UNIVERSITY OF CALIFORNIA SAN DIEGO

Robust Physical Design and Design Technology Co-Optimization Methodologies at Advanced VLSI Technology

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering (Computer Engineering)

by

Minsoo Kim

2023

University of California San Diego

2023

DEDICATION

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor, Professor Andrew B. Kahng, for serving as the chair of my committee and providing invaluable guidance throughout my Ph.D. journey.

I also want to thank my fellow labmates at the UCSD VLSI CAD Laboratory, including Nikhil Dakwala, Sayak Kundu, Jiantao Liu, Bodhisatta Pramanik, Ravi Varadarajan, Zhiang Wang, Mingyu Woo, Chien-Yi Yang and Dooseok Yoon, as well as former lab members, including Payal Agarwal, Dr. Wei-Ting (Jonas) Chan, Dr. Ahmed Taha Elthakeb, Dr. Kwangsoo Han, Dr. Chia-Tung (Mark) Ho, Dr. Seungwon Kim, Dr. Hyein Lee, Hsin-Yu (Desmond) Liu, Dr. Uday Mallappa, Tushar Shah, Siddharth Singh, Dr. Vaishnav Srinivas, Sriram Venkatesh, Dr. Lutong Wang and Dr. Bangqi Xu, for their assistance and thought-provoking discussions. Special thanks go to Dr. Tuck-Boon Chan, Dr. Ilgweon Kang, Dr. Daeyeal Lee, Dr. Jiajia Li and Dr. Dongwon Park for their guidance during my Ph.D. studies.

I am also grateful to my thesis committee members, Professor Chung-Kuan Cheng, Professor Sujit Dey and Professor Farinaz Koushanfar, for their time, encouragement and insightful comments.

I would like to acknowledge the invaluable feedback and support of my industrial collaborators, including Suhyeong Choi, Dr. Hamed Fatemi, Dr. Jose Pineda de Gyvez, Sunik Heo, Dr. Jinwook Jung, Chunghee Kim, Hayoung Kim, Dr. Jaehwan Kim, Dr. Gi-Joon Nam and Dr. S. C. Song.

I extend my gratitude to my old fellows from South Korea, Jinhwan Choi, Sangha Chung, Sangyeop Kim, Youngmin Kim, Seungheon Lee and Byeong-Geun Sun, as well as my beloved companions, Hyunwoo Baek, Woojae Park, Eunjae Shin and Sooyoung Son.

Finally, I would like to express my heartfelt appreciation to my parents, Hongjin Kim and Jungsuk Kang, my mother-in-law, Nankyung Kim, and the rest of my family for their endless love and encouragement. I owe a special debt of gratitude to my wife, Youri Jeong, whose unwavering love and support enabled me to complete this journey, and my lovely daughters, Joy and Skylar, who gave me the strength I needed every day with their love. This thesis is dedicated to my family.

The material in this thesis is based on the following publications.

Chapter 2 contains a reprint of Hamed Fatemi, Andrew B. Kahng, Minsoo Kim and Jose Pineda de Gyvez, “Optimal Bounded-Skew Steiner Trees to Minimize Maximum k-Active Dynamic Power”, *Proc. International Workshop on System-Level Interconnect Problems and Pathfinding*, 2020; and Chung-Kuan Cheng, Andrew B. Kahng, Ilgweon Kang, Minsoo Kim, Daeyeal Lee, Bill Lin, Dongwon Park and Mingyu Woo, “CoRe-ECO: Concurrent Refinement of Detailed Place-and-Route for an Efficient ECO Automation”, *Proc. IEEE International Conference on Computer Design*, 2021. The dissertation author is a main contributor to, and a primary author of, each of these papers.

Chapter 3 contains a reprint of Sunik Heo, Andrew B. Kahng, Minsoo Kim and Lutong Wang, “Diffusion Break-Aware Leakage Power Optimization and Detailed Placement in Sub-10nm VLSI”, *Proc. Asia and South Pacific Design Automation Conference*, 2019; and Sunik Heo, Andrew B. Kahng, Minsoo Kim and Lutong Wang and Chutong Yang, “Detailed Placement for IR Drop Mitigation by Power Staple Insertion in Sub-10nm VLSI”, *Proc. Design, Automation and Test in Europe*, 2019. The dissertation author is a main contributor to, and a primary author of, each of these papers.

Chapter 4 contains a reprint of Chung-Kuan Cheng, Andrew B. Kahng, Hayoung Kim, Minsoo Kim, Daeyeal Lee, Dongwon Park and Mingyu Woo, “PROBE2.0: A Systematic Framework for Routability Assessment from Technology to Design in Advanced Nodes”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and System* 41(5), 2022; Chidi Chidambaram, Andrew B. Kahng, Minsoo Kim, Giri Nallapati, S. C. Song and Mingyu Woo, “A Novel Framework for DTCO: Fast and Automatic Routability Assessment with Machine Learning for Sub-3nm Technology Options”, *Proc. IEEE Symposium on VLSI Technology*, 2021; and Suhyeong Choi, Jinwook Jung, Andrew B. Kahng, Minsoo Kim, Chul-Hong Park, Bodhisatta Pramanik and Dooseok Yoon, “PROBE3.0: A Systematic Framework for Design-

Technology Pathfinding with Improved Design Enablement”, in submission to *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023. The dissertation author is a main contributor to, and a primary author of, each of these papers.

All co-authors, listed in alphabetical order, including Professor Chung-Kuan Cheng, Dr. Chidi Chidambaram, Suhyeong Choi, Dr. Hamed Fatemi, Sunik Heo, Dr. Jinwook Jung, Professor Andrew B. Kahng, Dr. Ilgweon Kang, Hayoung Kim, Dr. Daeyeal Lee, Professor Bill Lin, Dr. Giri Nallapati, Dr. Chul-Hong Park, Dr. Dongwon Park, Dr. Jose Pineda de Gyvez, Bodhisatta Pramanik, Dr. S. C. Song, Dr. Lutong Wang, Mingyu Woo, Chutong Yang and Dooseok Yoon, have kindly approved the inclusion of the aforementioned publications in my thesis.

VITA

2011	B. S., Electrical Engineering, Yonsei University, Seoul, South Korea
2013	M. S., Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon, South Korea
2013–2017	Engineer, Samsung Electronics, Gyeonggi-do, South Korea
2021	C. Phil., Electrical Engineering (Computer Engineering), University of California San Diego
2023	Ph. D., Electrical Engineering (Computer Engineering), University of California San Diego

All papers coauthored with my advisor Professor Andrew B. Kahng have authors listed in alphabetical order.

PUBLICATIONS

Suhyeong Choi, Jinwook Jung, Andrew B. Kahng, **Minsoo Kim**, Chul-Hong Park, Bodhisatta Pramanik and Dooseok Yoon, "PROBE3.0: A Systematic Framework for Design-Technology Pathfinding with Improved Design Enablement", in submission to *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

Chia-Tung Ho, Alvin Ho, Matthew Fojtik, **Minsoo Kim**, Shang Wei, Yaguang Li, Brucek Khailany and Haoxing Ren, "NVCell 2: Routability-Driven Standard Cell Layout in Advanced Nodes with Lattice Graph Routability Model", *Proc. International Symposium on Physical Design*, 2023, pp. 44-52.

Chung-Kuan Cheng, Andrew B. Kahng, Ilgweon Kang, **Minsoo Kim**, Daeyeal Lee, Bill Lin, Dongwon Park and Mingyu Woo, "PROBE2.0: A Systematic Framework for Routability Assessment from Technology to Design in Advanced Nodes", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and System* 41(5) (2022), pp. 1495-1508.

Andrew B. Kahng, **Minsoo Kim**, Seungwon Kim and Mingyu Woo, "RosettaStone: Connecting the Past, Present and Future of Physical Design Research", *IEEE Design & Test* 39(5) (2022), pp. 70-78.

Chung-Kuan Cheng, Andrew B. Kahng, Ilgweon Kang, **Minsoo Kim**, Daeyeal Lee, Bill Lin, Dongwon Park and Mingyu Woo, "CoRe-ECO: Concurrent Refinement of Detailed Place-and-Route for an Efficient ECO Automation", *Proc. IEEE International Conference on Computer Design*, 2021, pp. 366-373.

Chidi Chidambaram, Andrew B. Kahng, **Minsoo Kim**, Giri Nallapati, S. C. Song and Mingyu Woo, "A Novel Framework for DTCO: Fast and Automatic Routability Assessment with Machine Learning for Sub-3nm Technology Options", *Proc. IEEE Symposium on VLSI Technology*, 2021, pp. 1-2.

Hamed Fatemi, Andrew B. Kahng, **Minsoo Kim** and Jose Pineda de Gyvez, "Optimal Bounded-Skew Steiner Trees to Minimize Maximum k-Active Dynamic Power", *Proc. ACM/IEEE International Workshop on System-Level Interconnect Problems and Pathfinding*, 2020, pp. 1-8.

Austin Rovinski, Tutu Ajayi, **Minsoo Kim**, Guanru Wang and Mehdi Saligane, "Bridging Academic Open-Source EDA to Real World Usability", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2020, pp. 1-7.

Vidya A. Chhabria, Andrew B. Kahng, **Minsoo Kim**, Uday Mallappa, Sachin S. Sapatnekar and Bangqi Xu, "Template-based PDN Synthesis in Floorplan and Placement Using Classifier and CNN Techniques", *Proc. Asia and South Pacific Design Automation Conference*, 2020, pp. 44-49.

Tutu Ajayi, Vidya A. Chhabria, Mateus Fogaça, Soheil Hashemi, Abdelrahman Hosny, Andrew B. Kahng, **Minsoo Kim**, Jeongsup Lee, Uday Mallappa, Marina Neseem, Geraldo Pradipta, Sherief Reda, Mehdi Saligane, Sachin S. Sapatnekar, Carl Sechen, Mohamed Shalan, William Swartz, Lutong Wang, Zhehong Wang, Mingyu Woo and Bangqi Xu, "Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project", *Proc. ACM/IEEE Design Automation Conference*, 2019, pp. 76:1-76:4.

Tutu Ajayi, David Blaauw, Tuck-Boon Chan, Chung-Kuan Cheng, Vidya A. Chhabria, David K. Choo, Matteo Coltella, Sorin Dobre, Ronald Dreslinski, Mateus Fogaça, Soheil Hashemi, Abdelrahman Hosny, Andrew B. Kahng, **Minsoo Kim**, Jiajia Li, Zhaoxin Liang, Uday Mallappa, Paul Penzes, Geraldo Pradipta, Sherief Reda, Austin Rovinski, Kambiz Samadi, Sachin S. Sapatnekar, Lawrence Saul, Carl Sechen, Vaishnav Srinivas, William Swartz, Dennis Sylvester, David Urquhart, Lutong Wang, Mingyu Woo and Bangqi Xu, "OpenROAD: Toward a Self- Driving, Open-Source Digital Layout Implementation Tool Chain", *Proc. Government Microcircuit Applications and Critical Technology Conference*, 2019, pp. 1105-1110.

Sunik Heo, Andrew B. Kahng, **Minsoo Kim**, Lutong Wang and Chutong Yang, "Detailed Placement for IR Drop Mitigation by Power Staple Insertion in Sub-10nm VLSI", *Proc. Design, Automation & Test in Europe Conference & Exhibition*, 2019, pp. 824-829.

Sunik Heo, Andrew B. Kahng, **Minsoo Kim** and Lutong Wang, "Diffusion Break-Aware Leakage Power Optimization and Detailed Placement in Sub-10nm VLSI", *Proc. Asia and South Pacific Design Automation Conference*, 2019, pp. 550-556.

Minsoo Kim, Chong-Min Kyung and Kang Yi, “An Energy Management Scheme for Solar-Powered Wireless Visual Sensor Networks Toward Uninterrupted Operations”, *Proc. IEEE International SoC Design Conference*, 2013, pp. 23-26.

ABSTRACT OF THE DISSERTATION

Robust Physical Design and Design Technology Co-Optimization Methodologies at Advanced VLSI Technology

by

Minsoo Kim

Doctor of Philosophy in Electrical Engineering (Computer Engineering)

University of California San Diego, 2023

Professor Andrew B. Kahng, Chair

The semiconductor industry has achieved remarkable progress by adhering to Moore's Law in the past few decades. As a result, technology has continuously scaled down and advanced to the 2nm and 3nm nodes by 2023. The consistent scaling of advanced technologies has made it possible to utilize them in various applications of modern IC designs, such as mobile, data center, automotive, graphics, the Internet of Things (IoT) and artificial intelligence (AI), which may demand high performance and/or ultra-low power consumption. However, the recent slowdown in the traditional Moore's Law scaling rate has presented significant challenges. Therefore, considerable efforts have been devoted to physical design and design-technology co-optimization to optimize the advantages of advanced technology nodes for different applications.

This thesis presents robust physical design and design-technology co-optimization methodologies that aim to maximize the benefits of advanced technologies and optimize power, performance, area and cost in modern IC design. The proposed methodologies are categorized into three main directions: (i) general physical design methodologies, (ii) technology-aware physical design methodologies and (iii) design-technology co-optimization methodologies.

To address challenges in modern IC design, this thesis presents two works: (i) bounded-skew Steiner tree optimization for clock tree synthesis to minimize active power and (ii) concurrent refinement of detailed place-and-route (P&R) for efficient engineering change order (ECO) automation.

To address challenges specific to advanced technology nodes, this thesis presents two works: (i) leakage power optimization with the awareness of local layout effects and (ii) detailed placement for IR drop mitigation by power staple insertions.

Finally, to address challenges in design-technology co-optimization at advanced technology nodes, this thesis presents three works: (i) PROBE2.0: A systematic framework for routability assessments, (ii) a routability study using the PROBE2.0 framework with 3nm technology configurations and (iii) PROBE3.0: A systematic framework for power, performance, area and cost explorations, with improved design enablement.

Chapter 1

Introduction

In the last several decades, the semiconductor industry has successfully followed Moore's Law as a guide for the long-term planning of advanced technology development. As a result, semiconductor technologies have consistently scaled down to the 2nm and 3nm nodes, with the number of transistors in a single chip now exceeding tens of billions due to aggressive transistor density scaling. These advanced technologies bring various capabilities to applications, including mobile, data center, automotive, graphics, the Internet of Things (IoT) and artificial intelligence, with significantly improved performance and/or reduced power consumption.

1.1 New Challenges

Despite the semiconductor industry's past success, the trend of Moore's Law scaling has recently slowed down. Figure 1.1 illustrates this slowdown, where the fin pitch, contacted poly pitch (CPP) and standard-cell height scaling rates have flattened over time. Consequently, the slowdown has presented numerous challenges across many applications, and both the semiconductor industry and academia have made significant efforts to mitigate the slowdown and address the challenges posed by various applications and advanced technology nodes.

1.1.1 Challenges for Modern IC Design

In modern Integrated Circuit (IC) designs, various applications target different power and performance levels. Figure 1.2 illustrates the major applications in modern IC design and

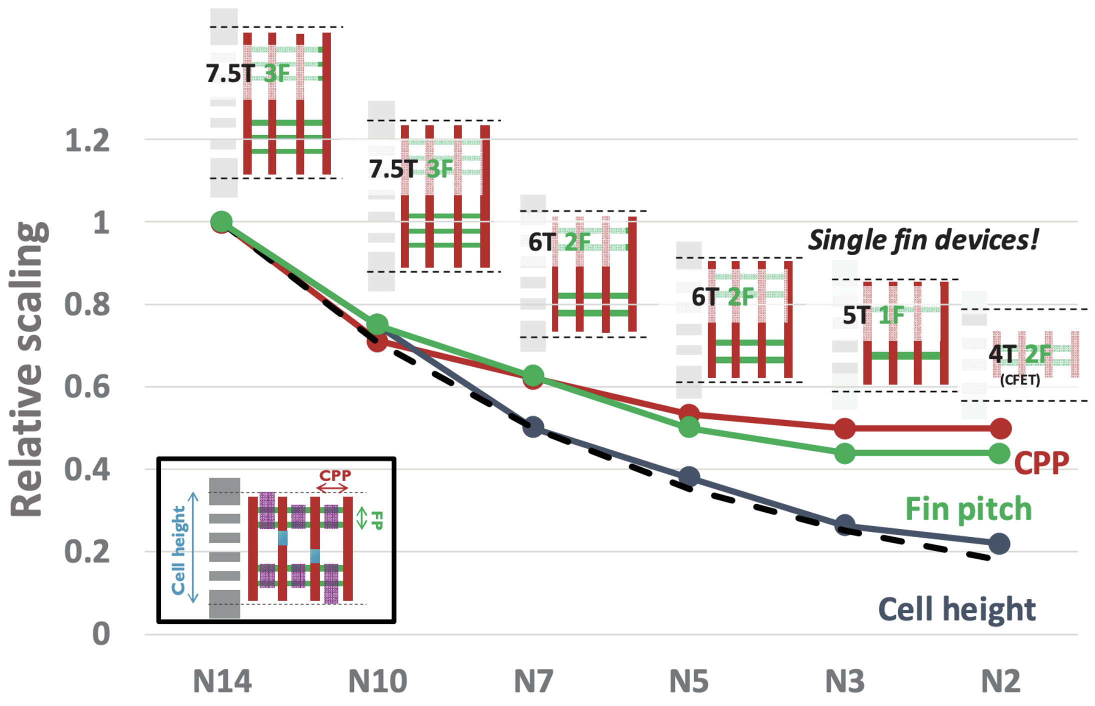


Figure 1.1. The scaling rates of fin pitch, contacted poly pitch (CPP) and standard-cell height have slowed down over time, and device architectures have also evolved from fin field-effect transistor (FinFET) to complementary field-effect transistor (CFET) [104].

their general power and performance requirements. High-performance computing demands high performance with high power consumption, while IoT aims to achieve long operational lifetimes with ultra-low power consumption. Across these applications, the ultimate goal of modern IC design is to optimize power, performance, area and cost (PPAC) within a reasonable turnaround time. However, as the target PPAC is dependent on the application and design, optimizing for various applications can be challenging in modern IC design.

Aggressive PPAC optimizations introduce further challenges in modern IC design. One such challenge is the demand for aggressive power reduction. Although low power methodologies have enabled significant power reductions while maintaining or even improving performance, the recent demand for ultra-low power consumption in automotive, mobile and IoT applications comes with strict power constraints. Another challenge is the routability problem and complex design rule check violations (DRC) caused by aggressive area reduction in physical design.

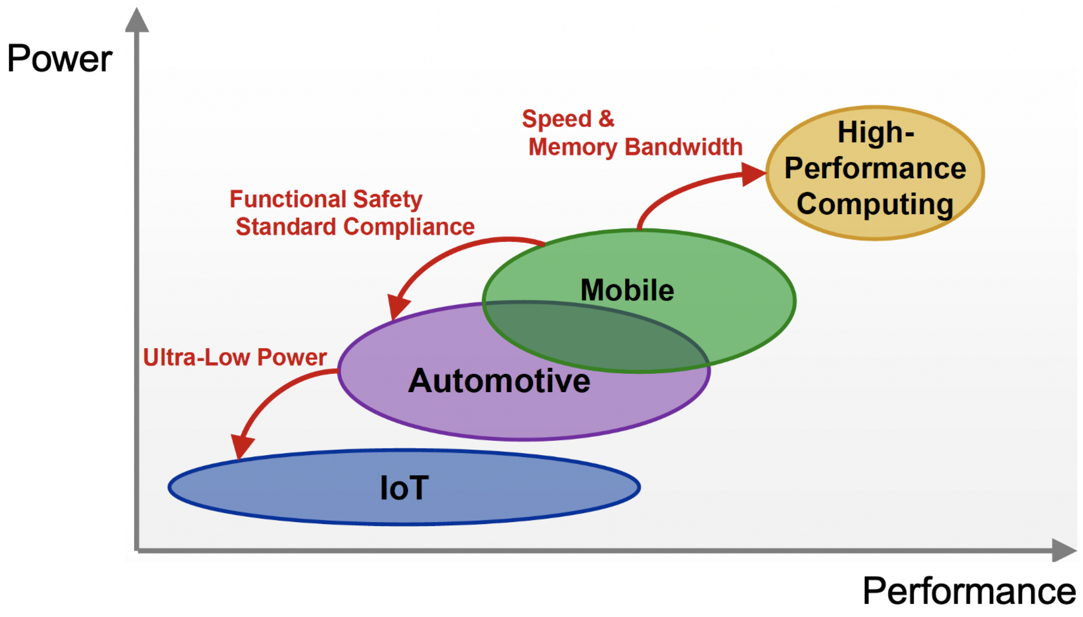


Figure 1.2. Power and performance requirements for applications [89].

The complexity of design rules and constraints in advanced nodes exacerbates this problem, necessitating additional human engineering resources to address the problem.

1.1.2 Challenges for Advanced Technology Nodes

As technology advances, aggressive scaling of critical dimensions and pitches poses new challenges in placement, clock tree synthesis, routing and timing analysis. One of these challenges arises due to the increasing impact of layout-dependent device parameter shifts. This local layout effect (LLE) becomes more significant at advanced nodes due to aggressive scaling. LLE can cause power and performance variations in devices, resulting from standard-cell architectures and/or placement in physical design. For example, in standard cells, diffusion breaks isolate two neighboring devices. In advanced nodes, these breaks can have a stress effect on timing and leakage power. Figure 1.3(a) illustrates the saturation voltage and effective current shifts caused by LLE from the second diffusion break. Figure 1.3(b) shows the interlayer dielectric (ILD) stress in a TCAD simulation, which depends on the spacing between the first and second diffusion breaks.

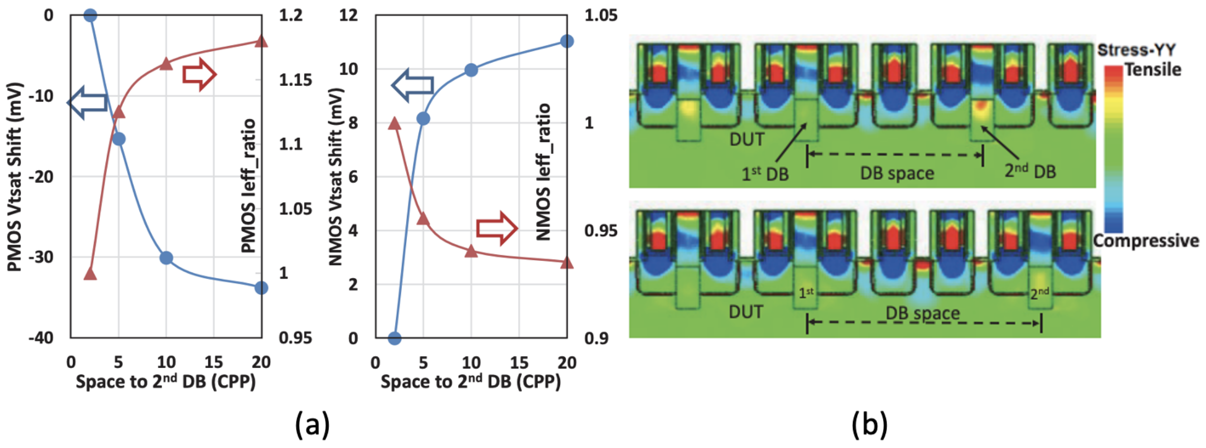


Figure 1.3. Local layout effect [128]. (a) Saturation voltage and effective current shifts caused by the second diffusion break effect. (b) Interlayer dielectric (ILD) stress in a TCAD simulation, which is a function of the spacing between the first and the second diffusion breaks.

Another challenge is the high resistance of interconnects due to aggressive BEOL metal pitch scaling. Figure 1.4 illustrates the unit resistance of interconnects for recent technologies, where the 5nm technology has a 9X larger unit resistance for the lowest BEOL metal layers (*Mx*) compared to the 16nm technology. This leads to severe timing degradation from interconnects and voltage (IR) drop problems.

1.1.3 Challenges for Design-Technology Co-Optimization

Design-Technology Co-Optimization (DTCO) is an optimization process that aims to maximize the benefits of advanced technology nodes by exploring most beneficial design and technology configurations. Fast and accurate PPAC explorations are key players in this process as they help to explore new transistor architectures, materials, standard-cell architectures and design knobs. In addition, *scaling boosters* are introduced at advanced technology nodes to maintain the traditional Moore's Law density scaling rate. These scaling boosters provide more design and technology configurations and broader solution spaces for the PPAC explorations. However, the exploration of these new technology options is becoming increasingly difficult and complicated, and requires thorough evaluations.

Today's DTCO methodologies face several limitations in advanced nodes. One of the

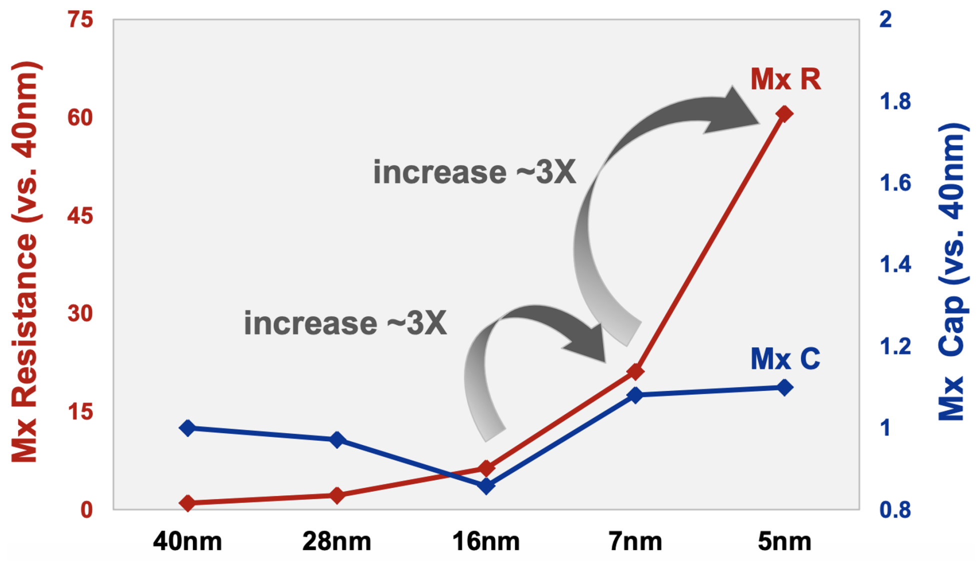


Figure 1.4. Resistance of BEOL metal layers (*Mx*) increases exponentially as technology advances [89].

major challenges is the limited exploration of the entire solution space due to a lack of human engineering resources. The process is not fully automated and requires significant manual efforts from human engineers, such as creating standard-cell layouts and process design kits (PDK). Another challenge is that the current ring oscillator-based PPAC explorations do not always match the results of real chips as they do not address all aspects of design, such as routability. Moreover, PPAC is design-dependent, and the results from ring oscillators cannot be applied to all types of designs. Finally, due to the competitive nature of the semiconductor industry, critical information related to advanced technologies remains under the proprietary domain of semiconductor companies, making it inaccessible to academia. As a result, academic researchers are unlikely to make significant innovations in DTCO research as they cannot conduct experiments with real values corresponding to modern IC technologies.

1.2 This Thesis

This thesis presents robust physical design and DTCO methodologies to tackle challenges posed by advanced technology nodes. The methodologies proposed in this thesis can be classified

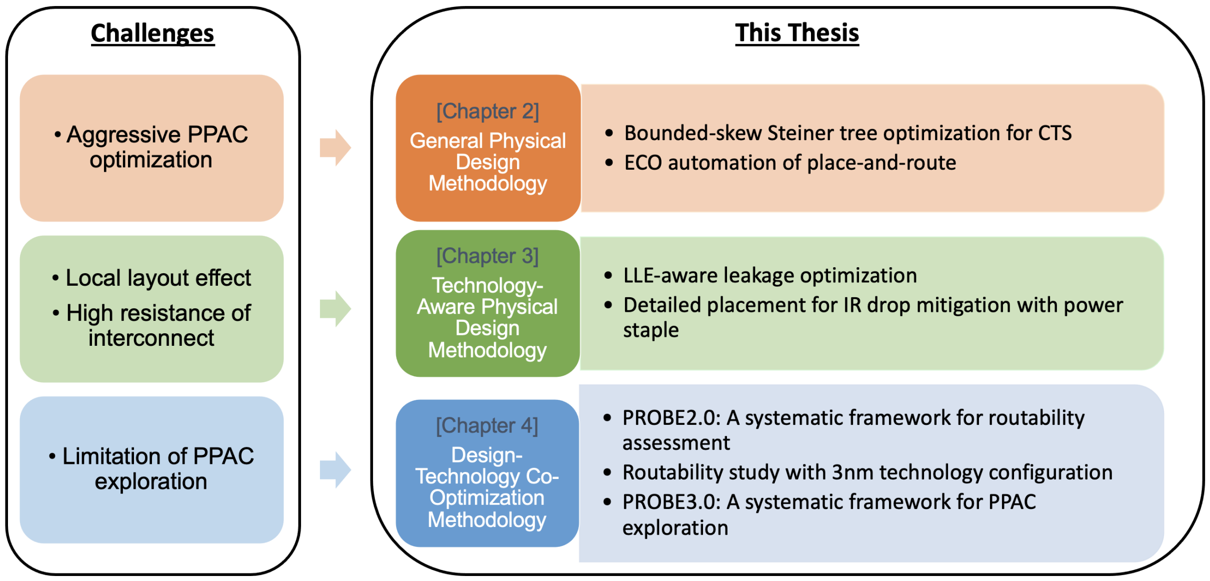

Challenges
• Aggressive PPAC optimization
• Local layout effect
• High resistance of interconnect
• Limitation of PPAC exploration
This Thesis
[Chapter 2] General Physical Design Methodology
• Bounded-skew Steiner tree optimization for CTS
• ECO automation of place-and-route
[Chapter 3] Technology-Aware Physical Design Methodology
• LLE-aware leakage optimization
• Detailed placement for IR drop mitigation with power staple
[Chapter 4] Design-Technology Co-Optimization Methodology
• PROBE2.0: A systematic framework for routability assessment
• Routability study with 3nm technology configuration
• PROBE3.0: A systematic framework for PPAC exploration

Figure 1.5. Overview of this thesis.

into three main categories, as illustrated in Figure 1.5.

- General physical design methodologies.
- Technology-aware physical design methodologies.
- Design-technology co-optimization (DTCO) methodologies.

First, to address the challenges for modern IC design, the *general physical design methodologies* chapter of this thesis presents approaches to optimize bounded-skew Steiner trees for clock tree synthesis to minimize active power, and to concurrently refine detailed placement and routing for efficient engineering change orders (ECO) automation.

Second, to address the challenges for advanced technology nodes, the *technology-aware physical design methodologies* chapter of this thesis presents approaches to optimize leakage power with the awareness of local layout effects, and to mitigate IR drop by detailed placement that enables more power staples to be inserted.

Last, to address challenges for DTCO, the *design-technology co-optimization* chapter of this thesis presents a systematic framework for routability assessments, a routability study by using the PROBE2.0 framework with 3nm technology configurations and a systematic framework for PPAC explorations with improved design enablements.

The reminder of this thesis is organized as follows.

- Chapter 2 presents two works in the context of general physical design methodologies. The first work addresses clock power optimizations in modern IC designs. With more memory hierarchies and various modules embedded in modern VLSI, on-chip memory accesses are increasingly required due to the high cost of off-chip memory accesses and the lower power density of memory fabrics. For memory-dominated chips, product specifications and electronic device designers focus on the maximum power consumption across all power usage scenarios, where a portion of memories are active and others are turned off by clock/power gates. We introduce and study *k-active dynamic power minimization* in bounded-skew trees, where we seek to minimize the maximum dynamic power consumption when at most *k* clock sinks are active in the memory-dominant designs. The experimental results give new insight into the tradeoff of maximum *k*-active dynamic power and wirelength, and of skew and wirelength. The second work addresses the challenges posed by aggressive area reduction in IC designs, which results in post-route DRCs caused by complex design rules at advanced technology nodes. We propose *CoRe-ECO*, a concurrent refinement framework for efficient automation of the ECO process. The framework efficiently resolves pin accessibility-induced DRCs by simultaneously performing detailed placement, detailed routing and cell replacement.

- Chapter 3 presents two works in the context of technology-aware physical design methodologies. The first work addresses *local layout effects* (LLE) in physical design. In a standard cell-based design, a diffusion break (DB) isolates neighboring devices and affects their delay and leakage power. In sub-10nm designs, the type of DB (single or double diffusion break) used in the library cell layout can significantly impact device performance and leakage. The 2^{nd} DB effect is a type of LLE in which the distance to the second-closest DB affects device performance. We develop 2^{nd} DB-aware leakage optimization and detailed placement heuristics. The second work addresses the high resistance of interconnects. High resistance of BEOL layers causes high IR drop at advanced nodes. To solve this problem, pre- or post-placed power staples are inserted in pin-access layers to connect adjacent power rails and reduce power delivery network resistance, at the cost of reduced routing flexibility, or reduced power staple insertion opportunity. We propose dynamic programming-based single- and double-row detailed placement optimizations to maximize the power staple insertion in a post-placement flow. We further propose metaheuristics to improve the quality of results.

- Chapter 4 presents three works in the context of design-technology co-optimization methodologies. We first present a systematic framework for routability assessments. The key challenge for DTCO is that large engineering efforts and long timelines are needed to develop design enablements and perform implementation studies in order to assess technology options. We describe a new framework that can systematically evaluate a measure of intrinsic routability, K_{th}, across both technology and design choices. We focus on routability since it is a critical factor in the scaling of area and cost. The experimental studies demonstrate the assessment of routability impacts for advanced-node technology and design options. We demonstrate learning-based K_{th} prediction to reduce runtime, disk space and commercial tool licenses needed to implement our framework. Also, we present a routability study with this framework with 3nm technology configurations to show block-level area scaling is reversed from cell-level scaling in cell height $<$150nm regime, using a machine learning (ML)-assisted DTCO methodology to optimize block-level area accounting for routability. Last, we present a systematic framework for design-technology pathfinding with improved design enablements. The goal of our work is to build a "configurable" and open-sourced PDK generation with promising scaling boosters, such as backside PDN and buried power rails, and study PPAC given technology and design parameters. In addition, we propose our ML-based parameter tuning flow and *clustering-based cell width-regularized placements* to generate more realistic artificial designs. We conduct experiments to present PPAC evaluation capabilities of the framework with our predictive technology and we achieve comparable results with studies from industry.

- Chapter 5 summarizes our conclusions in physical design and design-technology co-optimization.

Chapter 2

General Physical Design Methodology

This chapter presents two general physical design methodologies. The first work addresses clock power optimizations in modern IC designs. With more memory hierarchies and various modules embedded in modern VLSI, we introduce *k-active dynamic power minimization* problem in bounded-skew trees, where we seek to minimize the maximum dynamic power consumption when at most k clock sinks are active in the memory-dominant designs. The experimental results give new insight into the tradeoff of maximum k-active dynamic power and wirelength, and of skew and wirelength. The second work addresses the challenges posed by aggressive area reduction in IC designs, which results in post-route DRCs caused by complex design rules at advanced technology nodes. We propose *CoRe-ECO*, a concurrent refinement framework for efficient automation of the ECO process. The framework efficiently resolves pin accessibility-induced DRCs by simultaneously performing detailed placement, detailed routing and cell replacement.

2.1 Optimal Bounded-Skew Steiner Trees to Minimize Maximum k-Active Dynamic Power

Static Random-Access Memory (SRAM) is a key component of modern systems-on-chip (SOC), as a result of several inexorable trends. First, off-chip memory accesses entail long latencies and large energy consumption. Thus, modern SOCs have larger memory hierarchies

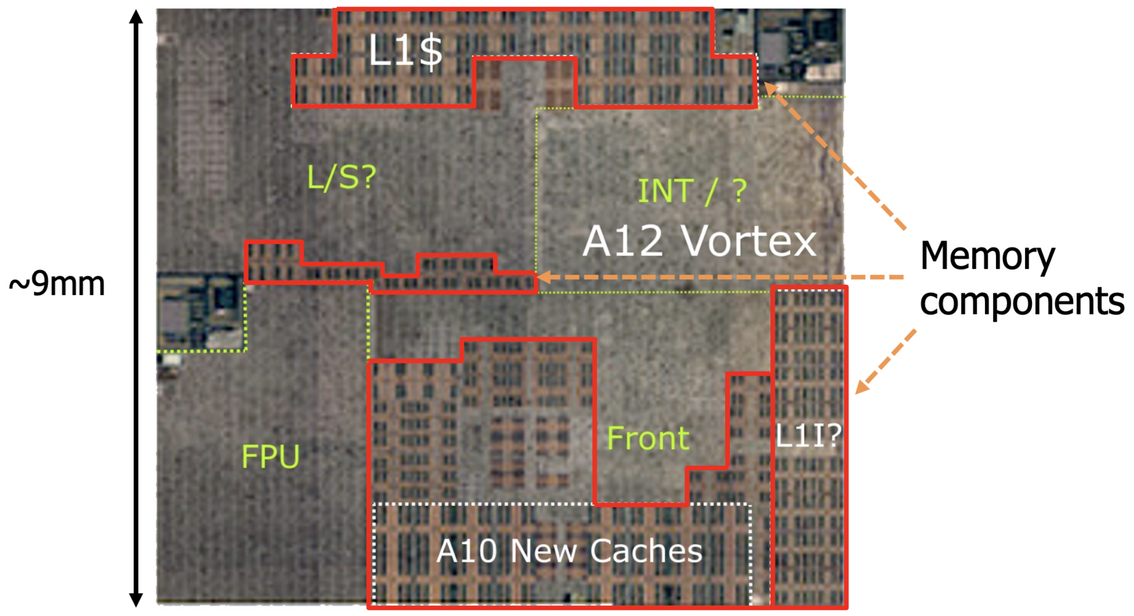


Figure 2.1. An example floorplan view of a memory-dominant chip [175]. The red boxes indicate memory components in the chip.

and integrate more modules, trading away high-cost off-chip memory accesses for lower-cost on-chip memory accesses. For example, microcontroller units contain multiple “masters”, such as central processing units (CPU), memory protection units (MPU), floating-point processing units (FPU), digital signal processors (DSP), etc. More than one of these masters may access memory components simultaneously in many applications. Second, the lower power density of memory fabrics can help reduce the need for “dark silicon”. Last, even as advanced technology nodes continue to be aggressively scaled, SRAM scaling has been slower than other (logic) scaling: device scaling (e.g., to single-fin transistors in 6T bitcells) causes susceptibility to variations and read/write instability [166]. Due to these trends, as well as emergence of new chip architectures for AI and machine learning, the area occupied by SRAM blocks now commonly approaches or exceeds 60% of the total die area in modern SOCs [43].

