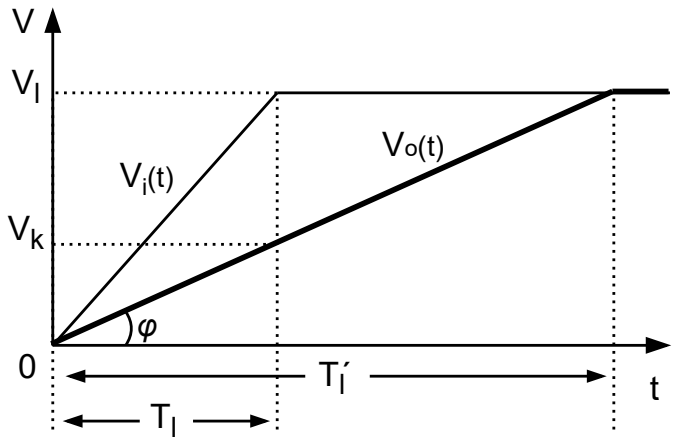


FIGURE 2.3: Output voltage waveform and slew calculation for a π-model with linear ramp input voltage waveform.

To compute the receiver pin capacitance up to a specific voltage threshold V_ℓ, which is denoted as $C_{p(\ell)}$, the slew at the output of the interconnect, T'_ℓ, must be calculated. The most accurate estimation may be obtained by performing interconnect transient analysis using the driver output waveform as excitation. However, this is prohibitive even for small circuits. An efficient MOR technique to accelerate interconnect simulation is presented in Chapter 3. However, the methodology presented in this section relies on closed-form formulas, as it also aims to be used in early-stage analysis. A closed-form formula for the interconnect output

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

slew can be derived as follows. Fig. 2.3 depicts an approximation of the output voltage waveform for a π-model, given a ramp input voltage waveform $V_i(t) = \frac{V_\ell}{T_\ell}t$, up to V_ℓ. In this case, the output voltage $V_o(t)$ can be calculated by:

$$\begin{aligned} V_o(t) &= V_i(t) - I_2(t)R \\ &= \frac{V_\ell}{T_\ell}t - \frac{V_\ell}{T_\ell}(C_{far} + C_{p(\ell)})\left(1 - e^{-\frac{t}{R(C_{far}+C_{p(\ell)})}}\right)R \end{aligned}$$

In the above equation, $I_2(t)$, which is given in (2.3), has been updated to include $C_{p(\ell)}$, which is parallel to C_{far}.

Therefore, at $t = T_\ell$, when the input voltage waveform crosses V_ℓ, the output voltage is given by:

$$V_k = \frac{V_\ell}{T_\ell}\left[T_\ell - R(C_{far} + C_{p(\ell)})\left(1 - e^{-\frac{T_\ell}{R(C_{far}+C_{p(\ell)})}}\right)\right] \tag{2.5}$$

Also, from Fig. 2.3, it can be seen that:

$$\tan(\phi) = \frac{V_k}{T_\ell} = \frac{V_\ell}{T'_\ell} \implies T'_\ell = \frac{V_\ell T_\ell}{V_k} \tag{2.6}$$

Substituting (2.5) in (2.6) yields:

$$T'_\ell = \frac{T_\ell}{1 - \frac{R(C_{far}+C_{p(\ell)})}{T_\ell}\left(1 - e^{-\frac{T_\ell}{R(C_{far}+C_{p(\ell)})}}\right)} \tag{2.7}$$

It is also worth mentioning that several moment-based metrics have been also evaluated for slew propagation (e.g., SS2M [20], BAK [21]), along with PERI [57] to extend them to ramp inputs. However, the metric of (2.7) leads to better accuracy results, while it does not require computation of moments.

Now, given T'_ℓ and the receiver output load, $C_{p(\ell)}$ is computed by accessing the CSM receiver capacitance LUTs (*e.g.,* CCS `receiver_capacitance_1/2_rise/fall` to derive $C1$, $C2$ values). In order to account for $C_{p(\ell)}$, the effective capacitance formula of (2.4) is finally updated to:

$$\begin{aligned} C_\ell &= C_{near} + (C_{far} + C_{p(\ell)}) \\ &\times \left[1 - \frac{R(C_{far} + C_{p(\ell)})}{T_\ell}\left(1 - e^{-\frac{T_\ell}{R(C_{far}+C_{p(\ell)})}}\right)\right] \end{aligned} \tag{2.8}$$

or in a compact form: $C_\ell = C_{near} + K_\ell(C_{far} + C_{p(\ell)})$, where

$$K_\ell = 1 - \frac{R(C_{far} + C_{p(\ell)})}{T_\ell}\left(1 - e^{-\frac{T_\ell}{R(C_{far}+C_{p(\ell)})}}\right)$$

Even though C_ℓ of (2.8) improves accuracy by considering the Miller effect, it is still insufficient to accurately estimate gate delay and output slew, as it is a single C_{eff} and assumes that driver output voltage is a linear ramp. As can be seen in Fig. 2.4, this approach totally ignores the nonlinear shape of the actual driver waveform obtained using SPICE. The resulting estimation error is much higher for driver output slew, as it is measured between the time instants when the output voltage waveform crosses the lower (V_{low}) and upper (V_{high}) thresholds, compared to driver delay which is measured between the time instants when the input and output voltage waveforms cross the delay threshold (V_{delay}). However, accurate

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

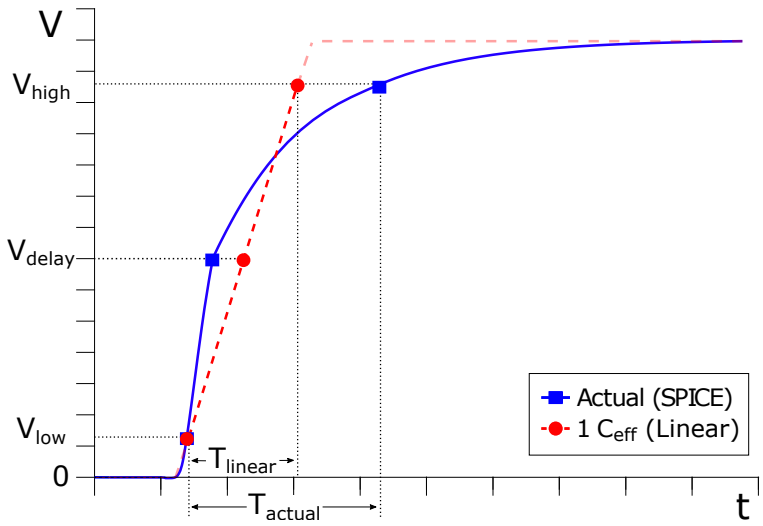


FIGURE 2.4: Comparison between the linear ramp voltage waveform computed using a single C_{eff} and the actual SPICE waveform.

driver slew computation is essential for interconnect delay and slew estimation, which impacts delay and slew estimation for the receiver gate(s).

2.4.2 Computation of multiple C_{eff}

To accurately approximate the nonlinear driver waveform, we compute a PWL ramp, exploiting library compatible CSMs. Fig. 2.5 demonstrates that this CSM waveform is able to closely match the actual waveform, leading to great accuracy in delay and slew estimation. To compute this waveform, we use multiple C_{eff} values, *i.e.,* one C_{eff} value per each linear segment. The effective capacitance $C_\ell^{\ell+1}$ for a specific voltage region $[V_\ell, V_{\ell+1}]$ of the driver output waveform can be derived by using the equivalent charge $Q_\ell^{\ell+1}$ equation, given by:

$$Q_\ell^{\ell+1} = \int_{T_\ell}^{T_{\ell+1}} I(t)\,\mathrm{d}t = Q_{\ell+1} - Q_\ell$$

Since $Q = CV$ and $V_\ell^{\ell+1} = V_{\ell+1} - V_\ell$, we have:

$$C_\ell^{\ell+1} = \frac{Q_\ell^{\ell+1}}{V_\ell^{\ell+1}} = \frac{Q_{\ell+1} - Q_\ell}{V_{\ell+1} - V_\ell} = \frac{C_{\ell+1}V_{\ell+1} - C_\ell V_\ell}{V_{\ell+1} - V_\ell} \tag{2.9}$$

Equation (2.9) describes C_{eff} in a specific region as a function of the C_{eff} values corresponding to the lower and upper thresholds of the region. Note that $C_\ell, C_{\ell+1}$ are computed using (2.8), to account for the Miller effect.

The detailed CSM waveform may also be used for a more accurate interconnect analysis, in order to compute a PWL receiver input waveform, using (2.7). This improves the estimation accuracy for receiver delay, output slew, and input pin capacitance. Furthermore, driver output slew $T_\ell^{\ell+1}$ and interconnect output slew $T_\ell^{(\ell+1)'}$, for a specific region $[V_\ell, V_{\ell+1}]$, can be derived as:

$$T_\ell^{\ell+1} = T_{\ell+1} - T_\ell \quad \text{and} \quad T_\ell^{(\ell+1)'} = T'_{\ell+1} - T'_\ell$$

As can be seen in Fig. 2.5, three C_{eff} values computed in three voltage regions, *i.e.,* $[0V_{dd}, V_{low}]$, $[V_{low}, V_{delay}]$, and $[V_{delay}, V_{high}]$, are typically sufficient to accurately compute driver delay and output slew. Computing a more detailed CSM waveform, using more C_{eff}

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

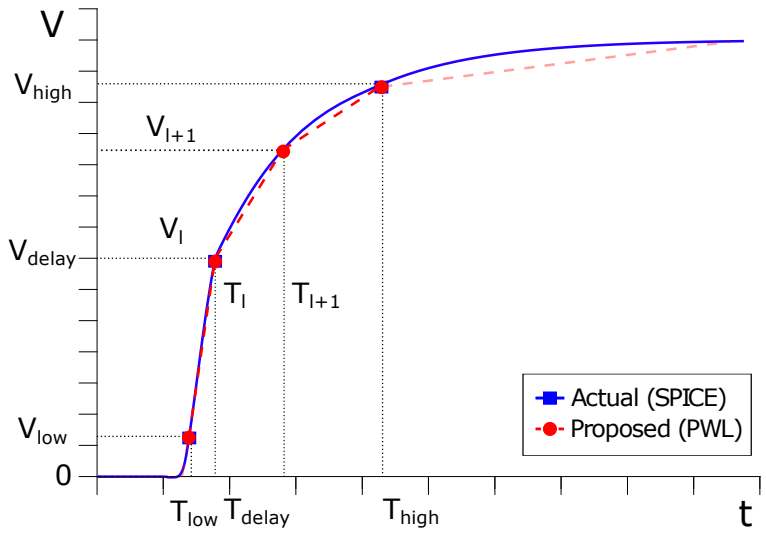


FIGURE 2.5: Comparison between the PWL voltage waveform computed using multiple C_{eff} and the actual SPICE waveform.

values, may lead to improved accuracy results for both gate/interconnect delay and slew, inducing a small performance overhead.

2.4.3 Algorithm for gate delay and output slew estimation, using CSMs and multiple C_{eff}

In this section, we present the iterative algorithm that implements our proposed approach. The purpose of this algorithm is to estimate both gate delay and output slew for a specific driver timing arc of a given <driver, π-model, receiver> stage. An example of such stage is shown in Fig. 2.1.

Given the driver input slew (tr_d^{in}) for the respective timing arc, a receiver output capacitive load (c_r^{out}) (usually set to C_{total}), the π-model parameters (C_{near}, R, C_{far}), and the non-controlling values for the driver and receiver side inputs, our method iteratively computes driver delay (d) and output slew (T_{low}^{high}), until output slew converges. This is done by computing the CSM driver output voltage waveform, and C_{eff}, in n regions provided as a set $V = \{V_a, ..., V_b\}$ of $n+1$ subsequent voltage thresholds. The C_{eff} values in these regions are stored into a set $C = \{C_a^{a+1}, ..., C_{b-1}^{b}\}$, while the time instants when driver output waveform crosses the specified voltage thresholds are stored into a set $T = \{T_a, ..., T_b\}$.

The details of the proposed algorithm are described in Algorithm 1. First, $C_\ell^{\ell+1}$ for each specified region $[V_\ell, V_{\ell+1}]$ is initialized to C_{total}, using the NLDM receiver input pin capacitance (steps 2-5). Second, the CSM output voltage waveform is computed, using C_{total} (step 6), and is used to obtain the initial estimation of d and T_{low}^{high} (step 7). In the main iterative refinement loop (steps 8-17), for each region $[V_\ell, V_{\ell+1}]$, the algorithm computes the receiver input slew values $T'_\ell, T'_{\ell+1}$ (step 10), in order to update the corresponding $C_{p(\ell)}$ values (step 11), and computes the new $C_\ell^{\ell+1}$ value to update C (steps 12-13). Then, driver output waveform, delay and output slew are re-calculated (steps 15-16). This iterative refinement is performed until T_{low}^{high} converges within a specified tolerance (*e.g.*, $|T_{low}^{high}(new) - T_{low}^{high}(old)| < tolerance$). Thus, the overall time complexity of our method is $O(|V|)$, where $|V|$ is the number of voltage thresholds. The CSM operations (LUT accesses, current-to-voltage transformations, interpolations, and driver waveform computation), which differ across various CSMs and are related to their characteristics, are of constant time complexity, since they do not depend on $|V|$.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

To compute the CSM driver output waveform, we developed Algorithm 2. Given a set of voltage thresholds, C_{eff} per region and the driver input slew tr_d^{in}, this algorithm computes the driver output waveform (*i.e.,* voltage for ECSM or current for CCS) for each region $[V_\ell, V_{\ell+1}]$, by setting the driver output load c_d^{out} to the corresponding $C_\ell^{\ell+1}$ (step 3), and accessing CSM LUTs (*e.g.,* CCS `output_current_rise/fall`) using (tr_d^{in}, c_d^{out}) (step 4). In case the CCS model is used, the corresponding voltage waveform is obtained by integrating the current waveform (*e.g.,* using the Trapezoidal rule) (steps 5-6). Then, $T_\ell, T_{\ell+1}$ are computed and T is updated (steps 7-8). Finally, after computing the CSM waveform, described by $\{T,V\}$, the algorithm computes the `reference_time` t_{ref}, by accessing CSM LUTs using tr_d^{in} (step 10).

Algorithm 1: Compute driver delay and output slew for a <driver, π-model, receiver> stage

Input: $V = \{V_a, ..., V_b\}, tr_d^{in}, c_r^{out}$
Output: d, T_{low}^{high}
1 **Function** `compute_driver_CSM_timing`$(V, tr_d^{in}, c_r^{out})$**:**
2 **foreach** *voltage region* $[V_\ell, V_{\ell+1}]$ *in* V **do**
3 $C_\ell^{\ell+1} = C_{near} + C_{far} + C_{nldm}$
4 update C with $C_\ell^{\ell+1}$
5 **end**
6 $\{T, t_{ref}\} =$ `compute_CSM_waveform`(V, C, tr_d^{in})
7 $\{d, T_{low}^{high}\} =$ `compute_CSM_delay_slew`(T, V, t_{ref})
8 **while** T_{low}^{high} *not_converged* **do**
9 **foreach** *voltage region* $[V_\ell, V_{\ell+1}]$ *in* V **do**
10 compute $T'_\ell, T'_{\ell+1}$ using (2.7)
11 compute $C_{p(\ell)}, C_{p(\ell+1)}$ by accessing CSM LUTs using (T'_ℓ, c_r^{out}) and $(T'_{\ell+1}, c_r^{out})$, respectively
12 compute $C_\ell^{\ell+1}$ using (2.9)
13 update C with $C_\ell^{\ell+1}$
14 **end**
15 $\{T, t_{ref}\} =$ `compute_CSM_waveform`(V, C, tr_d^{in})
16 $\{d, T_{low}^{high}\} =$ `compute_CSM_delay_slew`(T, V, t_{ref})
17 **end**
18 **return** $\{d, T_{low}^{high}\}$
19 **End Function**

Driver delay and output slew are computed using the operations described in Algorithm 3. Given the CSM output voltage waveform ($\{T, V\}$), this algorithm computes the time instants $T_{low}, T_{delay}, T_{high}$, when the output waveform crosses $V_{low}, V_{delay}, V_{high}$ (step 2). Then, using these values and the input `reference_time` t_{ref}, it computes d and T_{low}^{high} (steps 3-4).

At this point, we can elaborate on a key aspect regarding the efficient implementation of the proposed approach. The most computationally expensive step in our methodology is the CSM driver output waveform computation, described in Algorithm 2. This is because it involves interpolation between the closest precharacterized voltage waveforms, to compute the non-precharacterized waveform for arbitrary slew, capacitance values. Additionally, in the case of CCS, the closest current waveforms must be transformed to voltage waveforms prior to interpolation, which may also be costly.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Algorithm 2: Compute CSM driver output waveform in specified voltage regions, using multiple C_{eff}

Input: $V = \{V_a, ..., V_b\}, C = \{C_a^{a+1}, ..., C_{b-1}^{b}\}, tr_d^{in}$
Output: $T = \{T_a, ..., T_b\}, t_{ref}$

```
Function compute_CSM_waveform(V, C, tr_d^in):
    foreach voltage region [V_ℓ, V_ℓ+1] in V do
        c_d^out = C_ℓ^ℓ+1
        compute driver output waveform by accessing CSM LUTs using (tr_d^in, c_d^out)
        if (CSM used is CCS) then
            transform waveform from current to voltage
        compute T_l, T_ℓ+1 using voltage waveform and V_ℓ, V_ℓ+1
        update T with T_ℓ, T_ℓ+1
    end
    compute t_ref by accessing CSM LUTs using tr_d^in
    return {T, t_ref}
End Function
```

Algorithm 3: Compute driver delay and output slew, using CSM driver output waveform

Input: $T = \{T_a, ..., T_b\}, V = \{V_a, ..., V_b\}, t_{ref}$
Output: d, T_{low}^{high}

```
Function compute_CSM_delay_slew(T, V, t_ref):
    compute T_low, T_delay, T_high using CSM waveform {T, V}
    d = T_delay − t_ref
    T_low^high = T_high − T_low
    return {d, T_low^high}
End Function
```

To improve performance, the CCS current-to-voltage transformation and the computation of $T_\ell, T_{\ell+1}$ values for each precharacterized CSM waveform may be performed only once. In order to reduce memory requirements, we may compute and store only the required set of time instants T for the specified set of voltage thresholds V. This may be performed either off-line before the delay calculation for all precharacterized waveforms or only the first time we process each waveform. In case this is performed off-line for all standard cells, multiple threads may be used in parallel to speedup the procedure. Thus, to compute $T_\ell, T_{\ell+1}$ in Algorithm 2, we may interpolate between these time instants, instead of the entire waveforms.

The proposed approach may be extended to handle stages with distributed RC interconnects and multiple receiver gates, by exploiting the forward-backward traversal algorithm presented in [22], in order to update C (Algorithm 1, steps 10-15). This algorithm computes the delay of an RC interconnect, which is handled as connected π-models, assuming single slew and C_{eff} on each node. However, it can be modified to compute slew and C_{eff} per specified region. Specifically, during the forward traversal, the slew for each region may be propagated, using breadth-first search (BFS), from the driver output pin (source) toward the receiver input pins (sinks). For each π-model output node, $T_\ell^{\ell+1}$ may be computed using (2.7), using the $C_\ell^{\ell+1}$ of this node as C_{far} (considering also $C_{p(\ell)}$ for the π-models connected to sinks). Then, during the backward traversal, $C_\ell^{\ell+1}$ may be recalculated and

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

propagated backwards from sinks to source, to update C.

Moreover, it is worth mentioning that our algorithm may be integrated into the delay calculator of any gate-level Static Timing Analysis (STA) [3] or Dynamic Timing Analysis (DTA) [50] method based on library compatible CSMs.

2.5 Experimental Evaluation

To evaluate our method, we implemented Algorithm 1 using CCS as CSM, and three regions for C_{eff} and driver waveform computation. We also implemented six alternative methods (`M1`–`M6`) that differ in three key features, *i.e.,* (i) the driver model, where CCS or NLDM is utilized, (ii) the load model, where C_{total} or C_{eff} is used, and (iii) the receiver model, where CCS is used to consider the Miller effect, or NLDM is used otherwise. Table 2.1 summarizes the key differences of the investigated methods. Note that for the methods which use three regions (3 C_{eff}), without loss of generality, we set $V = \{0V_{dd}, 0.1V_{dd}, 0.5V_{dd}, 0.9V_{dd}\}$. We selected these regions because the standard cell library used for our experiments is precharacterized using $V_{low} = 0.1V_{dd}$, $V_{delay} = 0.5V_{dd}$, $V_{high} = 0.9V_{dd}$. Similarly, for the methods which use 1 C_{eff}, the single region $V = \{0V_{dd}, 0.5V_{dd}\}$ is used. Moreover, the convergence tolerance for T_{low}^{high} had been set to 10^{-3}.