Figure 2.1 shows a die shot of a recent SOC product in the mobile application processor space. The reference [175] notes that the width and height of the SOC are around 9mm, and that there is significantly more memory content relative to the previous-generation product.

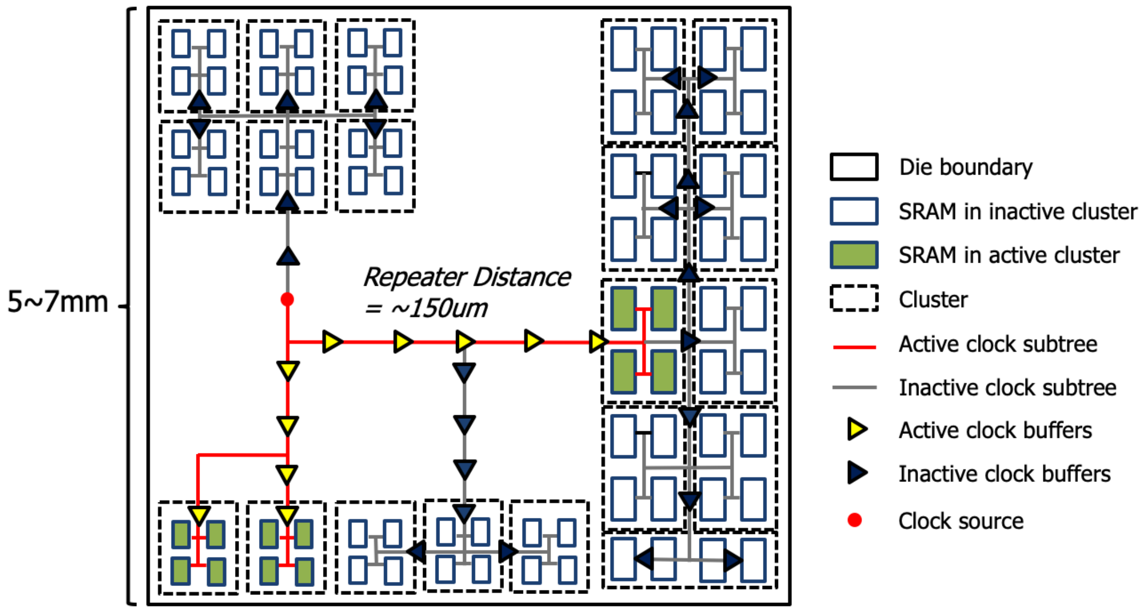


Figure 2.2. “Cartoon” of floorplan and clock tree structures for a memory-dominant SOC design, which motivates our *k*-active dynamic power minimization formulation.

Low-power design, always a significant challenge, becomes more important in modern SOCs. Clock distribution is an important aspect of any low-power methodology. And, reducing the power consumed by clock distribution to SRAMs is a growing concern in *memory-dominant* designs. Figure 2.2 shows a “cartoon” view of a *memory-dominant* SOC floorplan and a tree structure for clock distribution; the figure captures the physical design challenges of clock distribution to memory blocks. The figure also shows a motivation for our proposed *k*-active dynamic power minimization problem formulation.

With memory-dominant designs, memory clusters (sometimes referred to as “banks”) are defined by designers; SRAM instances in the same cluster are placed close together at the floorplan stage (four memories in a single cluster are highlighted in Figure 2.2). Importantly, in many low-power SOC products, the product specification guaranteed to OEM customers and other electronic product designers will focus on the **maximum** power consumption across all power usage scenarios wherein a portion of memories is “active” (marked in green in Figure 2.2) and all other memories are turned off by clock/power gates. Figure 30 of [157] shows current consumption of active memory partitions (clusters) for an MCU product. The figure shows that the maximum difference of current consumption across the partitions is around 30%. Thus, minimization of maximum power consumption for active memory clusters is a real and significant issue at the SOC product level. Figure 2.2 also illustrates the insertion of repeaters in long global segments of the clock distribution. In recent technology nodes, the repeater distance is typically at most 150 to 200μm, while die width and height can be 5 to 7mm or even more. The sizes of SRAM blocks and the SOC die, relative to repeater distances in advanced nodes, effectively "linearize" the calculation of clock power and buffered RC delay for top-level clock distribution.[1]

***k*-Active Dynamic Power Minimization Problem.** In usage scenarios where only *k* memory clusters are active and the rest are turned off by power/clock gates, the clock power consumption depends on the locations of active clusters and the wirelengths of active clock subtrees. Among all such scenarios, the worst-case power becomes the maximum *k-active dynamic power*. We thus propose and study the *k-active dynamic power minimization problem*, where we seek to minimize the maximum dynamic power consumption when at most *k* clock sinks are active. Based on the linearity of power consumption and buffered RC delay in top-level clock distribution, we can model *k*-active dynamic power and skew based on the union of source-sink paths to a given set of *k* active clock sinks.

***k*-Consecutive-Active Dynamic Power Minimization Problem.** To address potentially more realistic scenarios, we remove unrealistic power usage scenarios which overconstrain the optimization and result in suboptimal clock tree constructions. Figure 2.2 shows a scenario where there are two neighboring clusters, and a third cluster is placed far away from the others. Depending on the SOC architecture and fine-grain access given to applications, the optimization may not need to worry about scenarios where two simultaneously active clusters or sinks are placed far apart. We thus introduce the *k-consecutive-active dynamic power minimization problem* which simplifies constraints in the integer linear program (ILP) formulation that we develop below,

[1]Power and delay are linear in distance, when long global interconnects are buffered with a regular buffer interval [5].

since only sinks that are physically close can be active simultaneously. In our study below, we index clock sinks (SRAM clusters) based on geometric information, and establish k-active power constraints only for groups of k active sinks that have consecutive indices.

Flow-Based Integer Linear Program (ILP) for Bounded-Skew Tree. In the recent work of [51], the authors propose a flow-based ILP formulation to find optimal (spanning and Steiner) trees, given locations of clock source and sinks, along with skew constraints. We build on the work of [51] to address the k-active and k-consecutive-active dynamic power minimization problems. We also apply non-uniform grids to the previous ILP to obtain better results which more closely reflect routing channels in the SOC floorplan, and locations of memories. Finally, we study a scenario where we seek the optimal location of a *flexible* clock source rather than a fixed clock source; we show how to generally formulate the ILP for optimal tree construction when there are *flexible* and/or *forbidden* locations for the clock source.

We make the following contributions.

- Building on the work of [51], we use a flow-based integer linear programming (ILP) approach. We add new constraints to minimize a weighted sum of total clock distribution wirelengths and the maximum wirelengths of subtrees that reach k-active sinks. Our formulation addresses wirelengths of subtrees based on "k-active dynamic power".

- Our ILP formulation also addresses k-consecutive-active dynamic power of clock trees to remove unrealistic power usage scenarios.

- We apply non-uniform grids to the ILP problem to more accurately model terminal locations without incurring additional complexity increase.

- Going beyond pre-fixed clock source locations, we study effects of *flexible* as well as *forbidden* clock source locations. The former helps to reduce overall wirelength and wirelengths of subtrees with k-active sinks. The latter reflects scenarios where clock source locations are unavailable due to SOC floorplan constraints.

The remainder of this section is organized as follows. Section 2.1.1 summarizes related previous works. Section 2.1.2 describes our main problem formulation and its several variants. Section 2.1.3 describes the setup and results for our experiments. We conclude in Section 2.1.4.

2.1.1 Related Work

In this section, we review previous related works. Since we address power minimization for clock tree constructions, we first review the literature for zero-skew tree (ZST) and bounded-skew tree (BST) construction problems for clock trees. We also note several example approaches for low-power buffered clock tree construction in physical design.

Zero-Skew Tree (ZST) and Bounded-Skew Tree (BST) Construction. [10] [15] [38] [69] propose deferred-merge embedding (DME)-based zero-skew tree (ZST) constructions which subsequently became the foundation of many works for the bounded-skew tree construction problem. The works [30] [58] [70] address the bounded-skew Steiner tree construction problem with heuristic approaches. [19] uses hierarchical clustering for the ZST problem with dynamic programming. [132] proposes approximation algorithms for ZST and BST. Another part of the literature addresses optimal or approximate rectilinear Steiner minimal tree (RSMT) constructions. [3] proposes ILP (set covering) formulations to find a Steiner tree given a set of terminals and designated Steiner points. [98] finds optimal Steiner trees with specified topology satisfying lower and upper pathlength bounds, using linear programming. [121] proposes exact algorithms to find Steiner trees based on mathematical programming and dynamic programming. [28] proposes FLUTE, a fast lookup table-based RSMT algorithm. FLUTE is optimal when the number of terminals is less than ten, and for the range of practical instance sizes offers an extremely competitive runtime-quality tradeoff point for the RSMT problem. [51] studies the skew-cost tradeoff and suboptimality in previous interconnect tree constructions, and proposes an optimal flow-based ILP approach.

Low-Power Buffered Clock Tree. Many works have been proposed to minimize clock power in physical design. In some situations, considering only wirelengths in clock trees as a proxy

Table 2.1. Notations.

Notation	Meaning
p_i	i^{th} terminal ($p_i \in P$, p_1 is a root (source)).
v_i	i^{th} vertex ($v_i \in V$, $P \subseteq V$).
e_{uw}	A directed edge from vertex v_u to vertex v_w ($e_{uw} \in E$).
λ_{uw}	0-1 indicator of whether e_{uw} is in a tree T.
c_{uw}	Cost of edge e_{uw}.
f^i_{uw}	0-1 indicator of whether the flow to p_i goes through e_{uw}.
d^i_u	Pathlength at v_u along unique v_1-p_i path.
m_{uw}	Manhattan distance from v_u to v_w.
B	Skew bound.
L	Lower bound on pathlength from source v_1.
S_i	Pathlength of p_1-p_i path.
$S_{i,j}$	Pathlength of p_1-p_i-p_j tree.
$C_{i,j}$	Length of common edges of p_1-p_i path and p_1-p_j path.
$f^{i,j}_{uw}$	0-1 indicator of whether the flow to p_i and/or p_j goes through e_{uw}.
$PMax(1)$	Pathlength of the longest source-to-sink (p_i) path (p_1-p_i).
$PMax(2)$	Length of the longest source-to-two-sinks (p_i, p_j) subtree (p_1-$p_i \cup$ p_1-p_j).
M	A large constant.

for cost (power) is not enough for real-world designs due to buffers on interconnects. [120] addresses buffered clock power minimization by proposing an algorithm to design a tree topology and simultaneously insert buffers. [36] proposes a methodology for power-aware clock tree planning, named *LPClock*, which constructs clock trees heuristically and inserts clock gating cells to minimize clock power. [16] proposes a heuristic approach to construct a low-power zero-skew clock network by buffer and clock gate insertions, given terminal locations and activity information. By contrast, in this work, we focus on clock tree constructions in memory-dominant SOC designs. By considering SRAM blocks, SOC die sizes and buffer distances in advanced nodes, we observe that both the clock power and the signal propagation delay in clock subtrees can be modeled as linear phenomena.

We are not aware of previous works that study minimization of subtrees across subsets of k sinks. Below, we point out that traditional bounded-skew or skew-wirelength tradeoffs do not directly address this new objective. On the other hand, directly minimizing the cost of k-sink clock subtrees in memory-dominated SOC designs dimensions can reduce clock distribution power in actual operation of IC products. Our work can provide guidance for clock distribution in such designs.

2.1.2 Problem Formulation

In this section, we describe our problem formulation. First, we review the formulation of [51]; we then modify it to address the k-active dynamic power minimization problem. Second, we formulate the k-consecutive-active dynamic power minimization problem. Third, we incorporate non-uniform grids into the ILP formulation to more flexibly reflect clock source and sink locations. Last, we propose both *flexible* and *forbidden* clock source locations to further guide the choice of optimal clock source locations for reduced tree cost and/or subject to layout constraints. Table 2.1 shows the notations that we use in what follows.

Review of the ILP Formulation of [51]

We now review the ILP formulation for optimal bounded-skew spanning and Steiner tree construction in the work of [51]. Let x_i and y_i be coordinates of terminal p_i in the Manhattan plane. P denotes a set of terminals p_i over which a tree is to be constructed, where $i = \{1, 2, ..., |P|\}$. By convention, p_1 is a root (source) of the tree and the other terminals are leaves (sinks). A graph $G = (V, E)$ is constructed which includes terminals in P along with the Hanan grid [53] and additional *half-integer* grids (centered lines) between the predefined Hanan grids.[2] Every point at the intersection of two gridlines corresponds to a vertex $v_i \in V$, and directed edges e_{uw} from vertex v_u to vertex v_w are defined where v_u and v_w are neighboring vertices in the grid. Thus, each vertex v_i can have up to four incoming edges and four outgoing edges. The objective is to minimize the total cost of edges (i.e., total wirelength) in the constructed tree T.

[2]In the work of [51], optimal spanning and Steiner trees are constructed on Hanan grids or half-integer grids.

Minimize: $\sum_{u,w} \lambda_{uw} \cdot c_{uw}$

Subject to:

$$\lambda_{uw} \geq f_{uw}^{i}, \qquad \forall p_i \in P,\ e_{uw} \in E \tag{2.1}$$

$$\sum_{u} f_{uw}^{i} - \sum_{u} f_{wu}^{i} = \begin{cases} 1 & \text{if } v_w = v_1, \forall v_u \in V,\ p_i \in P,\ i \neq 1 \\ -1 & \text{else if } v_w = v_i \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

$$\sum_{u,w} c_{uw} \cdot f_{uw}^{i} \geq L, \quad \forall p_i \in P,\ i \neq 1 \tag{2.3}$$

$$\sum_{u,w} c_{uw} \cdot f_{uw}^{i} \leq L + B, \quad \forall p_i \in P,\ i \neq 1 \tag{2.4}$$

Given a set of terminals P including source p_1, and pathlength lower and upper bounds L and $L+B$, where B is a skew bound, the ILP finds a spanning or Steiner tree over P that satisfies the pathlength bounds while minimizing $\sum_{u,w} \lambda_{uw} \cdot c_{uw}$.

f_{uw}^{i} denotes a binary indicator of whether the flow to p_i goes through e_{uw}. Constraint (2.1) enforces $\lambda_{uw} = 1$ when any flow goes through e_{uw}. Constraint (2.2) enforces flow conservation at all vertices except for the source (where the outgoing flow exceeds the incoming flow by the number of sinks) and any sink (where the sum of incoming flow exceeds the outgoing flow by one).

Skew bounds are enforced by Constraints (2.3) and (2.4), which apply the lower bound (L) and the upper bound ($L+B$) to each source-to-sink pathlength.

$$\begin{cases} d_u^i = 0 & \text{if } v_u = v_1, \forall\, p_i \in P,\ i \neq 1 \\ d_u^i \geq L & \text{else if } v_u = v_i, \forall\, p_i \in P,\ i \neq 1 \\ d_u^i \leq L + B & \text{otherwise } \forall\, p_i \in P,\ i \neq 1 \end{cases} \tag{2.5}$$

$$d_u^i + M \cdot (1 - f_{wu}^i) \geq d_w^i + c_{wu},\ \ \forall v_u, v_w \in V, u \neq w,\ p_i \in P \tag{2.6}$$

$$d_u^i - M \cdot (1 - f_{wu}^i) \leq d_w^i + c_{wu},\ \ \forall v_u, v_w \in V, u \neq w,\ p_i \in P \tag{2.7}$$

Constraints (2.5) to (2.7) prevent cycles in the output, i.e., they enforce a tree solution topology. Without these constraints, cycles can be introduced by detouring to satisfy skew constraints. To avoid cycles, we define a variable d_u^i which denotes pathlength from the source to vertex v_u for a path to sink p_i. For Constraint (2.5), the pathlength is zero when vertex v_u is the source. The pathlength must be maintained between lower and upper bounds for any vertices. Constraints (2.6) and (2.7) are added for pathlength d_u^i. When $f_{wu}^i = 1$, a pathlength increases by the edge cost c_{wu}, which means that d_u^i will be infinite in the presence of a cycle, and will not satisfy Constraint (2.5).

$$\text{if } m_{1u} + m_{ui} > L + B,\ \ \forall v_u \in V,\ u \neq 1, ..., |P|,\ p_i \in P$$
$$f_{uw}^i = 0,\ f_{wu}^i = 0,\quad \forall e_{uw} \in E,\ e_{wu} \in E \tag{2.8}$$

$$\text{if } m_{1u} + m_{uu'} + m_{u'i} > L + B \;\;\&\&\;\; m_{1u'} + m_{u'u} + m_{ui} > L + B$$

$$f^i_{uw} + f^i_{u'w'} = 1, \quad f^i_{wu} + f^i_{u'w'} = 1, \; \forall e_{uw} \in E, \; e_{wu} \in E \tag{2.9}$$

$$f^i_{uw} + f^i_{w'u'} = 1, \quad f^i_{wu} + f^i_{w'u'} = 1, \; \forall e_{u'w'} \in E, \; e_{w'u'} \in E \tag{2.10}$$

Constraints (2.8) to (2.10) are added to improve ILP runtime by special handling of vertices and edges that are irrelevant to the desired tree solution. Constraint (2.8) makes flow variables f^i_{uw} and f^i_{wu} zero when the sum of the Manhattan distances from source p_1 to vertex v_u and from vertex v_u to sink p_i is larger than the upper bound $L+B$. This is because the pathlength of path p_1-p_i going through vertex v_u is always larger than the upper bound. Constraints (2.9) and (2.10) address a forbidden scenario involving two vertices. When the sum of the pathlengths from p_1 going through non-terminal vertices v_u and v'_u to sink p_i is larger than the upper bound $L+B$, then any flows for incoming and outgoing edges going through v_u and v'_u cannot exist simultaneously.

k-Active Dynamic Power Minimization

We extend the approach of [51] to address the k-active dynamic power minimization problem, as follows. We first modify the objective function from the ILP formulation, then we add six constraints to capture k-active dynamic power minimization problem for $k = 1$ and $k = 2$. Our objective function is to minimize the weighted sum of total cost of edges in an output tree T and the maximum k-active dynamic power. For k points other than p_1, we say that the k-active dynamic power is the total cost of edges in the union of the corresponding k source-to-sink paths in an output tree T. In addition, for a given output tree T, we define the maximum k-active dynamic power, $PMax(k)$, to be the maximum k-active dynamic power over all k-subsets of $\{p_2, \ldots, p_{|P|}\}$. Thus, our objective function and constraints can be formulated as follows.

$$\textbf{Minimize:}\quad \sum_{u,w} \lambda_{uw} \cdot c_{uw} + \alpha \cdot PMax(1) + \beta \cdot PMax(2)$$

Given a set of terminals P including source p_1, pathlength bounds L and $L+B$, and parameters α and β, we construct a Steiner tree over P that satisfies the pathlength bounds while minimizing $\sum_{u,w} \lambda_{uw} \cdot c_{uw} + \alpha \cdot PMax(1) + \beta \cdot PMax(2)$. The first term of the objective function is the total cost of the edges. $\lambda_{uw} = 1$ when edge $e_{u,w}$ is included in the output tree T. The second and third terms are the weighted k-active dynamic power where $k = 1, 2$ respectively. Based on the principle of inclusion-exclusion, we can extend to any k-active costs from $k = 1$ and $k = 2$, at the cost of additional wirelength summation terms and constraints that govern k-tuples of sinks.

$$S_i \leq PMax(1),\ \forall p_i \in P \tag{2.11}$$

$$S_{i,j} \leq PMax(2),\ \forall p_i, p_j \in P \tag{2.12}$$

$$\sum_{u,w} c_{uw} \cdot f^i_{uw} \leq S_i,\ \forall p_i \in P \tag{2.13}$$

$$S_i + S_j - C_{i,j} \leq S_{i,j},\ \forall p_i, p_j \in P \tag{2.14}$$

$$2 \cdot f^{i,j}_{uw} \leq f^i_{uw} + f^j_{uw} \tag{2.15}$$

$$\sum_{u,w} c_{uw} \cdot f^{i,j}_{uw} = C_{i,j},\ \forall p_i, p_j \in P \tag{2.16}$$

Constraints (2.11) to (2.16) are added for the $PMax(k)$ computation. In Constraint (2.11), S_i denotes a pathlength from the source p_1 to sink p_i and $PMax(1)$ denotes the maximum pathlength among all S_i. In Constraint (2.12), $S_{i,j}$ denotes a length of a source-sinks (p_i and p_j) subtree and $PMax(2)$ denotes the longest pathlengths among source-to-two-sinks (p_i and p_j) subtrees. Constraint (2.13) computes pathlength S_i from p_1 to p_i. In Constraint (2.14), $C_{i,j}$ is the common pathlength between the source-to-sink paths for p_i and p_j. The constraint is therefore with respect to the total edge length of a source-to-two-sinks (p_i and p_j) subtree. In Constraint (2.15), a new flow variable $f_{uw}^{i,j}$ is defined, which indicates whether edge e_{uw} is common to both of the source-to-sink paths to p_i and p_j. Constraint (2.16) computes common pathlength $C_{i,j}$ for sinks p_i and p_j.

k-Consecutive-Active Dynamic Power Minimization

In this section, we describe a k-consecutive-active dynamic power minimization problem. With this variant formulation, we capture locality of simultaneous memory accesses, and can thus relax constraints for unrealistic power usage scenarios which overconstrain the optimization.

In our study, we index clock sinks based on geometric information and consider locality according to adjacent indices. We treat the clock source as the origin, and associate each clock sink with its theta angle with respect to this origin. Sinks are indexed (in counterclockwise order) by theta angle.

We do not add new constraints to the ILP. Instead, we replace four constraints with the following:

$$S_{i,i+1} \leq PMax(2),\ \forall p_i, p_{i+1} \in P \tag{2.17}$$

$$S_i + S_{i+1} - C_{i,i+1} \leq S_{i,i+1},\ \forall p_i, p_{i+1} \in P \tag{2.18}$$

$$2 \cdot f_{uw}^{i,i+1} \leq f_{uw}^{i} + f_{uw}^{i+1} \tag{2.19}$$

$$\sum_{u,w} c_{uw} \cdot f_{uw}^{i,i+1} = C_{i,i+1},\ \forall p_i, p_{i+1} \in P \tag{2.20}$$

For the k-consecutive-active dynamic power minimization problem, Constraints (2.17), (2.18), (2.20) and (2.19) replace Constraints (2.12), (2.14), (2.15) and (2.16). In this way, constraints from the original k-active dynamic power minimization problem are relaxed for the k-consecutive-active dynamic power minimization problem. Instead of addressing all the possible combinations of k sinks, the relaxed constraints only address local groups of k sinks.

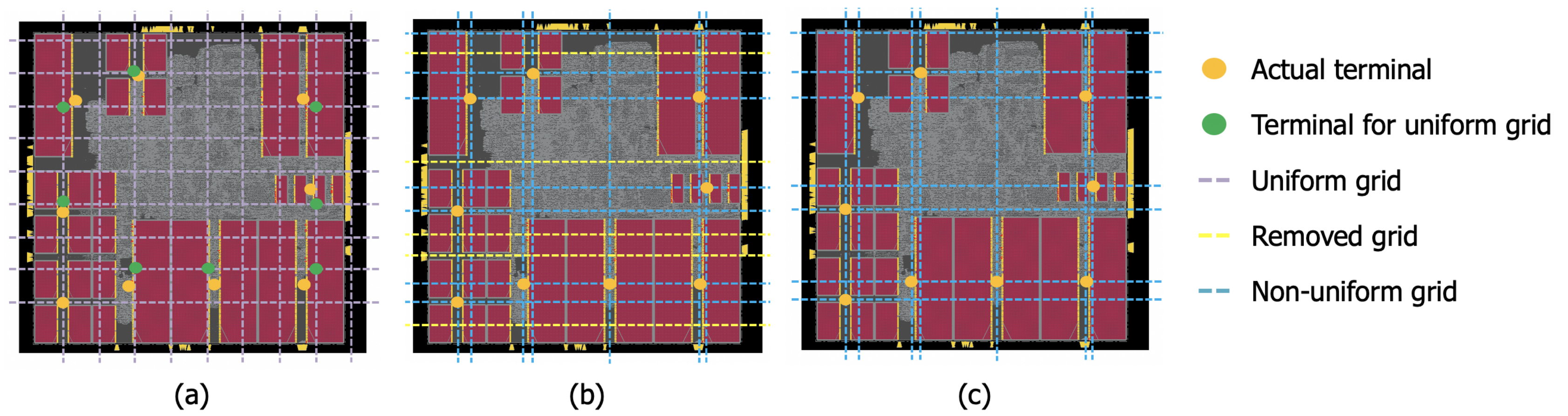


Figure 2.3. Terminals with uniform and non-uniform grids on an example floorplan. (a) Terminals with uniform grids. (b) Terminals with non-uniform grids based on channel intersection graph. (c) Terminals with non-uniform grids after the unusable grids.

Non-Uniform Grid

In this section, we demonstrate how non-uniform underlying grids enable the ILP to more flexibly capture locations of terminals and available routing resources in the Steiner tree construction. A non-uniform grid is created for a given floorplan and macro (SRAM) placement.[3] We consider channels between macros as well as clock pin locations (or midpoints of macros in clusters) of macros in our non-uniform grids; in the limiting case, the grid can be said to approach a *channel intersection graph* [33]. Removal of unusable edges in the grid can make the problem sparser and improve ILP runtimes.

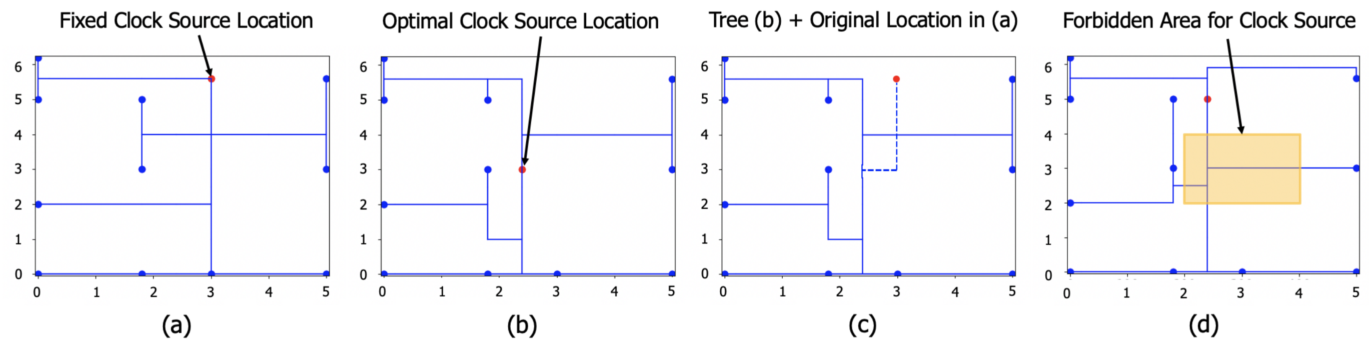
Figure 2.3 shows uniform and non-uniform grids on an example floorplan from a RISC-V based design.[4] Figure 2.3(a) shows uniform grids with locations of terminals. Note that we would like to accurately capture the locations of clock pins (terminals) on SRAMs. However, in a uniform grid these locations must be snapped to nearest grid intersections. This causes a discrepancy between desired terminal locations and the terminal locations actually used on the underlying uniform grids, leading to suboptimal tree solutions. On the other hand, Figure 2.3(b) shows a non-uniform grid based on the macro channels and clock source locations, with removal of unusable grids where terminals are not located. Figure 2.3(c) shows non-uniform grids after removing unusable grids. In this way, we can use desired terminal locations of clock pins for our tree construction and have more flexibility of terminal locations without adding more complexity to the ILP.

Flexible or Forbidden Clock Source Location

Given fixed locations of a clock source and sinks, we find an optimal solution to minimize costs while skew constraints are satisfied. However, in real designs, we might be able to move a clock source location unless fixed locations are hard constraints. For example, if a clock

[3]As in [51], a graph is constructed based on the Hanan grid and additional centered lines between the predefined grids.

[4]We synthesize netlists of *SweRV_wrapper* design [170] [176] using [171], and perform place-and-route using [144], in a foundry 14nm FinFET enablement. We perform macro placement by using [144] and manually adjust the macro placement. The design has 28 macros and 74K standard-cell instances.


Fixed Clock Source Location
Optimal Clock Source Location
Tree (b) + Original Location in (a)
Forbidden Area for Clock Source
(a)
(b)
(c)
(d)

Metric	(a)	(b)	(c)	(d)
WL	25.6	24.4	27.6	25.4
Skew	5	2	2	4.1
PMax(1)	8.6	5.6	8.8	7.6
PMax(2)	12.2	11.2	14.4	11.4
Cost	40.3	35.6	43.6	38.7

Figure 2.4. Example trees for flexible clock sources. (a) A tree with a fixed clock source. (b) A tree with a movable clock source, with optimal source location. (c) A tree with the tree from (b) and the original source location from (a). (d) A tree with a movable clock source that has a forbidden area.

generator is placed and fixed at a specific area or pin assignments for top-level implementations are given, we cannot move the clock source location. Otherwise, we can achieve better tree solutions by moving clock source locations. To this end, we describe enablement of flexible clock source locations.

To allow all vertices (except sinks) to be sources, we define a clock source at a new virtual vertex v_0. In the graph $G = (V, E)$ constructed, we add virtual edges from the virtual vertex v_0 to every vertex v_i. Thus, each vertex v_i has up to five incoming edges and four outgoing edges. We then set the cost of each virtual edge to zero. The vertex that ends up connected to the virtual vertex through a virtual edge becomes the actual clock source. We define two constraints for flexible clock source locations, as follows.

$$\sum_{0,u} \lambda_{0u} = 1, \ \forall e_{0u} \in E \tag{2.21}$$

$$\lambda_{0u} = 0, \ \forall v_u = v_i, p_i \in P, i \neq 1 \tag{2.22}$$

Constraint (2.21) ensures that only one virtual edge is used in tree solution T. Constraint (2.22) is to avoid flows from a virtual vertex (clock source) to any sinks with zero cost.

Figure 2.4 shows example trees ($|P| = 12$) with a fixed clock source and a movable clock source. Figure 2.4(a) shows an optimal tree with the fixed clock source at (3, 5.6). The total wirelength of the tree solution is 25.6 when lower bound $L = 3.44$ and upper bound $L + B = 8.6$. In this example, we set $\alpha = 1, \beta = 0.5$ in the weighted sum of total wirelength and k-active dynamic power. Given a virtual clock source, the tree solution in (b) makes a connection from the virtual source to the vertex at (2.4, 3), which becomes the actual clock source. Figure 2.4(c) shows a combined tree with the tree from Figure 2.4(b) and the original clock source location from Figure 2.4(a). As seen in the data table of Figure 2.4, the tree of Figure 2.4(c) has larger cost than the tree of Figure 2.4(a), as we expect (tree (a) is optimal for this particular source location).

On the other hand, even if clock sources can be flexibly moved, they might not be placeable at certain locations due to geometric constraints. To address this, the ILP can be modified to reflect a "forbidden area" wherein clock sources cannot be placed. The following constraint is added to define the forbidden area for clock sources.

$$\lambda_{0u} = 0,\ \forall v_u \in V_r, \tag{2.23}$$

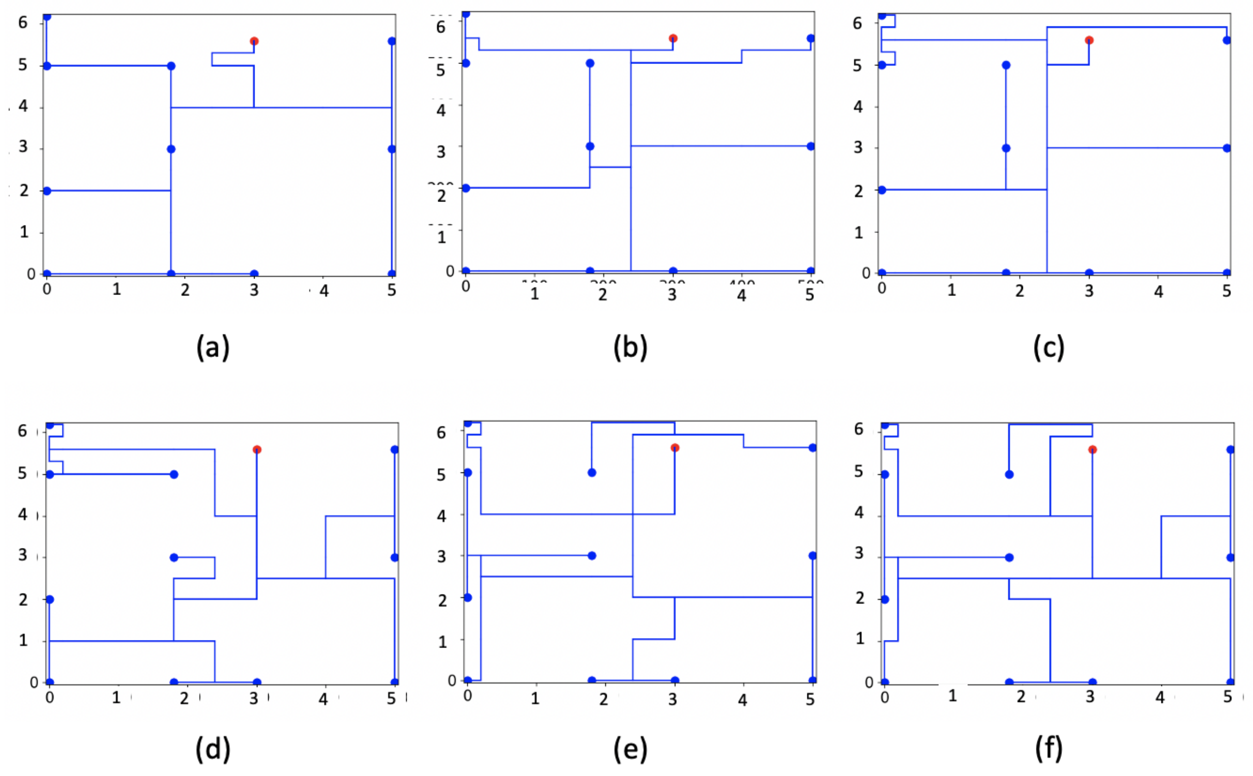
Constraint (2.23) prevents connections from the virtual edge to any vertices within a given forbidden area, where V_r denotes the set of vertices in the forbidden area. Figure 2.4(d) shows an example tree with flexible clock source locations and a forbidden area for the source. In this example, we set the forbidden area to be $2 \le x \le 4, 2 \le y \le 4$ – e.g., reflecting a situation where clock pins or clock generator cannot be placed in the middle of the die area.

2.1.3 Experimental Setup and Results

Experimental Setup

We implement our work in C++ with CPLEX 12.8.0 [155] and Gurobi 9.0.1 [153] as our ILP solvers. Our experiments are performed with four threads on a 2.6GHz Intel Xeon server. In our objective function, we set the weighting factors $\alpha = 0, 1, 100$ and $\beta = 0, 0.5, 100$. For lower bounds, we set lower bound $L = \{4,5,6,7,8,9\}$. For our runtime study, we set lower bound $L = M - B$ to avoid infeasiblility, where M denotes the maximum Manhattan distance between source and sink. Also, we define the skew bound B using M multiplied by 0.2, 0.4, 0.6, 0.8 (i.e., $B = 0.2 \cdot M, 0.4 \cdot M, 0.6 \cdot M, 0.8 \cdot M$). We also set unbounded cases with zero lower bound and infinite upper bound. In our experiments, we set two hours as the runtime limit for ILP.[5]

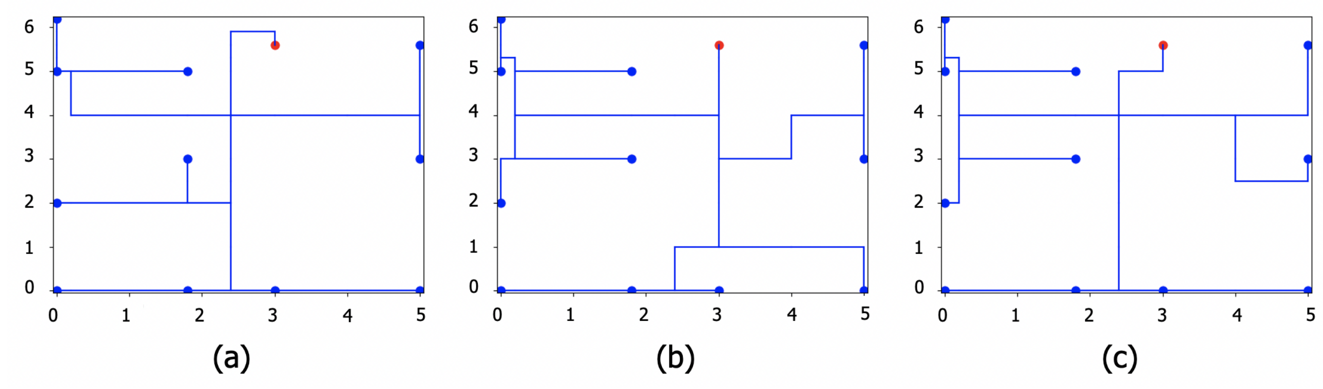
[5]The runtime limit is not applied to our runtime study in Table 2.2.



Metric	(a)	(b)	(c)
WL	24.4	26.3	27.3
Skew	4.8	4.7	3.8
PMax(1)	9.8	8.8	8.8
PMax(2)	15.8	12.1	12.8

Metric	(e)	(f)	(g)
WL	32.4	36.3	38.6
Skew	1.6	1.8	1.5
PMax(1)	8.6	8.8	8.6
PMax(2)	15.6	14.0	14.4

Figure 2.5. An example set of non-dominated trees for a 12-terminal pointset. The weighting factors and bounds are shown as (α, β, L, B). (a) (0, 0, 5, $0.6 \cdot M$). (b) (0, 100, 4, $0.6 \cdot M$). (c) (0, 100, 5, $0.6 \cdot M$). (d) (100, 0, 7, $0.4 \cdot M$). (e) (0, 100, 7, $0.4 \cdot M$). (f) (0, 100, 7, $0.2 \cdot M$).



Metric	(a)	(b)	(c)
WL	**26.6**	28.7	28.7
Skew	3.2	2.4	2.6
PMax(1)	9.4	**8.6**	8.8
PMax(2)	14.6	14	**13.6**

Figure 2.6. Example trees with various weighting factors in our objective function. (a) A tree solution to minimize the total wirelength. (b) A tree solution to minimize $PMax(1)$. (c) A tree solution to minimize $PMax(2)$.

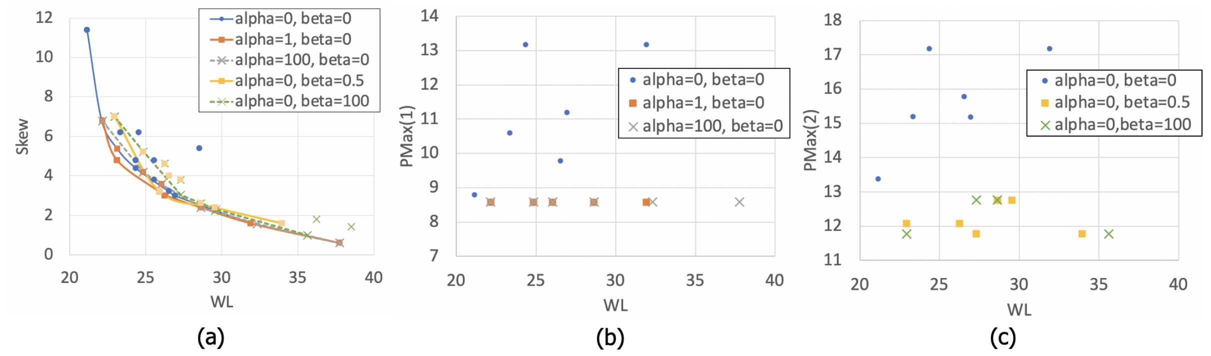


Figure 2.7. Scatter plots for (a) skew-wirelength, (b) $PMax(1)$-wirelength and (c) $PMax(2)$-wirelength for the 12-terminal pointset with various weighting factors α and β. Additional points that do not correspond to the figure legend are obtainable with other values of α, β, L and B.

Experimental Results

Study of Weighting Factors (α and β) in an Objective Function. We show example trees and scatter plots for skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength tradeoffs with various weighting factors α, β in our objective function. When $\alpha = \beta = 0$, the results are the same as those of [51]. In this way, we can compare our results to [51]. We observe that in practice, there are not many *non-dominated* trees for a given instance, across all values of α, β, lower and upper bounds. A tree T_1 is said to be *dominated* by tree T_2 if the wirelength, skew, $PMax(1)$ and $PMax(2)$ metrics of T_2 are all equal to or better than those of T_1. Figure 2.5 shows an example set of non-dominated solution trees for one instance in our experiment.

Figure 2.6 shows example trees with various weighting factors in our objective function. Given a 12-terminal pointset, lower bound $L = 6$ and skew bound $B = 0.4 \cdot M$, we find tree solutions where $\alpha = \beta = 0$, $\alpha = 100, \beta = 0$ and $\alpha = 0, \beta = 100$, respectively. We set lower bound $L = 4, 5, 6, 7, 8, 9$ in this experiment. Figure 2.6(a) shows the tree solution which minimizes the total wirelength of the tree. Figure 2.6(b) shows the tree solution which minimizes $PMax(1)$. The solution has the minimum total wirelength among all solutions with minimum $PMax(1)$. Compared to Figure 2.6(a), $PMax(1)$ decreases from 9.4 to 8.6 while the total wirelength increases. Figure 2.6(c) shows the tree solution which minimizes $PMax(2)$. The solution has the minimum total wirelength among all solutions with minimum $PMax(2)$. Compared to

Figure 2.6(a), $PMax(2)$ decreases from 14.6 to 13.6 while the total wirelength increases. From this example, we can see that solution trees can be optimized for $PMax(1)$ and $PMax(2)$ by varying the weighting factors in the objective function.

Figure 2.7 shows scatter plots for skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength. Figure 2.7(a) shows tradeoff curves with various weighting factors. Compared to the cases that minimize only total wirelength ($\alpha = 0$ and $\beta = 0$, i.e., as in the work of [51]), the tradeoff curves for nonzero weighting factors form a "bundle" of tradeoff curves. In terms of the skew-wirelength tradeoff, we see that considering k-active sinks does not affect skew much. Figure 2.7(b) gives a scatter plot for $PMax(1)$-wirelength, showing that $PMax(1)$ decreases when α is nonzero. Similarly, Figure 2.7(c) gives a scatter plot for $PMax(2)$-wirelength, showing that $PMax(2)$ decreases when β is nonzero. Figure 2.7(b) includes the solutions with nonzero α and Figure 2.7(c) includes the solutions with nonzero β to show improved $PMax(1)$ and $PMax(2)$, respectively.

Study of *k*-Consecutive-Active Sinks. As described above, we index sinks according to theta angle about the clock source, in counterclockwise order. Figure 2.8 shows scatter plots for skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength tradeoffs with k-active sinks and k-consecutive-active sinks. We set weighting factors $\alpha = 1, \beta = 0.5$ and lower bound $L = 4,5,6,7,8,9$. Figures 2.8(a) and (b) respectively show skew-wirelength and $PMax(1)$-wirelength for k-consecutive-active sink solutions, similar to what was presented for k-active sink solutions. Figure 2.8(c) shows that $PMax(2)$ values of k-consecutive-active sinks are less than those of k-active sinks. This is because $PMax(2)$ of k-consecutive-active sinks only considers local subtrees with consecutive indices of clock sinks. (Note that the k-consecutive-active sinks optimization will lead to different solutions than the k-active sinks optimization only when the β weighting term for $PMax(2)$ is nonzero.)

Study of Flexible or Forbidden Clock Source Locations. Figure 2.9 shows skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength scatter plots for flexible and fixed clock source locations for the 12-terminal pointset in Figure 2.4. In this experiment, we set weighting factors

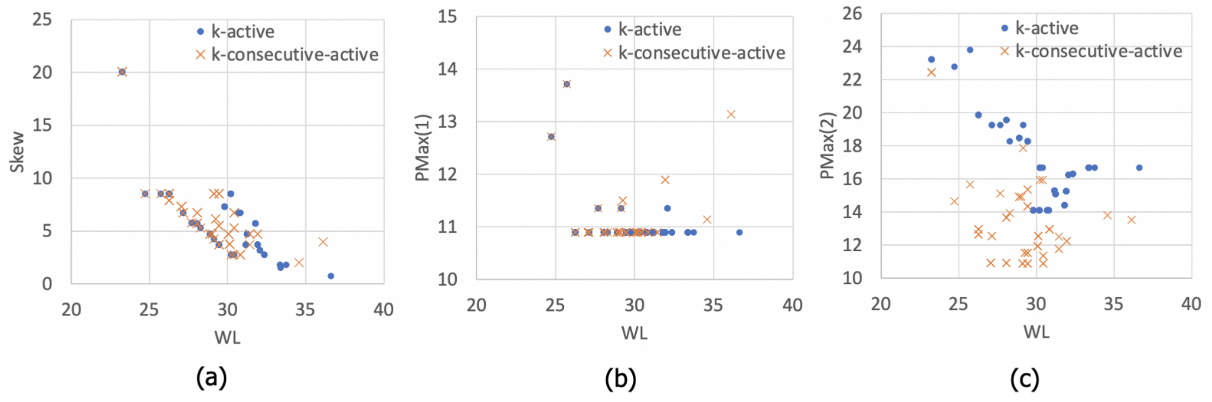


Figure 2.8. Scatter plots for (a) skew-wirelength, (b) $PMax(1)$-wirelength and (c) $PMax(2)$-wirelength for k-active sinks and k-consecutive-active sinks.

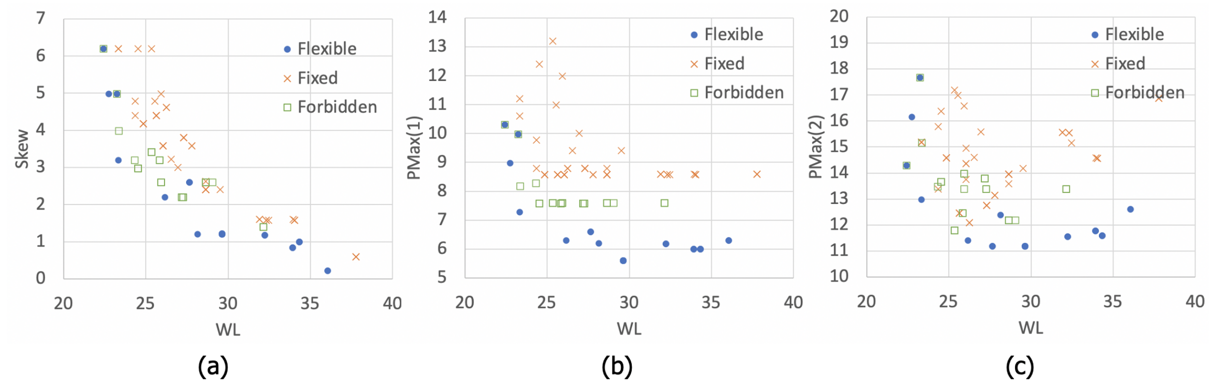


Figure 2.9. Scatter plots for (a) skew-wirelength, (b) $PMax(1)$-wirelength and (c) $PMax(2)$-wirelength for *fixed* and *flexible* clock source locations, as well as flexible clock source locations with a *forbidden* $2 \leq x \leq 4, 2 \leq y \leq 4$ area.

$\alpha = 1, \beta = 0.5$ and lower bound $L = 4,5,6,7,8,9$. Figures 2.9(a) to (c) shows that tree solutions with flexible clock source locations have better metrics (total wirelength, $PMax(1)$ and $PMax(2)$) than those with fixed clock source locations. Adding a forbidden area (reducing flexibility) worsens results in (c), but skew-wirelength, $PMax(1)$-wirelength and $PMax(2)$-wirelength remain superior to when the clock source location is fixed.

Study of Runtime. In order to compare ILP runtime, we define our pointsets with $P = \{8, 10, 12, 14, 16\}$, where $|P|$ is the number of terminals of trees. For this experiment, for each $|P|$ we randomly generate 10 sets of terminals within a seven by seven grid. We also generate random grid edge costs in horizontal and vertical directions (i.e., between consecutive

Table 2.2. Average ILP runtime when $\alpha = 1, \beta = 0.5, L = M - B$. M denotes the maximum distance between source and sink.

ILP solver	**Skew bound**	$\lvert P\rvert = 8$	$\lvert P\rvert = 10$	$\lvert P\rvert = 12$	$\lvert P\rvert = 14$	$\lvert P\rvert = 16$
Solver A	Unbounded	9.30	37.21	151.23	288.77	488.97
	$0.8 \cdot M$	10.07	12.17	32.69	56.57	100.31
	$0.6 \cdot M$	22.63	31.34	66.38	110.35	175.87
	$0.4 \cdot M$	71.29	96.41	153.19	1082.44	2454.79
	$0.2 \cdot M$	20.19	638.81	2912.25	14979.97	35566.68
Solver B	Unbounded	5.9	90.57	329.92	642.07	1454.97
	$0.8 \cdot M$	2.43	10.78	68.09	156.59	567.08
	$0.6 \cdot M$	7.87	91.53	619.80	757.56	1583.59
	$0.4 \cdot M$	96.93	1606.85	2091.18	11528.67	27122.51
	$0.2 \cdot M$	37.02	17800.59	24265.66	74113.28	107977.12

x- or y-coordinate values), from a uniform distribution over the interval [0.5, 2.0]. We use two different ILP solvers [153] [155] in this experiment.[6]

Table 2.2 shows the average ILP runtime. As shown in Table 2.2, ILP runtime increases when the number of terminals increases in most cases. Even though there are outliers due to infeasiblity of some of the problems, runtime increases when skew constraints get tighter (i.e., smaller skew bound).[7] The runtime for Solver A is smaller than for Solver B in most of the cases.

We also compare runtime for fixed and flexible clock source locations with non-uniform grids. Table 2.3 shows ILP runtime for fixed and flexible clock source locations with two different non-uniform grids, defined as follows. *Sparse* denotes non-uniform grids after removal of unusable edges as in Figure 2.3(c). *Dense* denotes non-uniform grids where unusable edges are retained, as in Figure 2.3(b). Due to the inclusion of a virtual source and edges, runs with flexible clock source location have longer runtimes than corresponding runs with fixed locations. We see that ILP solution runtime is reduced by removing unusable edges.

[6] In order to avoid benchmarking the tools, below we report tool names only as Solver A and Solver B.

[7] We cannot explain why the runtime for unbounded cases is smaller than for any bounded cases in [51]. In our studies, if we remove constraints for lower and upper bounds, the runtime slightly increases compared to the cases of $B = 0.8 \cdot M$ in Table 2.2.

Table 2.3. ILP runtime for fixed and flexible clock source locations in Figure 2.4 and non-uniform grids. We set variables $\alpha = 1, \beta = 0.5, L = M - B$. M denotes the maximum distance between source and sink.

Skew bound	**Sparse**		**Dense**	
	Fixed	**Flexible**	**Fixed**	**Flexible**
Unbounded	95.82	278.87	1964.88	2073.29
$0.8 \cdot M$	7.13	132.41	117.19	1581.06
$0.6 \cdot M$	34.89	432.06	500.11	3280.04
$0.4 \cdot M$	143.31	36781.35	2607.33	163050.29

2.1.4 Conclusion

In this work, we have proposed a new k-active dynamic power minimization problem that arises in clock distribution for memory-dominant SOC designs. We observe that in modern technologies the scale of SRAM blocks and SOC die, relative to global repeater distances, effectively "linearizes" the problem and makes it amenable to combinatorial optimization.

By extending the recent work of [51], we formulate an ILP for the k-active dynamic power minimization problem as well as a k-consecutive-active dynamic power minimization variant, The ILP minimizes the weighted sum of the total costs (wirelengths) and the maximum power across all sets of k active sinks. We also propose non-uniform grids to capture flexible locations of terminals in an output tree, and the handling of both flexible and forbidden clock source locations to improve solution quality and solver runtime. Our experimental results give new insights into the tradeoff for maximum k-(consecutive)-active dynamic power and wirelengths of subtrees as well as skew and total wirelengths.

Our future directions include (1) implementations of our tree solutions as a guide for clock tree synthesis in commercial place-and-route methodologies; (2) heuristics to combine ILP and, e.g., problem decomposition strategies to address larger-scale instances of the k-(consecutive)-active dynamic power minimization problem; and (3) runtime improvements of ILP solution, particularly with flexible clock tree locations.

2.2 CoRe-ECO: Concurrent Refinement of Detailed Place-and-Route for an Efficient ECO Automation

With the relentless scaling toward advanced technology nodes, increasingly sophisticated IC fabrication constraints (e.g., fewer routing tracks, higher pin density and complicated conditional design rules) bring rapidly increasing design complexity [110]. This leads to non-trivial challenges for physical design, particularly in the routing stage. The number of remaining design rule check violations (DRC) after place-and-route (P&R) has become one of the most crucial metrics for an automatic IC layout solution, since back-end designers must manually stitch/modify all DRCs via implementation of engineering change orders (ECO) at the post-layout stage. In particular, achieving the pin accessibility needed to resolve DRCs is a critical, time-consuming engineering task just before tapeout, and is therefore a critical bottleneck in the advanced-node IC design process [56] [103].

To mitigate pin accessibility-induced DRCs, several approaches are proposed to improve pin accessibility through detailed placement (DP) optimization [35] [66] [72] [88] [108] [130] and standard-cell layout optimization [23] [126]. The authors of [88] perform detailed placement optimization using a global routing solution as guidance, with pin accessibility modeled only in the form of pin density. Dynamic programming and deep learning-based detailed placement optimizations considering each pin's access are developed in [35] [66] [72] [130]. In [108], the authors introduce a measurement of inaccessible pins in a cell to optimize detailed placement. However, these models have limited capability of comprehending design rules in detailed routing (DR). The works of [23] and [126] have proposed pin accessibility-driven cell layout optimization frameworks for improving routability at block-level. Recently, the authors of [76] perform replacement of inaccessible cells with diverse cell layouts in terms of pin locations and number of access points, in the ECO stage. The applicability of these works is intrinsically limited due to the lack of holistic consideration for the block-level design steps (e.g., detailed placement and detailed routing).

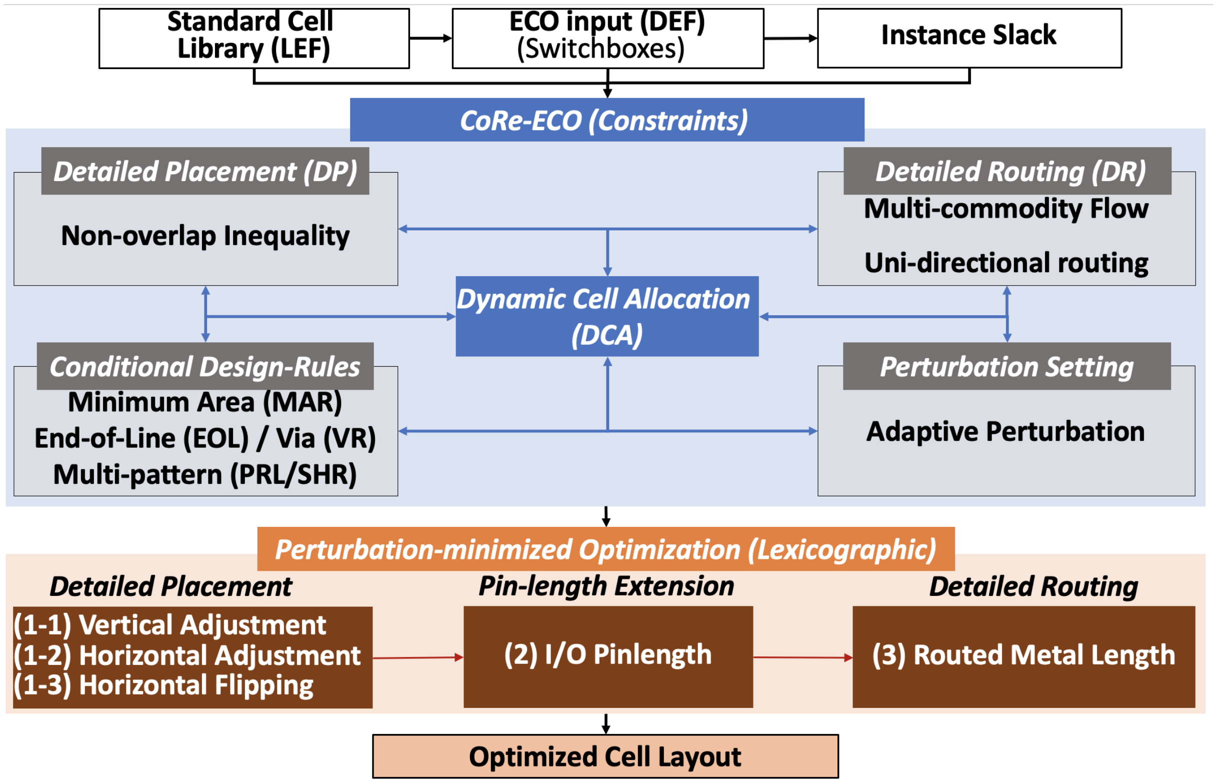


Figure 2.10. Framework overview.

In this work, we propose *CoRe-ECO*, a **Co**ncurrent **Re**finement framework which simultaneously performs incremental detailed placement and detailed routing, along with cell replacement, at the **ECO** stage. To our knowledge, this is the first work to present a concurrent co-optimization of detailed placement, detailed routing and cell replacement within block-level P&R. Our main contributions are summarized as follows.

- We propose a concurrent refinement framework which simultaneously performs detailed placement, detailed routing and cell replacement for each local window (i.e., switchbox) covering DRC locations, to determine ECOs at post-layout stage. We devise a novel dynamic cell allocation (DCA), merging the refinement steps into a single-step optimization.

- *CoRe-ECO* performs (i) placement adjustment such as horizontal/vertical shifting, horizontal flipping and cell swapping; (ii) pinlength extension with a recommendation of adopting alternative master cells while maintaining the same functionality; and (iii) routing optimization to seek the best-quality DRC-clean solution.

- *CoRe-ECO* minimizes perturbation of the given layout by utilizing a satisfiability modulo theories (SMT) solver, enabling multi-objective optimization.
- We validate the proposed *CoRe-ECO* framework with various testcases, demonstrating successful fixing of pin accessibility-induced DRCs through the proposed ECO flow along with total wirelength optimization.

The remaining sections are organized as follows. Section 2.2.1 describes the proposed *CoRe-ECO* framework. Section 2.2.3 discusses our experimental setup and results. Section 2.2.4 concludes the work.

2.2.1 CoRe-ECO Framework

This section introduces an overview of the proposed framework, grid-based P&R architecture, refinement operations, switchbox generation, perturbation-minimized optimization and SMT formulation.

Framework Overview

We formulate a conventional (sequential) layout refinement process as a constraint satisfaction problem (CSP) with variables and constraints to integrate placement adjustment and routing steps into a single multi-objective optimization problem. We adopt the SMT solver *Z3* [34] to solve the given optimization problem. Figure 2.10 illustrates an overview of our framework. Given standard-cell library, switchbox and instance slack information,[8] our framework simultaneously obtains the optimal solution that strictly satisfies the constraints integrated into our novel DCA scheme. Our notations are described in Table 2.4.

[8]We define *instance slack* as the worst slack among pins of a given instance.

Table 2.4. Notations for the proposed *CoRe-ECO* framework.

Notation	Meaning
T	Set of instances in a switchbox.
t	t^{th} instance.
l_t	0-1 indicator if instance t is flipped.
x_t	x-axis coordinate of lower-left corner of t.
y_t	y-axis coordinate of placement row of t.
y^t_{org}	Initial y-axis coordinate of placement row of t.
x^t_{org}	Initial x-axis coordinate of lower-left corner of t.
w_t	Width of instance t.
P^t	Set of internal pins of instance t.
p^t_i	i^{th} pin of instance t.
$a_{ext}(p^t_i)$	Set of extended vertices of pin p^t_i.
$V(V_i)$	Set of vertices in (i^{th} metal layer of) the routing graph G.
v	A vertex with the coordinate (x_v, y_v, z_v).
$a(v)$	Set of adjacent vertices of v.
$e_{v,u}$	An edge between v and u, $u \in a(v)$.
$w_{v,u}$	Weighted cost for metal segment on $e_{v,u}$.
N	Set of multi-pin nets in the given routing box.
n	n^{th} multi-pin net.
f^n_m	A two-pin subnet connecting a source and sink, i.e., a commodity.
$e^n_{v,u}$	0-1 indicator if $e_{v,u}$ is used for n.
$f^n_m(v,u)$	0-1 indicator if $e_{v,u}$ is used for commodity f^n_m.
$m_{v,u}$	0-1 indicator if there is a metal segment on $e_{v,u}$.
$C^n_m(v,u)$	Capacity variable for $e_{v,u}$ of commodity f^n_m.

Grid-based Place-and-Route Architecture

We define the grid-based placement and 3-D routing graph composed of four metal layers (i.e., M1-M4) as shown in Figure 2.11. Cell instances and I/O pins are aligned with M1 vertical tracks and gate poly of the standard cell. Inspired by [73], we adopt supernodes to cover the multiple candidates for each pin, either the I/O pin of a standard cell (i.e., P_{IN}) or the outer pin of a switchbox (i.e., P_{EX}). The location of P_{IN} is dynamically determined by the placement formulation and is associated with the flow formulation for routing through DCA. P_{EX} interconnects the internal pins inside the switchbox to the outer pins and is located along the

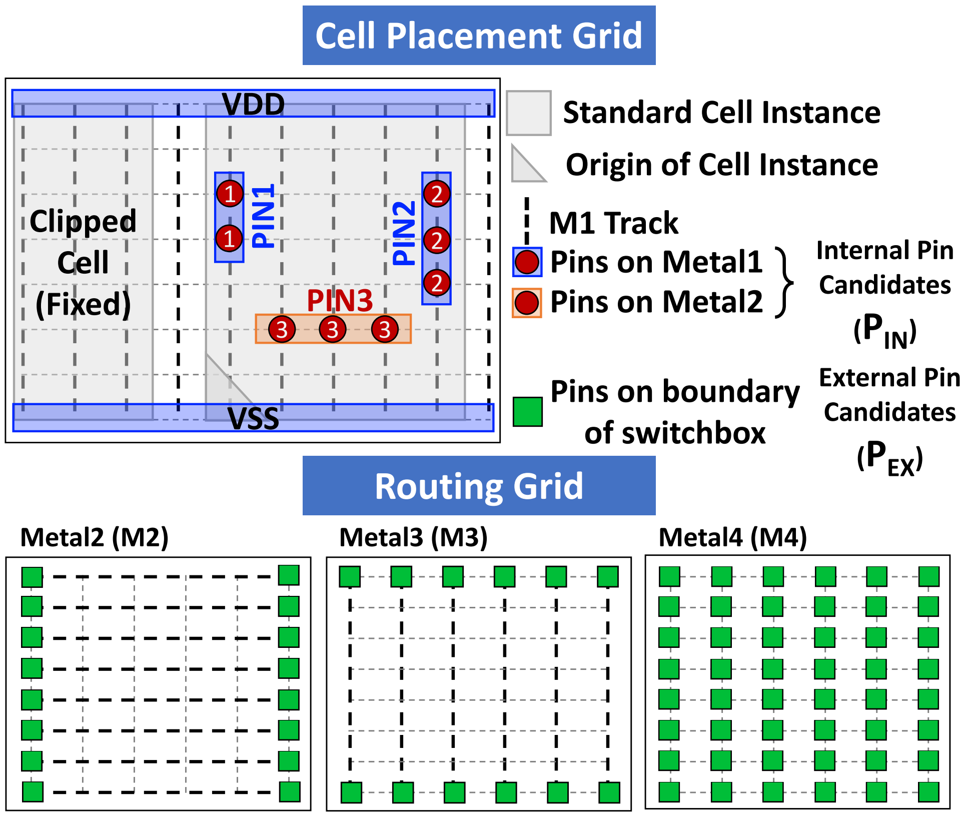


Figure 2.11. Grid-based place-and-route architecture.

boundary of the switchbox which corresponds to the pre-routed result.

The horizontal routing grid (i.e., M2 and M4) consists of eight tracks per placement row and the vertical routing grid (i.e., M3) is aligned with the cell placement grid.[9] Note that we focus on M2-M3 layers assuming that M2 and M3 have the same metal pitches, because our proposed framework targets the correction of the pin accessibility-induced DRCs. Thus, M4 layer only contains VIA34 elements as the external pins for connections to the upper layers.

Refinement Operation

CoRe-ECO uses placement adjustment and pinlength extension as refinement operations within its adaptive perturbation method.

[9]We assume an on-grid routing scheme for each routing layer, consistent with sub-7nm multi-patterning technologies and IC practitioners' restriction of preferred routing direction per each layer [142]. In this work, we do not support different pitches for M2, M3 and M4, which are out of scope.

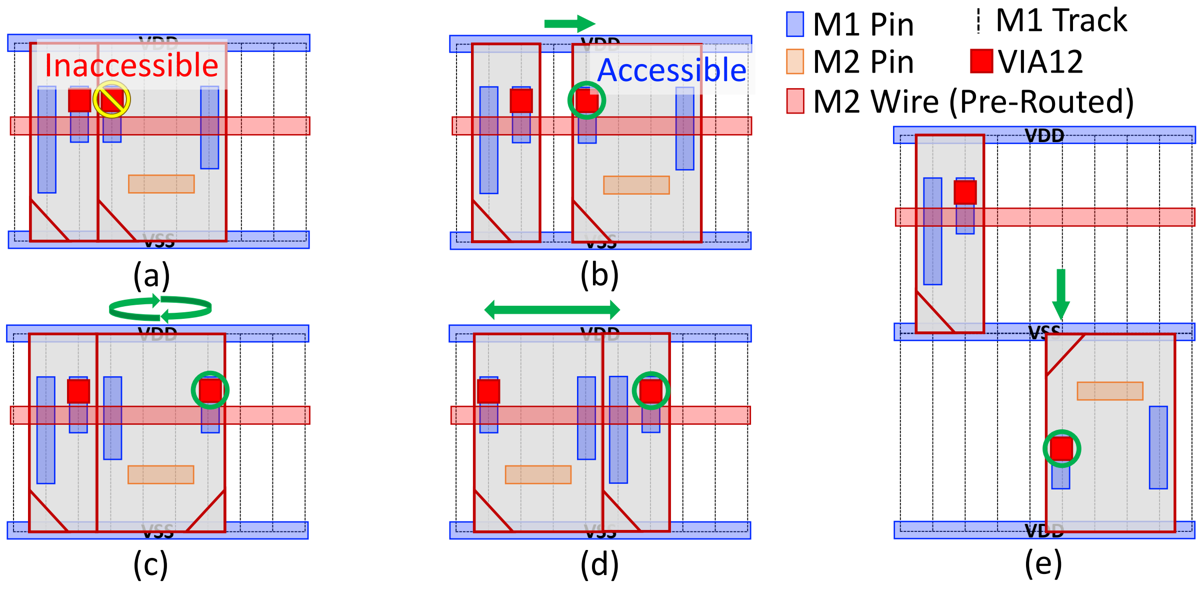


Figure 2.12. Placement adjustments. (a) Initial placement. (b) Horizontal shifting. (c) Horizontal flipping. (d) Cell swapping. (e) Vertical shifting.

Placement Adjustment. Figure 2.12 illustrates possible adjustments during detailed placement to solve the pin accessibility-induced DRC. When the given placement layout in a switchbox (Figure 2.12(a)) does not have feasible routing solutions due to an inaccessible M1 pin, *CoRe-ECO* adjusts the placement of instances in the switchbox by horizontal/vertical shifting, horizontal flipping and cell swapping as shown in Figures 2.12(b)-(e). Note that cell swapping is only performed between two adjacent instances.

Pinlength Extension. As a rule, master cells with minimum pinlength are preferred for use during initial P&R, e.g., to achieve better timing optimization. However, in the ECO stage, engineers should consider adopting alternative master cells for specific instances, so as to improve pin accessibility or to achieve the target design specification. While works of 20+ years ago [9] [11] [92] pursued *liquid library* approaches, today it is well-understood that a library cell must be qualified before it is used in a production chip. Thus, our framework suggests minimum-achievable pinlength extensions needed to fix DRCs: by only extending the metal segments of I/O pins, we enable engineers to adopt (i.e., swap in) alternative master cells that maintain the same functionality while minimizing the magnitude of undesired timing impact on

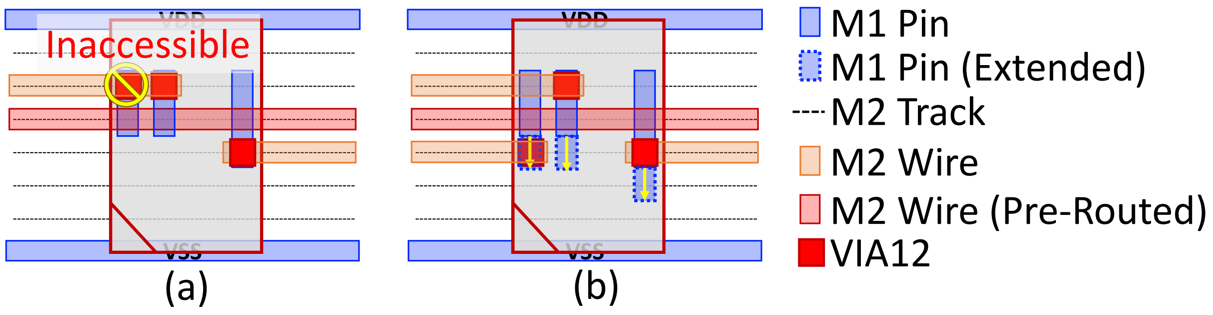


Figure 2.13. Pinlength extension. (a) Initial placement. (b) Pinlength extension.

each cell. Furthermore, our framework minimizes the effective number of alternative master cells by the perturbation-minimized optimization described in Section 2.2.2. Note that this simple pin-extension satisfies all the conditional design rules for generating standard-cell libraries that are described below in Section 2.2.3. Figure 2.13 illustrates pinlength extension that resolves a DRC caused by the inaccessible pin (Figure 2.13(a)) by extending the instance's I/O pins (Figure 2.13(b)) without detailed placement adjustments. We only apply the pin-extension for M1 pins because (i) we do not allow routing on M1 layer, and (ii) M2 pins are directly accessible from BEOL layers.

Adaptive Perturbation. We utilize instance timing slacks to set the perturbation range (i.e., the range of the vertical and horizontal adjustments) by considering timing margins of each instance. For the instances with the worst slacks, we fix the placement and the routed wires to prevent the deterioration of timing characteristics. For the rest of the instances, the applicable range of perturbation is set by the input parameter settings. Flipping of instances and extension of pinlengths are allowed for all instances except for the fixed instances.

2.2.2 Switchbox Generation

CoRe-ECO generates a switchbox by extracting the instance, pin, net and obstacle information from each local window covering DRCs. Figure 2.14(b) visualizes the generated switchbox representation from the local window depicted in Figure 2.14(a).

Instances. We separate the instances in the refinement region (i.e., blue dotted box) according

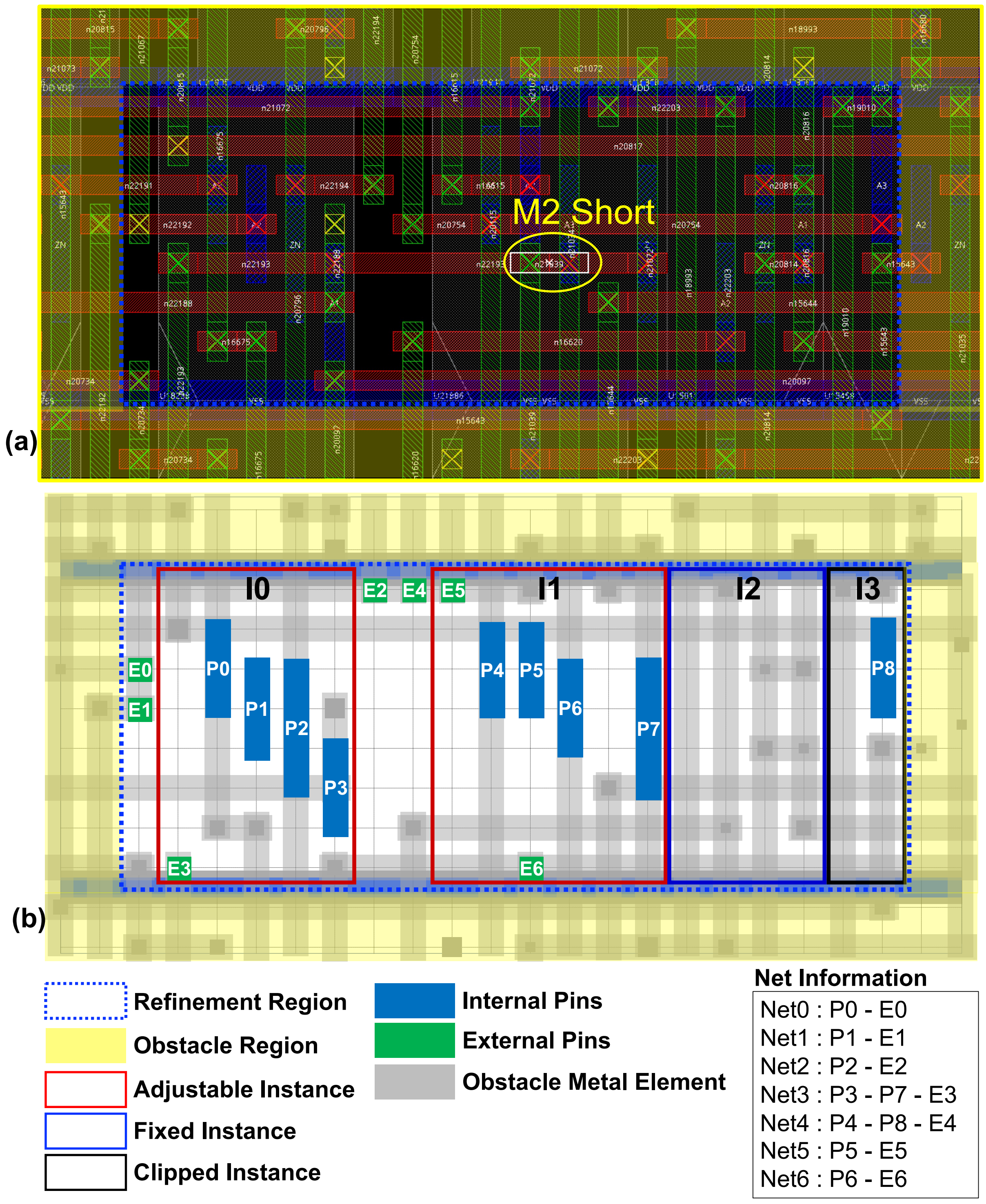


Figure 2.14. Switchbox generation. (a) Local window with one DRC, displayed by the commercial tool [144]. (b) Visualization of the generated switchbox.

to the given P&R results. We first fix the placement and the corresponding pre-routed nets of the instances (i.e., I2) whose timing slacks are less than a predefined worst-slack upper bound. Then, the instances inside the region are extracted as adjustable instances (i.e., I0, I1) that are allowed for the refinement operations. The clipped instances (i.e., I3) are partially included in the refinement region. Therefore, those instances are not adjustable, but their partial pins/nets are extracted for the routing optimization.

Pins/Nets. The I/O pins in the adjustable and clipped instances are extracted as internal pin candidates (i.e., P0 - P8). The external pins (i.e., E0 - E6) are extracted along the boundary of the refinement region if there is a connection from the internal pins to the outside of the refinement region. The net information defines new interconnections between internal pins and external pins.

Obstacle Elements. The switchbox has two types of obstacle elements (i.e., gray rectangles). First, we extract the routed metal elements in the obstacle region (i.e., yellow solid box) and set those elements as obstacles to check for DRCs on the boundary of the refinement region. Second, we consider the routed elements inside the refinement region as obstacles if those elements do not have any connections to the internal/external pins or they are connected to the fixed instances.