In more detail, `M1` uses NLDM and computes C_{total}, while it ignores the Miller effect. Among all the examined methods, `M1` is the only non-iterative method. All the other methods, *i.e.,* `M2`–`M6`, are implemented with modifications of the iterative method described in Algorithm 1. `M2` assumes a single C_{eff} (Algorithm 1, steps 9-14) and implements a function similar to `compute_CSM_waveform()`, that computes a ramp waveform with a fixed slew, by using the NLDM LUTs. However, it cannot model the Miller effect. `M3` considers the Miller effect but computes C_{total} using two input pin capacitance values, C1 and C2, for the receiver model (Algorithm 1, steps 12-13). The rest of the methods (`M4`, `M5`, `M6`, and ours) compute C_{eff}, however, differ in the number of driver voltage waveform regions selected to be matched, and Miller effect consideration. More specifically, `M4` (*i.e.,* the method of [22]) and `M5` (*i.e.,* a variant of [15]) compute C_p using NLDM, (Algorithm 1, step 11), and use 1 C_{eff} and 3 C_{eff}, respectively. Finally, `M6` is identical to our method, but applies interconnect transient simulation to estimate receiver input slew more accurately (Algorithm 1, step 10). To evaluate the accuracy of the above methods, we measured their Root

TABLE 2.1: Characteristics of the investigated methods and Root Mean Square Percentage Errors (RMSPE) against Synopsys HSPICE®

Method	Driver Model	Load Model	Receiver Model	RMSPE	
				Delay	Slew
`M1`	NLDM	C_{total}	NLDM	19.19%	21.60%
`M2`	NLDM	1 C_{eff}	NLDM	5.86%	8.92%
`M3`	CCS	C_{total}	CCS	19.08%	21.51%
`M4`	CCS	1 C_{eff}	NLDM	1.65%	10.81%
`M5`	CCS	3 C_{eff}	NLDM	1.99%	2.68%
`M6`*	CCS	3 C_{eff}	CCS	1.01%	2.10%
Ours	CCS	3 C_{eff}	CCS	1.32%	2.48%

* Computes receiver input slew using transient simulation.

Mean Square Percentage Error (RMSPE) against Synopsys HSPICE® [17]. For each timing metric x (delay or output slew), the RMSPE of each method, across all measurements, is

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

calculated by:

$$RMSPE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\left|\frac{\hat{x}_i - \bar{x}_i}{\bar{x}_i}\right| * 100\%\right)^2}$$

where $\hat{x}_i$ is the measurement of the examined method for stage i, $\bar{x}_i$ is the corresponding HSPICE® measurement, and n is the total number of measurements for all stages considering both rise and fall transitions (*i.e.,* $n = 2 \cdot \#stages$).

To compare the investigated methods for both accuracy and runtime, we integrated them into the TAU 2020 contest C++ framework [43], which generates representative <driver, π-model, receiver> stages. For our experiments, the driver and receiver gates are selected from the ASU ASAP 7 nm predictive Process Design Kit (PDK) [58], which is publicly available in the OpenROAD GitHub repository [59]. Some basic MOSFET SPICE model parameters of the PDK are provided in Table 2.2. However, the TAU framework only supports gates with one input/output pin, *i.e.,* buffers and inverters, and thus we also had to extend this framework to support multiple input/output gates (e.g., NAND, XOR, AOI, HA, DFFRS, DLatch) and account for any combinational, sequential, and asynchronous timing arc.*

TABLE 2.2: ASU ASAP 7 nm SPICE MOSFET parameters

SPICE Parameter	Value
Structure Selector ($GEOMOD$)	1 (triple-gate)
Channel Length (L)	21 nm
Fin Height (H_{fin})	32 nm
Fin Thickness (T_{fin})	6.5 nm
Threshold Voltage ($V_{th,n}$, $V_{th,p}$)	0.25 V, −0.2 V
Oxide Permittivity (ϵ_{ox})	34.53 pF/m
Physical Oxide Thickness (T_{oxp})	21 nm

The examined π-model loads are representative input admittance models of real IC interconnects of varying length, routed on different metal layers (up to 16 layers), and their resistance and capacitance values cover an exhaustive range of 0.2 to 100 kOhm and 0.0001 to 0.25 pF, respectively. Moreover, for the driver timing arc, we used an input voltage waveform with slew varying from 0.005 to 0.32 ns, which represents an ASU ASAP pre-driver gate. Finally, receiver output capacitance values in the range of 0.0004 to 1.473 pF are used, which along with driver input slew, cover the entire ranges used in the CCS model precharacterization.

2.5.1 Accuracy results

To compare the accuracy of the investigated methods, in terms of delay and output slew RMSPE against HSPICE®, a set of 50k stages was used. In Fig. 2.6, the horizontal axis corresponds to the time constant over input slew metric of the driver RC load (π-model and receiver capacitance), *i.e.,* $\frac{R(C_{far}+C_p)}{T_{low}^{high}}$, for all stages arranged in buckets. This metric was used to represent different RC and input waveform characteristics of driver loads. Fig. 2.6 clearly shows that the methods which use C_{total} as load model, *i.e.,* `M1` and `M3`, lead to extremely inaccurate results. For example, `M1` results in 3.27× and 2.42× greater RMSPE, for delay and slew, respectively, compared to `M2` (as shown in Table 2.1).

Moreover, Table 2.1 and Fig. 2.6 present a comparison of the NLDM and CCS gate models. In general, methods using CCS present high delay and slew accuracy. For example,

* Our delay calculator is available at https://github.com/digaryfa/UTH-Timer.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

M4 presents 1.65% delay RMSPE, by using CCS as driver model, while M2 results in 5.86% delay RMSPE, by using NLDM. However, M4 may lead to slightly higher slew error, as it uses NLDM as a receiver model.

At this point, we can evaluate the impact of multiple C_{eff} values on the accuracy of delay and slew estimation. As depicted in Table 2.1, the use of multiple C_{eff} values (*i.e.,* in M5, M6, and ours) does not significantly influence the delay accuracy, compared to M4 which uses the same driver model and a single C_{eff}. On the other hand, output slew accuracy can be dramatically improved using multiple C_{eff} values. As can be seen in Fig. 2.6, specifically in bucket [0.50, 0.74], M4 results in approximately 24% slew RMSPE, while our method achieves 4% error.

As for the impact of Miller effect on delay and slew calculation, our proposed method leads to better results compared to M5, which ignores this effect. Fig. 2.6 demonstrates that for small values of $\frac{R(C_{far}+C_p)}{T_{low}^{high}}$, *i.e.,* in the range [0, 0.09], the Miller effect has a significant impact on delay calculation, while slew calculation is slightly influenced. For example, in bucket [0, 0.01], our method achieves 0.87% delay RMSPE, compared to M5 which leads to 2.48% error.

Finally, we compare our method against M6, which provides the highest accuracy among all the examined methods. As shown in Table 2.1, our method results in 1.32% delay RMSPE and 2.48% slew RMSPE, while M6 leads to 1.01% and 2.1% errors, respectively. However, M6 is significantly slower than our methodology, as interconnect transient simulation is time-consuming, rendering this method prohibitive for large designs. Therefore, considering only the investigated methods that are efficient for gate-level timing analysis, our method achieves the best accuracy results. In addition, it is worth mentioning that the proposed iterative method achieves less than 2.6% and 4% delay and slew RMSPE, respectively, even from the first iteration.

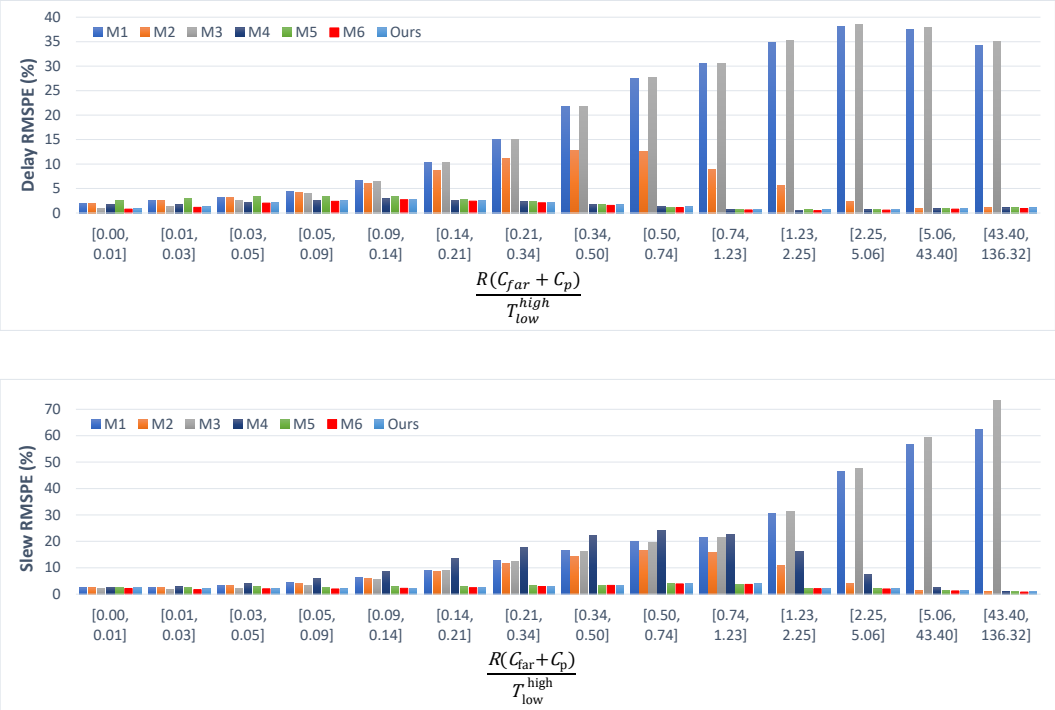


FIGURE 2.6: Gate delay and output slew RMSPE against Synopsys HSPICE® on a testcase with 50k stages.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

2.5.2 Runtime results

To examine the scalability of our method, we generated various testcases, from 10 to 200k stages. For runtime evaluation, we used a Linux workstation with a 3.60 GHz Intel® Core™ i7-4790 CPU (4 cores, 8 threads) and 16 GB memory. Note that all the examined methods, as well as the HSPICE® simulations, estimate the delay and output slew for all stages in parallel, using multiple threads. Table 2.3 reports the detailed runtimes of the investigated methods, while Fig. 2.7 demonstrates their scalability with the number of stages. As shown in Table 2.3, `M6` is prohibitive even for small number of stages, while its execution time for 200k stages is 626.32 seconds (2328× slower than ours). On the contrary, all the other methods are quite fast, as they compute driver delay and output slew for 200k stages in less than 0.27 seconds, while presenting similar scalability, as depicted in Fig. 2.7. In more detail, the NLDM-based methods, *i.e.,* `M1` and `M2`, need only 0.15 and 0.17 seconds for 200k stages, respectively, while `M3` requires 0.21 seconds. The runtime overhead of using either one or three C_{eff} values is negligible, as methods `M4` and `M5` indicate, *i.e.,* 0.21 and 0.24 seconds, respectively. The proposed method computes gate delay and output slew in 0.27 seconds for 200k stages, adding only a very small overhead compared to `M5` which ignores the Miller effect. In general, our iterative method converges in 2.3 iterations on average and always in less than 4 iterations.

Note that for optimal results, the appropriate methodology may be applied based on its runtime and the characteristics of the stage to be analyzed (*i.e.,* the relative bucket, as shown in Fig. 2.6).

TABLE 2.3: Runtime results of the examined methods for testcases with number of stages varying from 10 to 200k

#stages	Runtime (s)						
	`M1`	`M2`	`M3`	`M4`	`M5`	`M6`	**Ours**
10	0.002	0.002	0.002	0.002	0.002	0.033	0.002
100	0.002	0.002	0.002	0.002	0.002	0.307	0.002
1000	0.002	0.002	0.003	0.002	0.003	3.017	0.003
2500	0.005	0.005	0.005	0.005	0.006	5.701	0.006
5000	0.006	0.006	0.007	0.007	0.008	15.135	0.009
10000	0.010	0.011	0.013	0.012	0.014	30.617	0.016
25000	0.022	0.024	0.029	0.027	0.032	67.357	0.035
50000	0.040	0.044	0.056	0.053	0.062	155.803	0.069
100000	0.077	0.085	0.109	0.104	0.121	322.373	0.135
200000	0.149	0.167	0.215	0.207	0.242	626.327	0.269

2.6 Chapter Summary

In this chapter, we presented an iterative method for fast and accurate gate delay estimation. The proposed approach estimates the driver output voltage waveform and C_{eff} in multiple waveform regions, while considering their interdependence. In contrast to prior works, it exploits library compatible CSMs, employs closed-form formulas, and considers the impact of Miller effect. Its high accuracy and fast convergence make it appealing for use either within optimization loops in early design stage or signoff timing analysis. To evaluate our approach, we integrated our method into the TAU 2020 contest framework and generated 200k test stages, composed of representative π-models and ASU ASAP 7 nm FinFET gates.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

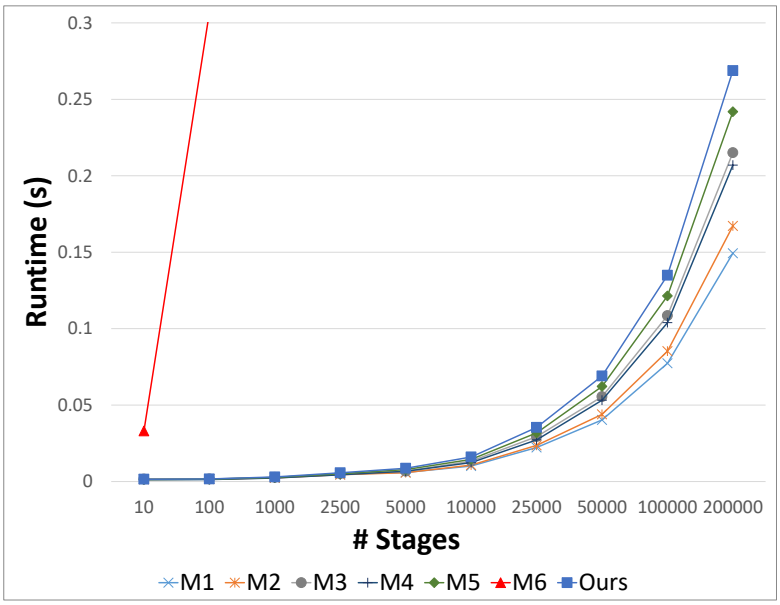


FIGURE 2.7: Graphical comparison between the runtimes of the examined methods for testcases varying from 10 to 200k stages.

Experimental results indicate that our approach achieves 1.3% and 2.5% delay and slew RMSPE, respectively, while converging in 2.3 iterations on average.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Chapter 3

Interconnect Delay Estimation via Sparsity-Aware Model Order Reduction

3.1 Introduction

In nanometer process technologies, interconnect delay has become the decisive factor of performance since it contributes almost 70% of the total path delay [60]. This is due to the increased interconnect average length and routing density, which in turn has resulted in increased interconnect parasitics. In fact, interconnect capacitance dominates gate capacitance, while the resistive nature of interconnects is also growing. On top of that, the parasitic RC networks are getting significantly larger (i.e., include more internal nodes and RC segments) to accurately model contemporary VLSI interconnects that may consist of many branches routed on different metal layers. Thus, we cannot neglect the impact of interconnect parasitics on timing analysis.

Accuracy is the primary consideration in timing analysis, especially during signoff. It is well known that existing fast moment-based delay estimation methods, such as the Elmore delay model [18], are inadequate for signoff timing analysis of interconnects since they rely on simplistic assumptions that are invalid in deep submicron technologies. Furthermore, another important aspect of timing analysis is the efficiency of the interconnect delay estimation method. Traditional transient analysis using SPICE-like simulators offers golden accuracy results but fails to meet the performance and memory requirements for full-chip analysis of current designs that include very large interconnects. However, during interconnect timing analysis, it is not necessary to fully simulate all internal state variables (i.e., node voltages and branch currents), as we only need to calculate the voltage responses at the output ports of the interconnect, given the excitations at the corresponding input ports. As a result, the very large-scale parasitic network can be replaced by a much smaller model with a similar response at the input/output ports. This process is called Model Order Reduction (MOR). MOR techniques are employed in timing analysis to provide a good compromise between accuracy and performance. However, all established MOR methodologies [23–25], as well as recent works such as our proposed approach [61], result in dense system matrices that render their simulation impractical since the simulation cost can easily overshadow the benefits obtained by dimension reduction.

Previous methods attempting to address the problem of dense matrices resulting from MOR have been proposed in the literature. In particular, Ye et al. [62] first divide the circuit nodes into a group corresponding to ports that must be preserved and a group of internal nodes that can be eliminated. Then, they find the Schur complement of the system and perform sparse matrix manipulations on it. The approach in [63] employs partitioning of the circuit into subcircuits and then applies a methodology similar to [62] on each individual

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

subcircuit. A major problem of the aforementioned methods is that they rely more or less on circuit partitioning, which requires circuit-specific information and its effectiveness is heavily dependent on the given circuit. On the other hand, there exists another class of MOR methods, namely node elimination methods, such as [64], which are based on successively eliminating nodes of the initial circuit and distributing the approximation error to the components that are incident to the remaining nodes. These methods inherently produce sparse ROMs, but their efficiency varies widely among different circuits, and they typically do not match the accuracy and reduction levels of projection-based MOR methods. A rigorous mathematical approach for the sparsification of dense MOR circuit matrices was proposed only recently in [65]. The aforementioned method employs a sequence of algorithms based on the computation of the nearest Laplacian matrix and the subsequent sparsification of the corresponding graph.

In this chapter, we present a sparsity-aware MOR methodology, based on [65], for efficient signoff timing analysis of VLSI interconnects. When the proposed methodology is integrated in an iterative optimization flow (instead of using closed-form delay metrics such as [18]), the convergence rate of the optimization can be improved due to the more accurate estimation of the critical paths [22]. On top of that, it can be embedded in a framework like [66] for the timing analysis of large interconnects with many ports (denoted hereafter as complex interconnects). In addition, the sparsified Reduced-Order Models (ROMs) have a straightforward realization to equivalent interconnect networks that can be dumped into a more compact Standard Parasitic Exchange Format (SPEF) file and be used in industrial design flows.

The rest of this chapter is structured as follows. Section 3.2 provides some background material on the interconnect modeling and delay estimation methods. Section 3.3 presents the proposed sparsity-aware MOR methodology for the timing analysis of complex interconnects. In Section 3.4, we demonstrate the performance and accuracy of the proposed methodology. Finally, Section 3.5 concludes this chapter.

3.2 Background

3.2.1 Electrical modeling of a VLSI interconnect

In digital designs, a wire connecting pins of gates (standard cells in EDA terminology) or macros is referred to as an interconnect. An interconnect typically has one driver cell (input port), while it can drive many fanout/receiver cells (output ports). During the physical design, interconnects are routed on multiple metal layers which have different geometric characteristics (e.g., width and thickness). For equivalent electrical representation, VLSI interconnects are typically represented by RC networks, which are extracted using industrial parasitic extraction tools such as Synopsys StarRC™ [67]. For complex large interconnects driving a big number of cells, an accurate representation may be obtained by breaking its total capacitance C_{total} and total resistance R_{total} into multiple segments, creating a distributed RC-tree model as shown in Fig. 3.1. Such RC networks of current VLSI circuits may contain up to hundreds of thousands of internal nodes and thousands of output ports. Typically, the effect of inductance can be ignored within the chip and is only considered for package-level and board-level analysis.

3.2.2 Interconnect delay estimation methods

Elmore Delay Model is the most widely known closed-form interconnect delay estimation method. For a distributed RC-tree (like the interconnect shown in Fig. 3.1), Elmore delay

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

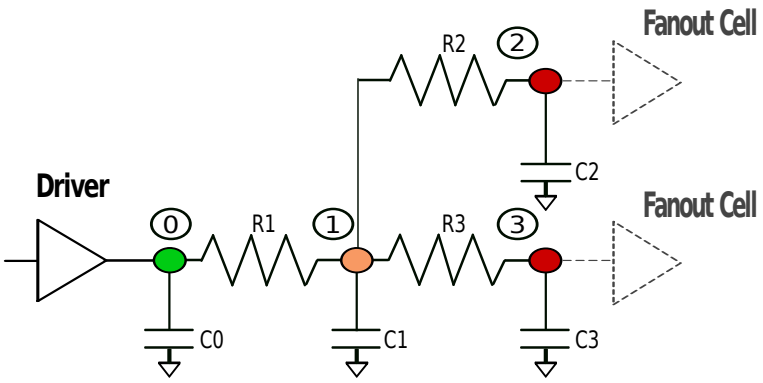


FIGURE 3.1: Distributed RC interconnect of a gate driving two fanout cells.

$T_{0\to i}$ from the input port 0 to the output port i is given by:

$$T_{0\to i} = \sum_{k\in N} R_{ki}C_k$$

where N is the set of all nodes in the RC network, C_k is the lumped capacitance at node k, and R_{ki} represents the total resistance of the common path between the paths from input port 0 to node k and from input port 0 to output port i.

Although Elmore delay model is simple and fast to evaluate since it has linear time complexity with respect to the interconnect size, it considers only a step input, ignoring effects that are input slew dependent. It is well established that this model can be off by orders of magnitude in some cases.

SPICE Transient Analysis of the Original Interconnect Model is the most accurate but time-consuming approach for output waveform calculation and delay estimation. Using the voltage waveform at the input port of the interconnect, which is computed during timing analysis of the previous gate (see Chapter 2), the voltage waveform on each output port can be obtained by SPICE transient analysis of the RC model. Given these waveforms, the corresponding delay is computed as the difference between the time instants when the output and input voltage cross the 50% of the supply voltage.

Let the RC model of an interconnect be composed of n total nodes (excluding ground), where p nodes correspond to input and output ports. Given the excitation source vector of the circuit, the node voltages can be obtained by solving the following system of ordinary differential equations, arising from the Modified Nodal Analysis (MNA) [68]:

$$\mathbf{G}\mathbf{x}(t) + \mathbf{C}\frac{d\mathbf{x}(t)}{dt} = \mathbf{u}(t) \tag{3.1}$$

where $\mathbf{G} \in \mathbb{R}^{n\times n}$ and $\mathbf{C} \in \mathbb{R}^{n\times n}$ are the node conductance and capacitance matrices, $\mathbf{x} \in \mathbb{R}^n$ is the vector of unknown node voltages, and $\mathbf{u} \in \mathbb{R}^n$ is the vector of excitations from independent current sources. Note that the vector of excitations $\mathbf{u}(t)$ has only one non-zero entry, corresponding to the node where the single independent current source is connected to, because the model of interconnects includes only one input port corresponding to the driver output, as we described in the previous subsection.

For the transient analysis, we can use the Backward-Euler numerical method to obtain a system of linear algebraic equations to be solved at any time instant [68]. Several efficient solution methods and preconditioning techniques have been proposed in the literature for solving such linear systems [69, 70]. However, even such efficient simulation techniques cannot provide the performance required in timing analysis of large interconnects with many ports.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Model Order Reduction techniques have been employed in the past, as an alternative to the time-consuming SPICE transient analysis of the original RC interconnect model of (3.1). MOR aims at approximating the original model by another model of reduced order $r << n$, such that the input-output behavior is preserved. In MOR, the approximation is performed through a process of projecting the system matrices onto lower-dimensional subspaces by suitable projection matrices $\mathbf{V}_\ell \in \mathbb{R}^{n\times r}$ and $\mathbf{V}_r \in \mathbb{R}^{r\times n}$, as follows:

$$\tilde{\mathbf{G}} = \mathbf{V}_\ell^T \mathbf{G} \mathbf{V}_r, \quad \tilde{\mathbf{C}} = \mathbf{V}_\ell^T \mathbf{C} \mathbf{V}_r$$

The biggest problem caused by MOR projections is that sparsity of the reduced-order matrices is lost, which may render any time or frequency domain simulation impractical and offset the benefits from order reduction.

3.2.3 Laplacian matrices

Laplacian matrices play a central role in the proposed methodology. Therefore, prior to the description of our methodology, we provide a definition of the Laplacian matrix.

Definition 1: Let $G = (V, E, w)$ be a weighted undirected graph that does not include self-loops, with a set of vertices (nodes) $V = \{1, 2, \ldots, n\}$, a set of edges $E = \{(i, j) | i, j \in V\}$, and weight function $w : E \to \mathbb{R}_{>0}$. The Laplacian of G is a matrix $\mathbf{G}_L \in \mathbb{R}^{n\times n}$ such that:

$$\mathbf{G}_L(i,j) = \begin{cases} \sum_{(i,k)\in E} |w(i,k)| & \text{, if } i = j \\ -w(i,j) & \text{, if } (i,j) \in E \\ 0 & \text{, otherwise} \end{cases} \tag{3.2}$$

Thus, any Laplacian matrix corresponds to a graph and vice versa. An example of a graph along with the corresponding Laplacian matrix is shown in Fig. 3.2.

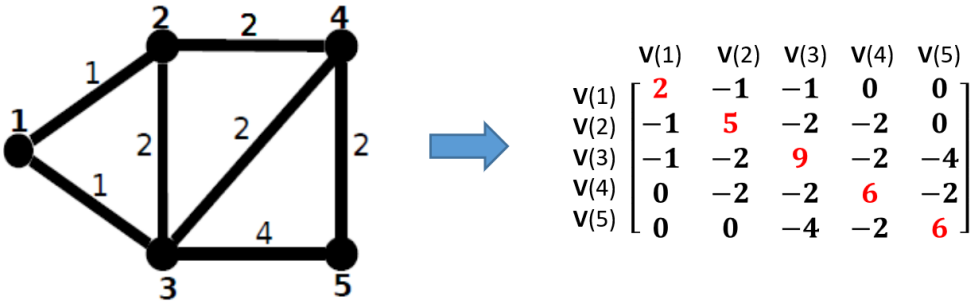


FIGURE 3.2: A weighted graph and the corresponding Laplacian matrix.

3.3 Proposed Approach

In this section, we present our sparsity-preserving MOR technique for timing analysis of VLSI interconnects with many ports. The proposed approach is based on a rigorous methodology for the sparsification of dense ROMs arising in MOR [65]. As described in Algorithm 4, this method firstly computes the nearest Laplacian matrices to the ROM matrices ($\tilde{\mathbf{G}}$, $\tilde{\mathbf{C}}$), let $\tilde{\mathbf{G}}_L$, $\tilde{\mathbf{C}}_L$, and then sparsifies those Laplacians by exploiting efficient spectral graph techniques, so that to preserve the eigenvalues of $\tilde{\mathbf{G}}_L$, $\tilde{\mathbf{C}}_L$ in between given error bounds. However, the projection to the nearest Laplacian matrices at step 2 (depending on the chosen MOR technique) may induce unacceptable error in the sparse ROM, especially when the number of ports is large.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

To this end, our proposed approach aims at improving the accuracy of the sparsification method by applying appropriate equation rearrangements and congruence transformations to the original model matrices ($\mathbf{G}, \mathbf{C}$) of (3.1), like those introduced in PACT [24], which obviate the need to approximate the dense reduced-order matrix $\tilde{\mathbf{G}}$ by its nearest Laplacian matrix after MOR. Although we use the PACT method in the following, since it inherently includes the aforementioned congruence transformations, our method can employ any MOR technique as long as these transformations are applied beforehand.

Algorithm 4: Sparsification of dense ROMs arising in MOR

Input: $\tilde{\mathbf{G}}$, $\tilde{\mathbf{C}}$
Output: $\tilde{\mathbf{G}}_{sp}, \tilde{\mathbf{C}}_{sp}$
Function `MORSparse` $(\tilde{\mathbf{G}}, \tilde{\mathbf{C}})$ **:**
  $[\tilde{\mathbf{G}}_L, \tilde{\mathbf{C}}_L]$ = Compute the nearest Laplacian matrices to $\tilde{\mathbf{G}}$, $\tilde{\mathbf{C}}$ (steps 2-5 in Algorithm 6 [65])
  $[\tilde{\mathbf{G}}_{sp}, \tilde{\mathbf{C}}_{sp}]$ = Sparsify $\tilde{\mathbf{G}}_L$, $\tilde{\mathbf{C}}_L$ (steps 6-7 in Algorithm 6 [65])
  return $[\tilde{\mathbf{G}}_{sp}, \tilde{\mathbf{C}}_{sp}]$
End Function

The method of PACT preserves the dominant poles of the original model admittance matrix in the ROM admittance matrix, given an error control parameter and a frequency band of operation. PACT first rearranges the equations of (3.1) so that the first p equations correspond to port nodes, while the rest of them correspond to internal nodes. Thus, (3.1) can be re-written as:

$$\begin{bmatrix} \mathbf{G}_P & \mathbf{G}_C^T \\ \mathbf{G}_C & \mathbf{G}_I \end{bmatrix} \begin{bmatrix} \mathbf{x}_P(t) \\ \mathbf{x}_I(t) \end{bmatrix} + \begin{bmatrix} \mathbf{C}_P & \mathbf{C}_C^T \\ \mathbf{C}_C & \mathbf{C}_I \end{bmatrix} \begin{bmatrix} \frac{d\mathbf{x}_P(t)}{dt} \\ \frac{d\mathbf{x}_I(t)}{dt} \end{bmatrix} = \mathbf{u}(t) \tag{3.3}$$

where $\mathbf{x}_P$ and $\mathbf{x}_I$ represent the p port nodes and the i internal node voltages, respectively, $\mathbf{G}_P \in \mathbb{R}^{p\times p}$ and $\mathbf{C}_P \in \mathbb{R}^{p\times p}$ represent the interconnections between the port nodes, $\mathbf{G}_I \in \mathbb{R}^{i\times p}$ and $\mathbf{C}_I \in \mathbb{R}^{i\times p}$ describe the interconnections between the internal nodes, and $\mathbf{G}_C \in \mathbb{R}^{p\times i}$ and $\mathbf{C}_C \in \mathbb{R}^{p\times i}$ represent the interconnections between the port nodes and the internal nodes. After the rearrangement of equations, PACT applies the following congruence transformations to matrices $\mathbf{G}$, $\mathbf{C}$ of (3.3):

$$\tilde{\mathbf{G}} = \mathbf{V}^T\mathbf{G}\mathbf{V} = \begin{bmatrix} \mathbf{G}_P - \mathbf{G}_C^T\mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_i \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{G}}_P & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_i \end{bmatrix}$$

$$\begin{aligned} \tilde{\mathbf{C}} = \mathbf{V}^T\mathbf{C}\mathbf{V} &= \begin{bmatrix} \mathbf{C}_P - \mathbf{B}^T\mathbf{A} - \mathbf{A}^T\mathbf{C}_C & \mathbf{B}^T\mathbf{L}^{-T}\mathbf{U} \\ \mathbf{U}^T\mathbf{L}^{-1}\mathbf{B} & \mathbf{\Lambda} \end{bmatrix} \\ &= \begin{bmatrix} \tilde{\mathbf{C}}_P & \tilde{\mathbf{C}}_C^T \\ \tilde{\mathbf{C}}_C & \mathbf{\Lambda} \end{bmatrix} \end{aligned} \tag{3.4}$$

with

$$\mathbf{V} = \begin{bmatrix} \mathbf{I}_p & \mathbf{0} \\ -\mathbf{A} & \mathbf{L}^T\mathbf{U} \end{bmatrix}$$

where $\mathbf{A} = \mathbf{G}_I^{-1}\mathbf{G}_C$, $\mathbf{B} = \mathbf{C}_C - \mathbf{C}_I\mathbf{A}$, $\mathbf{L}$ is the lower triangular matrix from the Cholesky factorization of $\mathbf{G}_I$ ($\mathbf{G}_I = \mathbf{L}\mathbf{L}^T$), $\mathbf{U}$ is the matrix with the eigenvectors as columns, $\mathbf{\Lambda}$ is a diagonal matrix with the eigenvalues from the eigendecomposition of matrix $\tilde{\mathbf{C}}_I = \mathbf{L}^{-1}\mathbf{C}_I\mathbf{L}^{-T}$ ($\tilde{\mathbf{C}}_I = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$), and $\mathbf{I}_p, \mathbf{I}_i$ are the identity blocks corresponding to the port and internal nodes, respectively.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Notice that both internal matrices of (3.4) (the $i \times i$ lower right submatrices of $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{C}}$) are diagonal, and thus the admittance matrix $\mathbf{Y}(s)$ of (3.3), which can be derived by obtaining the Laplace transform ($\mathbf{Y}(s)\mathbf{x}(s) = \mathbf{B}\mathbf{u}(s)$) and then eliminating $\mathbf{x}_I$ from the first p equations, is written as:

$$\mathbf{Y}(s) = \tilde{\mathbf{G}}_P + s\tilde{\mathbf{C}}_P - \frac{s^2\mathbf{r}_1^T\mathbf{r}_1}{1+s\lambda_1} - \ldots - \frac{s^2\mathbf{r}_n^T\mathbf{r}_n}{1+s\lambda_n} \tag{3.5}$$

where $\mathbf{r}_i$ is the i_{th} row of $\tilde{\mathbf{C}}_C$ and λ_i is the i_{th} diagonal of $\mathbf{\Lambda}$. As proved in [24], in case the terms associated with λ_i in (3.5) are dropped when $\lambda_i < \lambda_c$, the relative error of each of the individual elements of $\mathbf{Y}(s)$ is bounded on the complex axis for $|\omega| \leq \omega_c$ by $\epsilon \leq \epsilon_c$ if:

$$\epsilon_c = \omega_c\lambda_c + \omega_c^3\lambda_c^3 \tag{3.6}$$

Finally, the ROM is built by eliminating the rows and columns of $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{C}}$ for which the corresponding diagonal of $\mathbf{\Lambda}$ is less than λ_c or equivalently only the fractional terms in (3.5) that contribute a pole closer than $-\frac{1}{\lambda_c}$ to the imaginary axis are kept in $\mathbf{Y}(s)$. The value of λ_c is determined by inserting the user-specified error ϵ_c and maximum frequency ω_c in (3.6).

Looking at the model of (3.4), an important observation is that matrix $\tilde{\mathbf{G}}$ is of the Laplacian kind because it consists of the identity matrix and the Schur complement of matrix $\mathbf{G}_I$ in the original Laplacian matrix $\mathbf{G}$. A result proved in [71] states that the Schur complement of a block of a Laplacian matrix is also Laplacian. Therefore, in our proposed approach, matrix $\tilde{\mathbf{G}}$ is not required to be approximated by its nearest Laplacian matrix. As a result, the error induced by the approximation of matrices $\tilde{\mathbf{G}}$, $\tilde{\mathbf{C}}$ with their nearest Laplacian matrices $\tilde{\mathbf{G}}_L$, $\tilde{\mathbf{C}}_L$ (in step 2 of Algorithm 4) is reduced by avoiding the approximation of $\tilde{\mathbf{G}}$, which is commonly the most important among the two matrices. In this way, our approach significantly improves the accuracy of the general method of [65] (Algorithm 4), where both $\tilde{\mathbf{G}}$, $\tilde{\mathbf{C}}$ would need to be approximated by the nearest Laplacian matrices. Our improved sparsity-aware MOR technique is presented in Algorithm 5.

Algorithm 5: Sparsity-aware MOR technique for timing analysis of VLSI interconnects

Input: $\mathbf{G}$, $\mathbf{C}$
Output: $\tilde{\mathbf{G}}_{sp}, \tilde{\mathbf{C}}_{sp}$
1 **Function** `MORSparse`($\mathbf{G}$, $\mathbf{C}$)**:**
2 [$\tilde{\mathbf{G}}$, $\tilde{\mathbf{C}}$] = Apply the PACT algorithm [24] on $\mathbf{G}$, $\mathbf{C}$
3 $\tilde{\mathbf{G}}_L = \tilde{\mathbf{G}}$
4 $\tilde{\mathbf{C}}_L$= Compute the nearest Laplacian matrix to $\tilde{\mathbf{C}}$ (steps 3-5 in Algorithm 6 [65])
5 [$\tilde{\mathbf{G}}_{sp}$, $\tilde{\mathbf{C}}_{sp}$] = Sparsify $\tilde{\mathbf{G}}_L$, $\tilde{\mathbf{C}}_L$ (steps 6-7 in Algorithm 6 [65])
6 **return** [$\tilde{\mathbf{G}}_{sp}, \tilde{\mathbf{C}}_{sp}$]
7 **End Function**

3.4 Experimental Evaluation

For the evaluation of the proposed methodology, we developed an in-house gate-level Static Timing Analysis (STA) tool in C/C++. Our STA tool implements Algorithm 1 using Synopsys Composite Current Source (CCS) model for gate timing analysis. The CCS model offers an acceptable trade-off between time to compute and signoff accuracy in voltage waveform estimation at the gate output. The estimated voltage waveform is then used for the timing analysis of the driving interconnect.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Given the interconnect parasitics netlist, we built the matrices of the original model shown in (3.1). Since (3.1) assumes that an independent current source is connected to the input port, we transformed the CCS voltage waveform, which is in series with the first resistor of the interconnect, to the equivalent current source parallel to the resistor. At the next step, by applying the PACT MOR [24] and the proposed sparsity-aware MOR (Algorithm 5), we generated the PACT and MORSparse ROMs. Note that for all examined benchmarks, we selected $\omega_c = 5\,\text{GHz}$ and $\epsilon_c = 5\%$ in (3.6), which resulted in obtaining the upper left $p \times p$ submatrices ($\tilde{\mathbf{G}}_P$, $\tilde{\mathbf{C}}_P$) of the reduced matrices in (3.4).

In order to evaluate the accuracy and runtime of interconnect timing analysis using MORSparse, we implemented the Elmore delay model and SPICE transient analysis for the simulation of the original/full model and the PACT and MORSparse ROMs. Our SPICE simulator was verified against Synopsys HSPICE® [17]. For the solution of the linear systems and the rest of linear algebra operations, we used the Eigen library [72]. We tested the proposed methodology on the ISPD-2012 [73] benchmarks implemented in 45 nm process technology using the NanGate FreePDK library [74], while we executed all experiments on a Linux workstation with a 2.1 GHz Intel® Xeon® E5-1620V4 CPU (4 cores, 8 threads) and 16 GB memory.

To estimate the interconnect delays, we ran our STA tool using the Elmore delay method and SPICE simulation of the original model (full-SPICE), PACT ROM, and MORSparse ROM. For demonstration purposes, we selected one interconnect from each benchmark. Table 3.1 reports the corresponding simulation runtime, delay estimation Mean Relative Error (MRE) among all output ports with respect to full-SPICE simulation, as well as the sparsity ratio $\left(\frac{\textbf{zeros}\left(\tilde{\mathbf{G}}+\tilde{\mathbf{C}}\right)}{\textbf{rows}\left(\tilde{\mathbf{G}}+\tilde{\mathbf{C}}\right)\times\textbf{cols}\left(\tilde{\mathbf{G}}+\tilde{\mathbf{C}}\right)} \right)$ of the PACT and MORSparse ROMs. We can observe that a sparsity ratio over 96.5% is achieved for the MORSparse ROM, which leads to runtime speedups up to 30× over full-SPICE simulation and up to 3.2× over PACT ROM SPICE simulation. The runtime of the Elmore delay estimation is not reported since the Elmore delay can be computed once (before STA) and be retrieved directly during STA, as it is not affected by the characteristics of the interconnect input voltage waveform. Note that although these benchmarks do not include coupling capacitances, since the Elmore delay model cannot deal with crosstalk, MORSparse is expected to provide even higher speedups for such cases since both the simulation of the original model and the PACT method would be more difficult, as the related models would be much bigger while matrix $\mathbf{C}$ would not be a simple diagonal matrix.