Perturbation-Minimized Optimization

The proposed *CoRe-ECO* has multiple objectives associated with the refinement operations and routing problems. To honor the given (initial) detailed placement and detailed routing solution, we minimize the perturbations made by refinement operations as well as the total metal length. The vertical adjustment (ΔV) and horizontal adjustment (ΔH) are respectively defined as the total amount of vertical and horizontal shifts of the adjustable instances as shown in (2.24) and (2.25). The horizontal flipping (ΔF) is defined as the total number of flipped instances as shown in (2.26). The pinlength extension (ΔP) is defined as the sum of extended pinlengths as shown in (2.27). The routing (ML) is the sum of routed VIA/Metal elements (i.e., VIA12, M2, VIA23 and M3). Each element has the same weight in the calculation of ML because we separate the objective functions for each type of element as shown in (2.28). *CoRe-ECO* simultaneously optimizes these multiple objectives in light of the “lexicographic” order described in (2.29). In other words, the objectives are optimized according to the priority order given by **LexMin**; for each given objective, this effectively induces a single-objective optimization problem under the constraining condition that optimizes the higher-priority objectives.

$$\textbf{Vertical Adjustment } (\Delta\mathbf{V}) \textbf{ :} \sum_{t \in T} (|y_t - y^t_{org}|) \tag{2.24}$$

$$\textbf{Horizontal Adjustment } (\Delta\mathbf{H}) \textbf{ :} \sum_{t \in T} (|x_t - x^t_{org}|) \tag{2.25}$$

$$\textbf{Horizontal Flipping } (\Delta\mathbf{F}) \textbf{ :} \sum_{t \in T} l_t \tag{2.26}$$

$$\textbf{Pinlength Extension } (\Delta\mathbf{P}) \textbf{ :} \sum_{e_{p,r} \in E} \left(w_{p,r} \times m_{p,r}\right), \tag{2.27}$$

$$\forall p \in P^t, \forall t \in T, \forall r \in a_{ext}(p)$$

$$\textbf{Routing (ML\{\#VIA12, \#M2, \#VIA23, \#M3\}) :} \sum_{e_{v,u} \in E} m_{v,u} \tag{2.28}$$

E=Sets of each VIA12, M2, VIA23, and M3 Element

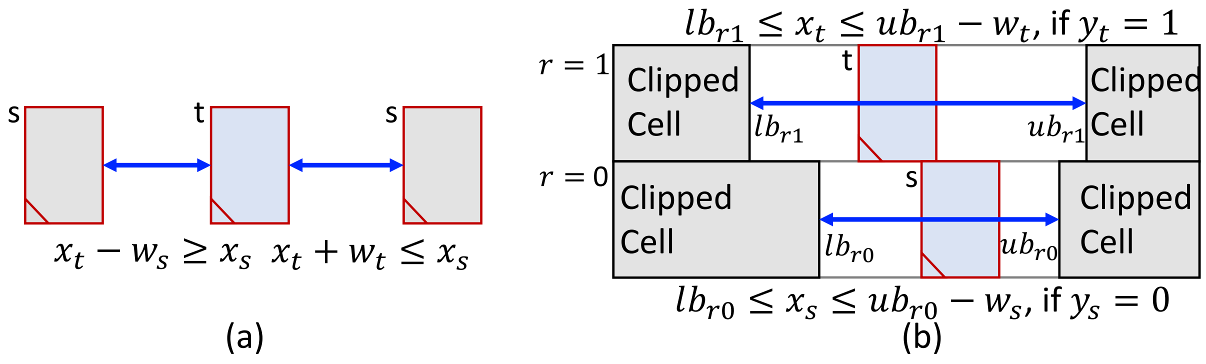


Figure 2.15. Placement constraints. (a) Relative positions between two instances in the same placement row. (b) Boundary condition of each placement row.

$$\textbf{LexMin: } \text{(a) } \Delta V, \text{ (b) } \Delta H, \text{ (c) } \Delta F, \text{ (d) } \Delta P,$$
$$\text{(e) ML } \{\text{(1) \#VIA12, (2) \#M2, (3) \#VIA23, (4) \#M3}\} \tag{2.29}$$

SMT Formulation

Placement Formulation. We utilize the conventional floorplanning approach (i.e., *Relative Positioning Constraint (RPC)*) for the placement problem [115]. All instance positions in a specific placement row can be represented by two RPCs as shown in Figure 2.15(a). At least one of the two inequalities holds for each pair $t \neq s$ through the SMT expression described in Algorithm 1. The maximum adjustable column boundary of each instance t is determined by the placement row y_t due to the different composition of clipped instances in each row as shown in Figure 2.15(b). These geometric conditions determine the position and the flip status of the instance.

Dynamic Cell Allocation (DCA). Every pin in each instance has its corresponding flow capacity variable $C_m^n(p,r)$ for certain net n and commodity m on the corresponding vertices of the placement grid, according to the shape and relative position of the pin in the instance as well as the possible adjustment range of each instance (see Figure 2.16(a)). When locations of instances are determined by the placement formulation, the flow capacity variables of each instance's

Algorithm 1. Set RPC constraint (instances t, s)

```
1: if $y_t = y_s$ then                              ▷ t and s are on the same placement row
2:     if $x_t \geq x_s + w_s$ then
3:         $x_t \geq x_s + w_s$;                     ▷ t is on the right side of s
4:     else if $x_t + w_t < x_s$ then
5:         $x_t + w_t \leq x_s$;                     ▷ t is on the left side of s
6:     else
7:         Unsatisfiable condition;
8:     end if
9: end if
```

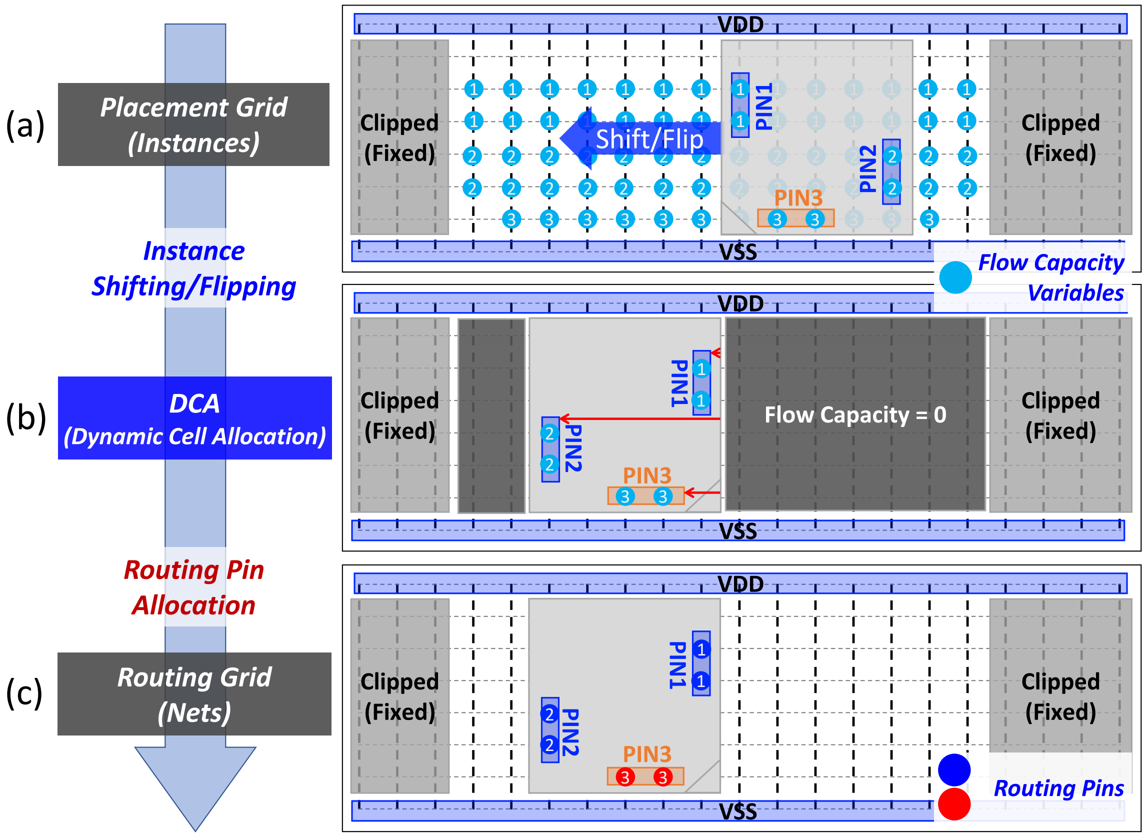


Figure 2.16. Dynamic cell allocation (DCA).

pins are conditionally assigned to the corresponding locations according to the placement status of each instance (i.e., shifted, flipped) as described in Algorithm 2. First, the coordinates of each pin are determined by the location of each instance and flip status (Lines 1-6). Then, all capacities $C_m^n(p,r)$ outside the range of each pin are assigned to zero (Lines 7-11) as depicted in Figure 2.16(b).

Algorithm 2. Set flow capacity control constraint $(C_m^n(p,r))$

```
/* x coordinate (resp. placement row) of a routing grid r: $x_r$ (resp. $y_r$) */
/* x coordinate (resp. placement row) of a pin p: $x_p$ (resp. $y_p$) */
/* p is either source or sink of a net n and commodity m */
/* origin's column, origin's row, flipping of a instance i: $i_x$,$i_y$,$i_f$ */
/* x offset from the instance origin of a pin p: $o_{x_p}$ (resp. $o_{y_p}$) */
1: if $i_f$ = false then
2:     $x_p = i_x + o_{x_p}$;
3: else
4:     $x_p = i_x - o_{x_p}$;
5: end if
6: $y_p = i_y + o_{y_p}$;
7: if $(x_r \neq x_p) \mid (y_r \neq y_p)$ then
8:     $C_m^n(p,r) = 0$;
9: else
10:    $C_m^n(p,r)$ is determined by routing formulation;
11: end if
```

$$f_m^n(v = p, u = r) \leq C_m^n(p,r), \quad \forall r \in a(p), \forall r \in V_0 \tag{2.30}$$

Equation (2.30) associates the flow variable $f_m^n(v,u)$ with the flow capacity variable $C_m^n(p,r)$. Each $f_m^n(v,u)$ is determined by the routing formulation when vertex v is the internal pin p, and the adjacent vertex u is the adjacent vertex r of p in M1 and M2 (i.e., V_1, V_2). This enables our routing formulation to recognize the feasible sets of r in V_1, V_2 layers as routing pins, as depicted in Figure 2.16(c).

Pinlength Extension. We generate flow variables for extendable pin candidates to enable pinlength extension when finding a routable solution, as illustrated in Figure 2.17. The extendable pins are generated in both up/down directions from the uppermost/lowermost vertices of each I/O pin. We set different weights for the extendable pin candidates, proportional to their distance from the nearest I/O pins. These weights are used as the priority in our objective function (Equation (2.27)) for minimizing the total length of the extended pins.

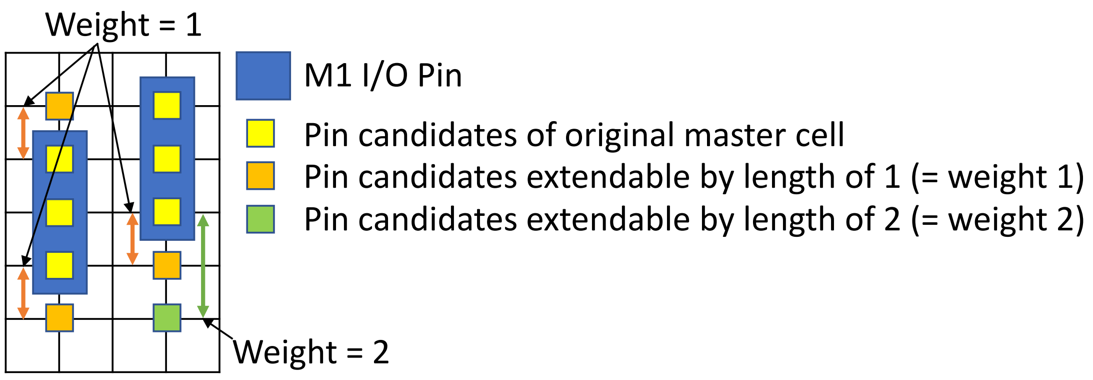

Weight = 1
M1 I/O Pin
Pin candidates of original master cell
Pin candidates extendable by length of 1 (= weight 1)
Pin candidates extendable by length of 2 (= weight 2)
Weight = 2

Figure 2.17. Weighted extendable pin candidates.

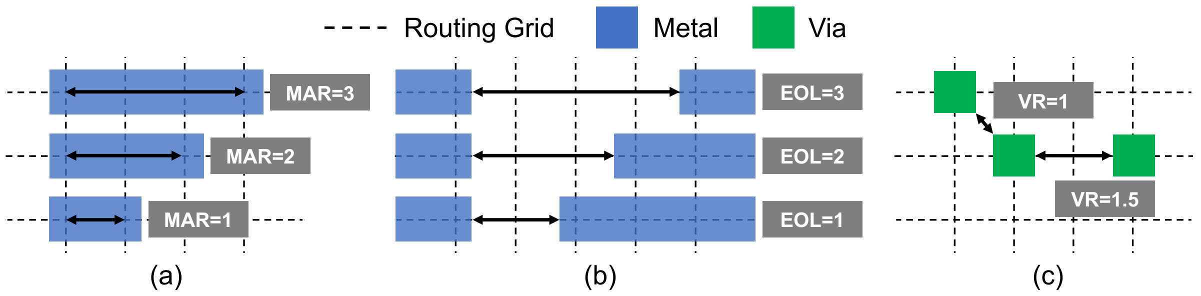


Figure 2.18. Grid-based conditional design rules: (a) MAR, (b) EOL and (c) VR.

Routing Formulation. We use multi-commodity network flow and conditional design rules to formulate the detailed routing problem, following the same principles as [100]. The flow formulation finds a routing path between the source and the sink for each commodity. Specifically, refined constraints for *commodity flow conservation* and *vertex exclusiveness* in uni-directional edges are implemented in our framework to reduce the search space of the routing formulation. The conditional design rules work as constraints to route using design-rule violation-free paths. *CoRe-ECO* implements three fundamental grid-based design rules,[10] namely, *Minimum Area (MAR)*, *End-of-Line Spacing (EOL)* and *Via Rule (VR)*, as illustrated in Figure 2.18. MAR (Figure 2.18(a)) defines the minimum number of grids that should be covered by the metal segments. EOL (Figure 2.18(b)) defines the minimum number of grids between two metal segments. VR (Figure 2.18(c)) defines the minimum distance (in L_2 norm) between vias.

[10]In this work, we assume sub-7nm technologies that are based on Extreme Ultraviolet (EUV) lithography as in the previous work [24]. However, our framework is applicable to additional multi-pattern-aware design rules, such as Parallel Run Length (PRL) and Step Height Rule (SHR).

Table 2.5. Standard-cell architectures.

Cell architecture			Design constraint	
#Fin	#Routing tracks	Cell height	Design rule	Pin accessibility
3	6	8T	EUV-Loose (EL)	MPO3
3	6	8T	EUV-Tight (ET)	MPO3
2	4	6T	EUV-Loose (EL)	MPO2
				MPO3
			EUV-Tight (ET)	MPO2
				MPO3

2.2.3 Experimental Setup and Results

We have implemented the proposed *CoRe-ECO* framework in *Perl/SMT-LIB* 2.0 and validated on a Linux workstation with Intel (R) Xeon E5-2560L 1.8GHz processor and 128GB memory. The SMT Solver *Z3* (version 4.8.5) [34] is used to produce the optimized solution.

Experimental Setup

Standard-Cell Library Preparation. Using an SMT-based cell layout automation [169], we prepare six types of standard-cell libraries with various cell architecture and design constraints presented in Table 2.5. We adopt two design rule sets that comprise combinations of specific design rule settings, as follows. EUV-Loose (EL) consists of MAR/EOL/VR = 1/1/1. EUV-Tight (ET) consists of MAR/EOL/VR = 1/2/1, inspired by [24]. We also generate two different types of cell libraries ensuring at least two and three I/O pin access points (i.e., MPO2 and MPO3). Then, for design enablement, we convert the primitive layout solutions of the SMT to LEF format. We assume contacted poly pitch (CPP), metal pitch (MP), and cell height of 40, 40 and 280nm, respectively.[11] We also generate three additional LEFs that have cells with pinlengths extended by 1, 2 and 3 grids, respectively, consistent with the cell height of the corresponding standard-cell library.

Place-and-Route (P&R). We validate our framework by using four open-source RTL designs AES, JPEG, LDPC [161] and IBEX [160]. We utilize M2-M7 layers as back-end-of-line (BEOL).

[11]Since the layouts are fully grid-based, we consider the CPP and MP as pitches of grids.

Table 2.6. Experimental statistics. RT = the number of routing tracks, DRSet = Design rule set, Impr./Incr. = Improvement/Increment ratio over ECO routing, #V/#H/#F/#P = the number of refined instances by vertical shifting/horizontal shifting/horizontal flipping/pinlength extension.

Design			Cell library			ECO routing		Proposed *CoRe-ECO* refinement								#ECO	Runtime
								#Remaining DRCs		WL		#Refinements					
Name	#Nets	#Cells	RT	DRSet	MPO	#DRCs	WL(μm)	#DRCs	Impr.	WL(μm)	Incr.	#V	#H	#F	#P	round	(h)
AES	13,958	13,694	6T	ET	3	40	46,272.16	2	95.0%	46,262.73	-0.020%	2	17	5	36	3	9.2
	13,912	13,648	4T	ET	2	74	45,968.76	13	82.4%	45,958.58	-0.022%	3	32	31	60	5	9.9
	13,938	13,674	4T	EL	3	41	45,797.96	0	100.0%	45,781.98	-0.035%	2	19	12	28	4	4.3
	13,802	13,538	4T	EL	2	146	45,035.54	56	61.6%	45,025.73	-0.022%	8	51	23	86	5	16
	13,867	**13,603**	**4T**	**ET**	**3**	**153**	**44,548.12**	**44**	**71.2%**	**44,528.05**	**-0.045%**	**25**	**104**	**45**	**79**	**4**	**18.2**
JPEG	70,543	70,518	6T	ET	3	79	144,905.98	30	62.0%	144,892.66	-0.009%	0	12	15	51	3	13.2
	71,491	**71,466**	**4T**	**EL**	**2**	**155**	**134,901.91**	**73**	**52.9%**	**134,893.09**	**-0.007%**	**3**	**30**	**14**	**73**	**4**	**25.3**
	71,177	71,152	4T	EL	3	37	133,558.22	18	51.4%	133,555.19	-0.002%	2	14	8	18	3	2.8
	70,932	70,907	4T	ET	2	198	132,739.50	132	33.3%	132,738.24	-0.001%	4	22	12	49	3	41.4
	70,008	69,983	4T	ET	3	68	136,852.58	42	38.2%	136,849.64	-0.002%	1	13	5	12	2	8.7
LDPC	57,133	55,081	6T	EL	3	21	732,917.12	8	61.9%	732,908.83	-0.001%	2	5	3	7	2	1.3
	57,138	55,086	4T	ET	2	12	737,957.04	7	41.7%	737,955.47	0.000%	0	3	1	1	1	0.24
	57,106	**55,054**	**4T**	**EL**	**3**	**90**	**751,812.13**	**53**	**41.1%**	**751,797.69**	**-0.002%**	**7**	**19**	**7**	**9**	**2**	**5.3**
IBEX	15,540	12,225	4T	EL	2	166	48,734.35	56	66.3%	48,730.00	-0.009%	12	40	16	80	5	15.8
	15,432	**12,117**	**4T**	**EL**	**3**	**62**	**48,426.72**	**6**	**90.3%**	**48,425.94**	**-0.002%**	**3**	**11**	**9**	**8**	**3**	**4.8**
	15,502	12,187	4T	ET	2	141	47,042.10	80	43.3%	47,045.31	0.007%	3	18	10	48	4	13.9
	15,179	11,864	4T	ET	3	93	49,346.74	32	65.6%	49,367.76	0.043%	0	9	14	13	3	5.1
Average						92.7	195695.11	38.4	**58.6%**	195689.23	**-0.003%**	**4.5**	**24.6**	**13.5**	**38.7**	**3.3**	**11.5**

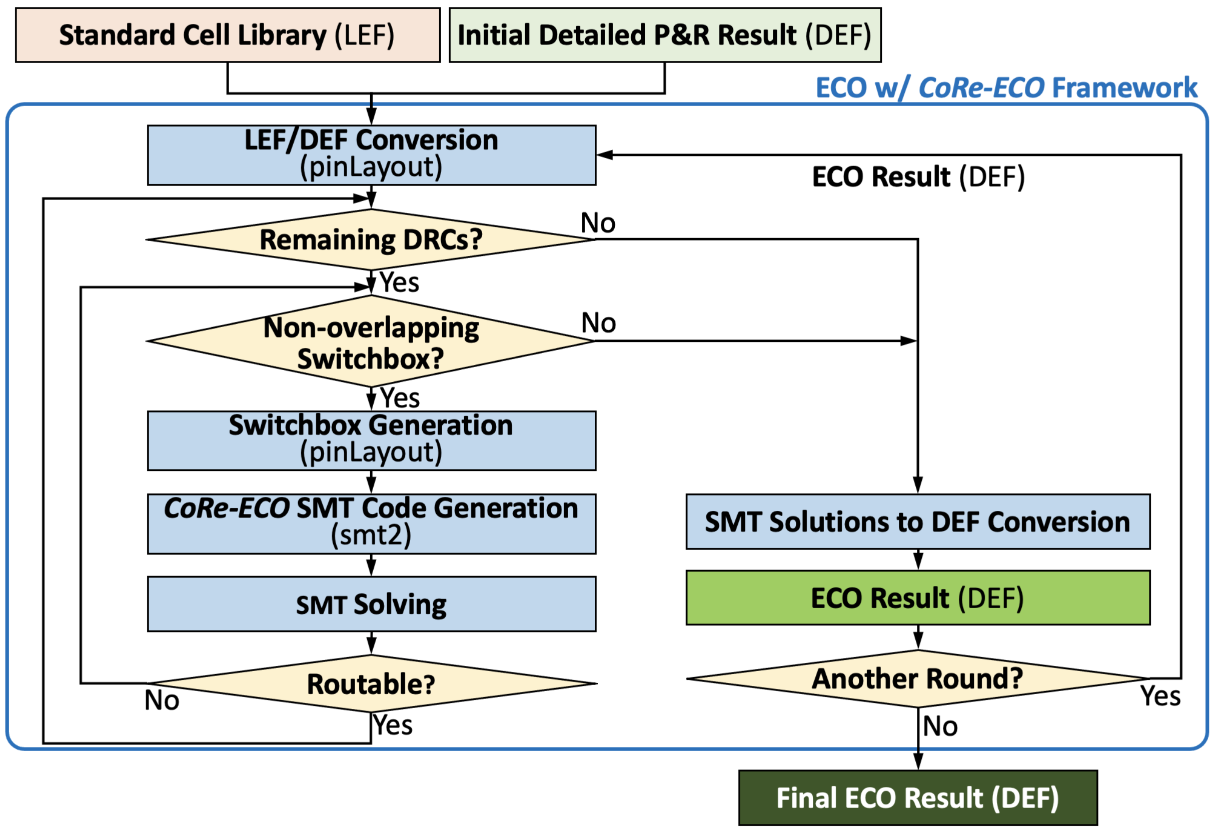

Standard Cell Library (LEF)
Initial Detailed P&R Result (DEF)
ECO w/ CoRe-ECO Framework
LEF/DEF Conversion (pinLayout)
ECO Result (DEF)
Remaining DRCs?
No
Yes
Non-overlapping Switchbox?
No
Yes
Switchbox Generation (pinLayout)
CoRe-ECO SMT Code Generation (smt2)
SMT Solutions to DEF Conversion
SMT Solving
ECO Result (DEF)
Routable?
No
Yes
Another Round?
Yes
No
Final ECO Result (DEF)

Figure 2.19. Overall ECO flow using *CoRe-ECO* framework.

We assume power/ground pins on the M1 layer for the initial detailed routing. However, since our proposed framework targets grid-based architecture and correction of the pin accessibility-induced DRCs, we focus on M2-M3 layers assuming that M2 and M3 layers have the same metal pitches. Two commercial tools [144] [172] are used to generate the initial P&R layouts and to execute the following ECO routing. In the commercial tool, we perform 20 iterations of ECO routing until the commercial tool is unable to further reduce the number of DRCs (i.e., #DRCs) for most of the benchmark cases. The blue lines in Figure 2.20 show the example trends of #DRCs through the ECO routing iterations for four representative cases from Table 2.6 (i.e., cases in bold). Note that we compare our work with the results of ECO routing because we are not able to fairly compare our work with the previous works, [35] [66] [72] [76] [88] [108] [130], due to (i) the different target design stage (i.e., detailed placement optimization vs. ECO) and (ii) the lack of exact experimental settings.

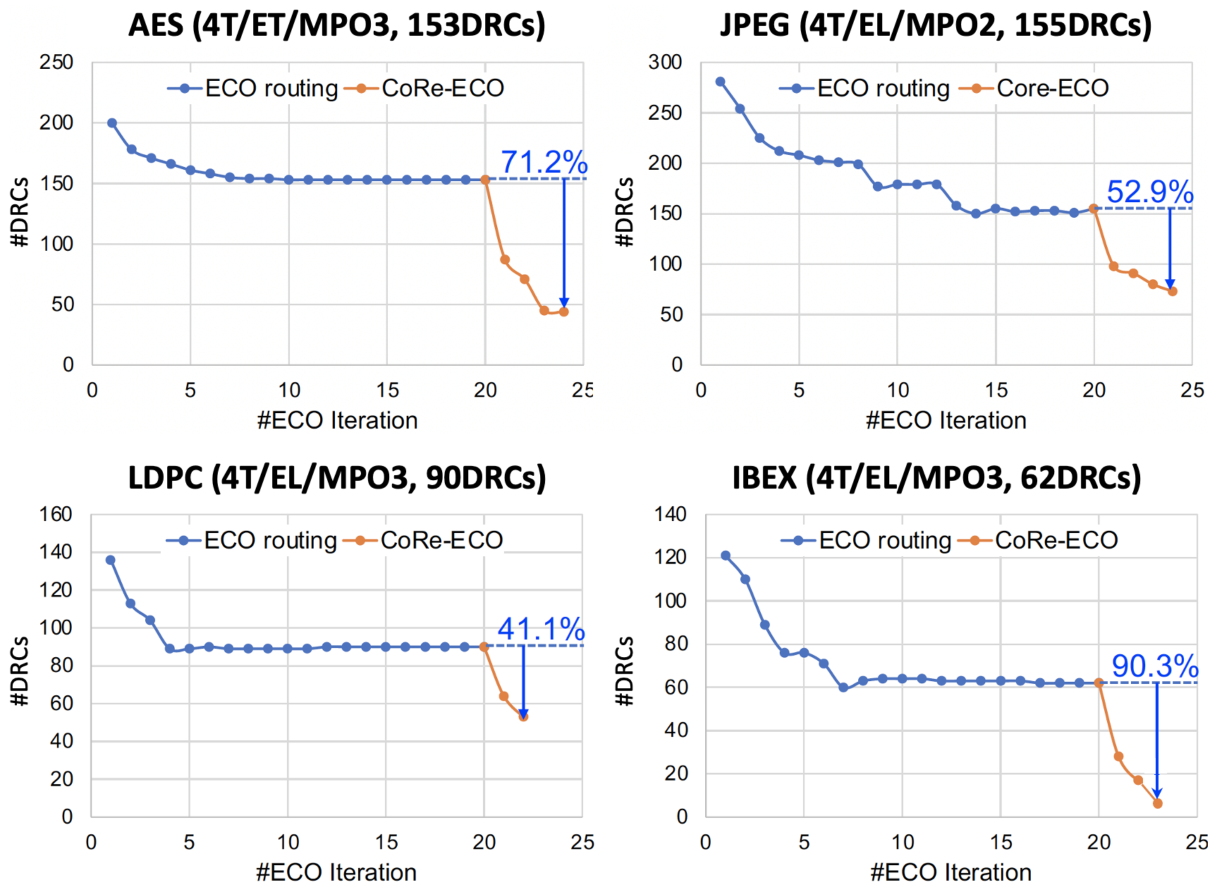


Figure 2.20. Example trends of the number of DRCs by the ECO routing and the proposed *CoRe-ECO* iterations.

Setting up the Perturbation Range. We set the 1% of worst slack cells and the routed nets connected to those cells as fixed instances and obstacle elements, respectively. For the rest of the instances, we set the perturbation range of the vertical and horizontal adjustments to two placement rows and eight poly pitches, respectively.

Design of Experiments

Figure 2.19 illustrates an overview of the ECO flow utilizing our *CoRe-ECO* framework. Given a cell library and initial detailed P&R result, the new ECO round starts with converting this layout information to a *pinLayout* format for the proposed framework in the LEF/DEF Conversion step. Then, if there exist any remaining DRCs in the non-overlapping regions, we rip up the region and generate a switchbox representation in the Switchbox Generation step. Note that the switchboxes in the same ECO round cannot overlap because our framework could change the P&R in each switchbox; the framework also refers to the horizontal/vertical obstacle regions for checking of design rules on the boundary of each switchbox. Given the switchbox representation, we generate an SMT code and solve the problem through the *CoRe-ECO* SMT Code Generation and SMT Solving steps. We iterate these refinement steps until we find a routable solution or there are no remaining DRCs or feasible switchboxes in non-overlapping regions. After the iterations, in the SMT Solutions to DEF Conversion step, we apply the DRC-clean solutions to the original DEF and generate a revised DEF for the next ECO round or publish as the final ECO result.

The ECO flow described above is fully automated, and each sequence (i.e., switchbox generation to SMT solving) can be executed in parallel through the multi-threaded operation. In this work, up to 24 threads are used for all testcases. For each DRC, our framework examines multiple switchboxes of various sizes (i.e., 10 - 25 vertical tracks and 1 - 5 placement rows) for several relative locations to the target DRC. The size of a routable switchbox for each DRC varies according to the existing P&R results, routing congestion, and locations of neighboring DRCs. To minimize the perturbation of the placement as well as the runtime of the SMT Solving step, our framework increases the size of the switchbox from the minimum (i.e., 10 vertical tracks $\times$ 1 row) to the maximum (i.e., 25 vertical tracks $\times$ 5 rows) in the Switchbox Generation step until it finds a routable solution or fails.

Experimental Results

Statistics on the Proposed ECO Flow. Table 2.6 summarizes the experimental statistics of the proposed ECO flow for benchmark cases which consist of four base design circuits synthesized with various cell libraries described in Table 2.5. In Table 2.5, #VTrack, #Row, #DRC, #Net and #Pin denote the number of vertical tracks, placement rows, DRCs, Nets and Pins in Switchbox, respectively. #TotalInst., #Adj.Inst. and #FixedInst. denote the number of total, adjustable and

Table 2.7. Detailed experimental results of the proposed ECO flow (AES, 4T, ET, MPO2, 74 DRCs).

Index	ECO round	Switchbox								#Cell refinement				SMT runtime(s)
		#Vtrack	#Row	#DRC	#Total Inst.	#Adj. Inst.	#Fixed Inst.	#Net	#Pin	#V	#H	#F	#P	
0	1st	18	2	1	5	3	2	15	33	0	2	0	1	11.0
1		19	1	2	4	3	1	9	22	0	0	0	2	1.3
2		18	2	1	4	3	1	13	29	0	1	1	1	8.5
3		14	1	1	3	2	1	6	17	0	0	0	1	0.6
4		14	1	2	3	2	1	6	15	0	0	0	0	0.6
5		21	2	1	7	3	4	15	43	0	0	0	3	25.4
6		17	1	1	3	3	0	7	17	0	2	0	1	3.4
7		15	2	1	4	2	2	9	21	0	1	0	0	28.8
8		14	1	1	3	2	1	8	17	0	0	0	1	0.7
9		14	1	1	2	0	2	6	14	0	0	0	0	0.4
10		18	2	1	8	5	3	16	41	0	0	3	3	11.8
11		16	1	1	2	1	1	6	13	0	0	0	1	0.6
12		15	1	2	4	2	2	6	16	0	0	1	2	0.7
13		22	2	3	9	6	3	19	42	0	3	2	5	20.2
14		16	1	1	3	3	0	8	18	0	2	1	2	1.4
15		14	1	1	2	2	0	5	11	0	1	0	0	0.4
16		14	1	1	2	1	1	4	12	0	0	0	0	0.5
17		14	1	1	2	2	0	4	10	0	0	1	0	0.4
18		14	1	1	2	1	1	4	9	0	1	0	0	0.4
19		14	1	1	2	2	0	6	14	0	0	0	0	1.0
20		18	2	1	6	3	3	16	38	0	0	1	2	5.7
21		14	1	1	3	2	1	6	14	0	2	1	0	0.5
22		16	1	1	3	2	1	7	16	0	1	2	2	0.8
23	2nd	18	1	1	4	3	1	10	25	0	0	1	1	2.2
24		14	1	1	3	2	1	7	15	0	0	0	1	0.7
25		14	1	1	2	1	1	3	7	0	0	0	0	0.3
26		15	2	1	5	2	3	12	29	0	0	0	0	13.7
27		14	1	1	2	1	1	7	18	0	0	1	1	0.8
28		14	1	1	2	1	1	5	11	0	1	0	0	0.4
29		19	2	1	7	6	1	15	42	0	1	1	3	54.6
30		16	2	1	5	4	1	16	34	0	1	0	0	5.6
31	3rd	29	2	1	10	8	2	27	63	0	1	1	5	588.9
32		18	5	2	13	11	2	34	77	0	0	3	2	859.9
33		17	2	7	7	5	2	14	38	2	3	2	2	6.3
34		19	3	3	10	7	3	24	54	0	1	1	4	109.1
35		20	3	1	9	8	1	25	67	0	1	4	4	1465.8
36		14	3	3	7	5	2	16	41	0	0	0	1	10.8
37		17	2	1	5	2	3	18	39	0	1	0	2	8.3
38		18	5	3	13	7	6	33	71	1	2	2	3	1841.8
39	4th	14	1	2	2	2	0	5	12	0	2	1	1	0.6
40		21	2	1	9	6	3	16	43	0	0	1	2	30.7
41	5th	16	2	1	7	3	4	12	28	0	2	0	1	6.5
Average		**16.6**	**1.7**	**1.5**	**5.0**	**3.3**	**1.6**	**11.9**	**28.5**	**0.1**	**0.8**	**0.7**	**1.4**	**122.2**
Total				61	208	139	69	500	1196	3	32	31	60	5131.84

fixed or clipped instances in Switchbox, respectively. #V, #H, #F and #P denote the number of refined instances by Vertical Shifting, Horizontal Shifting, Horizontal Flipping and PinLength

Extension, respectively. Column "ECO routing" represents the total number of DRCs (i.e., #DRCs) in target layers (i.e., M2-M3) and the total wirelength (i.e., WL) after the 20 iterations of ECO routing. The target DRCs mainly include "Cut Spacing" on M1-M2 layers and "Metal End-of-Line Spacing", and "Metal Short" on M2-M3 layers. Our *CoRe-ECO* framework reduces the remaining DRCs after ECO routing by 58.6% on average, with reductions ranging from 33.3% to 100.0%. Figure 2.20 shows the trend of #DRCs versus iterations of ECO routing (i.e., the blue line), along with the following *CoRe-ECO* flow (i.e., the orange line) for four representative cases of each base benchmark circuit from Table 2.6 (i.e., cases in bold). The figure demonstrates that our framework can further improve the routability with the concurrent cell refinements and the routing optimization. The reduction of the average total wirelength by 0.003% shows that our framework has successfully minimized the wirelength despite the refinement of cell placement and pinlength extension. We observe that the number of ECO rounds and the total runtime depend on #DRCs and the benchmark circuit configurations. For all benchmark cases, *CoRE-ECO* performs 3.3 ECO rounds on average with an average runtime of 11.5 hours.

Table 2.7 presents the detailed refinement results of the AES benchmark circuit with 4T/ET/MPO2 cell library and 74 DRCs. A total of five ECO rounds with 42 switchboxes have been performed to resolve 61 out of all 74 DRCs. The average number of vertical tracks, placement rows, and DRCs in the switchboxes are 16.6, 1.1, and 1.5, respectively. Each switchbox includes 5.0 total/3.3 adjustable/1.6 fixed or clipped instances and 11.9 nets/28.5 pins on average. Through the five rounds of ECO flow, 126 out of 139 adjustable instances have been perturbed in the placement or the length of pins. 6 out of 42 ECO cases have been fixed without any perturbation. And 7 and 7 cases require the extension of pinlengths or the re-placement of instances, respectively. The remaining 22 cases are routable by only changing both instance placement and pinlengths. The average runtime per switchbox is less than 3 minutes and switchboxes up to 18$\times$34 vertical/horizontal tracks, 7 adjustable instances, 33 nets and 71 pins (i.e., Index 38) have been solved within 31 minutes. Figure 2.21 shows the reduction

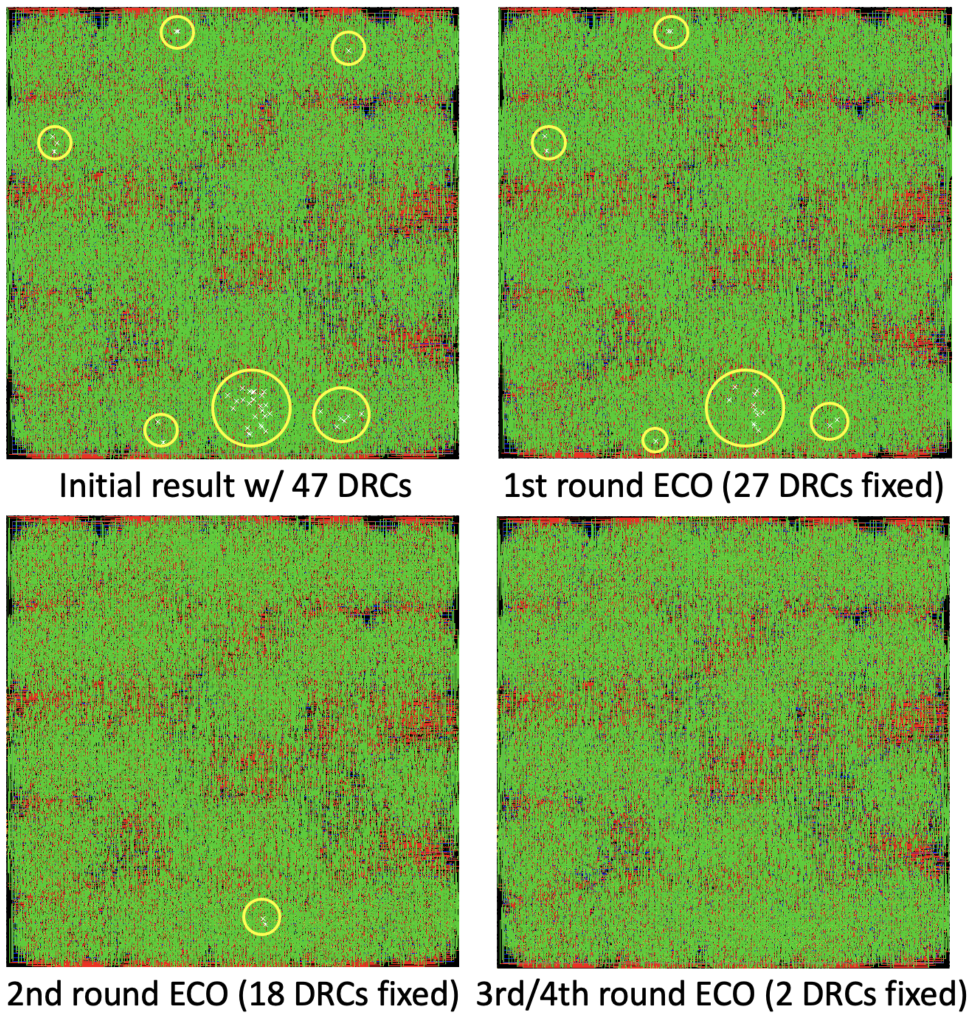

Initial result w/ 47 DRCs
1st round ECO (27 DRCs fixed)
2nd round ECO (18 DRCs fixed)
3rd/4th round ECO (2 DRCs fixed)

Figure 2.21. DRC reductions by *CoRe-ECO* rounds for AES (4T/ET/MPO2).

of DRCs in full-chip layouts by multiple ECO rounds utilizing *CoRe-ECO* framework, displayed by a commercial tool [144]. The yellow circles indicate regions with DRCs.

Example Refinement Operations. Figure 2.22 shows an example of refinement operations in our proposed ECO framework. Figure 2.22(a) depicts a switchbox of index 38 case in Table 2.7. The switchbox consists of 7 adjustable / 6 clipped (i.e. fixed) cell instances with 3 'M3 Short' DRCs in 18×34 vertical/horizontal tracks. Figure 2.22(b) illustrates the DRC-clean solution with the refinement operations (i.e., placement adjustment and pinlength extension) and the optimized routing in terms of the metal length. Note that the elements in gray color represent the obstacles inside the switchbox and that M1 I/O pins are not displayed in Figure 2.22(b). The pre-routed

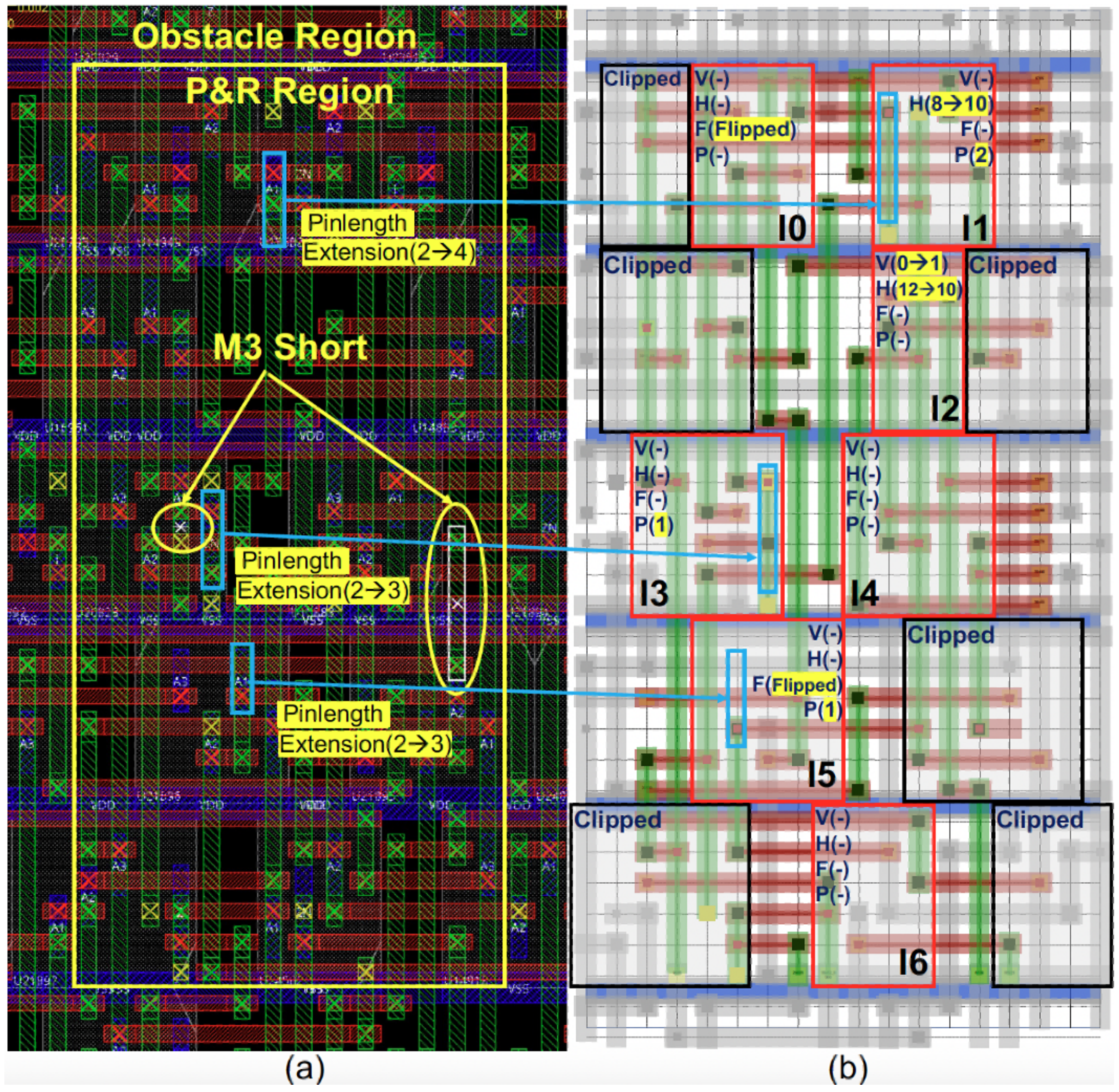


Figure 2.22. Example of refinement operations in the proposed ECO flow (Index 38 case in Table 2.7). (a) Switchbox with three DRCs. (b) Routable solution with placement adjustments and pinlength extensions.

wires, that (i) are connected to the fixed instances or (ii) have no internal connection inside the switchbox or (iii) exist outside the switchbox, are regarded as obstacles.

- **Placement Adjustment.** The placement of the instance I2 in Figure 2.22(b) has been adjusted from the placement row 0 to row 1 and horizontally shifted from the vertical track 12 to track 10. The instance I0 has shifted in the same placement row from the vertical track 8 to track 10. Instances I0 and I5 have been flipped on the same placement locations.

- **Pinlength Extension.** The I/O pins of the instance I1, I3 and I5 in Figure 2.22(b) have been extended by 1-2 grids to maximize the pin accessibility. In the proposed ECO flow, the respective master cell of each of these instances is replaced with the additional master cell with extended pinlengths, in the SMT Solutions to DEF Conversion stage.

2.2.4 Conclusion

We have described a novel concurrent refinement framework for the automated ECO flow. Our framework provides simultaneous and perturbation-minimized refinements of detailed placement-, detailed routing- and cell-optimized layout solutions to address the DRCs during the ECO stage. By ripping up and refining a local window of the whole layout design, *CoRe-ECO* is capable of achieving a DRC-clean layout solution. We have demonstrated that our framework successfully resolves an average of 58.6% (range: 33.3% to 100.0%) of remaining post-ECO route DRCs on M1-M3 layers, across a range of benchmark circuits with various cell architectures, with no adverse effect on total routed wirelength (average of 0.003% reduction).

2.3 Acknowledgments

Chapter 2 contains a reprint of Hamed Fatemi, Andrew B. Kahng, Minsoo Kim and Jose Pineda de Gyvez, “Optimal Bounded-Skew Steiner Trees to Minimize Maximum k-Active Dynamic Power”, *Proc. International Workshop on System-Level Interconnect Problems and Pathfinding*, 2020; and Chung-Kuan Cheng, Andrew B. Kahng, Ilgweon Kang, Minsoo Kim, Daeyeal Lee, Bill Lin, Dongwon Park and Mingyu Woo, “CoRe-ECO: Concurrent Refinement of Detailed Place-and-Route for an Efficient ECO Automation”, *Proc. IEEE International Conference on Computer Design*, 2021. The dissertation author is a main contributor to, and a primary author of, each of these papers.

I would like to thank my coauthors, Professor Chung-Kuan Cheng, Dr. Hamed Fatemi, Professor Andrew B. Kahng, Dr. Ilgweon Kang, Dr. Daeyeal Lee, Professor Bill Lin, Dr. Dongwon Park, Dr. Jose Pineda de Gyvez and Mingyu Woo.

Chapter 3

Technology-Aware Physical Design Methodology

This chapter presents two technology-aware physical design methodologies. The first work addresses local layout effects (LLE) in physical design. In a standard cell-based design, a diffusion break (DB) isolates neighboring devices, and the type of DB (single or double diffusion break) used in the standard cells can significantly impact device performance and leakage. The 2^{nd} DB effect is a type of LLE in which the distance to the second-closest DB affects device performance. We develop 2^{nd} DB-aware leakage optimization and detailed placement heuristics. The second work addresses the high resistance of interconnects which causes severe supply voltage (IR) drop problems at advanced nodes. To solve this problem, pre- or post-placed power staples are inserted in pin-access layers to connect adjacent power rails and reduce power delivery network (PDN) resistance, at the cost of reduced routing flexibility, or reduced power staple insertion opportunity. We propose dynamic programming-based single- and double-row detailed placement optimizations to maximize the power staple insertion in a post-placement flow. We further propose metaheuristics to improve the quality of results.

3.1 Diffusion Break-Aware Leakage Power Optimization and Detailed Placement in Sub-10nm VLSI

With aggressive lateral scaling of advanced CMOS technology, many challenges of standard-cell architectures, floorplan, placement, routing and timing signoff have been introduced into leading-edge IC design. Design today is challenged by numerous complex front-end-of-line (FEOL) and back-end-of-line (BEOL) design rules and layout restrictions. Moreover, aggressive area reduction via the standard-cell architecture leads to a breakdown of the traditional assumptions of composability and independence of cell-based layout. Notably, in sub-10nm VLSI, a given cell instance's timing and power can be strongly affected by the cell's neighbors in the placement; this is commonly referred to as *local layout effect* (LLE). Since standard-cell library models are provided for the cell itself, without regard to its neighbors in the placement, LLE induces performance modeling errors and/or added margins in signoff. Today's advanced-node design and signoff technologies must now carefully take LLEs into account.

New Standard-Cell Architecture with Diffusion Breaks. A *diffusion break* (DB) isolates two adjacent devices and is one of several critical aspects related to LLEs. For example, Slide 44 of the DAC-2018 tutorial [61], referring to such sources as [4], notes how continuous diffusion and diffusion breaks are an important lever for shrinking standard-cell area – at the cost of more complex design rules and LLEs. In modern sub-10nm enablements, there are two types of diffusion breaks: *single diffusion break* (SDB) and *double diffusion break* (DDB). SDBs occupy a smaller space than DDBs do, but incur greater LLE. On the other hand, while DDBs require more area and incur more leakage than SDBs, they make devices faster and are more immune to layout context. In this work, we ignore the 2^{nd} DB effect for DDB cells due to its negligible impact on leakage and timing. We study two types of standard cells, i.e., *SDB cells* and *DDB cells*. An SDB (resp. DDB) cell has an SDB (resp. a DDB) at both its left and right edges. In the following, we call a design with only SDB cells as *Type-I*, and a design with both SDB and DDB cells as *Type-II*.

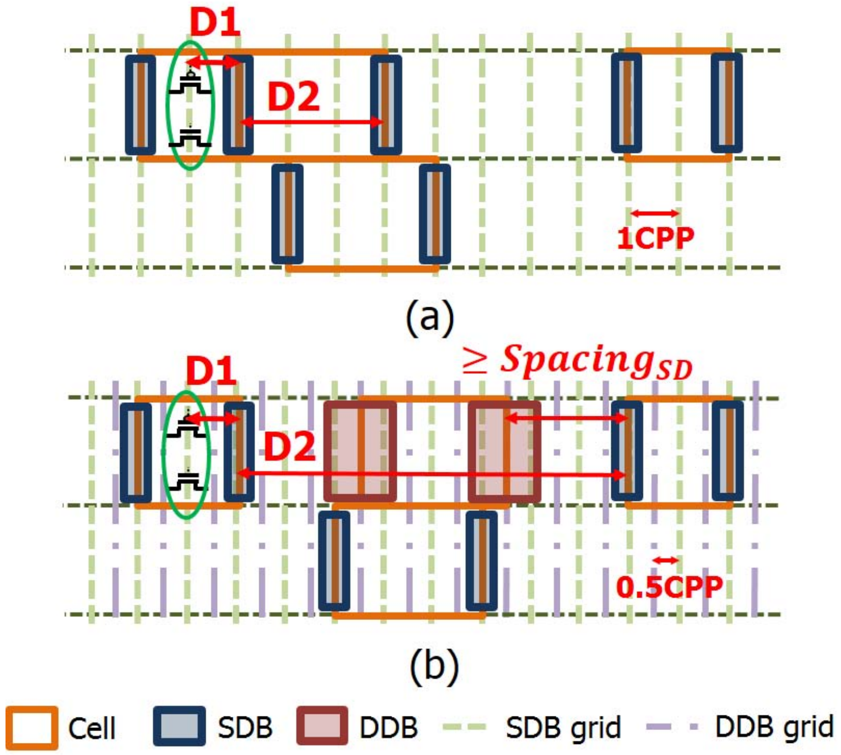


Figure 3.1. Examples illustrating DB types and placement constraints. (a) Layout with four SDB cells *(Type-I)*. (b) *Mixed-DB* layout with three SDB cells plus one DDB cell *(Type-II)*.

Figure 3.1 shows two legal placements with SDB and DDB cells. SDB and DDB cells can be placed within a single block as long as: (i) SDB (resp. DDB) cells all align to SDB (resp. DDB) grids, where the SDB grid has a half contacted poly pitch (CPP) offset from the DDB grid; and (ii) there is a process-specific minimum spacing requirement ($Spacing_{SD}$ in Figure 3.1) between neighboring SDB and DDB cells. In Figure 3.1, for a device in a green oval, D1 is the distance to the first placed SDB, and D2 is the distance between the first placed SDB and the second placed SDB. That is, D2 refers to the “2^{nd} *DB distance* of an SDB cell”, and we use the term “2^{nd} DB effect” to indicate the threshold voltage (Vt) shift due to D2 for an SDB cell. Note that the Vt shift due to the 2^{nd} DB effect causes a timing path with fixed cell instances and routing to be impacted by the *neighbors* of its cell instances. Since an SDB cell can have a second placed SDB on each of its left and right sides, in this work, we use the distance to the farther of the two placed 2^{nd} DBs as the D2 of a cell.

Design Impact from the 2^{nd} DB Effect. Table 3.1 shows results of a motivating analysis that demonstrates how much the 2^{nd} DB effect changes timing results and leakage power of designs.[12] We use two designs, AES and VGA, from OpenCores [161] and a reference flow of *Cadence Innovus Implementation System v17.1* [144] to generate post-P&R designs. We use 80% initial utilization and 0.35ns target clock period. We note that filler cell insertion is a critical step that determines 2^{nd} DB distances of cells and the impact of the 2^{nd} DB effect. Filler cells are inserted into empty spaces after place-and-route (P&R), typically starting with larger-width filler cells and finishing with smaller-width filler cells so as to greedily minimize instance count. However, filler cells with large width can create large 2^{nd} DB distances for any neighboring non-filler cells. So, in order to see the design impact of the 2^{nd} DB effect, we compare two filler cell insertion strategies: largest to smallest filler *(Standard)* and small filler only *(Small)*. As seen in Table 3.1, when we perform 2^{nd} DB-aware analysis for designs which come from a commercial P&R tool, 2^{nd} DB-aware leakage is larger than 2^{nd} DB-unaware leakage by 8.8% to 144.4%. In other words, actual leakage can be much larger than the leakage reported by signoff analysis that does not take into account the 2^{nd} DB effect. Further, after 2^{nd} DB-aware timing analysis, setup worst timing slacks are improved or similar, and hold worst timing slacks are not changed, due to small 2^{nd} DB impacts for hold corner. Overall, this preliminary study shows that there is substantial model-hardware miscorrelation in terms of leakage power because the 2^{nd} DB effect is not comprehended in cell library models. Since the physical location of the 2^{nd} DB matters when tracing timing impacts of optimizations, the physical layout design and signoff flows should be aware of the 2^{nd} DB effect to avoid this model-hardware miscorrelation.

In what follows, we focus on mitigation of analysis error – in particular, underestimation of leakage – that is caused by unawareness of the 2^{nd} DB effect. We assume a place-and-route methodology that inserts only Small fillers, to *a priori* minimize the leakage impact of the 2^{nd} DB effect. We only consider setup timing slacks during our optimization.

This Work. In this work, we study design impacts caused by the 2^{nd} DB effect and propose

[12]Details of our modeling of 2^{nd} DB effect are given in Section 3.1.2.

Table 3.1. Preliminary study of the 2nd DB effect. Design nomenclature specifies design name (AES or VGA), filler insertion strategy (Small (*sm*) or Standard (*std*)), and design type (*Type-I* or *Type-II*). WS denotes the worst slack of a design.

Design	**Hold WS** (ns)	**Setup WS** (ns)	**Leakage** (mW) (Δ%)
AES-*sm*-I	0.108 / 0.108	-0.056 / -0.047	0.387 / 0.742 (91.7%)
AES-*sm*-II	0.113 / 0.113	-0.055 / -0.055	0.419 / 0.456 (8.8%)
VGA-*sm*-I	0.080 / 0.113	0.016 / 0.022	1.486 / 3.186 (114.4%)
VGA-*sm*-II	0.086 / 0.086	-0.002 / -0.002	1.523 / 2.439 (60.1%)
AES-*std*-I	0.108 / 0.108	-0.056 / -0.045	0.387 / 0.909 (134.9%)
AES-*std*-II	0.113 / 0.113	-0.055 / -0.054	0.419 / 0.474 (13.1%)
VGA-*std*-I	0.080 / 0.080	0.016 / 0.024	1.486 / 3.632 (144.4%)
VGA-*std*-II	0.086 / 0.086	-0.002 / -0.001	1.523 / 2.640 (73.3%)

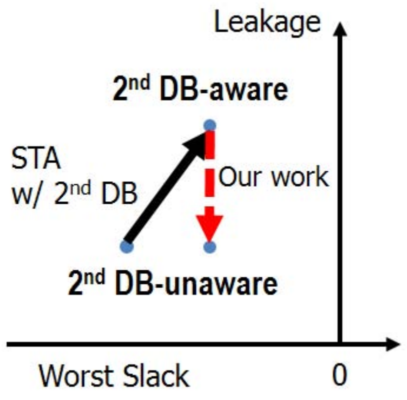


Figure 3.2. Target of our work. We seek to minimize *actual*, 2nd DB-aware leakage power with no timing degradation.

a 2nd DB-aware leakage optimization that uses cell relocation, gate sizing, Vt swapping and DB (Type) swapping to mitigate the P&R tool's lack of awareness of 2nd DB effect. Our main contributions are summarized as follows.

- We study design impacts of the 2nd DB effect for designs using only SDB cells (Type-I) and both SDB and DDB cells (Type-II).
- We propose a 2nd DB-aware leakage optimization and placement methodology that uses relocation, gate sizing, Vt swapping and DB swapping. Our algorithm honors the placement grid for each DB and a specified spacing constraint ($Spacing_{SD}$ in Figure 3.1) between SDB and DDB.

- We achieve 80% leakage recovery (i.e., of the leakage increment seen with correct, 2^{nd} DB-aware analysis) on average without changing design performance.

Figure 3.2 conceptually illustrates the target of our work. We aim to recover leakage increments caused by the 2^{nd} DB effect without any worsening of negative timing slacks.

The remainder of this chapter is organized as follows. Section 3.1.1 describes previous related works. Section 3.1.2 presents our problem statement, leakage model and optimization approach. Section 3.1.3 describes our experimental setup, key experiments and results. We conclude in Section 3.1.4.

3.1.1 Related Work

We categorize relevant previous works as: (i) gate sizing and (ii) local layout effect and placement.

Gate Sizing. Traditional gate sizing problems have been studied for many years. Objectives of the problems are to find the gate width and threshold voltage for each cell so that a circuit can achieve optimized timing, power, and area as a result. Gupta *et al.* [44] describe a sensitivity-based gate sizing algorithm to reduce leakage power. Hu *et al.* [57] and Kahng *et al.* [65] propose global timing recovery and leakage power reduction with a sensitivity-guided greedy sizing algorithm. Wei *et al.* [122] minimize total power using gate sizing and Vt assignment. Luo *et al.* [90] propose a combined methodology with placement and gate sizing. The work of [90] minimizes power by sequential placement, sizing and Vt swapping stages with a slack management scheme. The works of [62] [93] suggest heuristics for co-optimization of sizing and placement with minimum implant area constraints.

Local Layout Effect and Placement. In sub-10nm, various works address LLEs from a standard-cell perspective. Yang *et al.* [128] describe LLEs of 2^{nd} DB and gate-cut stress in 10nm high performance mobile SOCs. Zhao *et al.* [134] introduce gate-cut stress induced LLEs on 14nm FinFET, and Xie *et al.* [125] introduce SDBs and DDBs in actual FinFET devices. At the same time, many recent works aim to mitigate LLEs for detailed placement. Ha *et al.* [46]

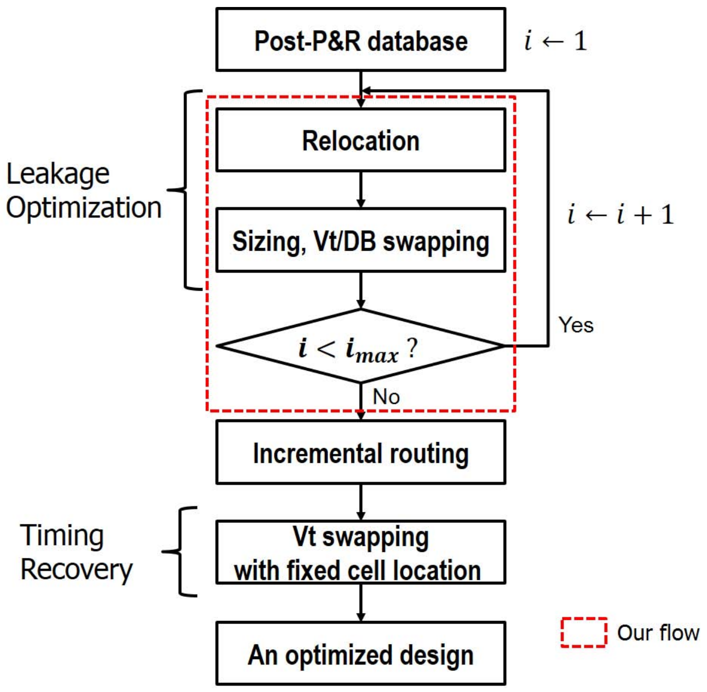


Figure 3.3. Overview of our overall optimization flow. The red box indicates steps that we implement. A commercial P&R tool is used for all other steps in the flow.

introduce a pre-placement methodology to mitigate LLEs from a long SDB created by vertically stacked SDB cells and a 2^{nd} DB-aware timing analysis scheme. Berthelon *et al.* [6] propose a design and technology co-optimization with strain-induced LLEs along with asymmetric and non-rectangular active area. Han *et al.* [50] propose a detailed placement method to minimize the number of abutments between cells, and Du *et al.* [37] deal with new placement constraints of a 16nm technology. Han *et al.* [48] [52] also develop a detailed placement algorithm to reduce the number of "steps", i.e., diffusion height differences, between adjacent cells. Overall, our work is distinguished from the previous works in that (i) we handle both sizing and placement with constraints from SDB and DDB, and (ii) we mitigate leakage increments caused by the 2^{nd} DB effect, which is introduced as a recent type of LLE.

3.1.2 2^{nd} DB-Aware Leakage Optimization and Placement

In this section, we first describe our problem statement and the timing and leakage model for the 2^{nd} DB effect. We then describe our methodology for detailed placement and leakage

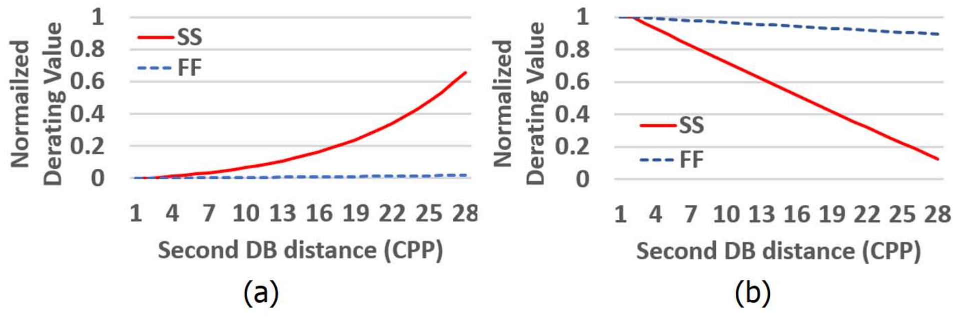


Figure 3.4. Normalized derating values for (a) leakage power and (b) cell delay according to the 2^{nd} DB distance.

optimization. Figure 3.3 shows the overall flow of our optimization. The input of the flow is an optimized post-route design database produced by a commercial P&R tool. Our leakage optimization is performed i_{max} iterations, and is followed by incremental routing. After routing, Vt swapping with fixed cell location recovers negative timing slacks caused by routing changes. This incremental routing and Vt swapping is performed by a commercial P&R tool.

Problem Statement

Given a post-P&R design, our goal is to swap cells and perturb placement so as to minimize leakage with 2^{nd} DB awareness.

Input: A post-P&R design database from a commercial tool.

Output: An optimized design with 2^{nd} DB awareness.

Objective: To minimize leakage power of the design.

Do: Relocation, gate sizing, Vt swapping and DB swapping.

Constraints: No total negative slack (TNS) degradation, no cell overlap, grid-based placement for each DB, and a spacing constraint between SDB and DDB.

Modeling for the 2^{nd} DB Effect

Delay and leakage power of devices are changed by the 2^{nd} DB effect. We introduce our model for 2^{nd} DB using derating values of Vt shift from the work of [128] and our collaborator [149]. Figure 3.4 shows the normalized derating values for leakage power and delay.

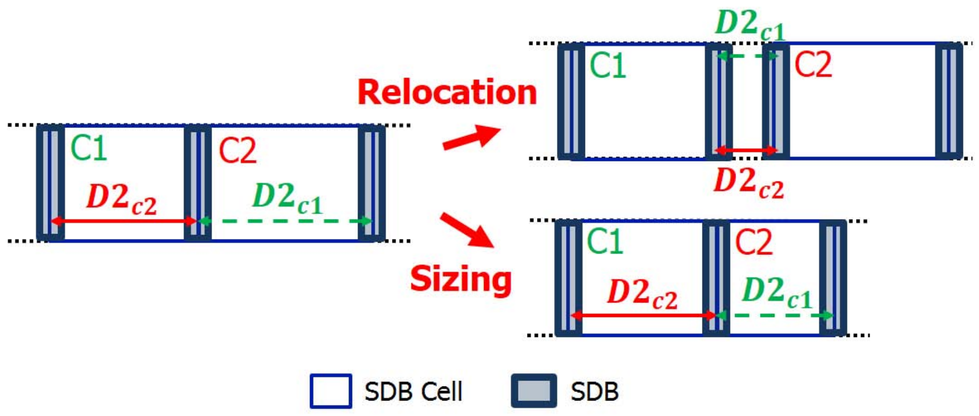


Figure 3.5. Examples of 2nd DB distance change due to cell relocation and sizing. The red color of $D2_{C2}$ indicates the 2nd DB distance of C2 and the green color of $D2_{C1}$ indicates the 2nd DB distance of C1.

According to [149], we use the following sets of process, voltage, and temperature conditions: (SS, 0.72V, -40°C) for setup corner and (FF, 0.88V, 125°C) for hold corner. Since the Vt shifts for PMOS and NMOS are different, we assume that the averaged Vt shift for cells is proportional to 2nd DB distance for SS and FF [149]. Advised by [149], we calculate the derating values for leakage power and delay of cells by assuming that leakage power exponentially increases with 2nd DB distance. In our library, we assume that a given DDB cell is 2CPP wider than the corresponding SDB cell, and that the 2nd DB effect maxes out at the maximum width of standard cells in our library. Furthermore, we follow design rules for the spacing constraint between neighboring SDB and DDB cells. Our derated libraries are created using a 2CPP-step of 2nd DB distance. We do not have a 2nd DB distance of 1CPP because the minimum cell width is 2CPP. And, we also round down odd-valued 2nd DB distances to even values because libraries with a step size of 1CPP incur more than 3$\times$ peak memory usage during our optimization.[13]

[13]When the 2nd DB-unaware leakage is 0.225mW for the AES Type-I design (60% utilization, 0.5ns clock period), the 2nd DB-aware leakage with the 1CPP step size is 0.319mW, and that with the 2CPP step size is 0.295mW; our optimization results (leakage recovery) are 99.0% and 97.1%, respectively.

Table 3.2. Notations.

Notation	Meaning
$m_k.c$	Cell for the k^{th} candidate.
$m_k.m$	Method for the k^{th} candidate.
$m_k.sf$	Sensitivity for the k^{th} candidate.
$m_k.\Delta leakage$	$\Delta leakage$ for the k^{th} candidate.
$\Delta leakage$	$\Delta leakage$ of a design.
M	Set of candidates.
$c.c_l$	Left neighboring cell of cell c.
$c.c_r$	Right neighboring cell of cell c.
$c.2nddb$	2^{nd} *DB* distance of cell c.
W_{min}	Minimum cell width.

Algorithm 3. 2^{nd} DB-aware relocation algorithm.

```
Procedure: $Relocation()$
Inputs: a netlist $D$, a placement of $D$
Outputs: an optimized netlist
1: $M \leftarrow \emptyset$ ; $k \leftarrow 0$
2: for cell $c$ in the netlist do
3:     if cell $c$ is neither a DDB cell nor a flip-flop then
4:         if cell $c$ is abutted with $c.c_l$ and whitespace on right side of cell $c \geq W_{min}$ then
5:             $m_k.c \leftarrow c$ ; $m_k.m \leftarrow$ move to the right by $W_{min}$
6:             update $c.2nddb, c.c_l.2nddb, c.c_r.2nddb$
7:             $m_k.\Delta leakage \leftarrow \Delta leakage$ ; $M \leftarrow M \cup m_k$ ; $k \leftarrow k+1$
8:         end if
9:         if cell $c$ is abutted with $c.c_r$ and whitespace on left side of cell $c \geq W_{min}$ then
10:            $m_k.c \leftarrow c$ ; $m_k.m \leftarrow$ move to the left by $W_{min}$
11:            update $c.2nddb, c.c_l.2nddb, c.c_r.2nddb$
12:            $m_k.\Delta leakage \leftarrow \Delta leakage$ ; $M \leftarrow M \cup m_k$ ; $k \leftarrow k+1$
13:        end if
14:    end if
15: end for
16: while $M \neq \emptyset$ do
17:    Pick $m_k$ with the maximum $m_k.\Delta leakage$ in $M$
18:    Commit $m_k$ ; $M \leftarrow M \backslash m_k$
19:    if TNS degrades then
20:        Revert
21:    end if
22: end while
```

2^{nd} DB-Aware Relocation

We first describe our sensitivity-based method to *relocate* cells with no TNS degradation. Cell location is critical for the 2^{nd} DB effect. Figure 3.5 illustrates how 2^{nd} DB distance varies due to cell relocation and sizing. In this figure, $D2_{C2}$ denotes the 2^{nd} DB distance of cell C2 and $D2_{C1}$ denotes the 2^{nd} DB distance of cell C1. If C2 is relocated to the right by 2CPP, then the 2^{nd} DB distances of both C1 and C2 decrease to 2CPP. If C2 is sized down and still abuts C1, then only $D2_{C1}$ decreases. Therefore, we can reduce the 2^{nd} DB distance by cell relocation, and recover leakage power. Algorithm 3 shows this *relocation* flow (Table 3.2 gives notations that we use). Line 1 initializes an empty set of candidate moves. In Lines 2-15, we calculate candidate moves per cell and the sensitivity per move. Here, we only consider moves of SDB combinational cells, as shown in Lines 2 and 3. In Lines 4-8, for each cell, we add a candidate right move (relocated to the right) and calculate the sensitivity (leakage recovery due to the move) if the cell has no whitespace on the left but whitespace on the right. Similarly, in Lines 9-13, we add a candidate left move and calculate the sensitivity. In Lines 16-21, we greedily commit the candidate move with the highest sensitivity as long as there is no TNS degradation and the placement is legal.

2^{nd} DB-Aware Sizing

Algorithm 4 describes our sensitivity-based methodology for gate sizing, Vt swapping and DB swapping. Similar to the methodology of 2^{nd} DB-aware relocation, we greedily perform either gate sizing, Vt swapping or DB swapping for each cell according to a pre-sorted list. Line 2 initializes an empty set of candidate moves. In Lines 3-20, we first populate the list with all candidate moves per cell, i.e., swapping to a higher Vt, downsizing and DB swapping. $\Delta leakage/\Delta slack$ is used as our sensitivity function and $\Delta slack$ denotes the timing slack difference for the swapped cell. Here, we only consider swaps of combinational cells, as shown in Line 4. In Lines 5-9, we add a candidate and perform the sensitivity calculations for downsizing if the cell is downsizable. Similarly, Lines 10-13 and Lines 14-17 add a candidate and calculate

Algorithm 4. 2^{nd} DB-aware leakage optimization.

```
Procedure: ReduceLeak()
Input: a netlist $D$, a placement of $D$
Output: an optimized netlist
 1: $Relocation(D)$
 2: $M \leftarrow \emptyset$ ; $k \leftarrow 0$
 3: for cell $c$ in the netlist $D$ do
 4:     if cell $c$ is not a flip-flop then
 5:         if cell $c$ is downsizable then
 6:             $m_k.c \leftarrow c$ ; $m_k.m \leftarrow downsize$
 7:             update $c.c_l.2nddb, c.c_r.2nddb$
 8:             $m_k.sf \leftarrow \Delta leakage/\Delta slack$ ; $M \leftarrow M \cup m_k$ ; $k \leftarrow k+1$
 9:         end if
10:         if cell $c$ is not a highest $Vt$ then
11:             $m_k.c \leftarrow c$ ; $m_k.m \leftarrow downVt$
12:             $m_k.sf \leftarrow \Delta leakage/\Delta slack$ ; $M \leftarrow M \cup m_k$ ; $k \leftarrow k+1$
13:         end if
14:         if cell $c.c_l$ or $c.c_r$ is a different DB type from cell $c$ then
15:             $m_k.c \leftarrow c$ ; $m_k.m \leftarrow changeDB$
16:             update $c.c_l.2nddb, c.c_r.2nddb$
17:             $m_k.sf \leftarrow \Delta leakage/\Delta slack$ ; $M \leftarrow M \cup m_k$ ; $k \leftarrow k+1$
18:         end if
19:     end if
20: end for
21: while $M \neq \emptyset$ do
22:     Pick $m_k$ with the maximum $m_k.sf$ in $M$
23:     Commit $m_k$ ; $M \leftarrow M \backslash m_k$
24:     if TNS degrades then
25:         Revert
26:     end if
27: end while
```

the sensitivity for Vt swapping and DB swapping, respectively. 2^{nd} DB distances are calculated for downsizing and DB swapping, as shown in Lines 7 and 16. In Lines 21-27, we then greedily commit the candidate move with the highest sensitivity as long as there is no TNS degradation.

Table 3.3. Experimental results for all the design blocks.

Design information			2^{nd} **DB-unaware**			2^{nd} **DB-aware**			**Our results**				**Runtime**		
Design	**#Inst.**	**Type**	**WS** (ns)	**TNS** (ns)	l_{base} (mW)	**WS** (ns)	**TNS** (ns)	l_{actual} (Δ%) (mW)	**WS** (ns)	**TNS** (ns)	l_{opt} (Δ%) (mW)	*Recovery*	**LeakOpt** (s)	**Routing** (s)	**Total** (s)
AES	13806	Type-I	0.011	0.000	0.225	0.012	0.000	0.295 (31%)	0.001	0.000	0.227 (1%)	97.1%	207	819	1026
	13279	Type-II	0.002	0.000	0.182	0.002	0.000	0.190 (4%)	-0.001	-0.002	0.180 (-1%)	$>$100%	228	740	968
MPEG	12380	Type-I	0.004	0.000	0.227	0.005	0.000	0.272 (20%)	-0.001	-0.003	0.245 (8%)	60.0%	257	902	1159
	12250	Type-II	0.004	0.000	0.228	0.004	0.000	0.264 (16%)	0.002	0.000	0.237 (4%)	75.0%	261	872	1133
JPEG	47061	Type-I	0.002	0.000	0.683	0.002	0.000	0.867 (27%)	0.001	0.000	0.745 (9%)	66.3%	1820	6911	8731
	39422	Type-II	0.003	0.000	0.697	0.003	0.000	0.765 (10%)	0.002	0.000	0.714 (2%)	75.0%	2039	7093	9132
VGA	67491	Type-I	0.011	0.000	1.203	0.012	0.000	1.786 (48%)	0.001	0.000	1.295 (8%)	84.2%	3819	29183	33002
	67546	Type-II	0.015	0.000	1.230	0.015	0.000	1.449 (18%)	-0.002	-0.010	1.256 (2%)	88.1%	3998	30799	34797

3.1.3 Experimental Setup and Results

Experimental Setup

We implement our heuristics in C++ with OpenAccess 2.2.43 [167] to support LEF/DEF [158]. We perform our experiments in a commercial 14nm FinFET technology with 9-track triple-Vt libraries. We apply our optimization to four design blocks, AES, MPEG, JPEG and VGA, from OpenCores [161]. We use two types of designs: Type-I has only SDB cells and Type-II has both SDB and DDB cells. We vary the target clock period from 0.3 to 0.5ns with a step size of 0.05ns, and we vary the block utilization from 60% to 80% with a step size of 5%. For each data point, we denoise the experiments using three runs, by varying the target clock period by $\pm$1ps. We report the median value of the leakage recovery. We use *Synopsys Design Compiler L-2016.03-SP4* [171] for synthesis and *Cadence Innovus Implementation System v17.1* [144] for P&R, with leakage optimization in the highest effort mode. All the timing results in this section are reported by OpenSTA [164]. Due to different timing results between the commercial P&R tool and OpenSTA, we use a correlation methodology from [96] to correlate the P&R tool to OpenSTA. All experiments are performed with eight threads on a 2.6GHz Intel Xeon server.