The accuracy results reported in Table 3.1 indicate that our methodology provides an accurate approximation of the interconnect delay, with a typical MRE of 4%, while the Elmore delay may deviate up to 288%. The output waveforms obtained by SPICE simulation of the original model and the MORSparse ROM are given in Fig. 3.3. It is apparent that the differences between the two waveforms are negligible.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

TABLE 3.1: Comparison of ROM sparsity, simulation runtime, and delay estimation accuracy between dense ROMs obtained by PACT and sparse ROMs obtained by MORSparse, with respect to full-SPICE simulation of the original model

Benchmark	#nodes	#ports	Model Sparsity		Runtime					Accuracy		
			PACT	MORSp.	Full-SPICE (ms)	PACT (ms)	MORsp. (ms)	PACT vs Full-SPICE Speedup	MORSp. vs Full-SPICE Speedup	Elmore MRE	PACT MRE	MORSp. MRE
vga_lcd	2788	264	52.9%	99.2%	901	44.1	29.1	20.4×	30.9×	264%	5.5e-3%	1.5%
netcard	1827	261	0%	96.6%	381	250.3	100.2	1.52×	3.8×	174%	1.2e-1%	4.3%
mem_ctrl	1677	294	41.5%	98.8%	221	49.5	23.2	4.5×	9.5×	288%	2.7e-3%	1.8%
leon2	1685	247	1.6%	98.3%	653	391	122	1.67×	5.4×	135%	7.5e-2%	6.8%
leon3mp	3345	501	21.1%	97.9%	696	96.2	100.5	7.7×	6.9×	180%	3.7e-2%	7.2%
b19	2050	309	0.6%	96.5%	1073	96.1	165	11.1×	6.5×	135%	1.4e-1%	3.6%

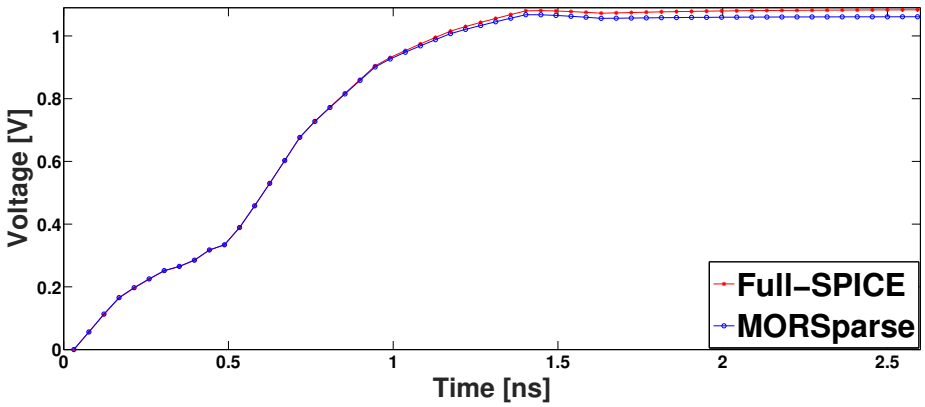


FIGURE 3.3: Voltage response at port inst_41217:A1 of vga_lcd.

3.5 Chapter Summary

In this chapter, we presented a fast and accurate methodology for the timing analysis of VLSI interconnects. The proposed methodology is based on a sparsity-preserving MOR technique to obtain sparse models of the interconnects to be simulated. When incorporated in a timing analysis tool, it can provide signoff accuracy and accelerate interconnect simulation by orders of magnitude. Experimental results on complex interconnects of the ISPD-2012 benchmarks showed that our methodology introduces a negligible typical MRE of 4%, while achieving runtime speedups up to 30× over SPICE simulation of the original model and up to 3.2× over SPICE simulation of the dense PACT ROM.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Chapter 4

EVT-based Worst-Case Delay Estimation Under Process Variation

4.1 Introduction

As we move toward the 45 nm device feature sizes and beyond, manufacturing process variation is becoming an ever-increasing concern for the design of high-performance VLSI circuits, introducing extensive spreads in the transistor and interconnect parameters [26, 75]. Gate and interconnect delay, which is a function of the aforementioned parameters, has to be considered random, distributed between a minimum and a maximum value. As a result, chip designers must take into account process variation so that they can ensure the reliable operation of the circuit. We can distinguish two approaches to verify timing under process variation. The first one and most often used in industry is the corner-based analysis [28]. In the corner-based methodology, the circuit is simulated at the corners of the design parameters where designers expect to find the worst-case delay. In the second method, timing verification is performed through Statistical Static Timing Analysis (SSTA) [29], looking for the distributions of ATs at the Primary Outputs (POs). These distributions provide useful information that enables the estimation of the worst-case AT and slack. SSTA can be performed either by distribution propagation [30] from any Primary Input (PI) to any PO or by Monte Carlo (MC) simulation [31]. The former method is faster but leads to less accurate results than the latter one.

Although previous works toward worst-case delay estimation are attractive and provide meaningful insights into the problem, they are either too slow for large designs or inaccurate since they are based on simplistic assumptions about the underlying gate/interconnect delay models or the distributions propagated across the circuit nodes during SSTA. More specifically, authors in [30] approximate the result of the non-linear MAX operation between two normally distributed random variables with a normal distribution. However, they do not test the effectiveness of their proposed method on a convincing set of benchmarks. In any case, the assumption they make is not supported by the stochastic process theory, and their approximation is expected to deviate even more from the actual distribution when complex gates with more than two inputs are considered. In [28] and [46], a Response Surface Methodology (RSM) is employed, which performs multiple simulations and curve-fitting to obtain the worst delay. However, due to an unprecedented increase in transistor and interconnect complexity (e.g., the large number of metal layers) in modern IC designs, RSM would require a prohibitive number of simulations in order to cover all the dimensions of the problem [47]. Furthermore, authors in [48] focus only on one stage and derive the worst-case delay condition, studying different interconnect structures and gate drive strengths. They provide useful guidelines for the selection of the interconnect capacitance and resistance values that result in the worst-case delay for the stage. However, they do not comment on how the worst-case delay condition can be extended to the whole circuit, as it is infeasible to extract a global worst-case condition due to the existing interdependencies between the delay

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

of a gate/interconnect and the parameters of the following gates/interconnect on a path.

To this end, in this chapter, we present a novel statistical methodology, which relies on MC simulation and Extreme Value Theory (EVT), to estimate the worst AT at the POs of VLSI circuits, under variations on parameters that determine gate and interconnect delay. Note that although during timing analysis we care about the AT distribution on every path endpoint, we focus on POs for simplicity and demonstration purposes. The contribution of our work is that the proposed methodology can provide fast yet accurate results irrespective of the timing models or any assumption about the distributions of AT at the circuit nodes. Moreover, it is suitable for both transistor-level and gate-level timing analysis. It is also worth mentioning that although we focus on maximum delay estimation, our approach may also be applied for minimum delay estimation, enabling both setup and hold timing analysis.

The rest of this chapter is organized as follows. Section 4.2 provides a brief introduction to EVT and presents how an upper end point of a bounded random variable can be estimated exploiting elements from EVT. Section 4.3 explains why it is infeasible to extract a global worst-case condition for the parameters that affect the delay under process variation apriori. Section 4.4 presents our methodology for worst-case delay estimation. Section 4.5 comments on the results obtained by applying the proposed methodology to the ISCAS benchmarks. Finally, conclusions are drawn in Section 4.6.

4.2 Background on EVT

4.2.1 Modeling extreme and rare events

EVT is a branch of probability theory that focuses on the study of extreme and rare events. There are two possible methods for modeling extreme statistics on the basis of a random sample $\mathbf{X}_n = \{X_1, X_2, \ldots, X_n\}$*, where X_is are independent, identically distributed (i.i.d.) random variables from the cumulative density function (cdf) F, namely the *Block Maxima* method and the *Peak Over Threshold* method. The *Block Maxima* method divides the data sample $\mathbf{X}_n$ into l blocks of size m and then holds the maximum from each block, making a new sample of *maxima* $\mathbf{M}_l = \{M_1, M_2, \ldots, M_l\}$. As described in [76], the sample of maxima follows a distribution with cdf that is given by:

$$F_M = P(X_1 \leq x, \ldots, X_m \leq x) = \prod_{i=1}^{m} P(X_i \leq x) = F^m(x) \tag{4.1}$$

The *Peak Over Threshold* method takes the r largest values from the data sample $\mathbf{X}_n$ that exceed a predetermined high threshold u and forms a separate sample of *exceedances* ($\mathbf{X}_{ex}$), again assuming that the sample of exceedances consists of i.i.d. random variables from the cdf F. As described in [76], the sample of exceedances follows a distribution with (conditional) cdf that is given by:

$$F_Z(z) = P(X - u \leq z | X > u) = \frac{F(z+u) - F(u)}{1 - F(u)} \tag{4.2}$$

4.2.2 Limiting distributions

We first need to define the concept of the *upper end point*, which plays a central role in the prediction of the worst-case delay.

* In this chapter, bold letters denote vector variables, while non-bold letters denote scalar variables.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Definition 2: The upper (or right) end point $\omega(F)$ of a cdf $F(x)$ is defined as the upper bound of the support of $F(x)$:

$$\omega(F) = sup\{x : F(x) < 1\} \tag{4.3}$$

The upper end point represents the maximum value that the associated random variable can acquire and becomes $\omega(F) = F^{-1}(1)$ if the random variable is bounded or $\omega(F) = +\infty$ in the opposite case.

The two fundamental theorems, upon which EVT relies, designate the limiting distributions of maxima sample defined in (4.1) when $m \to \infty$ and the limiting distribution of exceedances over threshold defined in (4.2) when $u \to \omega(F)$.

Theorem 1 *(Fisher-Tippet [77])*: The sample of maxima cdf (F_M), for given normalizing constants (i.e., location parameter μ_m and scale parameter σ_m), converges to the Generalized Extreme Value (GEV) as m tends to infinity:

$$\lim_{m\to\infty} F_M(\mu_m + \sigma_m) \to H_\xi = e^{-(1-\xi x)^{\xi^{-1}}} \tag{4.4}$$

where ξ is a parameter that determines the shape of H and depends on $F(x)$.

H can be classified, with respect to the shape parameter ξ, into one of the following cdfs:

Frechet:

$$H_{\xi<0}(x) = \begin{cases} 0 & , x \le \mu_m \\ e^{-(\frac{x-\mu_m}{\sigma_m})^{-\xi^{-1}}} & , x > \mu_m \end{cases}$$

where $\mu_m = 0$ and $\sigma_m = F^{-1}\left(1 - \frac{1}{m}\right)$.

Weibull:

$$H_{\xi>0}(x) = \begin{cases} e^{-(-(\frac{x-\mu_m}{\sigma_m})^{\xi^{-1}})} & , x \le \mu_m \\ 1 & , x > \mu_m \end{cases} \tag{4.5}$$

where $\mu_m = \omega(F)$ and $\sigma_m = \omega(F) - F^{-1}\left(1 - \frac{1}{m}\right)$.

Gumbel:

$$H_{\xi\to0}(x) = e^{-e^{-\frac{x-\mu_m}{\sigma_m}}}, \ x \in \mathbb{R} \tag{4.6}$$

where $\mu_m = F^{-1}\left(1 - \frac{1}{m}\right)$ and $\sigma_m = m\int_{F^{-1}(1-\frac{1}{m})}^{\omega(F)}(1 - F(y))dy$.

Theorem 2 *(Balkema and de Haan [78] and Pickands [79])*: The exceedances over threshold (X_{ex}) (conditional) cdf, for a given scale parameter σ_u, converges to the Generalized Pareto (GP) cdf as u tends to $\omega(F)$:

$$\lim_{u\to\omega(F)} F_Z\left(\frac{z}{\sigma_u}\right) \to GP_\xi(z) = 1 - (1 - \xi z)^{\xi^{-1}} \tag{4.7}$$

where ξ is a parameter that determines the shape of GP and depends on $F(x)$.

Depending on ξ, GP_ξ belongs to one of the following distribution families:

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Pareto:

$$GP_{\xi<0}(x) = \begin{cases} 0 & , x \leq u \\ 1 - \left(\frac{(x-u)+\sigma_u}{\sigma_u}\right)^{\xi^{-1}} & , x > u \end{cases}$$

where $\sigma_u = u$.

Beta:

$$GP_{\xi>0}(x) = \begin{cases} 1 - \left(-\frac{(x-u)-\sigma_u}{\sigma_u}\right)^{\xi^{-1}} & , u < x \leq u + \sigma_u \\ 1 & , x > u + \sigma_u \end{cases} \tag{4.8}$$

where $\sigma_u = \omega(F) - u$.

Exponential:

$$GP_{\xi\to 0}(x) = \begin{cases} 0 & , x < u \\ 1 - e^{-\frac{x-u}{\sigma_u}} & , x \geq u \end{cases} \tag{4.9}$$

where $\lim_{u\to\omega(F)} \frac{\sigma_u}{\xi(\omega(F)-u)} = 1$ when $\xi > 0$ and $\lim_{u\to\omega(F)} \frac{\sigma_u}{-\xi u} = 1$ when $\xi < 0$.

4.2.3 Estimation of a finite upper end point $\omega(F)$

An important fact derived from the limiting cdfs in (4.5, 4.8) is that a parent cdf with an infinite upper end point ($\omega(F) = +\infty$) can only have an extreme value distribution with $\xi < 0$, whereas a cdf with a finite upper end point ($\omega(F) < +\infty$) suggests an extreme value distribution with $\xi > 0$. Notice that Weibull and Beta cdfs reach 1 for all x values greater than a finite threshold. However, if $\xi \to 0^+$ ($\xi \downarrow 0$), then $\omega(F)$ cannot be efficiently or accurately estimated through the previous set of distributions and parameters constructed for the general case $\xi > 0$ [45], and we need to exploit either (4.6) or (4.9) with corresponding parameters related to the special case $\xi \downarrow 0$. The estimation of an upper end point $\hat{\omega}$(F) has been studied thoroughly in [80]. Below we present the formulas to get an upper end point estimate, as well as the corresponding confidence intervals to measure the accuracy of the estimate, from the sample of maxima or the sample of exceedances for the cases $\xi \downarrow 0$ and $\xi > 0$, respectively.

For the first case, $\xi > 0$, the upper end point estimate can be obtained as follows:

$$\hat{\omega}(F) = \hat{\sigma_u} + u \tag{4.10}$$

where $\hat{\sigma_u}$ is the Maximum Likelihood (ML) estimate of parameter σ_u that characterizes the Beta probability density function (pdf).

As we discussed in the previous subsection, the pdf of the Beta family (4.8) ($GP(x) = \frac{dGP}{dx}_{\xi>0}$) is the limiting distribution that asymptotically models the sample of exceedances over threshold when $\xi > 0$. The corresponding log-likelihood function of a Beta-distributed sample of size r is:

$$logL(\sigma_u, \beta) = \sum_{i=1}^{r} \left(log\frac{\beta}{\sigma_u} + (\beta - 1) log\left(-\frac{(X_i - u) - \sigma_u}{\sigma_u}\right)\right) \tag{4.11}$$

Maximization of (4.11), with respect to σ_u and the shape parameter β, yields estimates $\hat{\sigma_u}$ and $\hat{\beta}$. The confidence interval that corresponds to a confidence level of $(1-\delta) * 100\%$ is:

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

$$| \hat{\omega}(F) - \omega(F) | \leq \frac{z_{\delta/2}}{\sqrt{r}} \hat{\sigma_u} (\hat{\beta} - 1) \sqrt{\frac{\hat{\beta} - 2}{\hat{\beta}}} \tag{4.12}$$

where $z_{\delta/2}$ is the $\delta/2$ quantile point of the standard normal distribution $N(0,1)$ [81].

We follow the exceedances method instead of the maxima method when $\xi > 0$ for two reasons. The first one is that the Beta distribution for exceedances in (4.8) is a function of two parameters, whereas the Weibull distribution for maxima in (4.5) is a three-parameter function, and as a result the Beta log-likelihood function is more convenient to optimize. The second reason is that we expect the size of the exceedances sample to be greater than the size of the maxima sample and consequently to give better quality in the final estimate.

On the other hand, the lack of a parametric expression of $\omega(F)$ in the exceedances approach renders the maxima method, for which $\omega(F)$ appears as a parameter [see σ_m in (4.6)], our only option when $\xi \downarrow 0$. An estimate for the upper end point, when $\xi \downarrow 0$, can be given from the following formula:

$$\hat{\omega}(F) = \hat{\mu_m} + \frac{\hat{\sigma_m}}{1 + m\sqrt{\pi \log m}(\mathrm{erf}(\sqrt{\log m}) - 1)} \tag{4.13}$$

where $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$ is the well-known *error function* and $\hat{\mu_m}$, $\hat{\sigma_m}$ are the ML estimates of μ_m and σ_m of Gumbel, respectively.

The pdf of the Gumbel family (4.6), as we discussed in the previous subsection, is the limiting distribution that asymptotically models the sample of maxima when $\xi \downarrow 0$. The corresponding log-likelihood function of a Gumbel-distributed sample of size k is:

$$\log L(\mu_m, \sigma_m) = -\sum_{i=1}^{k} \left(\frac{X_i - \mu_m}{\sigma_m} + \exp(-\frac{X_i - \mu_m}{\sigma_m}) + \log \sigma_m \right) \tag{4.14}$$

Both $\hat{\mu_m}$ and $\hat{\sigma_m}$ can be obtained by the maximization of (4.14). A confidence interval for the estimate (4.13) (corresponding to a confidence level of $(1-\delta) * 100\%$) is given by:

$$| \hat{\omega}(F) - \omega(F) | \leqslant \frac{z_{\delta/2}}{\sqrt{k}} \frac{\hat{\sigma_n}\sqrt{6}}{\pi} \cdot \sqrt{(\gamma - 1)^2 + \frac{\pi^2}{6} + \frac{2(1-\gamma)}{s_m} + \frac{1}{s_m^2}} \tag{4.15}$$

where $s_m = 1 + m\sqrt{\pi \log m}(\mathrm{erf}(\sqrt{\log m}) - 1)$, $z_{\delta/2}$ is the δ/2-quantile point of the standard normal distribution $N(0,1)$, and $\gamma \simeq 0.5772\ldots$ is the *Euler gamma* constant.

In order to complete the discussion about the upper end point estimation, we have to show how to determine which of the two cases takes place, $\xi > 0$ or $\xi \downarrow 0$, given a sample $\mathbf{X}_n$, as the parent cdf $F(x)$ is, most likely, unknown. To this end, authors in [80] propose a test statistic $T(\mathbf{X})$ and a rejection region C_α, which corresponds to a given significance level α, to question the null hypothesis $H_0 : \xi \downarrow 0$ against the alternative hypothesis $H_1 : \xi > 0$. If $T(\mathbf{X})$ falls into C_α, then H_0 is rejected at this particular significance level. Given the cdf $F_T(x)$ of $T(\mathbf{X})$ under the null hypothesis, the critical region takes the form $C_\alpha = \{\mathbf{X} : T(\mathbf{X}) \geq F_T^{-1}(1-\alpha)\}$ so that the probability of $T(\mathbf{X})$ falling into C_α is equal to α. The test statistic presented below, when evaluated on the sample of exceedances (consisting of r units), tends asymptotically (as $r \to \infty$) to a normal distribution under the null hypothesis H_0:

$$T_{\mathbf{X}_{ex}} = \frac{\frac{(m_{\mathbf{X}_{ex}} - u)^2}{s^2_{\mathbf{X}_{ex}}}}{\frac{2}{\sqrt{r}}} \sim N(0,1) \tag{4.16}$$

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

where $m_{\mathbf{X}_{ex}}$ and $s_{\mathbf{X}_{ex}}$ are the mean and standard deviation of $\mathbf{X}_{ex}$, respectively. Accordingly, the critical region for an one-tailed test [81] is:

$$C_\alpha = \mathbf{X}_{ex} : T(\mathbf{X}_{ex}) \geq z_\alpha \tag{4.17}$$

where z_α is the α quantile point of the standard normal distribution $\sim N(0,1)$.

4.3 Background on Worst-Case Delay Analysis

The worst-case condition for one stage, when studied in isolation of the rest of the circuit, may provide the best-case conditions for the previous stage. In this section, we demonstrate why it is infeasible to extract a global worst-case delay condition apriori.

4.3.1 Impact of design parameter variations on maximum delay

The delay of a path (D) can be calculated by adding all the individual delays corresponding to gates and interconnects on the path. More specifically, the delay of the i-th gate on a path (d_i) is proportional to the total driving capacitance C_L, which is given by:

$$C_L = C_w + \sum_{j \epsilon \text{fanout}(i)} C_j(W_j),\ C_j \propto W_j \tag{4.18}$$

where C_w corresponds to the total wire capacitance seen by the i-th gate, W_j denotes the effective transistor width of each fanout gate j, and C_j corresponds to the input pin capacitance of each fanout gate j. It is very important to note that C_j is inversely proportional to the drain-source I_{ds} current of the driving transistors ($I_{ds} \propto (W, 1/L, f(V_{th}))$, where L is the corresponding effective channel length, V_{th} is the threshold voltage, and $f(V_{th})$ is a strictly decreasing function for the V_{th} range of interest) [82].