We measure how much recovery we achieve by our flow, as shown in Equation (3.1).

$$Recovery = 1 - \frac{l_{opt} - l_{base}}{l_{actual} - l_{base}} \tag{3.1}$$

We use l_{base} to denote the leakage value reported by the P&R tool at post-P&R stage, which is 2^{nd} DB-oblivious. We use l_{actual} to denote the 2^{nd} DB-aware leakage value at post-P&R stage. We use l_{opt} to denote the leakage value after our optimization. Thus, *Recovery* is the leakage recovery. Note that according to the metric *Recovery*, it is possible to have >100% leakage recovery. For example, if the 2^{nd} DB-aware analysis increases leakage from 0.182mW (l_{base}) to 0.190mW (l_{actual}), and our optimization reduces leakage to 0.180mW (l_{opt}), then we say that *Recovery* is 125%.

Design Space Exploration

In this section, we explore the sensitivity of results to design space choices. We explore three knobs for the experimental setup: (i) 2CPP padding for all flip-flops, (ii) DB swapping and (iii) multiple optimization iterations. We apply a minimum space of padding between flip-flops, because our optimization does not touch flip-flops. Flip-flops can be abutted with other cells but we require at least 2CPP space between flip-flops. We perform four sets of experiments to validate our experimental setup:

- **Baseline:** A design with 2CPP padding for flip-flops.
- **Expt. 1:** A design without 2CPP padding for flip-flops.
- **Expt. 2:** A design without DB swapping.
- **Expt. 3:** A design with four iterations ($i_{max} = 4$).
- **Expt. 4:** Sensitivity to timer setting (tolerance).

For the baseline, we use 2CPP padding for flip-flops and execute two iterations including DB swapping. Flip-flops are wide and they contribute a large portion of the 2^{nd} DB leakage increase to neighboring cells. Expt. 1 shows that leakage recovery is degraded by 29.7% without 2CPP padding for flip-flops. In Expt. 2, we compare our baseline to the optimization with DB swapping and show that 2.2% leakage recovery can be achieved with the addition of DB swapping. In Expt. 3, we perform our optimization in four iterations and compare the result to the baseline, where we only have two iterations of optimization. Four iterations give more leakage recovery in exchange for longer runtime. In Expt. 4, we explore the sensitivity to timer tolerance. The tolerance for the timer denotes a minimum percentage change in delay that causes propagated delays to be recomputed during incremental timing updates. We sweep the tolerance from 0% to 1% with a 0.1% step. As shown in Table 3.5, a design with 0.8% tolerance shows 67% runtime improvement and the same leakage recovery compared to a design with 0%

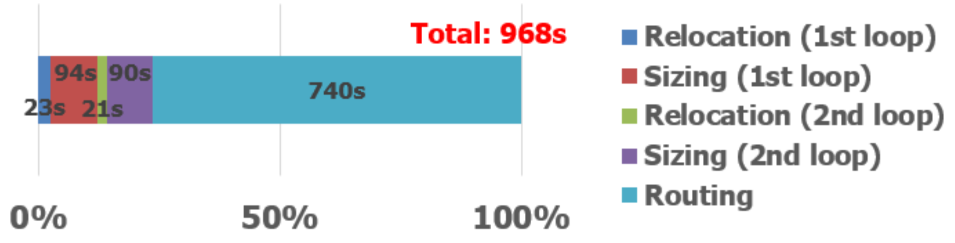


Figure 3.6. Runtime analysis for AES design. We use Type-II, 60% initial utilization and 0.5ns clock period. 1st *loop* denotes the first iteration of our optimization and 2nd *loop* denotes the second iteration. Sizing includes sensitivity calculations and cell moves for gate sizing, Vt swapping and DB swapping.

tolerance. Based on these results, we use 2CPP padding for flip-flops as our default P&R setup, two iterations of optimization including DB swapping, and 0.8% tolerance in incremental timing updates in our results.

Main Results

We conduct two overarching experiments to show results of our flow. The first experiment shows leakage optimization outcomes over four design blocks. The second experiment further studies the sensitivity of leakage recovery to target clock period and initial utilization of the design.

Leakage Optimization. The design information and experimental results are summarized in Table 3.3. The table reports (i) the number of instances in a design (#Inst.), (ii) design type as defined in Figure 3.1 (Type), (iii) worst setup slack (WS), (iv) total negative setup slack (TNS), (v) percentage of leakage increase from 2nd DB-unaware analysis (Δ%), (vi) runtime of leakage optimization including relocation and sizing (LeakOpt), and (vii) runtime of incremental routing by a commercial P&R tool (Routing). We use four design blocks – AES, MPEG, JPEG and VGA – with both Type-I and Type-II designs. Going from 2nd DB-unaware to 2nd DB-aware analysis, the leakage is increased by up to 43%. After our leakage optimization flow with incremental routing and timing recovery performed by the commercial tool, we achieve 80% leakage recovery on average, with negligible timing degradation. Figure 3.6 shows the runtime breakdown in our

Table 3.4. Experimental results for design space exploration. We use AES Type-II design with 60% initial utilization and 0.5ns target clock period.

Expt.	2^{nd} **DB-unaware**		2^{nd} **DB-aware**		**Our results**		
	TNS (ns)	l_{base} (mW)	**TNS** (ns)	l_{actual} (Δ%) (mW)	**TNS** (ns)	l_{opt} (Δ%) (mW)	*Recovery*
Baseline	0.000	0.182	0.000	0.190 (4%)	-0.002	0.180 (-1%)	>100%
Expt. 1	0.000	0.178	0.000	0.198 (11%)	-0.002	0.186 (4%)	60.0%
Expt. 2	0.000	0.182	0.000	0.190 (4%)	-0.002	0.183 (1%)	87.5%
Expt. 3	0.000	0.182	0.000	0.190 (4%)	-0.003	0.178 (-2%)	>100%

flow. We can see that 24% of runtime is spent on gate sizing, Vt swapping and DB swapping, including sensitivity calculations. 76% of runtime is spent on the incremental routing by the commercial P&R tool. Next, we verify the robustness of our optimization with different clock period and utilization.

Sensitivity to Target Clock Period and Utilization. Leakage increases due to the 2^{nd} DB effect and optimization results vary with initial utilization and target clock period. We sweep initial utilization and target clock period for each design so as to study design impacts and what our proposed methods can achieve. Figures 3.7(a) and (b) show leakage increments due to the 2^{nd} DB effect. Figures 3.7(c) and (d) show percentage recovery of the leakage after our optimization. In this experiment, 70% initial utilization is used for target clock period sweep, and 0.4ns clock period is used for initial utilization sweep. As target clock period decreases, the Type-I case has larger leakage increments caused by the 2^{nd} DB effect because cells are closely placed.[14] However, there are smaller changes for the Type-II case because DDB cells mitigate the 2^{nd} DB effect. On the other hand, as initial utilization increases, both the Type-I and Type-II cases have more or similar effects. Our proposed optimization, across the range of clock period and utilization values, achieves 80% leakage recovery on average. For the testcases, the recovery seen in Figures 3.7(c) and (d) is stable across clock periods and utilizations.

[14]With a tight timing constraint, there are more large cells and cells are more closely placed to reduce the net delay, resulting in larger 2^{nd} effect.

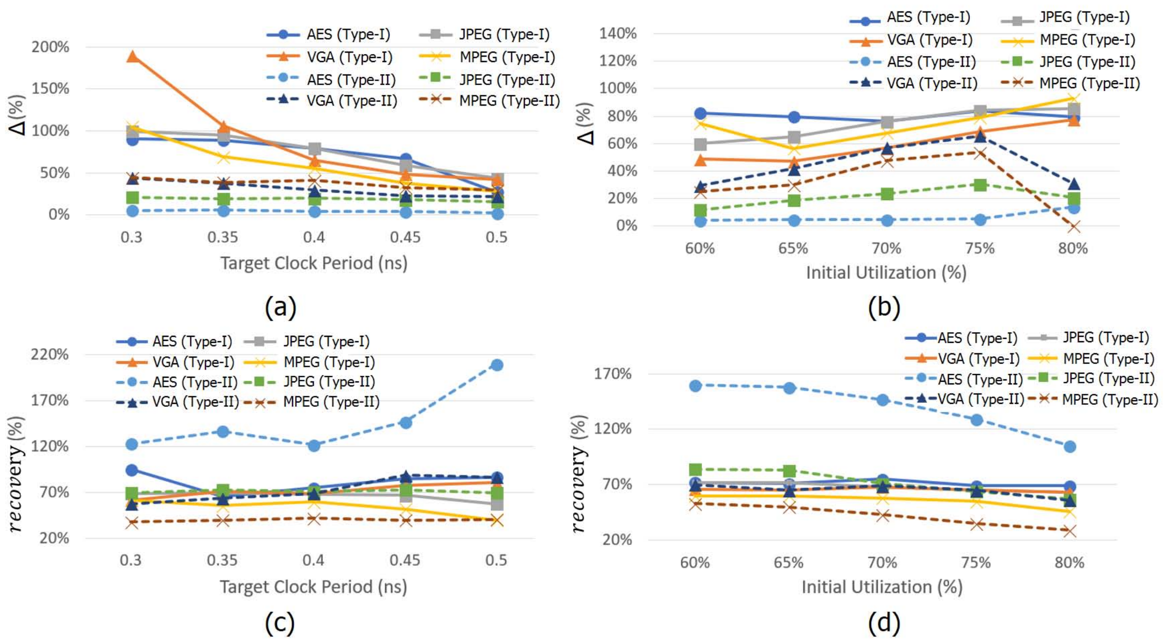


Figure 3.7. Studies of leakage power with varying target clock period and initial utilization. Leakage increment from the 2^{nd} DB effect vs. (a) target clock period and (b) initial utilization. Leakage recovery by our optimization vs. (c) target clock period and (d) initial utilization.

3.1.4 Conclusion

In this work, we study the 2^{nd} DB effect, which is one of the critical LLEs for physical design in sub-10nm technologies, along with mixed-DB design that can have both SDB and DDB cells. The 2^{nd} DB effect is a new challenge to the physical design flow, and designers must take this effect into account in order to mitigate model-hardware miscorrelation. We propose new heuristics to recover leakage power increments caused by the 2^{nd} DB effect, including the use of 2^{nd} DB-aware relocation, gate sizing, Vt swapping and DB swapping while satisfying placement constraints. Our work achieves 80% leakage recovery on average using the proposed flow. Directions for future work include: (i) detailed placement considering inter-row minimum implant width and spacing constraints due to the mix of SDB and DDB cells; (ii) more comprehensive study of sensitivity functions for improved leakage recovery considering multiple corners; and (iii) support of mixed-DB within one cell, e.g., PMOS has SDB and NMOS has DDB within one standard cell.

Table 3.5. Experimental results for sensitivity to the tolerance. The 2nd DB-aware leakage power of the design is 0.190mW. Reloc., 1st loop and 2nd loop denote runtime of the relocation step, and of the first and the second iterations of the optimization, respectively.

Tolerance	**1st loop**		**2nd loop**		**Our results**		
	Reloc. (s)	**Sizing** (s)	**Reloc.** (s)	**Sizing** (s)	**LeakOpt** (s)	**WS** (ns)	**Leak** (mW)
0%	23	320	22	316	681	-0.001	0.180
0.1%	23	152	21	149	345	-0.001	0.180
0.2%	23	142	22	140	327	-0.001	0.180
0.3%	23	137	22	132	314	-0.001	0.180
0.4%	23	134	22	130	309	-0.001	0.180
0.5%	23	128	21	125	297	-0.001	0.180
0.6%	23	119	22	117	281	-0.001	0.180
0.7%	23	107	22	103	255	-0.001	0.180
0.8%	23	94	21	90	228	-0.001	0.180
0.9%	23	90	22	88	223	-0.001	0.184
1.0%	23	87	22	84	216	-0.001	0.187

3.2 Detailed Placement for IR Drop Mitigation by Power Staple Insertion in Sub-10nm VLSI

Distribution of power (VDD) and ground (VSS) through the Power Delivery Network (PDN) is extremely challenging in modern VLSI design. Back-end-of-line (BEOL) resistance increases dramatically in sub-10nm VLSI [7] [129]; this increases delay and requires additional buffers to meet timing requirements. The resulting routing increases capacitive loads and power consumption, which causes greater supply voltage (IR) drop, and requires a denser PDN layout – which causes more congestion again. In sub-10nm technologies, due to this vicious cycle, adding more to the power mesh is not always the best mitigation of IR drop.

Insertion of *power staples*. Traditional PDN structures have power mesh on higher metal layers and power rails on one or two lower metal layers, with stacked vias in between. Insertion of *power staples* is a new technique for improving PDN robustness in sub-10nm technologies. *Power staples* are short pieces of metal connecting two or more adjacent (i.e., consecutive) VDD or VSS rails, to mitigate the IR drop. Figure 3.8 illustrates power staple insertions. In

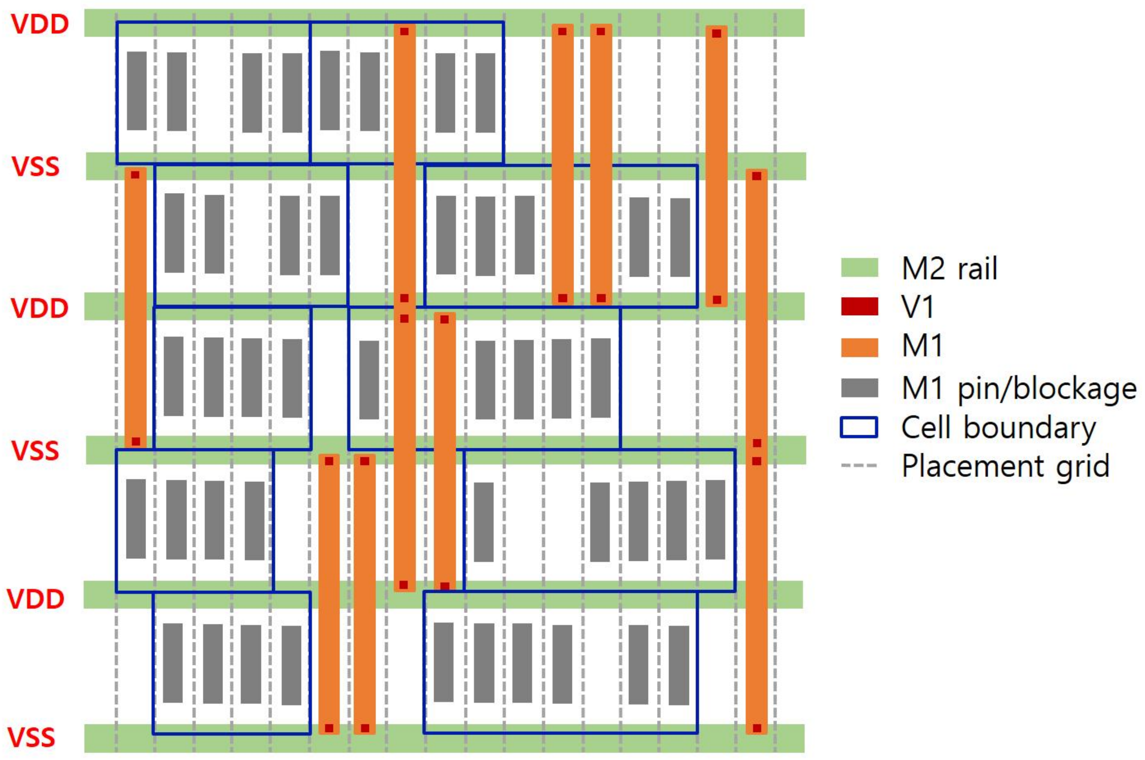


Figure 3.8. Illustration of power staple insertions.

this example, adjacent M2 rails (VDD-VDD, or VSS-VSS) are connected by power staples in M1. Since each power staple goes across at least two cell rows, vertical track availability in the context of standard-cell pins and pre-routes is crucial to achieve sufficient power staple insertion.

Present-day limitations and our approach. There are two basic types of power staple insertion flows: *pre-placement* and *post-placement*. A *pre-placement* flow inserts power staples at fixed intervals before placement. This strategy can achieve good IR drop, but creates regular hard placement blockages, and thus affects the placement flexibility. Alternatively, a *post-placement* flow inserts power staples opportunistically after placement, wherever long empty vertical segments (spanning at least two cell rows) are available. However, this approach is inherently limited by the placement solution: especially for high-utilization designs where empty tracks are usually not aligned between adjacent rows, the post-placement flow will result in fewer power staples and worse IR drop than the pre-placement flow.[15]

In this work, we present new single-row and double-row *dynamic programming* (DP)-based detailed placement optimizations to improve power staple insertion in *post-placement* flows. Our contributions are summarized as follows.

- We propose a single-row dynamic programming-based approach to maximize a given objective function which comprehends the benefit from placing power staples. The proposed algorithm preserves the original cell ordering within each row.
- We propose a double- (multi)-row dynamic programming-based approach to maximize the objective function. The multi-row algorithm can more aggressively optimize the placement while keeping cells in their original rows and preserving row ordering.
- We propose a heuristic weighting function that balances the number of power staples for VDD and VSS.
- We perform extensive studies on the sensitivity and scalability of our algorithms, and show up to 13.2% IR drop reduction compared to a *post-placement* flow without our optimization.

3.2.1 Related Work

Power staples and IR drop-aware placement. Power staple insertion is introduced in several works. [94] proposes a design-technology co-optimization (DTCO) framework with power staple insertion to mitigate IR drop. [131] introduces a "jumper" connection between vertically separated power rails to mitigate both dynamic IR drop and electromigration. [85] shows cell architectures using power staples, with both pre- and post-placement methodologies. Other works address IR drop during placement, e.g., [67] and [71] use analytic placement. [31] calculates

[15]This is not an apples-to-apples comparison: the pre-placement flow inherently opts for PDN robustness over layout density, and cannot achieve very high utilizations.

Table 3.6. Notations.

Notations	**Meaning**
C	Set of cells in a window of the initial placement.
c_k	k^{th} cell in the left-to-right ordered initial placement.
x_k	Initial x coordinate of the cell c_k. We define $x_0 = 0$ for convenience.
$[-x_\Delta, x_\Delta]$	Horizontal displacement range of the cells.
w_k	Width of the cell c_k. We define $w_0 = 0$ for convenience.
$[1, x_{max}]$	Valid sites for cell placement, starting from 1.
$d[i][j]$	The maximum value of the benefit function of the staples that can be placed when the cell c_i is at coordinate $x_{c_i} + j$. i ranges from 0 to n and j ranges from $-x_\Delta$ to x_Δ.
Additional notations for multi-row DP	
c_{rk}	The k^{th} cell, in left-to-right order, in row r.
w_{rk}	Width of the k^{th} cell in the left-to-right order in row r.
$D[i]\ [s_{1i}][l_1]\ [s_{2i}][l_2]$	The maximum value of the benefit function of the staples that can be placed from coordinate 1 to coordinate i, knowing that the nearest cell on the first row is $c_{s_{1}i}$ at coordinate $i - l_1$ and the nearest cell on the second row is $c_{s_{2}i}$ at coordinate $i - l_2$. Note that $s_{ri} = 0$ means that no cell in row r is placed.

current and voltage drop based on an admittance matrix, and proposes a cost function to guide placement perturbation.

Graph- and DP-based placement. Graph- and DP-based optimizations are widely used in detailed placement targeting various objectives. [68] proposes a series of methods in ordered single-row placement using a shortest-path algorithm to minimize perturbation or wirelength. [48] describes an optimal single-row and double-row DP optimization to address the *neighbor diffusion effect*. Their multi-row DP supports double-height cells, with cell relocation, reordering and flipping. [87] describes an ordered double-row placement with support of multi-height cells. A heuristic chain move algorithm is used to further improve the wirelength.

In summary, the works of [48] [68] [87] propose graph or dynamic programming models targeting various problems in detailed placement. They also support various cell movements and cell heights, which are not necessarily needed to optimize power staple insertion. Our work is distinguished in that (i) we formulate a single-row and double-row dynamic programming-based approach to maximize benefit functions that include the number and length of power staples; (ii) our double-row DP algorithm makes progress **site by site** instead of cell by cell as in [48], which allows precise calculation of power staple benefits (e.g., empty tracks available across multiple cell rows) per track; and (iii) we show the benefits of our optimization according to both #staples as well as IR drop in a *post-placement* power staple insertion flow.

Algorithm 5. Single-row optimization.

```
 1: Initialize for all legal $(i, j)$
 2:     $d[i][j] \leftarrow -\infty$, $d[0][j] \leftarrow 0$
 3: for $i = 0$ to $|C|$ do
 4:     for $j = -x_\Delta$ to $x_\Delta$ do
 5:         for all $(j') \in getNext(i, j)$ do
 6:             $i' = i + 1$
 7:             $t \leftarrow d[i][j] + getBenefit(x_{c_i} + w_{c_i} + j,\ x_{c_{i'}} + w_{c_i'} + j' - 1)$ - $\alpha \cdot disp(j')$
 8:             $d[i'][j'] \leftarrow \max(d[i'][j'], t)$
 9:         end for
10:     end for
11: end for
12: $finalBenefit \leftarrow \max(d[|C|][j])$
13: Return $finalBenefit$
14: Procedure $getNext$
15: Input: cell $i$ and displacement $j$ of cell $c_i$
16: Initialize $nextList \leftarrow \emptyset$
17: for $j' = -x_\Delta$ to $x_\Delta$ do
18:     $i' = i + 1$
19:     if $inbound(x_{c_{i'}} + j')$ $and$ $x_{c_{i'}} + j' \geq x_{c_i} + j + w_{c_i}$ then
20:         $nextList \leftarrow j'$
21:     end if
22: end for
23: Return $nextList$
```

3.2.2 Our Approach

Single-Row Optimization

Single-Row Optimization Problem: Given an initial legalized single-row placement and benefit table of power staples, perturb the placement to maximize power staple benefits to upper and lower rows while preserving the original cell ordering and placement legality.

For single-row dynamic programming, we place one cell at a time until all cells have been placed. Table 3.6 shows notations that we use in our formulation. For each cell c_k, cell index k is that cell's position in the row, from left-to-right. Given a cell set C, the leftmost cell is c_1 and the rightmost is $c_{|C|}$.

For each cell, x_{c_k} is the location of cell c_k in the original placement. The *displacement range* $[-x_\Delta, x_\Delta]$ means that we cannot place a cell more than x_Δ sites away from the original location. Thus, cell c_k can only be placed in the interval $[x_{c_k} - x_\Delta, x_{c_k} + x_\Delta]$.

We use a 2D array $d[i][j]$ to represent the best-to-now solution (benefit) when we have placed cell c_i at location $x_i + j$. Thus, we obtain the best solution when we have placed all $|C|$ cells. The array is of size $(|C|+1) \times (2 * x_\Delta + 1)$ where we initialize all $d[0][j]$ to have an initial benefit of zero. Since cell ordering is preserved, the best result can be found by simply finding the most beneficial solution across all $d[|C|][j]$.

Algorithm 5 describes our dynamic programming optimization. Lines 1-2 initialize all DP array entries to have a benefit of negative infinity, except that all $d[0][j]$ are initialized to have an initial benefit of zero. Lines 3-8 describe the main DP algorithm. The algorithm places each cell one at a time, until all cells are placed. From each current placement of i cells with cell c_i at $x_i + j$, we try to place cell c_{i+1} at all legal sites j'. Function $getNext(i, j)$ gets results for all legal sites j' so that cell c_{i+1} does not overlap with any previous cells. The benefit is updated incrementally using the $getBenefit$ function. To encourage smaller displacement for each cell relative to its initial placement location, we subtract a displacement cost with a displacement factor α. $d[i'][j']$ is updated whenever the current solution is better in terms of t. Line 12 gets the best single-row placement solution across all $d[|C|][j]$. We also store a pointer for each entry of the DP array so that the algorithm can trace back from $d[i'][j']$ to $d[i][j]$, from which $d[i'][j']$ gets updated. In procedure *getNext* of Algorithm 5, Line 16 initializes the legal location for cell $c_{i'}$ to be an empty list. Lines 17-20 add each available legal displacement j to the list.

The procedure $getBenefit$ takes the left and right coordinates of cell $c_{i'}$ and incrementally calculates the new power staples within this range. Since power staples should span at least two row heights, a simple benefit table could be as follows: a power staple length of one row height gets a benefit of zero; a power staple length between two and ten row heights gets a benefit of one (i.e., the power staple's length is extended by the two row-height stapling increment in the current cell row); and a power staple length larger than 10 row heights does not get any benefit (i.e., we do not encourage power staples longer than 10 row heights). Further discussion of the benefit table is given in Section 3.2.3.

Double-Row Optimization

We present our problem statement and double-row (extendable to multi-row) dynamic programming-based detailed placement.

Double-Row Optimization Problem: Given an initial legalized double-row placement and benefit table of power staples, perturb the placement to maximize power staple benefits to upper and lower rows without inter-row relocation while preserving the original cell ordering and placement legality.

Unlike our single-row dynamic programming algorithm, our double-row dynamic programming algorithm makes progress site by site instead of cell by cell. We use an array $D[i][s_1][l_1][s_2][l_2]$ to represent the solution and benefit up to site i. Recall that in single-row optimization, $d[i][j]$ represents a solution with $(i-1)$ cells placed on the left of $x_i + j$, and cell c_i placed at $x_i + j$. Here, a valid solution up to site i may have cells placed on the left of site i, or crossing site i. s_1 (resp. s_2) indicates the last placed cell c_{s_1} (resp. c_{s_2}) in row 1 (resp. row 2), and l_1 (resp. l_2) indicates the distance between the left boundary of c_{s_1} (resp. c_{s_2}) and site i.

Figure 3.9 gives an example. The long vertical line indicates the current site i, and we have placed one cell in row 1, and two cells in row 2. From the figure, the left boundary of the last placed cell in row 1 has a distance of four to site i, and the left boundary of the second placed cell in row 2 has a distance of two to site i. Thus, such a solution is represented by DP array entry $D[i][1][4][2][2]$.

To progress from $D[i][s_1][l_1][s_2][l_2]$ to $D[i+1]$, we must: (i) choose whether to place new cells in the two rows, and thus increase l_1 and/or l_2; and (ii) update the benefit incrementally by

Algorithm 6. Double-row optimization.

```
1:  **Initialize for all** $(i, s_1, l_1, s_2, l_2)$
2:      $D[i][s_1][l_1][s_2][l_2] \leftarrow -\infty$
3:      $D[0][s_1][l_1][s_2][l_2] \leftarrow 0$
4:  **for** $i = 0$ to $x_{max}$ **do**
5:      **for all** $(s_1, l_1, s_2, l_2)$ **do**
6:          **for all** $(i', s'_1, l'_1, s'_2, l'_2) \in getNext(i, s_1, l_1, s_2, l_2)$ **do**
7:              $t \leftarrow D[i][s_1][l_1][s_2][l_2] + getBenefit(i', s'_1, l'_1, s'_2, l'_2) - \alpha \cdot disp(s_1, s'_1, l'_1, s_2, s'_2, l'_2)$
8:              $D[i'][s'_1][l'_1][s'_2][l'_2] \leftarrow (max(D[i'][s'_1][l'_1][s'_2][l'_2], t)$
9:          **end for**
10:     **end for**
11: **end for**
12: $finalBenefit \leftarrow \max(d[x_{max}][|c_1|][l_1][|c_2|][l_2])$
13: **Return** $finalBenefit$
14: **Procedure** $getNext$
15: **Input:** $i, s_1, l_1, s_2, l_2$
16: **Initialize** $nextList \leftarrow \emptyset$
17: $i' \leftarrow i + 1$
18: **Only place a cell on the first row**
19: $s'_1 \leftarrow s_1 + 1$, $l'_1 \leftarrow 0$, $s'_2 \leftarrow s_2$, $l'_2 \leftarrow l_2 + 1$
20: **if** $legalPlace(s_1, l_1, s'_1)$ **then**
21:     $nextList.add \leftarrow (i', s'_1, l'_1, s'_2, l'_2)$
22: **end if**
23: **Only place a cell on the second row**
24: $s'_1 \leftarrow s_1$, $l'_1 \leftarrow l_1 + 1$, $s'_2 \leftarrow s_2 + 1$, $l'_2 \leftarrow 0$
25: **if** $legalPlace(s_2, l_2, s'_2)$ **then**
26:     $nextList.add \leftarrow (i', s'_1, l'_1, s'_2, l'_2)$
27: **end if**
28: **Place cells on both rows**
29: $s'_1 \leftarrow s_1 + 1$, $l'_1 \leftarrow 0$, $s'_2 \leftarrow s_2 + 1$, $l'_2 \leftarrow 0$
30: **if** $legalPlace(s_1, l_1, s'_1)$ *and* $legalPlace(s_2, l_2, s'_2)$ **then**
31:     $nextList.add \leftarrow (i', s'_1, l'_1, s'_2, l'_2)$
32: **end if**
33: **Place no new cells**
34: $s'_1 \leftarrow s_1$, $l'_1 \leftarrow l_1 + 1$, $s'_2 \leftarrow s_2$, $l'_2 \leftarrow l_2 + 1$
35: $nextList.add \leftarrow (i', s'_1, l'_1, s'_2, l'_2)$
36: **Return** $nextList$
```

adding the new power staples at site $i+1$ and subtracting any displacement cost of the new cells. Note that since we go over all sites, assuming we make progress from site i to $i+1$, we can limit the new cell placement (with its left boundary) to be exactly at site $i+1$ without sacrificing the solution space. A new cell placement at sites other than $i+1$, e.g., $i'+1$, will be handled when we make progress from i' to $i'+1$.

Algorithm 6 describes our double-row optimization. Lines 1-3 initialize the DP array entry to have a negative infinity benefit, except that all $D[0]$ entries have an initial benefit of zero. For the current DP entry $D[i][s_1][l_1][s_2][l_2]$ in Line 5, Lines 6-8 update the corresponding DP entry at site $i' = i+1$ for all legal $(i', s'_1, l'_1, s'_2, l'_2)$. The displacement function checks whether there are newly placed cells, and will only count the displacement for the newly placed cells. Then, the final solution is obtained by finding the most beneficial solution when we go over all sites, and all cells. In Algorithm 6, procedure *getNext* generates all legal $(i', s'_1, l'_1, s'_2, l'_2)$. From the current DP entry, Lines 18-21 generate all legal $(i', s'_1, l'_1, s'_2, l'_2)$ when we place a cell only on the first row; similarly, Lines 23-26, 28-31, and 33-35 generate all legal $(i', s'_1, l'_1, s'_2, l'_2)$ when we place a cell only on the second row, on both rows, or on no rows, respectively.

For dynamic programming optimization of more than two rows, we can just add DP array dimensions $[s_k][l_k]$ for each added row. However, the added dimensions may grow the total runtime and memory footprint beyond practical usage. In this work, we use double-row dynamic programming for optimization.

The size of the DP array for n rows is $O(x_{max}(s_{max}(2 * x_\Delta + w_{max}))^n)$, where w_{max} is the maximum cell width and s_{max} is the maximum value for dimension s_k. Experiments using our implementation reveal runtimes that are more obviously dependent on the number of *getNext()* function calls, as opposed to the size of the DP array. Our experiments also show that the number of *getNext()* function calls mainly depends on the number of cells in the optimization window and the displacement range x_Δ. This points to tuning of our DP implementation as a necessary future improvement.

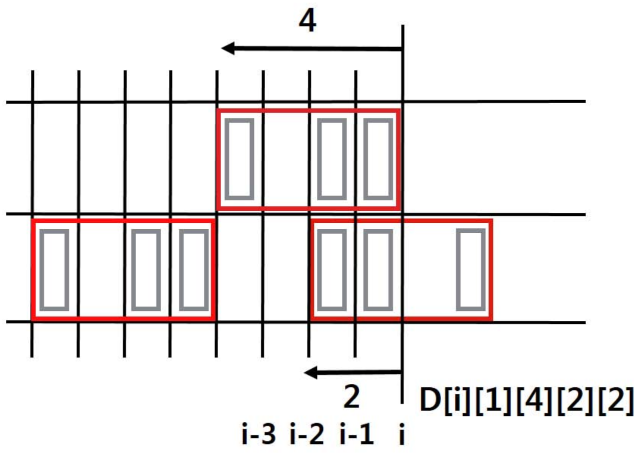


Figure 3.9. The case represented by D[i][1][4][2][2].

Overall Flow

Figure 3.10 shows the overall optimization flow. Given a post-P&R database, we first perform our dynamic programming-based detailed placement optimization to maximize the potential number of power staples, and then perform incremental routing and Vt swapping to fix routing and timing degradation after our optimization. Next, we perform power staple insertion to complete the PDN and perform a vectorless dynamic IR drop analysis to show the results. We note that our optimization and power staple insertion could also be performed in both post-place and post-CTS stages. However, to compare the QoR with, and without, our optimization using a single database, in this work we only perform our optimization in the post-route stage, as shown in Figure 3.10.

3.2.3 Experimental Setup and Results

We implement our dynamic programming in C++ with OpenAccess 2.2.43 [167] to support LEF/DEF [158]. We perform experiments in a 7nm FinFET technology with multi-height triple-Vt libraries from a leading technology consortium.[16] We apply our optimization

[16]We use the process, voltage, and temperature conditions (TT, 0.65V, $25^{\circ}C$) for both place-and-route and IR-drop analysis. We build the power mesh on M7 to M9 with $0.5\mu m$ width, $15.5\mu m$ pitch and $7\mu m$ offset, and M2 power rails. We use M1 and V12 for power staples, and M1 is not allowed for routing.

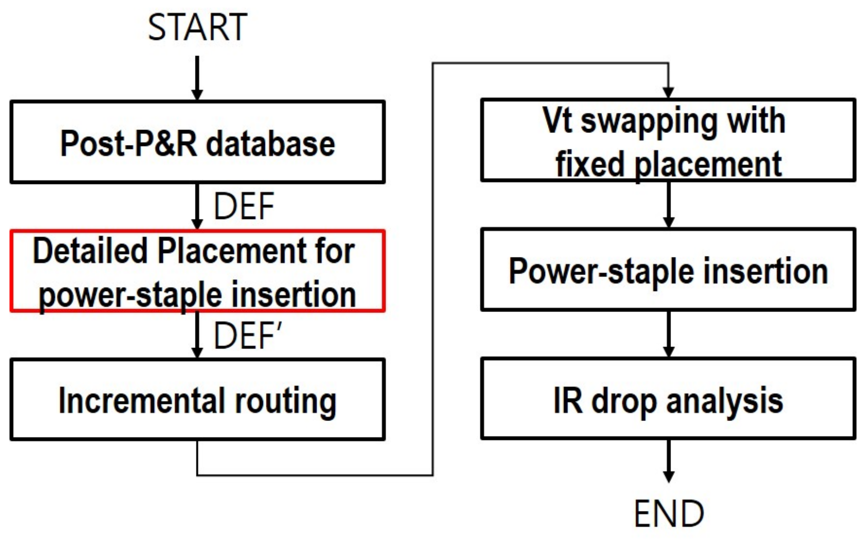


Figure 3.10. Overall flow of our work. The red box indicates steps that we implement. Commercial tools [144] [148] are used for all other steps.

to Arm Cortex-M0 and three design blocks (AES, JPEG and MPEG) from OpenCores [161]. Design information is summarized in Table 3.7.[17]

In this section, we first investigate sensitivity and scalability with respect to the displacement range. Second, we study effects of weighting factors, i.e., displacement factor and benefit table. Third, we propose a heuristic benefit table that is able to balance VDD and VSS staples. Fourth, we show experimental results across different design blocks and utilizations, with IR drop analysis. Last, we compare three examples of metaheuristics that can be implemented around our basic DP optimization. In the following experiments, we use our double-row optimization by default, and we show the comparison with single-row optimization.

Scalability/Sensitivity Study

Since the displacement range (x_Δ) determines the solution space, and also the size of the DP table data structure, we first study the sensitivity of #staples to the displacement range, and

[17]We synthesize designs using *Synopsys Design Compiler L-2016.03-SP4* [171], and perform place-and-route using *Cadence Innovus Implementation System v16.2* [144]. We perform vectorless dynamic IR drop analysis using *Cadence Voltus IC power integrity solution* [148]. Our dynamic programming-based optimizations are performed in a single thread on a 2.6GHz Intel Xeon server.

Table 3.7. Design information.

Design	#Inst	Clkp
M0	∼10K	500ps
AES	∼12K	500ps
MPEG	∼14K	500ps
JPEG	∼54K	500ps

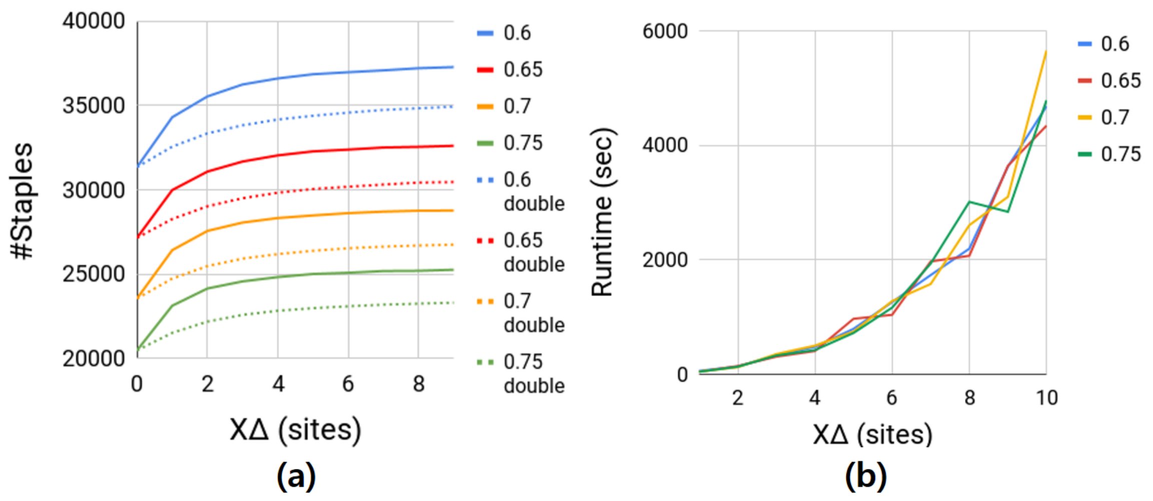


Figure 3.11. (a) Sensitivity of #staples to displacement range x_Δ and (b) sensitivity of runtime to displacement range (x_Δ).

the scalability of our algorithm when we increase the displacement range. In this experiment, we sweep x_Δ from 0 to 10 with a step of 1. To show the stability of our optimization with respect to block utilization, we use design block AES with four target utilizations: 0.60, 0.65, 0.70 and 0.75. In this experiment, since weighting factors do not affect the scalability, they have a default value of 0. The benefit table simply shows the total length of power staples that can be inserted.

Figure 3.11(a) shows the number of power staples vs. displacement range. The *x*-axis is the displacement range, and the *y*-axis is #staples. We can see that for values of displacement range > 5, there are diminishing returns in terms of #staples. Also, design blocks with a lower utilization tend to saturate with larger #staples, as one would expect. These data motivate future efforts to obtain a more detailed and accurate understanding of the relationship between target utilization and maximum IR drop, e.g., for use in early design exploration.[18] Note that the saturation of #staples versus displacement range depends on the library used. The dotted lines in Figure 3.11(a) show the (normalized) result when all widths are doubled (x-coordinates, and cell widths and pin locations in each library cell). With the wider cells, increasing the displacement range continues to give staple insertion benefits up to larger values of x_Δ.

Figure 3.11(b) shows the runtime vs. displacement range. The run with $x_\Delta = 9$ consumes $4\times$ runtime compared to the run with $x_\Delta = 5$, but only results in minor increase in #staples. Utilization generally does not affect the runtime, showing the robustness of our optimization. Thus, to balance the solution quality and runtime, we choose $x_\Delta = 5$ in all the following experiments.

Another study examines whether there is any benefit to breaking the optimization into multiple, more "gradual" phases. Table 3.8 shows how number of staples changes when we run two rounds of optimizations with total displacement range $(x_\Delta) = 6$. The outcome is worst when we perform two phases of optimization, each with displacement range $(x_\Delta) = 3$. The apparent takeaway is that it is better to consume available displacement range "in one shot".

Study of Weighting Factors

Our next experiment studies the impact of the displacement factor (α). We sweep the displacement factor with value 0, and from $10^{-3.5}$ to $10^{-0.5}$, with a multiplier of $10^{0.5}$. We still use the same design block AES with four different utilizations. Figure 3.12(a) shows the #staples vs. displacement factor and Figure 3.12(b) shows the total cell displacement vs. displacement factor.

From Figure 3.12, with a non-zero weight, total displacement decreases sharply while #staples does not change much. This indicates that we can preserve most of #staples while

[18]As our experiments show, there is less available increment in the number of placed staples when the utilization is higher. Decreasing (resp. increasing) utilization reduces current density while increasing (resp. decreasing) potential staple insertion. I.e., there is a compounding of IR drop mitigation (resp., degradation) effects, which is further compounded by changes to gate sizing and voltage-induced timing margins. Deriving a principled model of this dynamic is an open challenge.

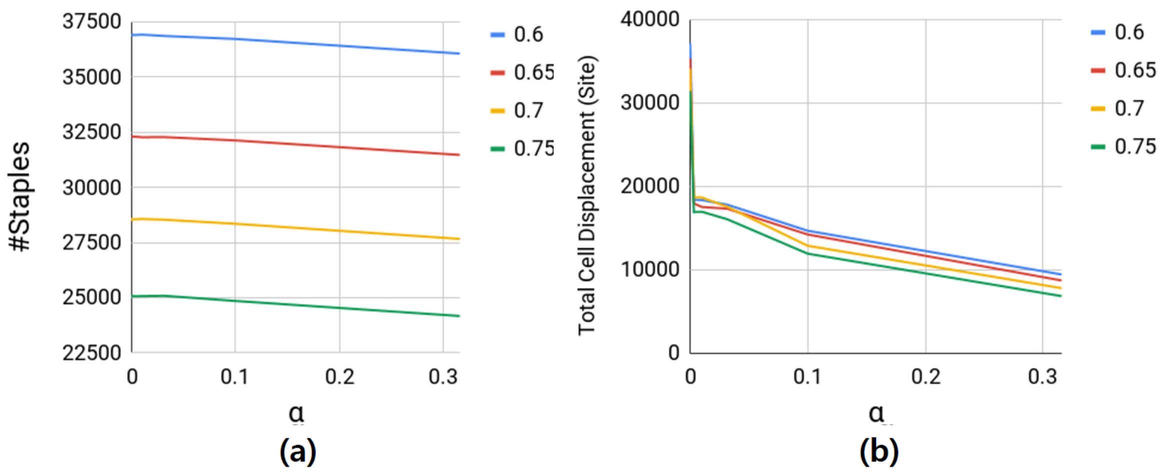


Figure 3.12. Sensitivities of (a) #staples and (b) total cell displacement, to the displacement factor (α).

Table 3.8. Total #staples placed when we apply two rounds of optimization with total displacement range $(x_\Delta) = 6$. Each row represents design AES with utilization between 0.60 and 0.75.

Util	x_Δ					
	6	**1+5**	**2+4**	**3+3**	**4+2**	**5+1**
0.60	36974	36857	36634	36493	36611	36748
0.65	32389	32240	32079	31936	32054	32199
0.70	28624	28540	28362	28252	28287	28432
0.75	25090	25939	24867	24722	24780	24915

reducing total cell displacement by using a non-zero α. A smaller total cell displacement is beneficial to minimize perturbation of the input layout and thus we use $\alpha = 0.01$ in all the following experiments. Note that this choice is based on our experiments, and that a different α may be preferred in a different technology node or design enablement.

Study of Benefit Table

We now discuss our investigations into the benefit table. First, we study the tradeoff between short and long staples, i.e., whether we should encourage fewer but longer staples, or more but shorter staples. Since our optimization changes the placement, to make a fair

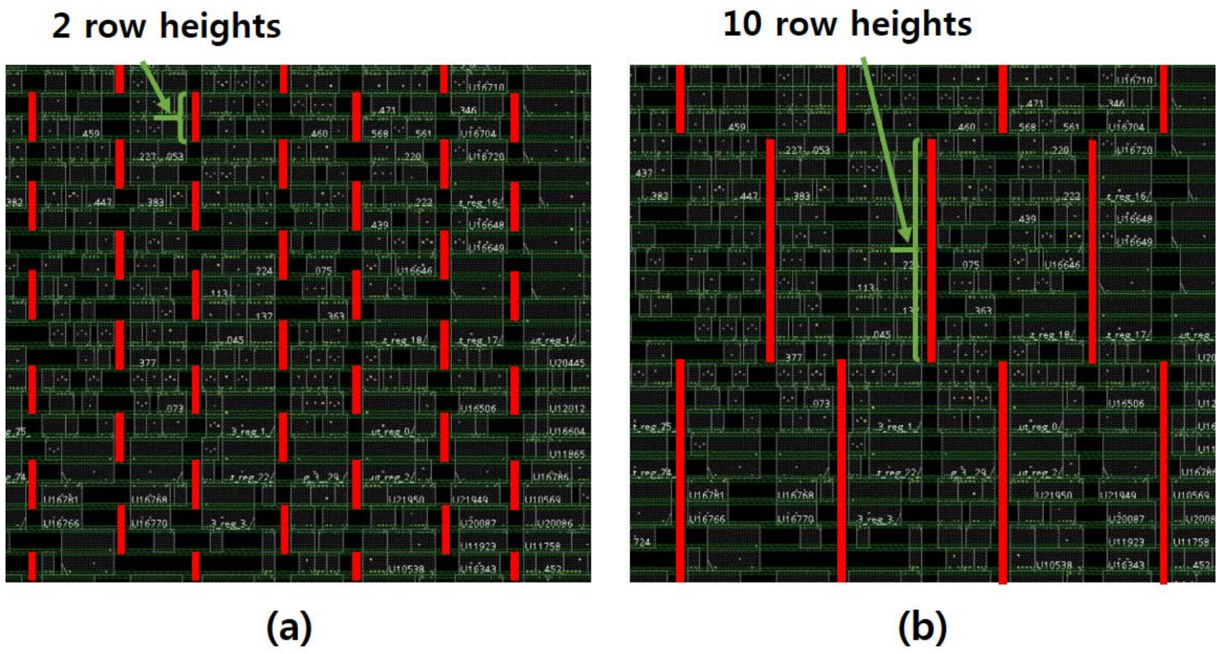


Figure 3.13. Design examples with different lengths of power staples. (a) Power staple length = 2 row heights. (b) Power staple length = 10 row heights.

comparison and investigate the tradeoff, we use the AES design with pre-placed power staples at fixed intervals (20CPP) and perform IR drop analysis.[19] Each run uses a different staple length, from 2 to 10 row heights, with a step size of 2 row heights. We only use VDD power staples in this study.

As shown in Figure 3.13, the total length of all power staples is kept essentially constant across different configurations. We also ensure that the initial input placement is identical across all runs, with power staple tracks always free from pins, blockages and pre-routes. As shown in Table 3.9, the IR drop is not sensitive to the staple length.

Table 3.9. IR drop vs. staple length.

Staple length	2	4	6	8	10
Worst IR drop (mV)	67.9	67.8	67.7	67.7	67.6

[19]*CPP* = contacted poly pitch. In this work, we use a technology enablement where 1CPP = 42nm.

Now we propose our heuristic benefit table setting to improve the balance between #VDD and VSS staples. Since the double-row dynamic programming is performed every two rows in a window, without overlapping (e.g., optimize 1^{st} and 2^{nd} rows, then 3^{rd} and 4^{th} rows), the optimization is always aligned with either VDD rails or with VSS rails, resulting in potentially biased power staples (e.g., staples of one rail are systematically more likely to occur than those of the other rail). To overcome this issue, we propose a reconfiguration of the benefit table so that a single-height staple gets a non-zero benefit (β), with all other staple lengths getting a benefit of one (up to ten row heights). Such single-height staples may get extended in the next optimization window so that there is a chance to re-balance VDD/VSS staples. We have experimentally swept β from 0 to 1.0, with a step size of 0.1, with results shown in Figure 3.14(a). As β increases, VDD/VSS staples become balanced without sacrificing the total #staples. We use $\beta = 0.7$ in all following experiments.

Metaheuristics

In this section, we compare three strategies: (i) single-row optimization (SR); (ii) double-row optimization with overlap (Meta-1); and (iii) two-pass double-row optimization without overlap (Meta-2). In (ii), we perform the double-row optimization every two rows with one row overlapped (e.g., we optimize row 1 and row 2, then row 2 and row 3, then row 3 and row 4, etc.). In (iii), the first pass starts at the odd rows, and then the second pass starts at the even rows (e.g., we optimize row 1 and row 2, row 3 and row 4, etc. in the first pass, and row 2 and row 3, etc. in the second pass).

Therefore, effectively each row is covered once in SR, and twice in Meta-1 and Meta-2. We show the experimental results in Figure 3.14(b). Compared to Baseline, where there is no detailed placement optimization, the methods SR, Meta-1 and Meta-2 respectively achieve 20.1%, 22.3% and 22.7% more inserted staples, averaged over all four target utilizations. Thus, we use Meta-2 for our main experimental results.[20]

[20]Note that there is generally no benefit from making multiple passes over the rows of the layout with the same total length of staples. Our ongoing efforts seek richer combinations of metaheuristics to further improve results.

Table 3.10. Experimental results using double-row optimization.

Design	**Util**	**#Staples**		**IR drop (mV,Avg)**		**IR drop (mV,Worst)**		**ΔWNS**	**Runtime**
		Init	Final (Δ%)	Init	Final (Δ%)	Init	Final (Δ%)	(ns)	(s)
M0	60%	29109	33423 (+14.8%)	29	28 (-3.4%)	76	66 (-13.2%)	-0.009	2666
	65%	25295	29254 (+15.7%)	28	28 (0.0%)	58	60 (+3.4%)	-0.016	2249.5
	70%	22233	25628 (+15.3%)	29	29 (0.0%)	77	76 (-1.3%)	-0.008	2450
	75%	19700	22889 (+16.2%)	29	29 (0.0%)	80	80 (0.0%)	-0.017	2058
AES	60%	31355	37813 (+20.6%)	23	23 (0.0%)	51	47 (-5.6%)	-0.001	2542.5
	65%	27165	33164 (+22.1%)	22	22 (0.0%)	54	51 (-5.6%)	-0.001	2127
	70%	23571	29129 (+23.6%)	22	22 (0.0%)	63	62 (-1.6%)	-0.001	3142
	75%	20496	25530 (+24.6%)	22	22 (0.0%)	71	68 (-4.2%)	-0.006	2520
MPEG	60%	75781	80951 (+6.8%)	24	24 (0.0%)	64	60 (-6.3%)	0.000	3700
	65%	67836	72685 (+7.1%)	24	24 (0.0%)	57	60 (+5.3%)	0.000	4088.5
	70%	61152	65362 (+6.9%)	25	25 (0.0%)	62	59 (-4.8%)	0.000	4410
	75%	55426	58928 (+6.3%)	30	30 (0.0%)	81	79 (-2.5%)	-0.002	3709
JPEG	60%	217467	244333 (+12.4%)	28	27 (-3.6%)	88	84 (-4.5%)	-0.001	16679
	65%	193089	218483 (+13.2%)	26	27 (+3.8%)	79	76 (-3.8%)	0.000	13869.5
	70%	171857	195781 (+13.9%)	28	28 (0.0%)	69	70 (+1.4%)	-0.007	15069
	75%	152805	174618 (+14.3%)	29	28 (-3.4%)	73	68 (-6.8%)	0.000	12563.5

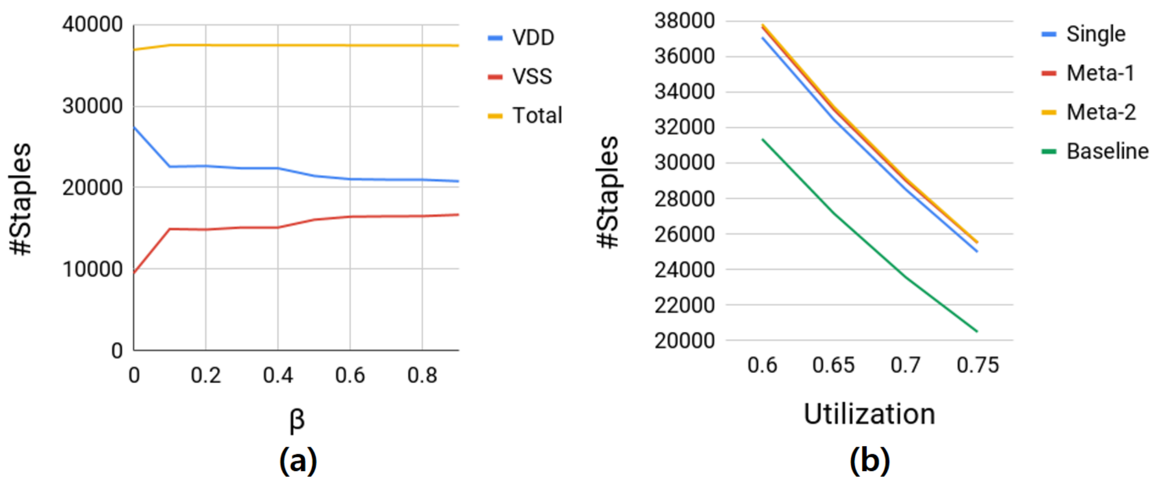


Figure 3.14. (a) Number of staples (total, VDD and VSS) that can be placed, versus balance factor (β) (AES with utilization 0.60) and (b) number of staples that can be placed, versus utilization by different strategies (AES with utilization between 0.60 and 0.75).

Main Results

We execute our Meta-2 optimization using all design blocks with the aforementioned parameter settings, and report #staples, average and worst IR drop before and after optimization, along with Δ worst negative slack (WNS) and runtime, at four utilizations per design block. Table 3.10 shows both the initial and post-optimization values of metrics (IR drop values are in units of mV). Figure 3.15 shows vectorless dynamic IR drop heatmaps for AES with 0.60 utilization.

We observe that our optimization can increase #staples by anywhere from 6.2% to 24.6%, and reduce IR drop by up to 13.2%, compared to *post-placement* staple insertion without our optimization. Furthermore, ΔWNS is similar before and after optimization, showing the effectiveness and robustness of our optimization. On the downside, IR drop can increase after our optimization (e.g., the case of M0 with 0.65 utilization); this is because the design's current map changes with placement perturbation and timing recovery steps, even as we add more power staples. Hence, directions for improvement include better control of the commercial P&R tool, and/or implementing our own legalization and ECO capabilities. We also note that

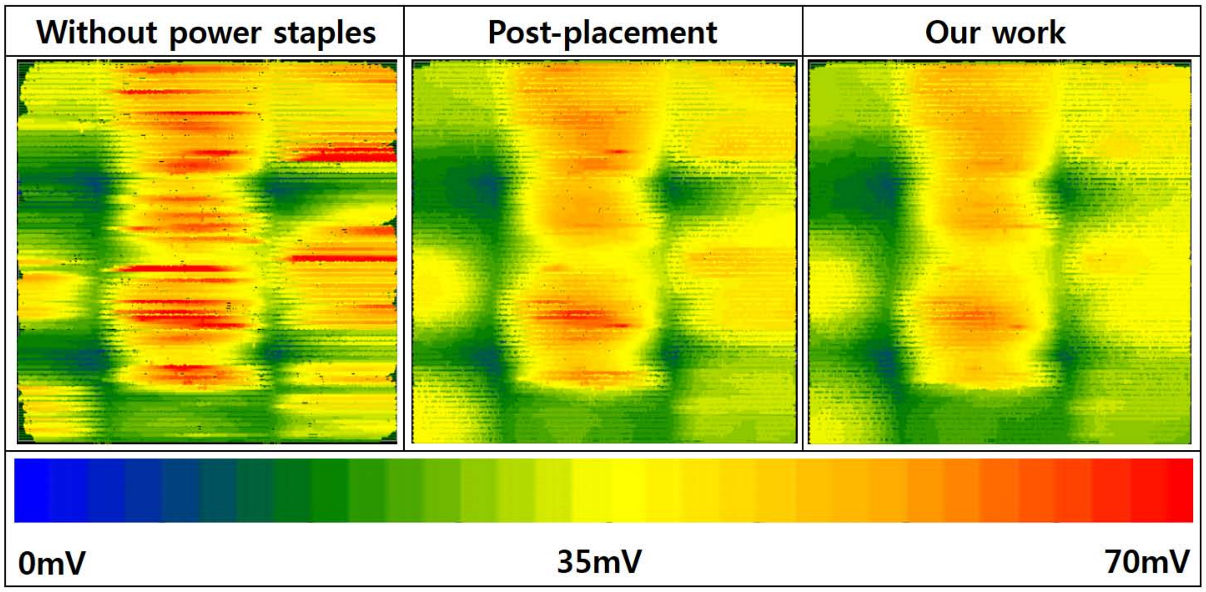


Figure 3.15. Heatmaps of vectorless dynamic IR drop for AES.

current runtimes are long, reflecting unoptimized implementation that pays heavily for *getNext()* function calls.

3.2.4 Conclusion

We have presented novel DP-based single- and double-row detailed placement optimizations with metaheuristics to maximize power staple insertion in a post-placement flow. We perform extensive studies on the scalability and sensitivity to parameters, impact of the benefit table, as well as balance of VDD/VSS staples. Compared to the traditional post-placement flow, we achieve up to 13.2% (10mV) reduction in IR drop, with almost no WNS degradation compared to a pre-placement flow. Directions for future research include (i) better exploration of the benefit table; (ii) understanding in greater detail the dynamics of the relationship between power staples and IR drop; (iii) further exploration of metaheuristics, (iv) runtime speedup techniques, and (v) exploration of power mesh structures with different width, pitch and offset.

3.3 Acknowledgments

Chapter 3 contains a reprint of Sunik Heo, Andrew B. Kahng, Minsoo Kim and Lutong Wang, “Diffusion Break-Aware Leakage Power Optimization and Detailed Placement in Sub-10nm VLSI”, *Proc. Asia and South Pacific Design Automation Conference*, 2019; and Sunik Heo, Andrew B. Kahng, Minsoo Kim and Lutong Wang and Chutong Yang, “Detailed Placement for IR Drop Mitigation by Power Staple Insertion in Sub-10nm VLSI”, *Proc. Design, Automation and Test in Europe*, 2019. The dissertation author is a main contributor to, and a primary author of, each of these papers.

I would like to thank my coauthors, Sunik Heo, Professor Andrew B. Kahng, Dr. Lutong Wang and Chutong Yang.

Chapter 4

Design-Technology Co-Optimization

This chapter presents three design-technology co-optimization methodologies. The first work presents a systematic framework for routability assessments, which can systematically evaluate a measure of intrinsic routability, K_{th}, across both technology and design choices. We focus on routability since it is a critical factor in the scaling of area and cost. The experimental studies demonstrate the assessment of routability impacts for advanced-node technology and design options. We demonstrate a machine learning (ML)-based K_{th} prediction flow to reduce runtime, disk space and commercial tool licenses needed to implement our framework. The second work presents a routability study that uses the framework with 3nm technology configurations. We demonstrate that block-level area scaling is reversed from cell-level scaling in cell height $<$150nm regime, using an ML-assisted DTCO methodology to optimize block-level area accounting for routability. The third work presents a systematic framework for design-technology pathfinding with improved design enablements. The goal of our work is to build a "configurable" and open-sourced process design kits (PDK) generation with promising scaling boosters, such as backside PDN and buried power rails, and study PPAC given technology and design parameters. In addition, we propose our ML-based parameter tuning flow and *clustering-based cell width-regularized placements* to generate more realistic artificial designs. We conduct experiments to present PPAC evaluation capabilities of the framework with our predictive technology and we achieve comparable results with studies from industry.

4.1 PROBE2.0: A Systematic Framework for Routability Assessment from Technology to Design in Advanced Nodes

Due to the slowdown of dimension scaling relative to the trend of the traditional Moore's Law, scaling boosters, such as backside power delivery networks (BSPDN) and buried power rails (BPR), are introduced at advanced technology nodes. Since scaling boosters play an important role to optimize power, performance, area and cost (PPAC) of IC products in advanced technologies, accurate and fast evaluations and predictions of PPAC are critical at an early stage of technology development. Also, use of scaling boosters makes evaluations and predictions of technology more difficult since they introduce a large number of knobs that simultaneously contribute to PPAC improvement.

Design-Technology Co-Optimization (DTCO) is now a well-known key element to develop advanced technology nodes and designs in modern VLSI chip design. Today's DTCO spans assessment and co-optimization across almost all components of semiconductor technology and design enablement. As described in Figure 4.1, the DTCO process consists of three stages: *Technology*, *Design Enablement*, and *Design*. First, the technology stage includes modeling and simulation methodologies for process and device technology. Second, the design enablement stage includes creation of required process design kits (PDK) for the ensuing design stage; these include device models, standard-cell libraries, routing technology files and interconnect parasitic (RC) models. Last, the design stage includes logic synthesis and place-and-route (P&R) based on the generated PDKs from the design enablement stage.

To evaluate and predict technology and design at advanced nodes, all three stages must be correctly performed, and PDKs must be generated from technology and design enablement stages. However, the DTCO process is not simple: feedback from design stage to technology stage takes weeks to months of turnaround time, along with immense engineering efforts. Also, based on the design feedback, additional PDKs may need to be generated at the design enablement

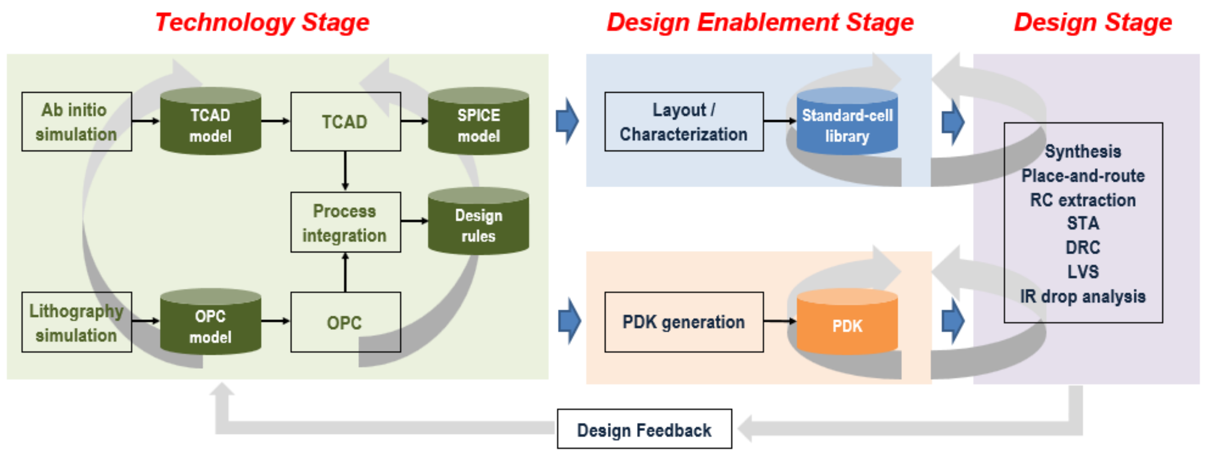


Figure 4.1. The DTCO process consists of three main stages: *Technology*, *Design Enablement*, and *Design*. Figure is redrawn from [174].

stage, which requires additional weeks to months. In order to reduce the turnaround time and maximize the benefit of the DTCO process, a fast and accurate DTCO methodology is needed to assess PPAC with reasonable turnaround time, and to more precisely guide multi-million dollar decisions at an early stage of technology development.

Density is the overarching metric for enablement of system-level benefits through scaling [123], and directly determines the Area and Cost aspects of PPAC. In our work, we focus on *routability* implications of technology choices, since routability and density are intimately tied together. The challenge of routability arises as back-end-of-line (BEOL, i.e., metal layers) technology fails to scale down in step with front-end-of-line (FEOL, i.e., device layers). Also, cell heights of standard-cell architectures are a crucial lever for density scaling [109], but small standard-cell heights challenge area routing and pin accessibility. Furthermore, high BEOL resistances require denser power delivery networks (PDN), which occupy more routing resources and harm routability.

Contributions of This Work. The goal of the work described in this section is to enable faster and more comprehensive evaluation of technology options, early in the technology development process. In particular, we describe a framework for *systematic assessment of routability* across the combined space of technology options and design enablement options.

Many measures of routability have been developed and applied over the past decades. These span the use of congestion maps, metrics of pin accessibility, machine learning-based congestion predictors, and other techniques, as we review in Section 4.1.1 below. However, these previous methods to assess routability do not solve two root causes of the long feedback loop in DTCO (Figure 4.1). The first root cause is that efficient simultaneous exploration of technology and design options in DTCO is blocked by the effort and expense of the design enablement stage. Producing layouts and characterizations for standard-cell libraries requires an enormous amount of engineering cost and time, due to complex constraints such as transistor-level placement, in-cell routing, and pin accessibility. Today's DTCO relies on limited, heuristic layout synthesis (e.g., manual layout of 15-60 key cells) to assess a given set of technology options. The second root cause is that routability assessment methods have mostly focused on assessing *implementations* of designs (e.g., to predict routability of the placement of a particular netlist), rather than assessing *design enablements*.

Our present work attacks both of the above-mentioned root causes of long design feedback loops in DTCO. To do this, we build on two threads of recent works: (1) automatic standard-cell layout generation using Satisfiability Modulo Theories solvers [26] [82], and (2) assessment of intrinsic routability of BEOL stack options via the K_{th} metric [64]. We review these works in Section 4.1.1. As demonstrated below, our framework is able to provide assessments of intrinsic routability across a range of technology and design parameters reflecting sub-7nm technologies. Our main contributions are summarized as follows.

- We describe a systematic and complete framework to evaluate routability across key parameters of technology and design. Our framework is generalizable and flexible; it enables rapid evaluation of hundreds of technology and design enablement options within hours or days, providing a valuable tool for early technology development.
- We propose a pin shape selection strategy based on the *remaining pin access (RPA)* [108], along with a top-metal-only pin shape selection strategy, at our design enablement stage.

We also extend methods of [26] [82] to automatically produce more realistic standard-cell libraries (LEF format [158]) in terms of power and ground pins, contacted poly pitches (CPP) and metal pitches.

- We extend the method of [64] to assess routability across configurations of technology and design, rather than only BEOL stack options. We study both *cell-level* and *design-level* routability, and show advantages of using *knight's tour-based* artificial netlist topology generation in cell-level routability assessment. (A knight's tour is a sequence of knight's moves in a chessboard that visits each square exactly once. Section 4.1.3 below explains its use in our methodology.)
- We achieve seamless integrations with commercial P&R tooling, via automated generation of power-ground hookups in cell layouts, and routing technology files to reflect modified design rules.
- We demonstrate accurate learning-based K_{th} prediction that reduces runtime, disk storage and tool license overheads of our framework.

Our work is organized as follows. Section 4.1.1 reviews related previous works. Section 4.1.2 gives an overview of our framework, along with the parameterizations and other details of how we generate standard-cell layouts and design enablements. Section 4.1.3 describes our methodologies for routability assessments and learning-based K_{th} prediction. Section 4.1.4 shows our experimental setup, key experiments and results. We give conclusions and directions of future work in Section 4.1.5.

4.1.1 Related Work

We now review relevant previous works. Broadly, these can be categorized into works on (1) standard-cell layout generation and (2) routability assessment. Our review includes the works of [26] [82] and of [64] which have provided a basis for our present work.

Standard-Cell Layout Generation

Automatic Standard-Cell Layout Generation. Standard-cell layout synthesis can help library design teams explore cell architectures with holistic consideration of transistor placement, in-cell routing, complicated design rules, and pin accessibility. The methods of [45] [136] provide co-optimization of transistor placement and in-cell routing, but do not consider such aspects as multi-patterning design rules that are seen in advanced technology nodes. [59] [83] [118] propose standard-cell layout automation frameworks for sub-7nm technologies, but these works incorporate multiple heuristic approaches with no guarantees of optimality. Park *et al.* [82] unify transistor-level placement and routing with dynamic pin allocation, and apply a satisfiability modulo theories (SMT) solver to achieve optimal layout solutions.

Pin Accessibility-Aware Standard-Cell Layout. One of the most difficult design features for standard-cell layout generation is the pin accessibility, which is challenged by the limited number of tracks and complicated design rules. The works of [105] [127] define metrics for pin accessibility within their objectives for standard-cell layout optimization. Seo *et al.* [108] propose the *remaining pin access* (RPA) metric to capture pin access interference from access points of neighboring pins. Cheng *et al.* [26] devise "at-least-k" Boolean constraints that guarantee a minimum number of pin openings (access points) per pin in the cell layout.

The SMT-based Standard-Cell Layout Generation of [26] [82]. Our present work builds on the SMT-based parametric standard-cell generation framework of [26] [82]. This framework takes in three main inputs. (1) **Cell architectures**: number of routing tracks and transistor fins, and track pitches. (2) **Netlist**: component connectivity of library cell. (3) **Design rules**: parametric conditional design rules depending on cell architecture. Given these inputs, a cell layout is produced that is optimal with respect to cell area, M2 track use (routability) and routed metal length, in lexicographic order of these criteria. A unifying *dynamic pin allocation* (DPA) constraint integrates additional design constraints such as transistor placement, in-cell routing, conditional design rules and pin accessibility constraints. This yields a constraint satisfaction formulation that produces an optimized cell layout via a single multi-objective optimization.

We observe that these previous works do not provide necessary enablements of commercial standard-cell P&R, such as LEF [158] generation, PDN generation, and routing technology file generation. Nor do these works support control parameters for standard-cell layout generation that are relevant in sub-7nm nodes. Below we describe a complete framework to support both standard-cell layout generation and associated P&R enablement. Our layout generation uses RPA-based and top-metal-only pin shape selection methods that improve pin accessibility.

Routability Assessment

Routability is a hard constraint in the modern (fixed-die) place-and-route context. Thus, many previous works have studied routability-driven placement, as well as ripup-and-reroute methods in global routing. For example, [25] [54] [75] [86] all propose routability-driven placement based on congestion maps derived from early trial or global routing. Pin accessibility of a given standard-cell instance also affects routability. The above-mentioned work of [108] describes pin accessibility-aware detailed placement based on the RPA metric. However, in this discussion we do not focus on placement and routing optimizations, but rather on methods for *assessment* of routability.

Routability Analysis and Prediction. Tseng *et al.* [117] propose a systematic framework with P&R tools to check routability, aiming to improve placement outcomes. The authors propose standard cell-level and placement-level routability scores to generate cell spacing constraints. [49] proposes an optimal ILP-based detailed router and evaluate feasibility (routability) of routing clips based on an ILP solver. The authors of [73] [99] [100] use Boolean satisfiability (SAT) to analyze routability under conditional design rules. [99] [100] furthermore extract minimal unsatisfiable subsets to diagnose bottlenecks when designs are proven to be unroutable.

Several recent works propose machine learning (ML)-based routability predictions. Zhuo *et al.* [135] propose a new routability prediction model based on supervised learning in placement. The works of [21] [124] predict routability based on convolutional neural networks (CNN) and support vector machines (SVM), respectively. Chan *et al.* [13] also propose SVM-based routability prediction, but aim to evaluate routability for various BEOL stack options.

The PROBE Routability Measurement Utility of [64]. Our present work builds on the "PROBE" framework of [64], which gives a measure of inherent routability of BEOL stack options. PROBE begins with a placement solution that is easy to route – e.g., a regular mesh placement of a mesh netlist topology. PROBE then iteratively swaps the locations of random pairs of neighboring placed cells, progressively "tangling" the placed netlist until the routing fails with more than some threshold number of post-route DRC violations. The number of random neighbor cell swaps performed, normalized to the number of instances in a design, is denoted by K. The number of swaps beyond which routing fails is denoted as the *K threshold* (K_{th}), and captures *intrinsic* routing capacity (e.g., of a given BEOL stack).

Figure 4.2(a) shows the scope of PROBE. Given a placement, a set of BEOL stack options $\{B_1, B_2, ..., B_i\}$ can be ranked in terms of routability. The framework supports two types of placements, shown in Figure 4.3. *Mesh-like* placements do not reflect any specific design; they consist of an array of instances of a given 2-input or 3-input cell. Connections are made between neighbors, inducing a near-meshlike netlist topology. *Cell width-regularized* placements are design-specific, and are produced by commercial P&R tools. However, the standard cells in the placements are all given the same cell width to avoid illegal placements after neighbor cell swaps.

We observe that [64] is applied only to BEOL stack options, and does not cover the rich space of FEOL technology and design enablement options. Moreover, the near-meshlike topology can produce only a limited range of routed wirelength and Rent parameter values that may not match realistic values. The framework we describe below supports DTCO with routability assessment across technology and design enablement options. We use a knight's tour-based construction that can better reflect actual design attributes.

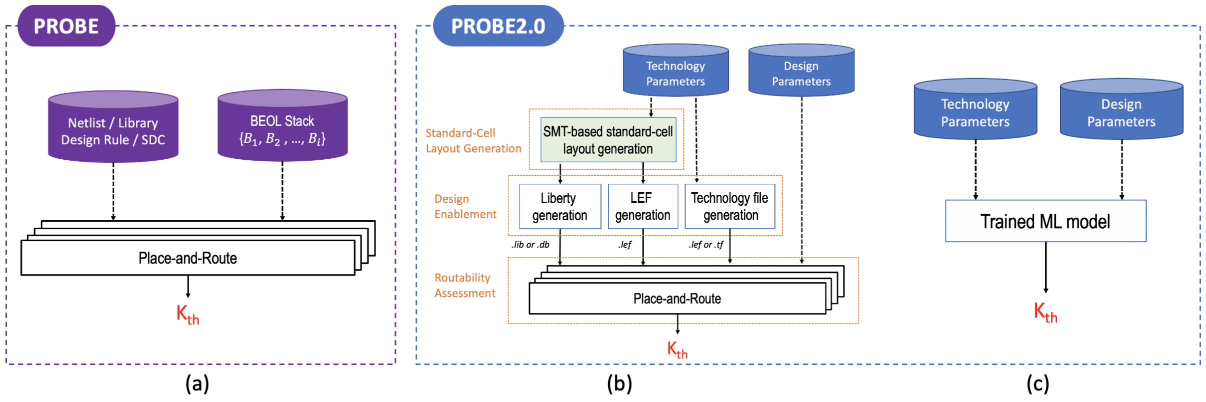


Figure 4.2. Overall flows for PROBE [64] and PROBE2.0. (a) PROBE only evaluates BEOL stack options. (b) The PROBE2.0 flow includes standard-cell layout generation [26] [82], design enablement and routability assessment. (c) The PROBE2.0 flow uses a trained learning-based model to predict K_{th}.