Under process variation, parameters that characterize the electrical behavior (W, L, V_{th}) of the transistor deviate from the nominal values, in some cases substantially, and thus we have to treat them as random variables distributed between a lower and an upper bound, when analyzing circuit performance. A sound assumption made in other similar studies about the distribution of W, L, and V_{th} is that are normally distributed between $\pm 3\sigma$. Hence, we consider $W\ \epsilon\ [W_0 - 3 * \sigma_w, W_0 + 3 * \sigma_w]$, $L\ \epsilon\ [L_0 - 3 * \sigma_L, L_0 + 3 * \sigma_L]$, and $V_{th}\ \epsilon\ [V_{th0} - 3 * \sigma_{Vth}, V_{th0} + 3 * \sigma_{Vth}]$ as normal random variables with mean values W_0, L_0, and V_{th0} and standard deviations σ_w, σ_L, and σ_{Vth}, respectively.

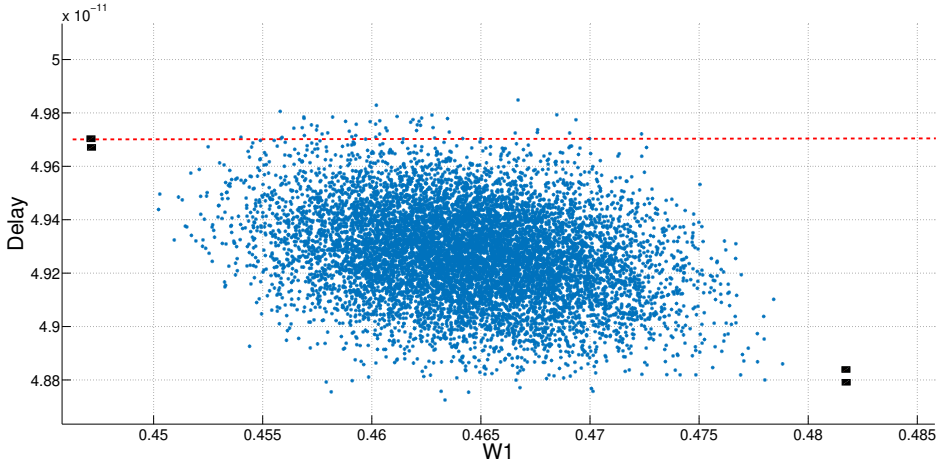
Therefore, when the effective width of a gate on a path is increased, the driving current is increased, and as a result, its delay is decreased. On the other hand, the delay of the previous gate is increased because it bears a higher load. Also, σ_{Vth} is inversely proportional to the square root of W and L ($\propto 1/\sqrt{WL}$), according to Pelgrom's rule [83]. Thus, when L is increased, $\sigma_{V_{th}}$ and I_{ds} are decreased, which indicates that the delay of the gate increases and the worst-case delay condition coming from V_{th} ($V_{th0} + 3 * \sigma_{Vth}$) becomes smaller at the same time.

Note that similar arguments hold for interconnects. Interconnect capacitance and resistance are a function of parameters that are determined by the interconnect structure such as width, thickness, and spacing, which in turn can be considered as bounded random variables normally distributed. For example, Fig. 3 in [48] highlights that the condition of the maximum interconnect capacitance is opposite to the one of the maximum resistance and RC constant, an observation aligned to the previous discussion about transistor parameters.

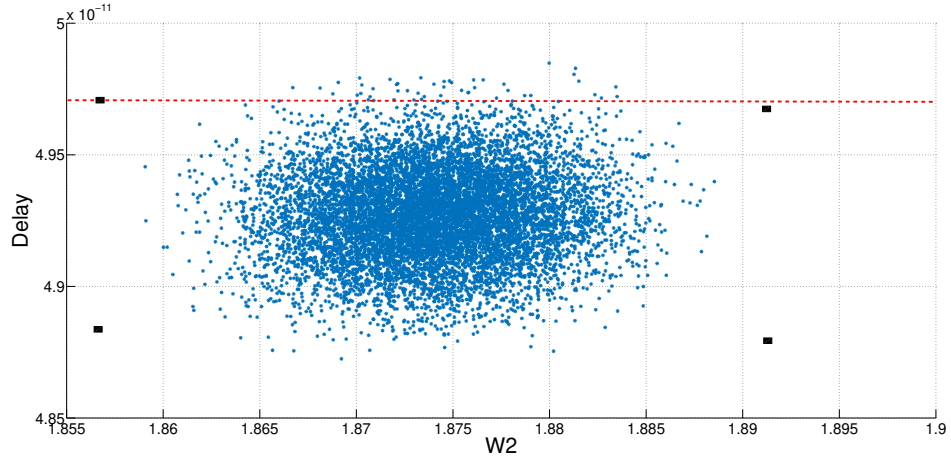
Fig. 4.1 shows the maximum delay of a path consisting of two inverters in a row, taken from the memory decoder assumed by Cacti [84], where $\sigma_{w_1} = 5\%$ and $\sigma_{w_2} = 5\%$ of minimum transistor size. Notice that the maximum delay is located somewhere between the

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

corners of $\mathbf{W} = \{W_1, W_2\}$, and thus it cannot be determined in advance. Even if the worst delay was located at the corners of W space, a corner-based analysis would be prohibitive since the number of all possible process parameters combinations grows exponentially with the circuit size.



(A) Projection of the 3D scatter plot of a two-inverters path delay on the W1 axis.



(B) Projection of the 3D scatter plot of a two-inverters path delay on the W2 axis.

FIGURE 4.1: Scatter plot of a two-inverters path delay derived from an MC simulator for 10,000 different (W1, W2) pairs. The delays at (W1, W2) corners are highlighted with black squares. Notice that there are a lot of points over the red-dotted line that indicates the maximum delay from the set of delays corresponding to $(W1, W2)$ corners, namely $\{(W1_{min}, W2_{min}), (W1_{min}, W2_{max}), (W1_{max}, W2_{min}), (W1_{max}, W2_{max})\}$.

4.4 Proposed Approach

Following the discussion of the previous section, this work focuses on finding the maximum (upper end point) of a bounded random variable on the basis of a random sample. This problem has been tackled successfully in [80] (for maximum supply current estimation). In this section, we review the previous methodology and comment on how it can be applied on the problem of maximum delay estimation. In addition, we share some initial thoughts for further enhancements of the proposed methodology.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

The first step of the methodology entails an MC simulation of n trials to acquire the statistical sample of ATs at the POs ($\mathbf{AT}_n = \{AT_1, AT_2, \ldots, AT_n\}$), where each MC trial is performed with a random $\mathbf{P} = \{P_1, P_2, \ldots, P_{\#dp}\}$ sample (P_i is assigned to the i_{th} design parameter, which in turn belongs to either a gate or interconnect) drawn from the joint probability $F_{\mathbf{P}} \sim (P_1, P_2, \ldots, P_{\#dp})$. The second step is to sort the units of the AT sample in ascending order and pick those unit samples that exceed a threshold u (makes the sample of exceedances $\mathbf{X}_{ex}$). The threshold u has to be selected so that the sample of exceedances reside in the tail of the AT distribution. Therefore, the units of $\mathbf{X}_{ex}$ have to be as many as required to perfectly approximate the parameters of GP pdf, which is the limiting distribution of the sample of exceedances. On the other hand, when a very small threshold u is selected in order to include more sample units of the initial AT sample, then there is the peril of including units that do not belong in the tail of AT distribution. A rule of thumb is to set the threshold u so that the upper 10% of the initial AT sample is included to $\mathbf{X}_{ex}$. More rigorous procedures for automatic selection of u can be found in [85] and [86]. Based on the $\mathbf{X}_{ex}$ sample, we calculate the test statistic T of (4.16) and compare it with the critical value of z_α (for significance level α) to decide whether we accept or reject the hypothesis $\xi \downarrow$. Experiments performed on various random samples drawn from an Exponential distribution have resulted to a critical value of $z_\alpha \simeq 7$, which corresponds to a significance level of $\alpha = 10^{-12}$ (detailed proof of the previous result can be found in Appendix C of [80]). Subsequently, in case the hypothesis $\xi \downarrow$ is true, we divide the initial sample of AT into l blocks of size m sub-samples and pick the maximum unit from each sub-sample forming the sample of maxima. Then, we calculate the ML estimates $\hat{\sigma}$ and $\hat{\mu}$ of the corresponding Gumbel parameters σ, μ as the solution given from the maximization of (4.14) over the sample of maxima, and finally, we determine the maximum AT for the previous estimates and provide the confidence interval for the chosen confidence level $1-\delta$ from (4.13) and (4.15), respectively. On the other hand, if the results of test statistic T indicate that we have to reject the hypothesis $\xi \downarrow$ ($T \geq 7$), we then find the ML estimates $\hat{\sigma}$ and $\hat{\beta}$ of the corresponding Beta parameters σ, β as the solution given from the maximization of (4.11) over the sample of exceedances, and finally, we determine the maximum AT for the previous estimates and provide the confidence interval for the chosen confidence level $1-\delta$ from (4.10) and (4.12), respectively. Algorithm 6 summarizes the steps of the proposed methodology for maximum AT estimation.

Algorithm 6: Statistical method for worst-case delay estimation under process variation

```
Generate n P samples and assign them to the corresponding gates and interconnects
Perform an MC simulation with n trials to acquire the statistical sample of ATs at the
  POs (AT_n = {AT_1, AT_2, ..., AT_n})
Sort the units of AT sample in ascending order and pick the units that are over a
  threshold u (sample of exceedances)
Evaluate the test statistic T of (4.16) for the sample of exceedances acquired in the
  previous step
if (T < 7) then
    Derive the sample of maxima from the sample of ATs
    Estimate the Gumbel parameters by maximizing (4.14) over the sample of maxima
    Determine the maximum AT estimate using (4.13) and the confidence interval for
      the chosen confidence level 1 − δ using (4.15)
else
    Estimate the Beta parameters by maximizing (4.11) over the sample of exceedances
    Determine the maximum AT estimate using (4.10) and the confidence interval for
      the chosen confidence level 1 − δ using (4.12)
end
```

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

4.4.1 Machine learning enhancements

The proposed methodology may be enhanced with machine learning techniques to determine the size n of MC simulations, which dominantly affects the time complexity of the method, as well as the threshold u. For example, we can acquire the sample of exceedances at step 3 by exploiting the Statistical Blockade algorithm [87]. This algorithm builds a classifier based on an initial input variables training set of size $n_0 << n$ (in our methodology the **P** sample) and a sample of the corresponding metric of interest (in our case the AT sample), derived after the simulation of the underlying system (in our case SSTA), to test whether a unit sample of the metric would fall within the tail of the parent distribution for every new unit sample of input variables (**P**), beyond the first n_0 samples, without performing simulation. By doing so, for a good classifier, we acquire a sample of exceedances that falls within the tail of the parent distribution with greater probability than simply selecting the upper 10% of the initial AT sample, thus speeding up the methodology while also improving accuracy. Note that in the $\xi \downarrow$ case, we cannot rely on the sample of exceedances. For this reason, we keep track of the **P**s rejected by the Statistical Blockade algorithm, we run the missing simulations, and then we follow the rest of the steps corresponding to the $\xi \downarrow$ case. Again, since we would have rejected many more P samples than those leading to AT samples in the tail, we would also achieve both greater accuracy and speedup, as n becomes a function of u.

4.5 Experimental Evaluation

For the experimental evaluation, we developed an SSTA tool in C++, based on MC simulations, to gather the statistical sample of ATs at different POs. Our SSTA tool embeds models that allow us to variate only the effective transistor width of a gate (W). However, this does not prevent us from deriving sound conclusions about the effectiveness of the proposed methodology, as the methodology does not depend on the number of design parameters that affect timing but on the statistical sample at the outputs. For timing verification where the worst slack is required, the extension is trivial by substituting the ATs sample in the proposed methodology with the SLACKs sample, as the i.i.d. assumption is not violated for this sample. We applied our methodology on a subset of the ISCAS85/89 [88–90] benchmarks implemented in 45 nm process technology using the NanGate FreePDK library [74]. The statistical timing models used by our SSTA tool are provided by the TAU 2013 variation-aware timing analysis contest [91]. For the evaluation, we used a Linux workstation with a 3.60 GHz Intel® Core™ i7-4790 CPU (4 cores, 8 threads) and 16 GB memory. Below we summarize the steps that we followed to run the experiments:

- We run SSTA with n trials to obtain the statistical sample of ATs at each PO ($\mathbf{AT}_k = \{AT_{k,1}, AT_{k,2}, \ldots, AT_{k,n}\}$, where k denotes the k-th PO) of the circuit under test (steps 1-2 of Algorithm 6).
- We apply steps 3-12 of Algorithm 6 on $\mathbf{AT}_k$ to get an estimate of maximum AT_k ($\hat{AT}_k$) and the corresponding confidence interval $\mid \hat{AT}_k - AT_k \mid$.
- Finally, we evaluate the current relative error $\left(\text{RE} = \dfrac{\mid \hat{AT}_k - AT_k \mid}{\hat{AT}_k}\right)$ of the worst AT_k estimate.
- If RE is below a predefined target relative error ($\text{RE}_{target} = 5\%$), we acquire more samples using the SSTA tool and re-apply our methodology until the desired accuracy is achieved.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

The generation of $\mathbf{W}_{\#gates}$ samples in step 1 of our methodology can be done very fast by randomly picking a corner W_k for each gate. The accuracy of the estimated ATs in step 2 is entirely up to the timing analysis tool and varies from SPICE-level to gate-level. Also, the computational time required to complete step 2 depends on the efficiency of the timing engine employed. However, the runtime of steps 3-12 is very small compared to the runtime of MC simulation, and thus each $\mathbf{AT_k}$ sample can be re-arranged into sub-samples of various sizes in order to obtain better estimates.

The results from the execution of the above steps on three sequential (s27, s35932, s38417) and three combinational (c17, c6288, c7552) designs are reported below. In this experiment, each MC trial is performed with a random $\mathbf{W}_{\#gates} = \{W_1, W_2, \ldots, W_{\#gates}\}$ sample, where each W is normally distributed between $\pm 3\sigma$. Table 4.1 shows the maximum of the **AT** sample and the estimated $\hat{AT}$ (denoted by "EVT max") at selected POs for each circuit under consideration, when RE_{target} is within 5% for 99% confidence level. Additionally, it reports the **AT** sample size and the achieved RE.

It is worth pointing out that the number of random samples required to meet this relative estimation error is slightly increased for POs that belong on paths of higher impedance interconnects compared to those including low impedance global interconnects (e.g., clock networks) routed on the upper-level metal layers, where the resistance is significantly smaller. The resulting sample sizes for the above benchmarks, using both high and low impedance interconnects, are presented in Table 4.2. In the former case, we extracted detailed *RC* parasitics in a Standard Parasitic Exchange Format (SPEF) file, while in the latter case, we assumed a zero wire delay model. Note that in both cases the estimated AT at the POs is within 5% of the true maximum AT at a cost of a few thousand MC simulations.

TABLE 4.1: Required sample size, sub-sample size, sample maximum AT, and estimated maximum AT on a subset of POs, targeting a relative estimation error within 5% for 99% confidence level

Benchmark	Primary Output	Sample Size	Sub-Sample Size	Sample max (ns)	EVT max (ns)	Relative Error
s27	G17	2500	25	0.265	0.270	0.47%
s35932	DATA_9_0	5000	25	0.513	0.694	4.25%
	DATA_9_19	5000	25	0.518	0.702	4.26%
s38417	g16297	2500	50	0.132	0.146	2.99%
	g25420	10000	25	0.470	0.820	0.75%
c17	N22	2500	25	0.053	0.054	0.44%
	N23	2500	25	0.051	0.052	0.07%
c6288	N5971	10000	25	2.304	2.691	1.62%
	N6280	5000	25	3.428	3.940	2.11%
c7552	N10718	5000	25	0.917	0.940	0.42%
	N10729	5000	25	0.671	0.809	2.73%

To evaluate the efficiency of our statistical method, we compare it against an exhaustive MC simulation, which corresponds to the slowest but most accurate analysis (a.k.a. full factorial design), for a test benchmark. In the exhaustive approach, we have to simulate all the combinations of gate widths, while when we apply our method, we choose n random samples of them. If we allow W to take a value within a continuous range, then the number of all combinations is not bounded. For this reason, each gate width W is set on its $\{-3\sigma, +3\sigma\}$ corner. As a result, the maximum AT derived by exhaustive MC simulation is a lower bound of the actual maximum AT, and thus the reported RE is an upper bound of the actual one.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

TABLE 4.2: Sample size required in order to achieve 5% relative estimation error for 99% confidence level. Higher impedance interconnects require more MC simulations than lower impedance interconnects

Benchmark	Sample Size (with interconnect)	Sample Size (w/o interconnect)
s27	2500	2500
s35932	5000	2500
s38417	10000	2500
c17	2500	2500
c6288	10000	5000
c7552	5000	5000

In order to perform this experiment, we pick a design with 30 gates where the exhaustive MC is feasible, since even for small designs of 100 gates the runtime would be tremendous (2^{100} MC trials). Due to the lack of availability of such a well-known circuit, we implemented a synthetic benchmark, based on the ISCAS c432 benchmark, which consists of 30 gates.

We first demonstrate the accuracy of our methodology and report RE on a single selected PO (N223) of the test design. In Table 4.3, we present how our methodology approaches the theoretical maximum AT at N223. From the table, we can clearly observe that the more samples we pick to apply our method, the better RE is achieved.

Note that the reported RE stands for the upper bound of the computed confidence interval, and thus for worst-case timing analysis, we have to add this error to the estimated maximum AT at the PO.

TABLE 4.3: Relative error on N223 for different sample/sub-sample sizes and 99% confidence level

Sample Size	Sub-Sample Size	Sample max (ns)	EVT max (ns)	Relative Error
2500	50	0.16784	0.16997	0.540%
5000	25	0.16833	0.16987	0.289%
10000	50	0.16811	0.16956	0.235%
20000	50	0.16814	0.16818	0.049%
50000	25	0.16814	0.16972	0.088%
100K	25	0.16848	0.16971	0.062%
1M	50	0.16848	0.16890	0.016%
10M	50	0.16848	0.16953	0.007%
100M	25	0.16848	0.17068	0.002%

Fig. 4.2 depicts the probability distribution at node N223 after MC simulation of 10 million trials. The following distribution is a representative one for all POs in all examined benchmarks and corresponds to the case $\xi \downarrow$ in the proposed methodology. Also, following the previous observation, another point to mention is that in all experiments, we find that a number of 50 sub-samples in block maxima modeling yields estimates with RE within 5% (at a confidence level of 99%) for any PO irrespective of the number of stages in the corresponding path.

The runtime efficiency of the proposed method is reported in Table 4.4. For this comparison, we executed the previous experiment for N223 and measured the runtime (denoted by "EVT runtime"). We conclude that our method achieves up to six orders of magnitude (*233426*×) better performance compared to an exhaustive MC simulation. In case we target

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

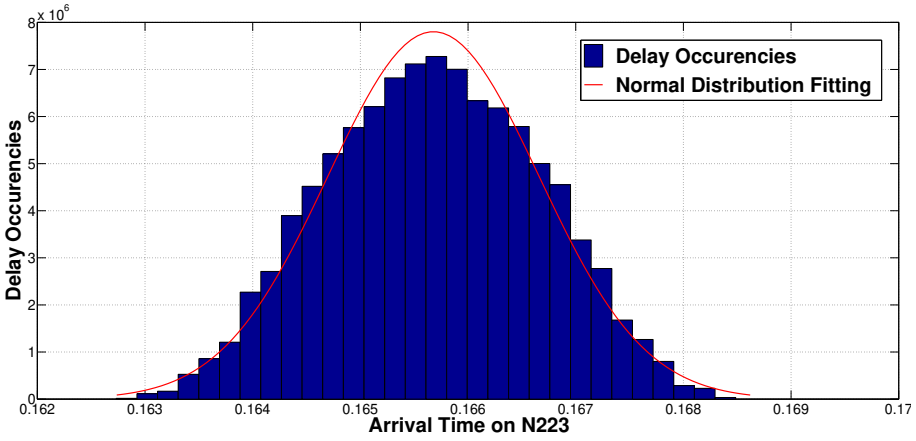


FIGURE 4.2: Probability distribution of AT on N223 after an MC simulation of 10M trials.

a 5% RE_{target} for all POs, the proposed method is five orders of magnitude ($58355\times$) faster than an exhaustive MC, given that the maximum required sample size is 10000 (as shown in Table 4.2).

TABLE 4.4: Runtime comparison between the proposed methodology and an exhaustive MC simulation

Sample Size	Exhaustive MC Runtime (s)	EVT Runtime (s)	Speedup
2500	14472	0.062	233420×
5000	14472	0.124	116710×
10000	14472	0.248	58355×
20000	14472	0.496	29177×
50000	14472	1.239	11680×
100K	14472	2.479	5838×
1M	14472	24.898	582×
10M	14472	248.527	59×
100M	14472	2478.140	5.85×

4.6 Chapter Summary

In this chapter, we presented a novel statistical methodology based on MC simulation and EVT, as a substitute for the conventional corner-based and the traditional statistical analysis, which can provide accurate estimates of the worst ATs at the POs, taking into account process variation. Our methodology does not make any assumption about the gate or interconnect timing model or the distribution of AT at the POs, and thus, it can be used from transistor-level to gate-level abstraction in timing verification. Experimental results demonstrate that we can achieve an RE between the true maximum and the estimated maximum that is below 5% with only a few MC trials.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Chapter 5

Dynamic Timing Analysis Using Event-Driven Simulation

5.1 Introduction

The advent of the aggressive technology scaling era has introduced extensive Process, Voltage, Temperature, and Aging (PVTA) variations, which may result in up to 50% delay variability [26, 32]. As a result, this trend renders circuits prone to timing failures and prevents them from meeting the desired performance specifications.

Conventionally, manufacturers address such failures by adopting timing guardbands, which essentially provide sufficient timing margins to account for any PVTA variation-induced delay increase [32]. However, such margins are considered to be overly pessimistic, since they are estimated statically based on rare operating conditions. In fact, conventional Static Timing Analysis (STA) [3] may be effective in quickly revealing the most critical paths but assumes worst-case inputs at each circuit node, thus ignoring the data-dependent excitation of each path [32, 92]. This approach ultimately forces circuits to operate at a much lower frequency or at a higher supply voltage than what they could potentially achieve [26].

To circumvent such overheads, recent schemes try to reveal any Dynamic Timing Slack (DTS) that may exist in activated paths by applying Dynamic Timing Analysis (DTA) [33, 35–39]. On one side, many of these works [35, 36, 39] try to exploit the available DTS for upscaling the frequency [35, 39], changing the cycle time [36], or reducing the supply voltage [38]. On the other side, other works exploit DTA in order to estimate timing failure rates while considering the processed data [33, 37].

A common characteristic of the above works is that the applied DTA is based on delay-annotated gate-level simulation tools (e.g., Siemens ModelSim™ [40]). Although such tools account for the paths that are dynamically activated, they still assume fixed, worst-case delays which are estimated by Graph-Based Analysis (GBA). In fact, GBA provides a pessimistic delay estimation that is based on STA assumptions [3]. None of the above works evaluate the impact of GBA on DTS estimation. A few recent works [38, 41] focus on improving the runtime of Graph-Based DTA (GB-DTA) rather than improving the DTA accuracy. This means that existing approaches may be leaving a large amount of timing slacks hidden and unexploited.

In this chapter, we primarily aim at unveiling the unexploited DTS ignored by all previous approaches based on GB-DTA, by taking into consideration the actual data-dependent path delays. The main contributions of this work can be summarized as follows:

- We develop a framework to unveil DTS that has been underestimated by prior works. To achieve this, we implemented a tool that is based on Event-Driven DTA (ED-DTA), which has been established as the most accurate DTA method [93].

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

- We compare the DTS computed by ED-DTA with the DTS estimated by a GB-DTA method which is based on delay-annotated gate-level simulation. Our results indicate that ED-DTA leads to 2.35% on average and up to 194.51% more DTS compared to GB-DTA, in terms of relative difference. Considering only the critical activated paths, the average DTS improvement is increased to 11.2%.
- We demonstrate the impact of underestimated DTS on clock frequency and timing failures. First, we measure the Point of First Failure (PoFF) between ED-DTA and GB-DTA, indicating the frequency improvement. Second, we estimate the timing errors manifested under potential variation-induced worst-case delay increase levels for ED-DTA and GB-DTA.

The rest of this chapter is organized as follows. Section 5.2 discusses the background and the motivation of our work, while Section 5.3 describes the proposed approach and the implemented workflow. Section 5.4 presents the experimental results. Conclusions are drawn in Section 5.5.

5.2 Background

5.2.1 Static timing analysis

Although several aspects of timing modeling and analysis (including gate/interconnect delay calculation and SSTA) have been already discussed in the previous chapters, in this section we further explain the timing characteristics and graph representation of a circuit, and we provide a more detailed description of STA to support this chapter.

Typically, a digital circuit consists of circuit elements (i.e., gates or macros) connected with interconnects and can be represented as a graph composed of nodes (i.e., primary input/output ports and gate pins) and edges (i.e., timing arcs) that are connecting these nodes. Each distinctive directed connection of timing arcs, thus nodes, forms a *timing path*. Each of the numerous timing paths require some time to propagate the signal, which essentially defines the delay of each path. This path delay (D) is computed by adding up the delays (d_i) of all timing arcs included in the path. Note that each timing arc delay is a function of input $slew$, which is defined as the amount of time required for the signal to transition from high-to-low or low-to-high. The input $slew$ on a timing arc also determines the $slew$ on the output node of the arc. The delay and output slew of each specific gate or interconnect timing arc may be calculated using our proposed approaches described in Chapters 2, 3.

One of the most essential steps in the design of any circuit is the identification of the worst-case critical path and the estimation of its delay, which eventually determines the maximum operational frequency of the circuit. Such a step requires to carry out STA, which is typically done using an early-late split, where each path has an early (lower) bound and a late (upper) bound on its delay to account for various parametric variations [3]. STA propagates the upper and lower slew bounds on the nodes through the timing arcs. Based on that, it calculates the earliest and the latest timing instants that a signal reaches a circuit node. These timings are quantified as earliest and latest *arrival time* (at), while the limits imposed on a circuit node for proper logic operation are quantified as earliest and latest *required arrival time*(rat). The difference between the required arrival time and signal arrival time at a circuit node determines the $slack$, which quantifies how well the timing constraints are met. That is, a positive slack means the required time is satisfied, and a negative slack means the required time is in violation. The rat and $slack$ are defined in the following equations, taking into consideration the setup constraint t_{setup} which denotes the amount of time that the signal should be stable before the active clock edge on every Flip-Flop (FF) to ensure proper operation:

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

$$rat_{setup} = rat_D^{late} = at_{CK}^{early} + T_{clk} - t_{setup} \tag{5.1}$$

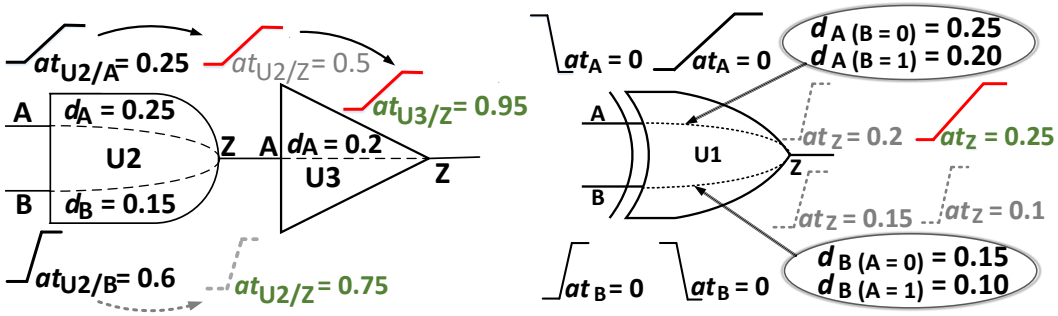
$$slack_{setup} = slack_D^{late} = rat_D^{late} - at_D^{late} \tag{5.2}$$

where T_{clk} is the clock period, and CK and D denote the clock and the data pin of the testing FF, respectively.

To provide a quick estimation of the worst setup timing slack among all paths, which is adequate for determining T_{clk}, GBA is applied which assumes worst-case (upper) arrival time and slew bounds on each node [3]. GBA which is essentially the default STA method used in commercial tools (e.g., Synopsys PrimeTime®), as well as in our in-house STA/SSTA tools mentioned in the previous chapters, may provide quick estimations but ignores the lower or intermediate arrival times and slew bounds. As a result, GBA may lead to an overly pessimistic slack estimation that is substantially different than the actual slack, which depends on input data.

5.2.2 Dynamic timing analysis

It has been recently estimated [33] that roughly 99% of activated critical paths are triggered by less than 10% of all operations and the chance to experience the worst-case input conditions and the upper slew bounds assumed by STA is very low [34]. These findings have turned the attention of many studies into the estimation of the so-called DTS that may exist within any path depending on the dynamically changing processed data [33,35–39]. Such works aim at revealing any unexploited room for occasional frequency up-scaling [35,36,39] or voltage down-scaling [38] and for estimating timing error rates in speculative processors [33,37]. The majority of such studies rely on frameworks that use delay-annotated gate-level simulation tools (e.g., ModelSim™ [40]). However, the annotated delays inherently impose significant pessimism in slack estimation, since they are extracted (in the form of a Standard Delay Format [SDF] file) from tools that are based on GBA (e.g., PrimeTime®). Therefore, although the developed frameworks can account for the data-dependent path activation, they may still underestimate the available DTS.



(A) Slew merge points (B) Worst-case input state assumptions

FIGURE 5.1: Main sources of pessimism in graph-based analysis.

To better understand the source of potential slack underestimation, let us take a better look at few shortcomings of GBA along with few examples. For simplicity, in the following examples, the delay of interconnect timing arcs is considered to be zero.

1) Slew merge points. When two *slew* values arrive at the same node of the graph, GBA propagates forward the worst *slew* and *at*. The most common example of a slew merge point is an output pin of a gate, where multiple timing arcs terminate. In Fig. 5.1 (A), we explain how the *late* timing information of an AND gate is propagated across the timing graph. In this example, we can see that the worst (max) *slew* (highlighted in red) and the worst (max)

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

at (highlighted in green) on pin $U3/A$ are propagated through different paths. As a result, the $delay$ from $U2/B$ to $U3/Z$ appears to be bigger than the real physical delay of the path, since the timing arc delay $d_{U3/A \rightarrow U3/Z}$ has been determined by the $slew$ of a different path.

2) Worst-case input state assumptions. Another major source of pessimism in GBA, is the worst-case input state assumptions. In more detail, some gates (e.g., XOR, XNOR, MUX) are state-dependent. This means that a specific transition (rise or fall) on the output pin of a gate may occur for different input transition combinations. Fig. 5.1 (B) demonstrates the $late$ timing propagation for an XOR gate. In this case, GBA propagates the worst timing information, resulting in even more pessimism in the delay estimation of upstream paths. Note that the worst propagated $slew$ (highlighted in red) at the output pin $U1/Z$ corresponds to the rise $slew$ propagated from $U1/A$ when $U1/B$ is at logic 0.

To demonstrate the inaccuracy involved in GB-DTA methodologies, let us give an example of DTS estimation for a specific activated path. In Fig. 5.2, GBA analysis has been performed on a timing graph, where the example gates of Fig. 5.1 have been combined together. For this initial logic state of the circuit, when the signal on $U1/A$ makes a falling transition ($1 \rightarrow 0$), the path from $U1/A$ to $FF2/D$ is activated. However, recall that the worst $slew$ on $U1/Z$ has been chosen for the rising ($0 \rightarrow 1$) transition of $U1/A$, as shown in Fig. 5.1 (B), which determines the worst-case annotated delays $d_{U2/A \rightarrow U2/Z}$ and $d_{U3/A \rightarrow U3/Z}$. Thus, $at^{late}_{FF2/D}$ is calculated based on those delays. Note that the $at^{early}_{FF2/CK}$ is assumed to be always equal to the worst-case value calculated in GBA, since the clock path usually consists only of series of inverters/buffers and does not include slew merge points. It is also important that, except from $at^{late}_{FF2/D}$, $rat^{late}_{FF2/D}$ induces additional pessimism in DTS estimation since the t_{setup} value is calculated based on the worst $slew$ propagated on $FF2/D$, during GBA. Assuming $at^{early}_{FF2/CK} = 0\,\text{ns}$, $T_{clk} = 1.2\,\text{ns}$, and $t_{setup} = 0.2\,\text{ns}$, GB-DTA estimates the DTS $slack^{late}_{FF2/D} = 1 - 0.65 = 0.35\,\text{ns}$, based on (5.1, 5.2), indicating that the slack is overly pessimistic compared to the path's real physical DTS. This indicates that there is a need to reveal the amount of slack that remains unexploited by existing GB-DTA frameworks [33, 35–39] and up to what extent it may affect PoFF, the dynamic frequency scaling, and the failure rate estimation.

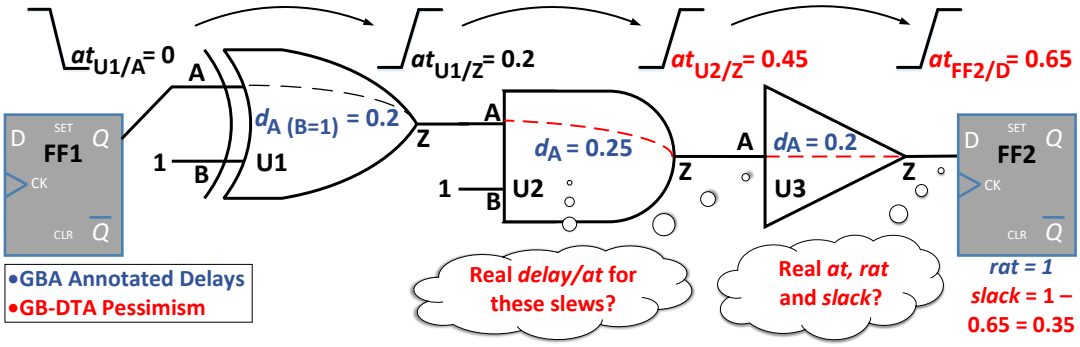


FIGURE 5.2: Graph-based analysis pessimism in GB-DTA.

5.3 Proposed Approach

In this section, we present our approach for unveiling DTS that may have been underestimated by GB-DTA methods used in prior works. To this end, we describe the principles and operation of our developed ED-DTA tool and its integration within a state-of-the-art flow used to evaluate DTS.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

5.3.1 Event-Driven DTA (ED-DTA)

Our approach is based on event-driven timing simulation, which is considered as the most accurate DTA method [93]. In the context of ED-DTA, a logic transition at a circuit node is modelled as an event. Therefore, every event is characterized by the logic value, the *slew*, and the *at* of the corresponding transition. This allows to track every single node excitation that may take place within any timing path of the circuit as opposed to graph-based methods. In ED-DTA, events are processed in the correct time order by building and traversing a time-sorted event list. During the dynamic simulation, the event with the smallest *at* is chosen to be propagated forward to its fanout nodes followed by the rest of the events.

Our event-driven approach uses both functional and timing models of the gates. An event arriving at an input pin of a gate causes the functional evaluation of the gate using its functional model. Whenever the logic value on the output pin of a gate changes, a new event is created and has to be placed in the appropriate point in the event list to ensure causality. For this purpose, the gate's timing model is used to calculate the timing characteristics (*slew*, *at*) for this output node and schedule the generated event, according to its *at*, for later processing.

The key idea behind the proposed method is that the *slew* and *at* of the generated event depend only on the triggering input event. On top of that, the implemented gate timing analysis does not make worst-case assumptions but takes into account the current logic state of all input-output pins. As described in Section 5.2, correct *slew* and *at* propagation along a timing path is crucial since it determines the delay of timing arcs and the *slew*, *at*, t_{setup}, *rat*, and *slack* on the path endpoint (i.e., primary output port or data pin of a sequential element).

To better understand the impact of our approach on the accuracy of DTS estimation, let us take a closer look on the example shown in Fig. 5.3, where we consider the same setup timing check as shown in Fig. 5.2. In our approach, when the signal in $U1/A$ makes a falling transition ($1 \rightarrow 0$), an event is created on $U1/A$ (Event 1) which triggers a rising transition ($0 \rightarrow 1$) on $U1/Z$ and the scheduling of the corresponding event (Event 2). In contrast to GBA, the real output *slew* is calculated and stored on the triggered event to be propagated forward. To this end, when ED-DTA processes the event on $U1/Z$, the real output *slew* on $U2/Z$ and timing arc delay $d_{U2/A \rightarrow U2/Z}$ are calculated, to schedule the event on $U2/Z$ (Event 3) with its accurate timing characteristics (*slew* and *at*). Following the described event propagation procedure, an event reaches the path endpoint $FF2/D$ (Event 4) at $at_{FF2/D} = 0.48$ ns. Assuming that the propagated $slew_{FF2/D}$ (which is smaller than the worst *slew* propagated by GBA) now results in $t_{setup} = 0.1$ ns, and that $at_{FF2/CK} = 0$ ns and $T_{clk} = 1.2$ ns, ED-DTA estimates the correct DTS $slack_{FF2/D} = 1.1 - 0.48 = 0.62$ ns, based on (5.1, 5.2), which is significantly bigger than the worst-case value estimated by GB-DTA.

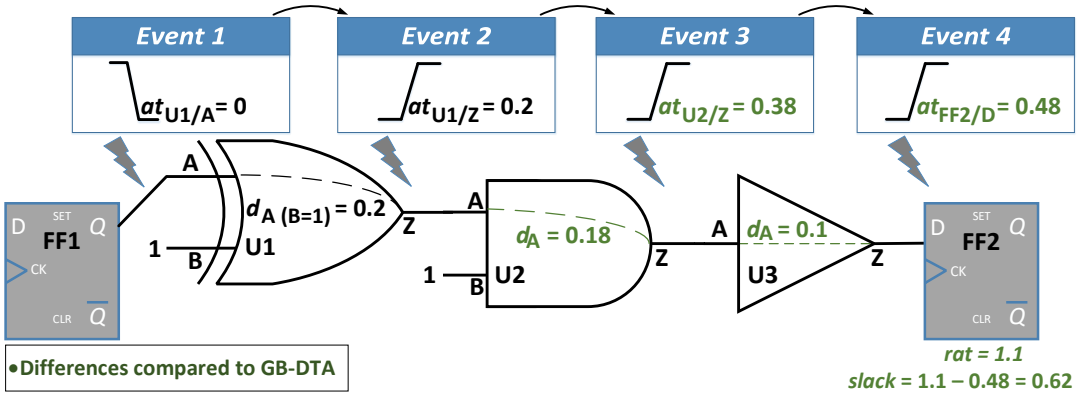


FIGURE 5.3: Accurate DTA using event-driven simulation.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

The developed ED-DTA tool operates on the gate level and represents the circuit with a timing graph, as described before. In our implementation, an event is characterized by the node on which the transition occurs, the *slew*, the *at*, and the logic value of the transition. Note that the timing and functional models of the gates are retrieved from the standard cell library which was used to implement the design.

The details of our ED-DTA tool are described in Algorithm 7. The tool takes the gate-level netlist and the signal activity information (in the form of a Value Change Dump [VCD] file) and performs both setup and hold timing checks. After creating the timing graph, it reads the VCD file to schedule input-vector events in the event-list and initializes the circuit to its steady state. During event-driven simulation, the events with the smallest *at* are propagated through the timing arcs to their fanout nodes. If the fanout node is a path endpoint, the tool computes the setup and hold DTS, and an error is reported in case a violation occurs.

Algorithm 7: Event-Driven DTA (ED-DTA)

```
1  Read the gate-level netlist and create the timing graph
2  Load the VCD file and schedule input events in the event list
3  Initialize the circuit to its steady state
4  while event_list_not_empty do
5      Propagate events scheduled on current_time to their fanout nodes
6      if (fanout node is a path endpoint) then
7          Calculate DTS and perform setup/hold timing check
8      else
9          Evaluate logic on the output node of the fanout gate
10         if (output node logic value has changed) then
11             Calculate output slew and at, and schedule a new event in the event list
12     end
13     Remove obsolete events from the event list
14     Advance current_time
15 end
```

5.3.2 Realization of the proposed approach

The ED-DTA tool described above is integrated within a state-of-the-art Electronic Design Automation (EDA) workflow along with the traditional graph-based DTA and STA methods. The overall workflow is depicted in Fig. 5.4, where our modifications compared to the conventional EDA flow are highlighted in orange. As can be seen, our workflow consists of a design and an analysis phase.

The first step of the design phase is logic synthesis which is followed by the Place and Route (PnR) steps. In this work, each design is synthesized, placed, and routed with the NanGate 45nm standard cell library [74], using the Synopsys Design Compiler® [94] and Cadence Innovus™ [95] tools, respectively. Note that these steps are performed utilizing optimizations that aim at achieving maximum performance.

The first step of the analysis phase is to verify that the design has met the timing closure. To this end, we developed a Graph-Based STA (GB-STA) tool which provides accurate results compared to commercial tools. The main reason for using our in-house GB-STA tool, instead of a commercial tool, is to ensure that all the examined timing analysis techniques rely on the same delay calculation engine. Both GB-STA and ED-DTA tools implement Algorithm 1 using the Synopsys Composite Current Source (CCS) [5] timing model for gate delay and output slew estimation. On the interconnect side, both methods adopt the Elmore

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

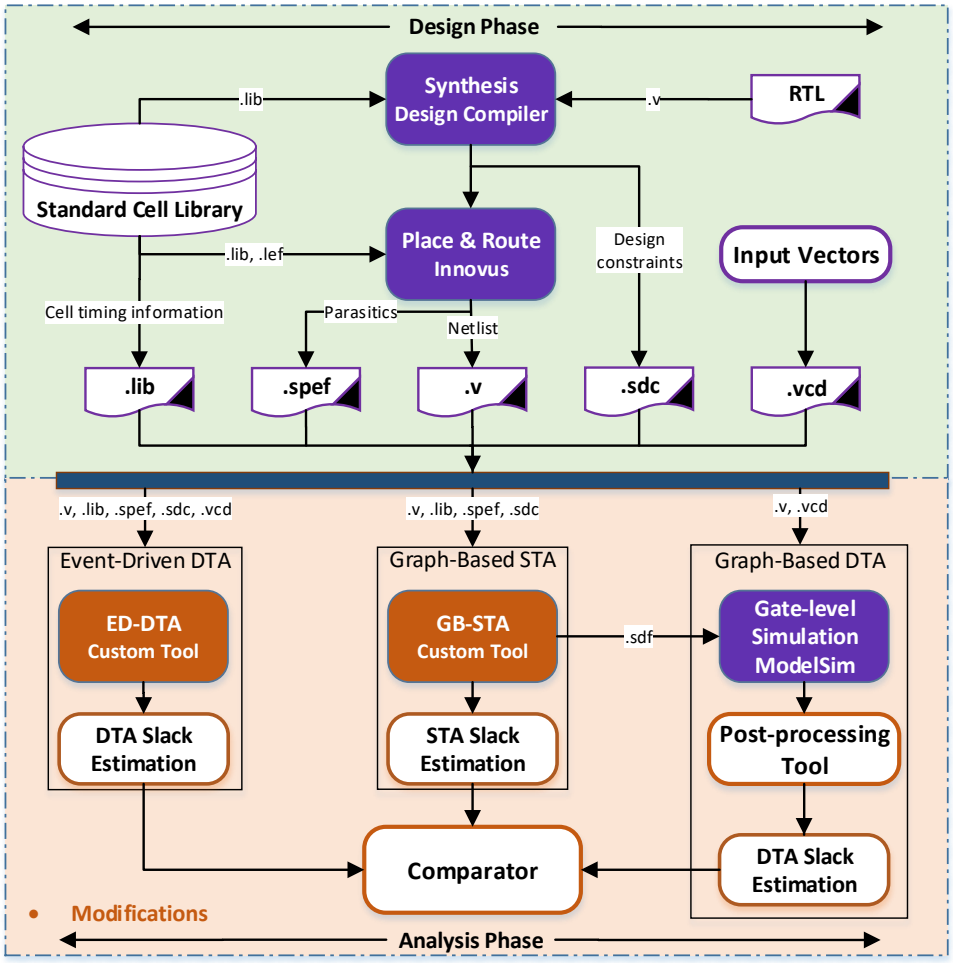


FIGURE 5.4: Workflow of the proposed approach.

delay model [18] and the closed-form slew metric presented in Chapter 2. Note that wire delay and slew estimation could be improved by employing our accurate interconnect timing analysis method presented in Chapter 3.

The GB-STA tool also provides the capability to annotate the worst-case wire and gate delays in an SDF file, which is essential for GB-DTA. For the sake of this analysis, we use post-layout gate-level simulation supported by ModelSim™. To enable characterization of DTS, ModelSim™ outputs a VCD file that contains information about the value changes occurred during simulation. More specifically, in the VCD file, we monitor the clock primary input port, primary output ports, and inputs (data and clock) of all FFs. This file is then provided to a custom post-processing tool for DTS estimation. For each path endpoint, this tool identifies the arrival time of the last event in each clock cycle and relates it to the arrival time of the next active clock edge, to estimate the DTS using (5.1, 5.2) for the setup timing check. Similar computations are also performed for the hold timing check. At the final step of the analysis phase, we compare the DTS measured by ED-DTA with the one estimated by GB-DTA.

To verify that the ED-DTA tool implements the functionality of the specification correctly, we compare the logic values on each primary output port extracted by the ED-DTA tool against ModelSim™.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

TABLE 5.1: Static Timing Slack (STS), Dynamic Timing Slack (DTS), and DTS improvement achieved by ED-DTA for various benchmarks

Benchmark	#gates	T_{clk} (ns)	STS_{GB} (ns)		DTS_{GB} (ns)		DTS_{ED} (ns)		AE_{DTA} (ns)		DTS Improvement ($DTS_{imp.}$)	
			min	mean	min	mean	min	mean	mean	max	mean	max
s344	91	0.34	0.008	0.085	0.042	0.199	0.046	0.203	0.002	0.011	1.19%	20.53%
fir	146	0.7	0.001	0.131	0.014	0.284	0.017	0.289	0.003	0.011	1.11%	27.73%
brent.kung.32b	319	0.38	0.003	0.046	0.003	0.122	0.005	0.127	0.003	0.014	3.15%	194.51%
kogge.stone.32	557	0.34	0.003	0.022	0.003	0.073	0.005	0.079	0.004	0.015	5.41%	86.27%
sobel	564	0.93	0.003	0.249	0.043	0.546	0.064	0.556	0.008	0.036	1.57%	43.55%
c6288	2488	2.94	0.006	0.608	0.615	1.974	0.724	2.024	0.049	0.158	2.81%	20.53%
multiplier.32b	6343	0.96	0.007	0.104	0.043	0.443	0.061	0.455	0.011	0.043	2.43%	38.11%
neural.network	13236	1.14	0.001	0.486	0.137	0.785	0.144	0.794	0.008	0.035	1.11%	13.34%

5.4 Experimental Evaluation

In this section, we evaluate our methodology for revealing the overly pessimistic estimation of DTS, existing in GB-DTA methods. First, we compare the setup DTS estimated by ED-DTA (DTS_{ED}) with the DTS extracted by conventional GB-DTA (DTS_{GB}). Then, we explore potential gains of ED-DTA by exploiting the available DTS. For such an analysis, we used the c6288 and s344 benchmarks from the ISCAS85/89 suites [88–90]; and the fir, brent.kung.32b, kogge.stone.32b, sobel, multiplier.32b, and neural.network benchmarks from the AxBench suite [96]. These benchmarks represent a variety of algorithms that covers a wide range of domains, i.e., arithmetic computation, machine learning, signal and image processing. Since DTS depends on the input data, we also extracted 100k randomly generated input vectors for each benchmark.

5.4.1 Evaluation of DTS

To evaluate the efficacy of our approach, we first estimated DTS_{ED}, DTS_{GB}, and the Static Timing Slack of GB-STA (STS_{GB}). Then, we measured the absolute error between DTS_{ED} and DTS_{GB} for every activated path i, as: $AE_{DTA}(i) = |DTS_{GB}(i) - DTS_{ED}(i)|$. To quantify how big or small AE_{DTA} is relatively to DTS_{GB}, we report the DTS improvement ($DTS_{imp.}$) achieved by ED-DTA, which is defined as: $DTS_{imp.}(i) = \left|\frac{DTS_{GB}(i)-DTS_{ED}(i)}{DTS_{GB}(i)}\right| \cdot 100 = \frac{AE_{DTA}(i)}{|DTS_{GB}(i)|} \cdot 100$, where i denotes a specific activated path.

Table 5.1 lists these observations along with the size and clock period (T_{clk}) for each benchmark. Note that the average (mean) DTS_{GB} is up to 4.3× bigger than the average STS_{GB}. According to this table, the average AE_{DTA} reaches up to 0.049 ns, while the maximum (max) AE_{DTA} observed is equal to 0.158 ns. As shown, the worst-case assumptions in GB-DTA lead to considerably high inaccuracy. In particular, ED-DTA provides varying degrees of DTS improvement, with the maximum $DTS_{imp.}$ ranging from 13.34% in the case of neural.network to 194.51% for brent.kung.32b. Overall, the proposed ED-DTA reveals on average 2.35% more DTS than GB-DTA. Fig. 5.5 depicts the distribution of DTS improvement across all activated paths for the considered benchmarks. Note that the minimum (min) DTS improvement is evaluated to zero across all benchmarks. This can be attributed to the fact that there are activated paths with exactly the same timing behavior and hence identical DTS_{GB} and DTS_{ED}.

Intuitively, input data that activate critical paths increase $DTS_{imp.}$. This can be attributed to the fact that critical paths, which usually consist of more slew merge points and input-state dependent gates compared to less critical paths, are more likely to be affected by the two main sources of pessimism in GB-DTA (see Section 5.2).

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

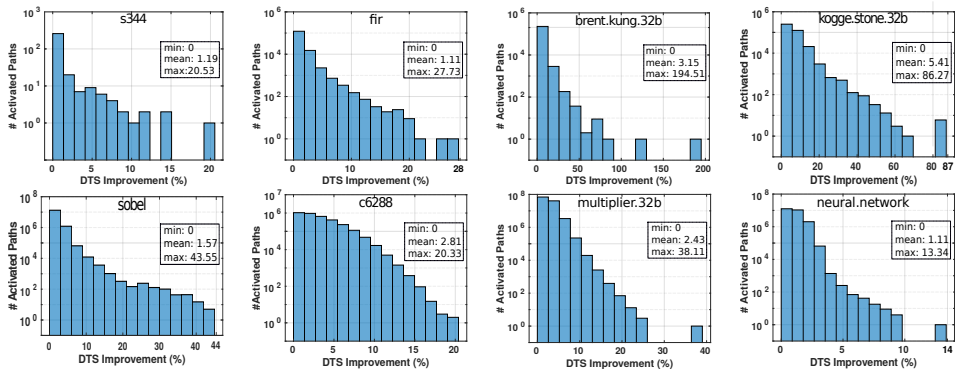


FIGURE 5.5: Distributions of DTS improvement achieved by ED-DTA, relatively to GB-DTA, across all dynamically activated paths.

5.4.2 DTS & dynamically activated critical paths

To experimentally verify this intuition, we extracted the top 5% critical activated paths (5% of the activated paths with the smallest slack). Fig. 5.6 shows the average and maximum DTS improvement across all benchmarks when only the critical paths are considered. We observe that the average $DTS_{imp.}$ has been increased to 11.2%. Additionally, the maximum $DTS_{imp.}$ of critical paths is equal to the maximum $DTS_{imp.}$ of all activated paths, verifying our intuition that critical paths impose the worst-case inaccuracy in GB-DTA.

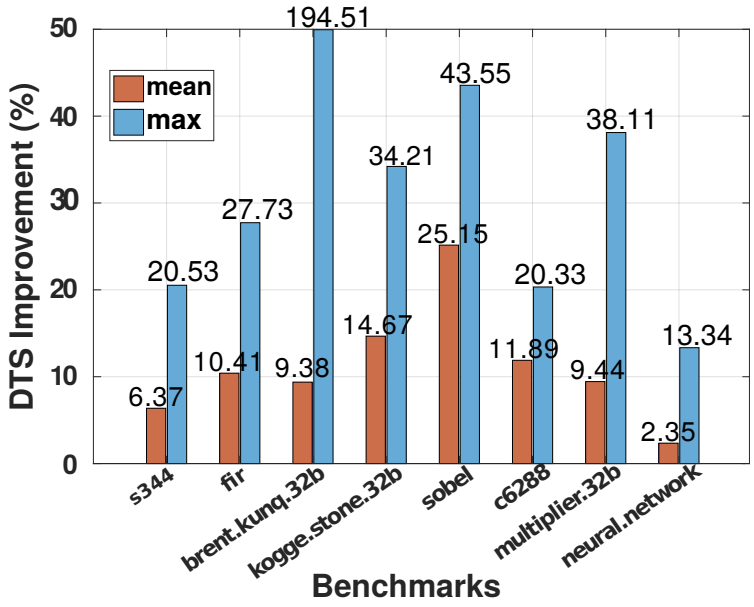


FIGURE 5.6: Mean and maximum DTS improvement achieved by ED-DTA considering only the top 5% critical activated paths.

5.4.3 Dynamic frequency scaling & timing failures

The underestimated slacks limit the gains achieved by works that exploit DTS [33,35–37,39]. Such works leverage the available DTS to reduce the clock period up to PoFF [35, 36, 39] or estimate the error rates under potential PVTA variation-induced delay increase levels [33,37].

PoFF and the number of timing errors strongly depend on the available DTS (a negative slack means that the path will fail). Fig. 5.7 shows the slack distributions obtained by

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

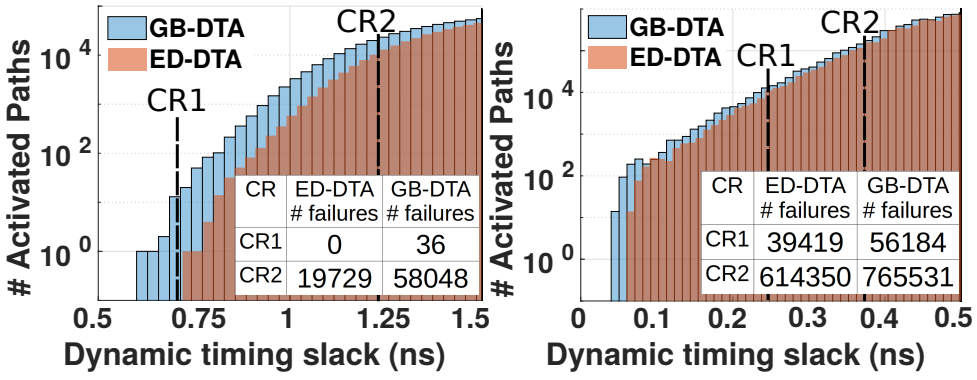
ED-DTA and GB-DTA for the c6288 and sobel benchmarks. For the sake of simplicity, the horizontal axis shows only the slacks that are less than 50% of T_{clk} (see Table 5.1). In this figure, it can be seen that PoFF varies considerably under GB-DTA and ED-DTA. Particularly, in the case of c6288, based on GB-DTA, PoFF is measured when T_{clk} was reduced to 2.325 ns (i.e., $DTS_{GB} = 0.615$ ns). Conversely, ED-DTA under c6288 incurs PoFF at $T_{clk} = 2.216$ ns (i.e., $DTS_{ED} = 0.724$ ns). In the case of sobel, PoFF under GB-DTA and ED-DTA occurs when $T_{clk} = 0.887$ ns ($DTS_{GB} = 0.043$ ns) and $T_{clk} = 0.866$ ns ($DTS_{ED} = 0.064$ ns), respectively. Considering all the examined benchmarks, ED-DTA results in 1.89% on average and up to 6.22% higher clock frequency than GB-DTA. It is very important to mention that this improvement corresponds to the minimum achievable clock frequency increase since PoFF is measured based on the most critical path across all clock cycles of the running application.

Recently, instruction-based or cycle-based frequency scaling schemes have been proposed [35, 36]. To estimate the potential gains that can be achieved by applying ED-DTA to such schemes, we extracted the most critical activated path on each cycle, which determines PoFF and the achievable frequency for this cycle. For each benchmark, we measured the average (across all cycles) frequency improvement (in terms of relative change over the nominal frequency) achieved by GB-DTA and ED-DTA. Overall, ED-DTA leads to 3.77% on average and up to 10.42% higher average frequency increase than GB-DTA.

GB-DTA not only leads to pessimistic estimation of DTS but also exhibits a substantial number of excited paths with such inaccurate slacks. This finding indicates that there is an increased probability of timing failures in GB-DTA under a potential delay increase. To better illustrate this, let us assume two Clock Reduction (CR) levels that represent potential PVTA variation-induced worst-case delay increase. Specifically, we reduced T_{clk} by 25% (referred to as CR1) and 40% (referred to as CR2). CR1 and CR2 are consistent with the levels of variation-induced delay increase that have been reported in the literature [26, 92]. As depicted in Fig. 5.7, under c6288 and CR1, ED-DTA manifests no timing failures, while GB-DTA incurs 36 timing failures. At CR2, GB-DTA results in 2.94× more failures than ED-DTA. Similar results are obtained in the case of sobel, where GB-DTA manifests 1.43× more timing failures compared to ED-DTA at CR2.

5.4.4 ED-DTA runtime

For the performance evaluation of our approach, we used a Linux workstation with a 2.1 GHz Intel® Xeon® E5-2620V4 CPU (8 cores, 16 threads) and 16 GB memory. ED-DTA runtimes range from 9.4 seconds for s344 up to 8019 seconds for multiplier.32b. Note that the proposed ED-DTA tool has been developed for single-thread CPU execution, since the primary purpose



(A) c6288 (B) sobel

FIGURE 5.7: DTS distributions obtained by ED-DTA and GB-DTA.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

of this work is to unveil the unexploited DTS ignored by GB-DTA methods, rather than to improve DTA performance. However, ED-DTA runtime can be improved by up to three orders of magnitude by utilizing Graphics Processing Units (GPUs) to exploit the available parallelism [97].

5.5 Chapter Summary

In this chapter, we presented a framework to unveil the pessimism in DTS estimation imposed by conventional GB-DTA methods which inherently rely on worst-case assumptions. To achieve this, we developed an accurate ED-DTA tool that considers the actual data-dependent path delays. Experimental results show that our ED-DTA approach unveils 2.35% on average and up to 194.51% more DTS, compared to GB-DTA. When only the critical activated paths are considered, the average DTS improvement is increased to 11.2%. We also demonstrated that the proposed approach enables a further increase of the clock frequency by up to 10.42% compared to existing frequency scaling schemes and reveals that timing failures can be up to 2.94× less than those estimated by existing failure estimation techniques.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

Publications

The research conducted for this Ph.D. dissertation resulted in the following peer-reviewed conference and journal publications:

Conference publications:

1. **Dimitrios Garyfallou**, Ioannis Tsiokanos, Nestor Evmorfopoulos, Georgios Stamoulis, and Georgios Karakonstantis, "Accurate Estimation of Dynamic Timing Slacks using Event-Driven Simulation", in *Proc. of the 21st International Symposium on Quality Electronic Design (ISQED)*, pp. 225–230, 2020, DOI:10.1109/ISQED48828.2020.9137017.

2. **Dimitrios Garyfallou**, Charalampos Antoniadis, Nestor Evmorfopoulos, and Georgios Stamoulis, "A Sparsity-Aware MOR Methodology for Fast and Accurate Timing Analysis of VLSI Interconnects", in *Proc. of the 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, pp. 89–92, 2019, DOI:10.1109/SMACD.2019.8795262.

3. Charalampos Antoniadis, **Dimitrios Garyfallou**, Nestor Evmorfopoulos, and Georgios Stamoulis, "EVT-based worst case delay estimation under process variation", in *Proc. of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1333–1338, 2018, DOI:10.23919/DATE.2018.8342220.

Journal publications:

1. **Dimitrios Garyfallou** *et al.*, "Gate Delay Estimation with Library Compatible Current Source Models and Effective Capacitance", *IEEE Trans. on Very Large Scale Integration (VLSI) Systems,* vol. 29, no. 5, pp. 962–972, 2021, DOI:10.1109/TVLSI.2021.3061484.

Moreover, our research led to the following peer-reviewed conference and journal publications that are not directly related to the content of this dissertation:

Conference publications:

1. Olympia Axelou, **Dimitrios Garyfallou**, George Floros, "Frequency-Limited Reduction of RLCK Circuits via Second-Order Balanced Truncation", in *SMACD/PRIME 2021; International Conference on SMACD and 16th Conference on PRIME*, pp. 1–4, 2021.

2. Chrysostomos Chatzigeorgiou*, **Dimitrios Garyfallou***, George Floros*, Nestor Evmorfopoulos, and Georgios Stamoulis, "Exploiting Extended Krylov Subspace for the Reduction of Regular and Singular Circuit Models", in *Proc. of the 26th Asia South Pacific Design Automation Conference (ASP-DAC)*, pp. 773–778, 2021, DOI:10.1145/3394885.3431589.

* Indicates equal contributution.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

3. **Dimitrios Garyfallou**, Nestor Evmorfopoulos, and Georgios Stamoulis, "A Combinatorial Multigrid Preconditioned Iterative Method for Large Scale Circuit Simulation on GPUs", in *Proc. of the 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, pp. 209–212, 2018, DOI:10.1109/SMACD.2018.8434892.

4. **Dimitrios Garyfallou**, Nestor Evmorfopoulos, and Georgios Stamoulis, "Large scale circuit simulation exploiting combinatorial multigrid on massively parallel architectures", in *Proc. of the 7th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp. 1–4, 2018, DOI:10.1109/MOCAST.2018.8376654.

Journal publications:

1. Pavlos Stoikos, **Dimitrios Garyfallou**, George Floros, Nestor Evmorfopoulos, and Georgios Stamoulis, "The Extended and Asymmetric Extended Krylov Subspace in Moment-Matching-Based Order Reduction of Large Circuit Models", *Integration, the VLSI Journal*, (**under review**).

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

References

[1] R. Goering, "What's Needed to "Fix" Timing Signoff? ," in *Panel of the 50th Design Automation Conference (DAC)*, 2013.

[2] A. B. Kahng, J. Lienig, I. L. Markov, and J. Hu, *VLSI Physical Design: From Graph Partitioning to Timing Closure*. Springer Science & Business Media, 2011.

[3] J. Bhasker and R. Chadha, *Static Timing Analysis for Nanometer Designs: A Practical Approach*. Springer Science & Business Media, 2009.

[4] C. Bittlestone, A. Hill, V. Singhal, and A. NV, "Architecting ASIC libraries and flows in nanometer era," in *Proc. of the 40th Design Automation Conference (DAC)*, pp. 776–781, 2003.

[5] Synopsys - Composite Current Source (CCS). Accessed: Sep. 15, 2021. [Online]. Available: http://www.opensourceliberty.org/ccspaper/

[6] Cadence - Effective Current Source Model (ECSM). Accessed: Sep. 15, 2021. [Online]. Available: https://www.cadence.com/en_US/home/alliances/standards-and-languages/ecsm-library-format.html

[7] J. Qian, S. Pullela, and L. Pillage, "Modeling the "Effective Capacitance" for the RC Interconnect of CMOS Gates," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 13, no. 12, pp. 1526–1535, 1994.

[8] A. B. Kahng and S. Muddu, "Improved effective capacitance computations for use in logic and layout optimization," in *Proc. of the 12th International Conference on VLSI Design*, pp. 578–583, 1999.

[9] B. N. Sheehan, "Library Compatible C_{eff} for Gate-Level Timing," in *Proc. of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 826–830, 2002.

[10] ——, "Osculating Thevenin model for predicting delay and slew of capacitively characterized cells," in *Proc. of the 39th Design Automation Conference (DAC)*, pp. 866–869, 2002.

[11] S. Abbaspour and M. Pedram, "Calculating the effective capacitance for the RC interconnect in VDSM technologies," in *Proc. of the 8th Asia South Pacific Design Automation Conference (ASP-DAC)*, pp. 43–48, 2003.

[12] F. Dartu, N. Menezes, and L. T. Pileggi, "Performance computation for precharacterized CMOS gates with RC loads," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 15, no. 5, pp. 544–553, 1996.

[13] Y. Zhou, Z. Li, R. N. Kanj, D. A. Papa, S. Nassif, and W. Shi, "A More Effective C_{eff} for Slew Estimation," in *Proc. of the International Conference on Integrated Circuit Design and Technology (ICICDT)*, pp. 1–4, 2007.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

[14] J. M. Wang, J. Li, S. Yanamanamanda, L. K. Vakati, and K. K. Muchherla, "Modeling the Driver Load in the Presence of Process Variations," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 25, no. 10, pp. 2264–2275, 2006.

[15] P. Feldmann, S. Abbaspour, D. Sinha, G. Schaeffer, R. Banerji, and H. Gupta, "Driver waveform computation for timing analysis with multiple voltage threshold driver models," in *Proc. of the 45th Design Automation Conference (DAC)*, pp. 425–428, 2008.

[16] IEEE International Roadmap for Devices and Systems (IRDS) - 2020 Update - More Moore. Accessed: Sep. 15, 2021. [Online]. Available: https://irds.ieee.org/editions/2020/

[17] Synopsys - HSPICE®. Accessed: Sep. 15, 2021. [Online]. Available: https://www.synopsys.com/verification/ams-verification/hspice.html

[18] R. Gupta, B. Tutuianu, and L. T. Pileggi, "The Elmore delay as a bound for RC trees with generalized input signals," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 16, no. 1, pp. 95–104, 1997.

[19] F. Liu, C. V. Kashyap, and C. J. Alpert, "A delay metric for RC circuits based on the Weibull distribution," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 23, no. 3, pp. 443–447, 2004.

[20] K. Agarwal, D. Sylvester, and D. T. Blaauw, "Simple metrics for slew rate of RC circuits based on two circuit moments," in *Proc. of the 40th Design Automation Conference (DAC)*, pp. 950–953, 2003.

[21] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI.* Addison-Wesley Pub. Co., 1990.

[22] R. Puri, D. S. Kung, and A. D. Drumm, "Fast and accurate wire delay estimation for physical synthesis of large ASICs," in *Proc. of the 12th Great Lakes Symposium on VLSI (GLSVLSI)*, pp. 30–36, 2002.

[23] R. W. Freund, "SPRIM: structure-preserving reduced-order interconnect macromodeling," in *Proc. of the International Conference on Computer Aided Design (ICCAD)*, pp. 80–87, 2004.

[24] K. J. Kerns and A. T. Yang, "Stable and efficient reduction of large, multiport RC networks by pole analysis via congruence transformations," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 16, no. 7, pp. 734–744, 1997.

[25] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by Pade approximation via the Lanczos process," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 14, no. 5, pp. 639–649, 1995.

[26] P. Gupta *et al*, "Underdesigned and opportunistic computing in presence of hardware variability," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 32, no. 1, pp. 8–23, 2013.

[27] M.-L. Chen *et al.*, "A FinFET with one atomic layer channel," *Nature communications*, vol. 11, no. 1, pp. 1–7, 2020.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

[28] H. Zhang, T. Chen, M. Y. Ting, and X. Li, "Efficient design-specific worst-case corner extraction for integrated circuits," in *Proc. of the 40th Design Automation Conference (DAC)*, pp. 386–389, 2009.

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

[30] H. Terada, T. Fukuoka, A. Tsuchiya, and H. Onodera, "Accurate Estimation of the Worst-case Delay in Statistical Static Timing Analysis," *IPSJ Trans. on System LSI Design Methodology*, vol. 1, pp. 116–125, 2008.

[31] L. Scheffer, "The count of Monte Carlo," in *the International Workshop on Timing Issues in the Specification and Synthesis of Digital Systems (TAU)*, 2004.

[32] G. Karakonstantis, A. Chatterjee, and K. Roy, "Containing the Nanometer "Pandora-Box": Cross-Layer Design Techniques for Variation Aware Low Power Systems," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS)*, vol. 1, no. 1, pp. 19–29, 2011.

[33] J. Xin and R. Joseph, "Identifying and predicting timing-critical instructions to boost timing speculation," in *Proc. of the 44th Annual International Symposium on Microarchitecture (MICRO)*, pp. 128–139, 2011.

[34] R. G. Dreslinski, M. Wieckowski, D. T. Blaauw, D. Sylvester, and T. N. Mudge, "Near-threshold computing: Reclaiming moore's law through energy efficient integrated circuits," *Proceedings of the IEEE*, vol. 98, no. 2, pp. 253–266, 2010.

[35] A. Rahimi *et al.*, "Application-Adaptive Guardbanding to Mitigate Static and Dynamic Variability," *IEEE Trans. on Computers*, vol. 63, no. 9, pp. 2160–2173, 2014.

[36] J. Constantin, L. Wang, G. Karakonstantis, A. Chattopadhyay, and A. Burg, "Exploiting dynamic timing margins in microprocessors for frequency-over-scaling with instruction-based clock adjustment," in *Proc. of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 381–386, 2015.

[37] X. Jiao *et al.*, "CLIM: A Cross-Level Workload-Aware Timing Error Prediction Model for Functional Units," *IEEE Trans. on Computers*, vol. 67, no. 6, pp. 771–783, 2018.

[38] H. Cherupalli, R. Kumar, and J. Sartori, "Exploiting Dynamic Timing Slack for Energy Efficiency in Ultra-Low-Power Embedded Systems," in *Proc. of the 43rd Annual International Symposium on Computer Architecture (ISCA)*, pp. 671–681, 2016.

[39] I. Tsiokanos, L. Mukhanov, and G. Karakonstantis, "Low-Power Variation-Aware Cores based on Dynamic Data-Dependent Bitwidth Truncation," in *Proc. of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 698–703, 2019.

[40] Siemens EDA - ModelSim™. Accessed: Sep. 15, 2021. [Online]. Available: https://eda.sw.siemens.com/en-US/ic/modelsim/

[41] H. Cherupalli and J. Sartori, "Scalable N-worst algorithms for dynamic timing and activity analysis," in *Proc. of the International Conference on Computer Aided Design (ICCAD)*, pp. 585–592, 2017.

[42] D. Garyfallou *et al.*, "Gate Delay Estimation With Library Compatible Current Source Models and Effective Capacitance," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 5, pp. 962–972, 2021.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

[43] TAU 2020 Timing Contest - Delay Calculator using Current Source Models. Accessed: Sep. 15, 2021. [Online]. Available: https://sites.google.com/view/taucontest2020/

[44] D. Garyfallou, C. Antoniadis, N. Evmorfopoulos, and G. Stamoulis, "A Sparsity-Aware MOR Methodology for Fast and Accurate Timing Analysis of VLSI Interconnects," in *Proc. of the 16th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, pp. 89–92, 2019.

[45] J. Galambos, *The Asymptotic Theory of Extreme Order Statistics*. R. E. Krieger, 1987, Second Edition.

[46] A. Dharchoudhury and S. Kang, "Worst-case analysis and optimization of VLSI circuit performances," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 14, no. 4, pp. 481–492, 1995.

[47] X. Lu, Z. Li, W. Qiu, D. M. H. Walker, and W. Shi, "PARADE: PARAmetric Delay Evaluation under Process Variation," in *Proc. of the 5th International Symposium on Quality Electronic Design (ISQED)*, pp. 276–280, 2004.

[48] T. Fukuoka, A. Tsuchiya, and H. Onodera, "Worst-case delay analysis considering the variability of transistors and interconnects," in *Proc. of the International Symposium on Physical Design (ISPD)*, pp. 35–42, 2007.

[49] C. Antoniadis, D. Garyfallou, N. Evmorfopoulos, and G. Stamoulis, "EVT-based worst case delay estimation under process variation," in *Proc. of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1333–1338, 2018.

[50] D. Garyfallou, I. Tsiokanos, N. Evmorfopoulos, G. Stamoulis, and G. Karakonstantis, "Accurate Estimation of Dynamic Timing Slacks using Event-Driven Simulation," in *Proc. of the 21st International Symposium on Quality Electronic Design (ISQED)*, pp. 225–230, 2020.

[51] J. F. Croix and D. Wong, "Blade and razor: cell and interconnect delay analysis using current-based models," in *Proc. of the 40th Design Automation Conference (DAC)*, pp. 386–389, 2003.

[52] N. Menezes, C. Kashyap, and C. Amin, "A "true" electrical cell model for timing, noise, and power grid verification," in *Proc. of the 45th Design Automation Conference (DAC)*, pp. 462–467, 2008.

[53] C. Knoth, H. Jedda, and U. Schlichtmann, "Current source modeling for power and timing analysis at different supply voltages," in *Proc. of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 923–928, 2012.

[54] P. R. O'Brien and T. L. Savarino, "Modeling the driving-point characteristic of resistive interconnect for accurate delay estimation," in *Proc. of the International Conference on Computer Aided Design (ICCAD)*, pp. 512–515, 1989.

[55] R. W. Freund, "Krylov-subspace methods for reduced-order modeling in circuit simulation," *Journal of Computational and Applied Mathematics*, vol. 123, no. 1-2, pp. 395–421, 2000.

[56] I. Keller, K. H. Tam, and V. Kariat, "Challenges in gate level modeling for delay and SI at 65nm and below," in *Proc. of the 45th Design Automation Conference (DAC)*, pp. 468–473, 2008.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

[57] C. Kashyap, C. Alpert, F. Liu, and A. Devgan, "Closed-form expressions for extending step delay and slew metrics to ramp inputs for RC trees," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 23, no. 4, pp. 24–31, 2004.

[58] L. T. Clark *et al.*, "ASAP7: A 7-nm finFET predictive process design kit," *Microelectronics Journal*, vol. 53, pp. 105–115, 2016.

[59] ASU - ASAP7 PDK. Accessed: Sep. 15, 2021. [Online]. Available: https://github.com/The-OpenROAD-Project/asap7/

[60] C. L. Ratzlaff and L. T. Pillage, "RICE: rapid interconnect circuit evaluation using AWE," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 13, no. 6, pp. 763–776, 1994.

[61] C. Chatzigeorgiou, D. Garyfallou, G. Floros, N. Evmorfopoulos, and G. Stamoulis, "Exploiting Extended Krylov Subspace for the Reduction of Regular and Singular Circuit Models," in *Proc. of the 26th Asia South Pacific Design Automation Conference (ASP-DAC)*, pp. 773–778, 2021.

[62] Z. Ye, D. Vasilyev, and J. R. Phillips, "Sparse Implicit Projection (SIP) for reduction of general many-terminal networks," in *Proc. of the International Conference on Computer Aided Design (ICCAD)*, pp. 736–743, 2008.

[63] R. Ionutiu, J. Rommes, and W. H. A. Schilders, "SparseRC: Sparsity Preserving Model Reduction for RC Circuits With Many Terminals," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 30, no. 12, pp. 1828–1841, 2011.

[64] B. N. Sheehan, "TICER: Realizable reduction of extracted RC circuits," in *Proc. of the International Conference on Computer Aided Design (ICCAD)*, pp. 200–203, 1999.

[65] C. Antoniadis, N. Evmorfopoulos, and G. Stamoulis, "Efficient sparsification of dense circuit matrices in model order reduction," in *Proc. of the 24th Asia South Pacific Design Automation Conference (ASP-DAC)*, pp. 255–260, 2019.

[66] S. Abbaspour, M. Pedram, A. Ajami, and C. Kashyap, "Fast Interconnect and Gate Timing Analysis for Performance Optimization," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 12, pp. 1383–1388, 2006.

[67] Synopsys - StarRC™. Accessed: Sep. 15, 2021. [Online]. Available: https://www.synopsys.com/implementation-and-signoff/signoff/starrc.html

[68] C.-W. Ho, A. Ruehli, and P. Brennan, "The modified nodal approach to network analysis," *IEEE Trans. on Circuits and Systems*, vol. 22, no. 6, pp. 504 – 509, 1975.

[69] D. Garyfallou, N. Evmorfopoulos, and G. Stamoulis, "A Combinatorial Multigrid Preconditioned Iterative Method for Large Scale Circuit Simulation on GPUs," in *Proc. of the 15th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, pp. 209–212, 2018.

[70] ——, "Large scale circuit simulation exploiting combinatorial multigrid on massively parallel architectures," in *Proc. of the 7th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp. 1–4, 2018.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

[71] D. E. Crabtree, "Applications of M-matrices to non-negative matrices," *Duke Mathematical Journal*, vol. 33, no. 1, pp. 197 – 208, 1966.

[72] G. Guennebaud, B. Jacob *et al.*, "Eigen v3," 2010. [Online]. Available: http://eigen.tuxfamily.org

[73] M. M. Ozdal, C. Amin, A. Ayupov, S. Burns, G. Wilke, and C. Zhuo, "The ISPD-2012 discrete cell sizing contest and benchmark suite," in *Proc. of the International Symposium on Physical Design (ISPD)*, pp. 161–164, 2012.

[74] NanGate FreePDK45 Generic Open Cell Library. Accessed: Sep. 15, 2021. [Online]. Available: https://si2.org/open-cell-library/

[75] V. Vishvanathan, C. Ravikumar, V. Menezes, B. T. Park, and C. R. Nagar, "Design technology challenges in the sub-100 nanometer era," *the periodical of the VLSI society of India–VLSI Vision*, vol. 1, no. 1, 2005.

[76] R.-D. Reiss, M. Thomas, and R. Reiss, *Statistical analysis of extreme values*. Springer Science & Business Media, 1997.

[77] R. A. Fisher and L. H. C. Tippett, "Limiting forms of the frequency distribution of the largest or smallest member of a sample," in *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 24, no. 2, pp. 180–190, 1928.

[78] A. A. Balkema and L. De Haan, "Residual Life Time at Great Age," *The Annals of probability*, vol. 2, no. 5, pp. 792–804, 1974.

[79] J. Pickands III *et al.*, "Statistical Inference Using Extreme Order Statistics," *the Annals of Statistics*, vol. 3, no. 1, pp. 119–131, 1975.

[80] N. E. Evmorfopoulos, G. I. Stamoulis, and J. N. Avaritsiotis, "A Monte Carlo approach for maximum power estimation based on extreme value theory," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 21, no. 4, pp. 415–432, 2002.

[81] G. G. Roussas, *A course in mathematical statistics*. Elsevier, 1997.

[82] N. H. Weste and D. Harris, *CMOS VLSI design: A Circuits and Systems Perspective*. Pearson Education India, 2015.

[83] M. J. Pelgrom, A. C. Duinmaijer, and A. P. Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 24, no. 5, pp. 1433–1439, 1989.

[84] S. J. Wilton and N. P. Jouppi, "Cacti: An enhanced cache access and cycle time model," *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 31, no. 5, pp. 677–688, 1996.

[85] H. Drees and E. Kaufmann, "Selecting the optimal sample fraction in univariate extreme value estimation," *Stochastic Processes and their Applications*, vol. 75, no. 2, pp. 149–172, 1998.

[86] A. Guillou and P. Hall, "A diagnostic for selecting the threshold in extreme value analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 63, no. 2, pp. 293–305, 2001.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131

[87] A. Singhee and R. A. Rutenbar, "Statistical Blockade: Very Fast Statistical Simulation and Modeling of Rare Circuit Events and Its Application to Memory Design," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 28, no. 8, pp. 1176–1189, 2009.

[88] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmark circuits and a targeted translator in FORTRAN," in *Proc. of the International Symposium on Circuits and Systems (ISCAS)*, pp. 1929–1934, 1985.

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

[90] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," in *Proc. of the International Symposium on Circuits and Systems (ISCAS)*, pp. 1929–1934, 1989.

[91] TAU 2013 Timing Contest - Variation-Aware Timing Analysis. Accessed: Sep. 15, 2021. [Online]. Available: https://sites.google.com/site/taucontest2013/

[92] D. Bull *et al.*, "A Power-Efficient 32 bit ARM Processor Using Timing-Error Detection and Correction for Transient-Error Tolerance and Adaptation to PVT Variation," *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 46, no. 1, pp. 18–31, 2011.

[93] A. Krstic, Yi-Min Jiang, and Kwang-Ting Cheng, "Pattern generation for delay testing and dynamic timing analysis considering power-supply noise effects," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 20, no. 3, pp. 416–425, 2001.

[94] Synopsys - Design Compiler®. Accessed: Sep. 15, 2021. [Online]. Available: https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/design-compiler-graphical.html

[95] Cadence - Innovus™ Implementation System. Accessed: Sep. 15, 2021. [Online]. Available: https://www.cadence.com/en_US/home/tools/digital-design-and-signoff/soc-implementation-and-floorplanning/innovus-implementation-system.html

[96] A. Yazdanbakhsh, D. Mahajan, H. Esmaeilzadeh, and P. Lotfi-Kamran, "AxBench: A Multiplatform Benchmark Suite for Approximate Computing," *IEEE Design & Test*, vol. 34, no. 2, pp. 60–68, 2017.

[97] E. Schneider, M. A. Kochte, and H.-J. Wunderlich, "Multi-level timing simulation on GPUs," in *Proc. of the 23rd Asia South Pacific Design Automation Conference (ASP-DAC)*, pp. 470–475, 2018.

Institutional Repository - Library & Information Centre - University of Thessaly
11/06/2024 18:01:18 EEST - 3.15.189.131