4.1.2 The PROBE2.0 Framework

Figure 4.2(b) shows our PROBE2.0 framework. It takes technology and design parameters as primary inputs, and consists of three major stages. (1) The **standard-cell layout generation stage** is based on input technology parameters, and is performed using an extension of an SMT-based standard-cell layout generation [26] [82]. It produces purely grid-based pin locations and cell boundaries. (2) The **design enablement stage** begins with the generated standard-cell layouts, and is also performed according to the input technology parameters. Design enablement generates LEF [158], Liberty [159] and routing technology files. LEF file generation converts the primitive form of layouts to LEF format. The conversion considers real-world constraints for the stability of standard-cell characteristics. (3) The **routability assessment stage** uses a *knight's tour-based* topology as well as open-source designs with the PROBE approach [64]. Also, Figure 4.2(c) shows how the PROBE2.0 flow can be realized with learning-based K_{th} prediction, where a trained machine learning model enables more efficient routability assessment.

The remainder of this section describes aspects of standard-cell architecture, technology and design parameters, and design enablement in the PROBE2.0 framework. Our routability assessment and learning-based K_{th} prediction flow are explained in Section 4.1.3.

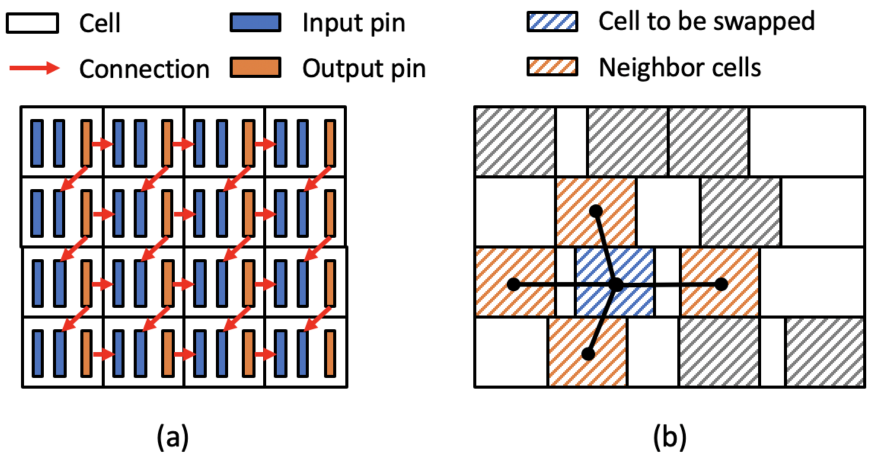


Figure 4.3. Placements for PROBE [64]. (a) A mesh-like placement based on a 2-input cell. (b) A cell width-regularized placement. The orange-striped cells are considered to be neighbors of the blue-striped cell. The blue-striped cell is swapped with a randomly-selected neighbor.

Standard-Cell Architecture

Figure 4.4 shows a grid-based standard-cell architecture and technology parameters of standard cells, as used in our work. We follow the 7nm standard-cell architectures in [32] [119] to generate the grid-based P&R graph with four layers TS/PC, M0, M1, and M2. TS/PC and M0 layers are included in FEOL layers and M1 and M2 layers are included in BEOL layers. Next, we give detailed definitions of the eight technology parameters and five design parameters that PROBE2.0 supports as user inputs.

Definitions of Technology Parameters

Technology parameters include various options for process technology as well as design enablement.

(1) *Fin*: The number of fins for devices of standard cells. We use 2 and 3 for Fin [32] [119].

(2) *CPP*: Contacted poly pitch for standard cells. We use 48 and 54nm for CPP [109] [119].

(3) *MP*: Metal pitch for M2 and M3 routing layers. We use 24, 32 and 40nm for MP [109] [119].

(4) *RT*: The number of available M2 routing tracks for standard cells. We use 4, 5 and 6 for RT [32] [119].

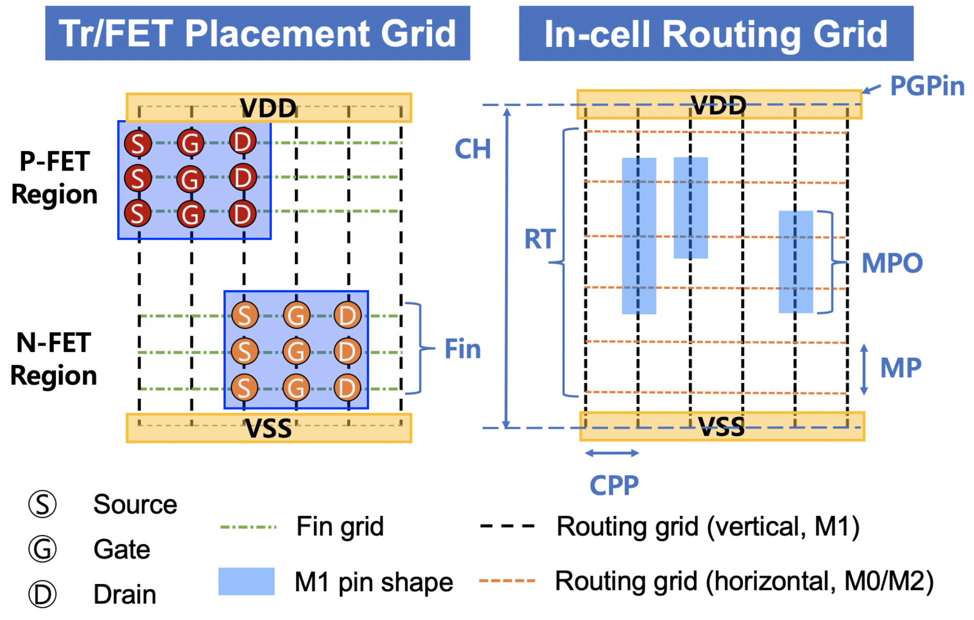


Figure 4.4. A grid-based standard-cell architecture and technology parameters for standard cells.

(5) *PGpin*: Pin types for power and ground of standard cells. We support three types of power and ground pins: *M1* [119], *M1+M2* [17], and *BPR* [107]. M1 denotes power and ground pins on the M1 layer. M1+M2 denotes power and ground pins on both M1 and M2 layers. BPR has no power and ground pins on BEOL layers. M1 and M2 power and ground pins have width equal to twice the minimum width on Mx (M1, M2 and M3) layers. (Thus, since minimum width and minimum spacing are each equal to half the metal pitch MP, the power and ground pins have widths equal to the Mx pitch.) Also, in enablement of BPR, the width of M1 power pins is the same as the minimum width of Mx. This is because commercial P&R tools require power and ground pins to be connected to PDN. Note that the minimum-width M1 power and ground pins do not affect routability since the minimum routing layer is M2.

(6) *CH*: Cell height of standard cells, expressed as a number of M2 routing track (T) pitches. For example, if M2 routing track pitch is 40nm and the cell height is 240nm, then CH is 6. We use and refer to standard cells with 5T, 6T, 7T and 8T [32] [119] cell heights. Note that CH depends on the combination of RT and PGpin. For example, if RT is 4 and PGpin is M1, then CH is 6. If RT is 4 and PGpin is BPR, then CH is 5.

(7) *MPO*: The number of minimum pin openings (access points) per pin. For example, if MPO is 2, every pin must have at least two access points for the generated standard-cell layout. We use the values of 2 and 3 for MPO [26] [82].
(8) *DR*: Design rule sets. In this work, all design rules are defined based on routing grids. We assume that our technologies are based on Extreme Ultraviolet (EUV) lithography. We define three fundamental grid-based design rules for Mx layers, *DR-MAR*, *DR-EOL* and *DR-VR* [26] [82]; we also define two design rule *sets*, namely, *EUV-Loose* (EL) and *EUV-Tight* (ET).

Figure 4.5 illustrates the three design rules. (i) DR-MAR denotes *minimum area* rules, as shown in Figure 4.5(a). When a metal shape occupies only one grid point and DR-MAR is 1, this violates the DR-MAR rule. I.e., a metal shape must be long enough to occupy at least two routing grid points. (ii) DR-EOL denotes *end-of-line* rules, as shown in Figures 4.5(b) and (c). When edges of two co-linear metal shapes are placed next to each other and DR-EOL is 1, this violates the DR-EOL rule. I.e., there must be at least one unoccupied routing grid point between edges of metal shapes. When DR-EOL is 2, there must be at least two empty routing grid points. (iii) DR-VR denotes *via restriction* rules, as shown in Figure 4.5(d). The figure shows the prohibited locations for other vias, relative to the placement of a given via. When DR-VR is 1, only four neighbors are blocked by the DR-VR rule.

Our two *design rule sets* each comprise combinations of specific design rule settings, as follows. EUV-Loose (EL) consists of DR-MAR = 1, DR-EOL = 1, and DR-VR = 1. EUV-Tight (ET) consists of DR-MAR = 1, DR-EOL = 2, and DR-VR = 1).

Last, we note that in this work, we assume metal enclosures of vias are 10nm in a preferred direction and 0nm in a non-preferred direction. Also, many practical design rules can be framed using our grid-based design rules. For example, rules for end-to-end spacing and minimum enclosures can be captured with the DR-EOL rule. We note that a wide range of via rules, including center, edge and corner spacing rules, can be captured with the DR-VR rule.

Based on the cell architecture of Figure 4.4 and the above technology parameters, our

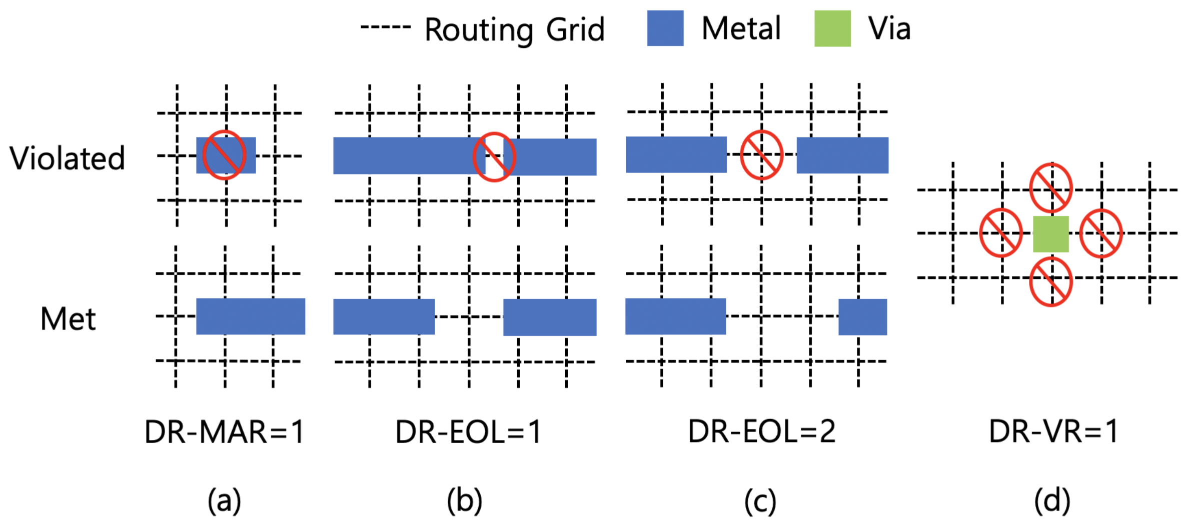


Figure 4.5. DR-MAR, DR-EOL and DR-VR design rules.

Table 4.1. Standard-cell architectures in our experiments.

Fin	RT	PGpin	CH
2	4	M1/M1+M2	5
		BPR	6
3	5	M1/M1+M2	6
		BPR	7
3	6	M1/M1+M2	7
		BPR	8

standard-cell layout synthesis framework can generate layouts for various cell architectures. In particular, our studies use six types of cell architectures with the combinations of Fin, RT and PGpin and CH shown in Table 4.1.

Definitions of Design Parameters

Our framework uses the following **design parameters**.

(1) *BEOL*: Metal stack options. We define *9M*, *10M*, *11M* and *13M* BEOL stack options based on scaling down from a commercial 14nm technology. Recall that Mx (1X layer) pitch, i.e., MP above, is a technology parameter that we can vary in routability exploration. To scale down a 14nm BEOL technology to sub-7nm technologies, we define 2X, 3.2X, 9X and 18X layer pitches

based on 40nm as the 1X pitch; this reflects advanced-node stacks as well as considerations such as litho/cost “cliff” ($\sim$80nm pitch limit for single-exposure 193i patterning). We calculate the *routing resource* of [64] for each BEOL stack option. The *routing resource* is defined as $\sum_b (1/pitch_b)$ where b denotes a metal layer and $pitch_b$ is the pitch of b. Essentially, this sums available routing tracks over all routing layers. Table 4.2 summarizes layer counts per pitch and *routing resource* (R) for the BEOL stack options that we study.

(2) *PDN*: Power delivery network options. These include traditional PDN options with different layers and pitch, and backside PDN as a scaling booster for advanced technology. We define four PDN options: *Backside*, *Sparse*, *Middle* and *Dense*. Table 4.3 shows the detailed information of these PDN options.

(3) *Tool*: Commercial tools to perform P&R [144] [172], referred to only as *Tool A* and *Tool B* to comply with vendor license agreements.

(4) *Util*: Placement utilization (0.6, 0.7, 0.8).

(5) *Design*: Designs studied in routability assessment. We use knight’s tour-induced artificial topologies, along with four open-source designs (AES, LDPC, JPEG, VGA) from OpenCores [161]. The respective instance counts of AES, LDPC, JPEG and VGA are approximately 13K, 56K, 69K and 72K.

Table 4.4 summarizes the technology and design parameters that we use in our experiments. Note that in our framework, the parameter list is flexible and readily extendable. This enables accommodation of new technology requirements or new scaling booster options. For example, sets of smaller CPP and MP values, including with non-unit “gear ratio” values such as 2:1 or 3:2 (e.g., CPP relative to vertical M1 pitch), can be evaluated with the PROBE2.0 framework. This serves a real DTCO and technology pathfinding problem in industry for sub-3nm technologies, through the end of the lateral scaling roadmap. Also, we can easily add new designs and/or PDN strategies as design parameters, for richer assessments.

Table 4.2. Four BEOL stack options. *R* denotes *routing resource*.

BEOL	**1X**	**2X**	**3.2X**	**9X**	**18X**	*R*
9M	3	4	NA	NA	2	104.4
10M	3	4	2	NA	1	117.8
11M	3	4	2	NA	2	120.1
13M	3	2	4	2	2	116.3

Table 4.3. Details of PDN options. All numbers are pitches in units of μm for each layer. "*P*" indicates that we use a given layer only for PDN at maximum area density, and do not allow the layer to be used for signal routing. The width of M5 stripes is $0.96\mu m$ and the width of M6/M8 stripes is $1.296\mu m$.

PDN	**9M**		**10M**		**11M**		**13M**	
Backside	NA		NA		NA		NA	
Sparse	M5	10	M5	10	M5	10	M5	10
	M6	10	M6	10	M6	10	M8	10
	M8	P	M9	P	M10	P	M12	P
	M9	P	M10	P	M11	P	M13	P
Middle	M5	20	M5	20	M5	20	M5	20
	M6	20	M6	20	M6	20	M8	20
	M8	P	M9	P	M10	P	M12	P
	M9	P	M10	P	M11	P	M13	P
Dense	M5	40	M5	40	M5	40	M5	40
	M6	20	M6	40	M6	40	M8	40
	M8	P	M9	P	M10	P	M12	P
	M9	P	M10	P	M11	P	M13	P

Design Enablement

Our design enablement produces ready-to-use standard libraries and required inputs for P&R. We generate LEF format from the primitive layout produced by SMT-based cell layout generation. Layouts are fully grid-based, with CPP and MP technology parameters defining the grid pitches. Importantly, we propose two pin shape selection schemes: RPA-based pin shape selection and top-metal only pin shape selection.

RPA-Based Pin Shape Selection. The SMT-generated cell layouts can have multiple distinct

Table 4.4. Technology and design parameters in our experiments.

Type	Parameter	Option
Technology	Fin	2, 3
	CPP	54, 48
	MP	24, 32, 40
	RT	4, 5, 6
	PGpin	BPR, M1, M1+M2
	CH	5, 6, 7, 8
	MPO	2, 3
	DR	EUV-Loose, EUV-Tight
Design	BEOL	9M, 10M, 11M, 13M
	PDN	Backside, Sparse, Middle, Dense
	Tool	Tool A, Tool B
	Util	0.6, 0.7, 0.8
	Design	A knight's tour, AES, LDPC, JPEG, VGA

pin shapes for a single pin. Access to such pins must be carefully handled to avoid instability of timing and power models of the standard cells. Figure 4.6(a) shows the initial SMT-generated layout of an OAI21_X1 cell. In the figure, pin *ZN* has two M1 shapes, ZN_a and ZN_b. When connections are made to different pin shapes (i.e., ZN_a or ZN_b), at least one of the cases will not match the cell's characterized timing and power model. For example, when output pin *ZN* is connected through the M1 pin shape ZN_a, the cell delay is 10ps, but when the connection is made through ZN_b, the cell delay is 8ps. This instability with respect to the cell timing/power model is unacceptable in modern design enablements. Therefore, when a pin has multiple candidate shapes, our framework chooses one of these shapes to use in the standard-cell layout that is produced.

Given our focus on routability, we apply pin shape selection based on the *Remaining Pin Access* (RPA) pin accessibility metric [108]. Figure 4.6 shows an example of our pin shape selection. Accessibility of a given pin is affected by other pins within a distance d_{int}. We set d_{int} as 1.0 for the EL design rule set, and 2.0 for the ET design rule set. We then calculate *RPA* values for each pin shape. From [108], pin access points are defined as $a_{p,m}$ where p is the corresponding pin (i.e., single pin shape) and m is the position (index) in terms of M2 routing tracks. $\mathscr{A}(p)$ denotes the set of access points for a given pin p. Then, $\mathscr{N}(a_{p,m})$ denotes the set of *neighboring* pin access points that (1) do not belong to p and (2) are within distance d_{int} of p on metal layer m. For example, $\mathscr{A}(A2) = \{a_{A2,1}, a_{A2,2}\}$ and $\mathscr{N}(a_{A2,2}) = \{a_{A1,2}, a_{ZN_b,2}\}$ in Figure 4.6(a). The used pin access (UPA) of pin p is defined as

$$UPA(p) = \sum_{a_i \in \mathscr{A}(p)} \sum_{a_j \in \mathscr{N}(a_i)} 1/|\mathscr{A}(p(a_j))| \tag{4.1}$$

Finally, RPA of pin p is defined as

$$RPA(p) = |\mathscr{A}(p)| - UPA(p) \tag{4.2}$$

For example, $RPA(A2) = 2.0 - (0.5 + 0.5) = 1.0$ when we apply Equations (4.1) and (4.2) to the example of Figure 4.6(a). In the same figure, the RPA values are 2.0 and 1.0 for ZN_a and ZN_b, respectively. Since the RPA value of ZN_a is larger than that of ZN_b, we choose the ZN_a pin shape for ZN and make ZN_b into an obstacle on M1, as shown in Figure 4.6(b). We empirically confirm the benefits of this RPA-based pin shape selection for routability, as measured by the K_{th} metric, in Section 4.1.4.

Top-Metal-Only Pin Shape Selection. When standard cells have multiple pin shapes on multiple metal layers, we incorporate a *top-metal-only* pin shape selection step. In the example of Figure 4.7, the initial layout of the OAI21_X1 cell has three pin shapes for the *ZN* pin; two on M1 (and V1) and one on M2. In this multiple-layer situation, we do not calculate RPA values for the pin shapes. Instead, we choose the M2 pin shape for *ZN*, and the two M1 pin shapes (and V1) become obstacles. This methodology brings several benefits. (1) Replacing M1 (and V1) pin shapes with obstacles benefits overall pin accessibility by reducing the complexity of pin access: P&R tools can solve the pin access problem more easily and with fewer DRCs. (2) As with the above-described pin shape selection, we avoid instability of timing and power with respect

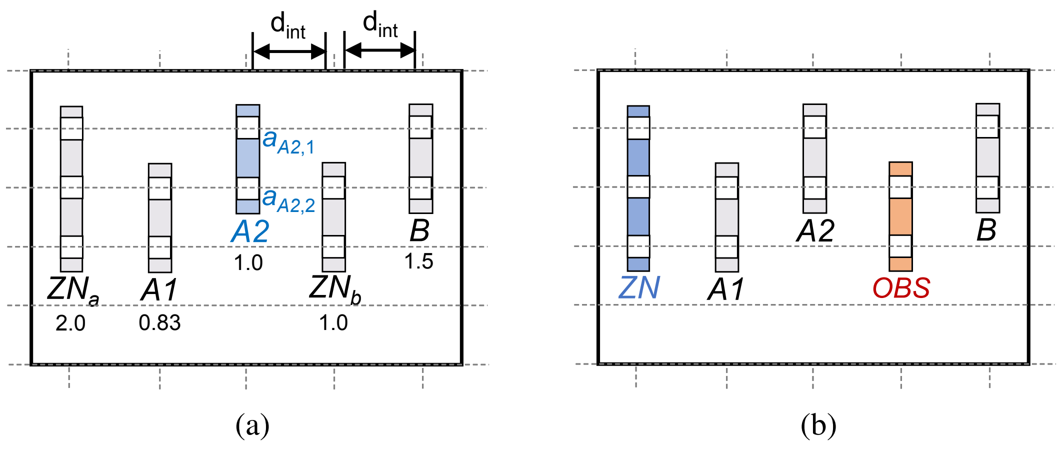


Figure 4.6. Example of pin shape selection in a standard cell (2Fin, 5RT, 2MPO and EUV-Loose). (a) Initial layout of OAI21_X1 from SMT-based layout generation. (b) The OAI21_X1 layout after RPA-based pin shape selection.

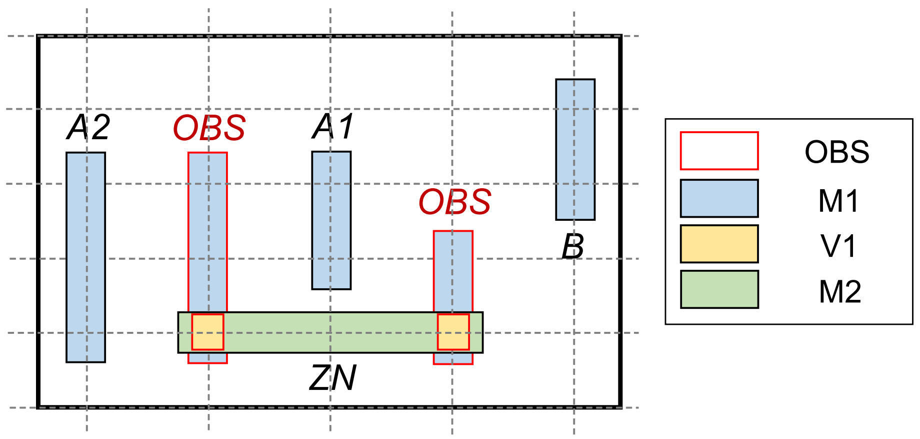


Figure 4.7. Example of standard-cell layout with top-metal-only pin shape selection. We show an initial layout of AOI21_X1 (2Fin, 5RT, 2MPO and EUV-Loose). In this case, the *ZN* pin has only the M2 pin shape, and the M1 and V1 shapes become OBS.

to characterized models. (3) Our top-metal-only pin shape selection mitigates susceptibility to electromigration (EM) by removing M1 pin shapes (e.g., Posser *et al.* [101] show a cell-internal signal EM problem in accordance with output pin position). (4) Last, changing M1 (and V1) pin shapes into obstacles improves accessibility of other neighboring M1 pins.

Power and Ground Pin Generation. To make SMT-produced layouts usable by P&R tools, power and ground pins must be added. Figure 4.8 shows how we define power and ground pins for standard cells. For BPR, the width of power and ground pins is equal to the minimum M1

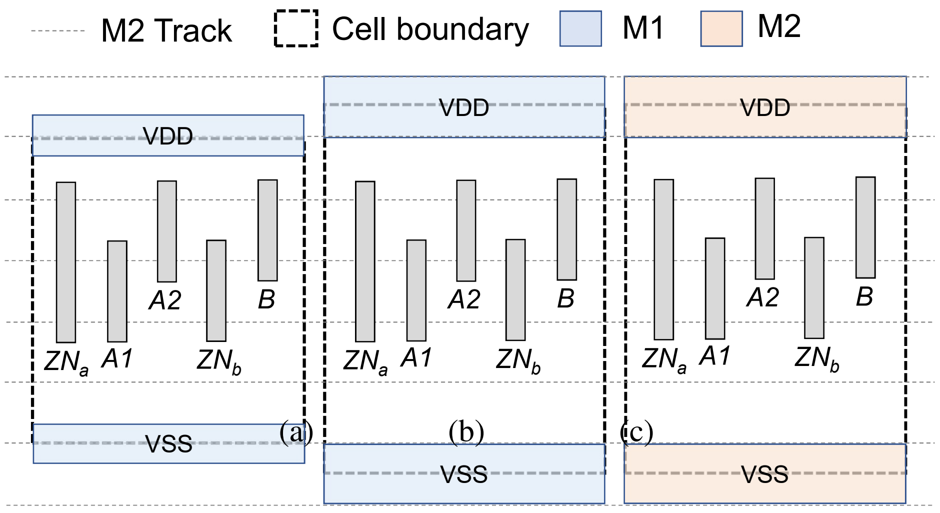


Figure 4.8. Power and ground pin (PGpin) examples: (a) BPR, (b) M1 and (c) M1+M2. The signal pin shapes (gray) are the same regardless of PGpin.

width. For M1 and M1+M2, it is double this width. M2 routing grids of standard cells with BPR have an offset by half of the M2 track pitch. The figure also shows how the height of standard cells is determined by the RT and PGpin parameters.

Liberty and Technology File Generation. The remaining parts of our design enablement generate Liberty and technology files. We use dummy Liberty files to avoid any potential errors in the commercial P&R tools. Our technology file generation takes as inputs the technology parameters CPP, MP and DR, and converts design rules into corresponding file formats (e.g., LEF and routing technology files) to enable use of commercial P&R tools.

4.1.3 Routability Assessment and Learning-Based K_{th} Prediction

We now describe how we perform both *cell-level* and *design-level* routability assessment using the PROBE2.0 framework. *Cell-level* routability assessment aims to evaluate the intrinsic routability of an individual standard cell's layout. *Design-level* routability assessment uses a real design testcase to evaluate the intrinsic routability of a design enablement and/or methodology.

Recall from Section 4.1.1 that the PROBE approach [64] starts from an initial placement of an initial netlist, then progressively increases K, the amount of "tangling", until the router exceeds a given DRC limit at the threshold K_{th}. The K_{th} value is specific to a given library along with technology and design parameters. To find K_{th}, multiple P&R runs are launched with various values of K, as illustrated in Figure 4.2(b).

Cell-Level Routability Assessment Based on Knight's Tour

In advanced technologies, pin accessibility of individual standard-cell masters has become a critical challenge with decreasing cell heights. We therefore propose *cell-level routability assessment* to evaluate routability at the individual cell level. The previous work of [64] uses baseline *mesh-like placements* to rank BEOL stacks by routability. These induce nearly square-mesh netlist topologies constructed using a single cell master from a fixed library. By contrast, our goal is to assess routability for a given choice of technology and design parameters, leveraging our standard-cell generation capability. The mesh-like placements used in [64] have key drawbacks: (i) they do not closely reflect the wirelength and Rent parameter attributes of real design testcases, and (ii) their construction encompasses only 2- and 3-input cells. In this work, we overcome these drawbacks by using a *knight's tour* to induce the initial placed netlist that is "tangled" to find K_{th}.

A *knight's tour* is a sequence of *knight's moves* that visits each square on a chessboard exactly once. A knight moves from its current square to a square that is two by one (or one by two) squares away, in any direction. In our approach, a square in an M by N chessboard corresponds to a placed cell in an initial placement of M by N instances of a given master. A knight's tour ordering induces connections – i.e., a netlist – within this placement. More precisely, for a given layout region with M rows and N columns of placed instances, and a k-input cell master, we create a knight's tour-based topology as follows.

(1) In the layout region with M rows by N columns of instances, we generate a knight's tour ordering. For this, we use the well-known Warnsdorff-rule algorithm [139].

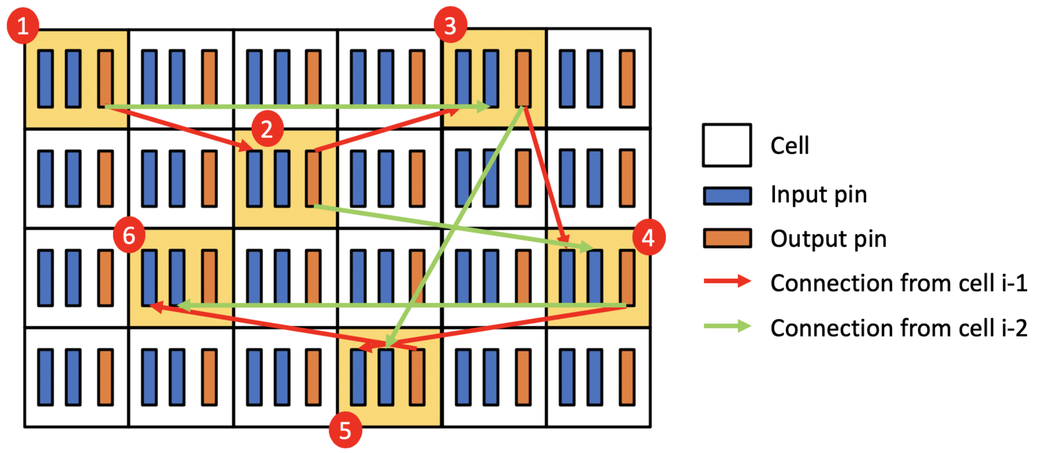


Figure 4.9. Example of knight's tour-based connections in a placement of 2-input cells. Numbers in red circles indicate the knight's tour ordering.

(2) An instance of the k-input cell master that is at position i in the knight's tour is assigned fanins from cells at positions $i-1$ to $i-k$ in the knight's tour ordering. For a given 2-input cell i, there will be a fanin connection from the output of cell $i-1$ to the first input of cell i, and a fanin connection from the output of cell $i-2$ to the second input of cell i.
(3) We end up with an initial placement ("chessboard") having M rows and N columns of instances, and an artificial netlist having connections induced by the knight's tour as described above.

Figure 4.9 shows an example of a knight's tour topology with 2-input cells. The red arrows indicate the connections between cell i and cell $i-1$. The green arrows indicate the connections between cell i and cell $i-2$.

Our studies confirm improved flexibility and correspondences to real design netlists when we apply the knight's tour-based approach. Table 4.5 shows placed wirelengths of the AES design, and of mesh-like placements [64] and knight's tour-induced netlists. INV_X1, NAND2_X1, NAND3_X1 and NAND4_X1 cell masters are used, with default technology and design parameters as specified in Section 4.1.4. Particularly when based on the NAND3_X1 master, the knight's tour topologies can give more realistic wirelengths than mesh-like placements.

Table 4.5. Placed design information for AES, mesh-like placements and knight's tour based topologies. Mesh-like placements and knight's tours have 115×115 placed instances. Area denotes total instance area in the designs. Area is in units of μm^2 and wirelength (WL) is in units of μm.

Design	Cell	Area	#Insts	WL
AES	-	1088	13221	46880
Mesh	INV_X1	Not supported		
	NAND2_X1	799	13225	11001
	NAND3_X1	999	13225	13932
	NAND4_X1	Not supported		
Knight's Tour	INV_X1	599	13225	10264
	NAND2_X1	799	13225	23610
	NAND3_X1	999	13225	41103
	NAND4_X1	1199	13225	62251

(As noted above, the mesh-like placement construction does not support use of 1- or 4-input cell masters.)

We also note that mesh-like placements have a simple topology wherein connections are only between neighbor cells in the initial placement. This inflexibility can weaken correspondences to real-world designs. Notably, the Rent parameter p of the (2-dimensional) mesh topology is 0.5 [47] [81] according to Rent's rule. We have studied achievable Rent parameters for mesh-like placements and knight's tour-based topologies by using RentCon [165]. Table 4.6 shows that Rent parameters of real circuits (AES, LDPC, JPEG, VGA) range from 0.617 to 0.891. On the other hand, the mesh-like placements p values from 0.500 to 0.585, while our proposed knight's tour-based placements have p values from 0.676 to 0.820. Again, the knight's tour construction can more closely reflect real circuits. Based on these studies, we use the knight's tour-based topology construction in our cell-level routability assessment.

Design-Level Routability Assessment

To perform design-level routability assessments, we begin with open-source RTL designs from OpenCores [161] and use commercial logic synthesis to obtain gate-level netlists.

Table 4.6. Rent parameters for mesh-like and knight's tour-based placed netlists. We use *rectangle sampling-based methods I and II* for the evaluation method, *Types I and II* for the pin counting method, and *geometric mean* for the averaging method in [165].

Source	Design	Rent parameter
OpenCores [161]	AES	[0.726, 0.832]
	JPEG	[0.617, 0.813]
	VGA	[0.656, 0.879]
	LDPC	[0.812, 0.891]
PROBE [64]	**Mesh (NAND2_X1)**	**[0.530, 0.535]**
	Mesh (NAND3_X1)	**[0.500, 0.585]**
PROBE2.0	**Knight's tour (NAND2_X1)**	**[0.676, 0.761]**
	Knight's tour (NAND3_X1)	**[0.708, 0.820]**

Following [64], we evaluate design-level routability based on *cell width-regularized* placements (Figure 4.3(b)) of these netlists. For each standard-cell library, we modify LEF such that all combinational cells are bloated to have the same width as the maximum-width cell among all 38 combinational cells in the library. (Our designs also instantiate D flip-flops, which have the largest cell width in our generated standard-cell libraries. We fix the locations of flip-flops in designs after placement and do not swap them.)

Cell width regularization is proposed in the work of [64]. Its purpose is to avoid illegal placements (with cell overlap) when neighbor cells are swapped during "tangling". Without the cell width regularization, the neighbor-cell swap operations might cause cell overlap due to different cell widths. Of course, the standard P&R flow does not use width-regularized cells, which raises the question of whether use of bloated cells could lead to misleading design-level routability assessments. Our experimental study of K_{th} obtained with bloated cells, versus maximum achievable utilization obtained with unbloated cells, is reported and offers some reassurance in this regard. Aside from this, we use bloated cells for two main reasons. (1) Our cell-level routability assessment focuses on intrinsic routability of cell layouts without any bloating of cell widths. (2) Despite the extra widths of cells, we can still assess pin accessibility for cells. Bloated cells still retain their original order and shapes of pins. So, the intrinsic routability for standard cells can still be measured. Below, design-level routability assessments with various technology and design parameters are reported.

Learning-Based K_{th} Prediction

Although K_{th} provides a useful means of routability assessment, it has three potential logistic disadvantages: large runtimes, large data footprints, and high consumption of commercial EDA licenses. First, as we perform neighbor-swaps to increase routing difficulty for a given placement, the increasing number of DRCs leads to large runtimes. This is because P&R tools perform increasingly expensive search-and-repair iterations in the detailed router. Furthermore, although we set a maximum runtime per each P&R run,[21] our overall runtime burden is still large, since the total number of tool runs itself is large. For example, when we perform P&R with $K = 1$ to 30 to obtain K_{th}, and there are just 100 sets of standard-cell libraries to be evaluated, up to 3000 P&R runs are required for this assessment. Second, large amounts of disk space may be used by our framework. For example, just 3000 runs of P&R with the VGA design occupy around 300GB of disk space in our experiments. Third, in many contexts the number of available commercial P&R tool licenses will also be limited.

To mitigate these potential disadvantages, we have developed *learning-based K_{th}* prediction. That is, to reduce the number of tool runs (thus saving runtime, disk space and license usage), we predict K_{th} via machine learning (ML) techniques, without performing all P&R runs. Figures 4.2(b) and (c) show PROBE2.0 flows without, and with, K_{th} prediction. The flow without K_{th} prediction in Figure 4.2(b) requires generation steps and multiple P&R runs. On the other hand, the flow with prediction in Figure 4.2(c) finds K_{th} using only inference with a trained ML model. We use 12 input features in our ML modeling, based on the technology and design parameters in Section 4.1.2. Table 4.7 summarizes the types, names, and data types of the input features that we use in the K_{th} prediction. From the original set of technology parameters, we

[21]We assume that runs with #DRCs under/around the threshold are finished within the maximum runtime. We set maximum runtimes based on the instance counts of the designs. In this work, we use maximum runtimes of two hours for a knight's tour and AES, and three hours for LDPC, JPEG and VGA.

Table 4.7. Input features of the K_{th} prediction.

Type	Name	Data	Notes
Technology	CPP	Integer	
	MP	Integer	
	PGpin	Category	BPR, M1, M1+M2
	CH	Integer	
	MPO	Integer	
	DR	Category	EL, ET
Design	BEOL	Integer	
	PDN	Category	Backside, Sparse, Middle, Dense
	Tool	Category	Tool A, Tool B
	Util	Float	[0.0, 1.0]
	Inst	Integer	
	Net	Integer	

omit Fin and RT from our feature list since Fin and RT are derivable from PGpin and CH. From the original set of design parameters, we omit "Design" and add the number of instances (*Inst*) and the number of nets (*Net*) as design features in our model.

We present our experimental results for the ML-based K_{th} prediction. Importantly, our ML models are trained in less than an hour using approximately 1968 data points (80% out of a total of 2460 data points); inference then requires only seconds for any technology-design parameter combination. By contrast, collecting a single K_{th} value for a given technology-design parameter combination requires not only the generation of standard-cell layouts and design enablements, but also up to 30 P&R runs that each consume significant average and maximum tool runtimes of 0.4 and 3.0 hours, respectively. Thus, runtime overhead is 12 (resp. 90) hours on average (resp. at most) per K_{th} value. In light of such large runtimes, the primary goal of our ML-based K_{th} prediction is to reduce the number of required P&R runs by predicting unseen data (that is, K_{th} values) with reasonably small errors. In Section 4.1.4 below, we also perform an experiment to show the tradeoff of model accuracy versus training data overheads.

Table 4.8. Default parameters in our experiments.

Design parameter		Technology parameter	
Name	**Value**	**Name**	**Value**
Fin	3	BEOL	10M
CPP	54	PDN	Middle
MP	40	Tool	A
RT	5	Util	0.7
PGpin	M1	Design	AES
CH	7	-	-
MPO	2	-	-
DR	EL	-	-

4.1.4 Experimental Setup and Results

We demonstrate routability assessment capability of PROBE2.0 via five main experiments.

- Expt. 1: Cell-level assessment with a knight's tour.
- Expt. 2: Design-level assessment with technology parameters.
- Expt. 3: Design-level assessment with design parameters.
- Expt. 4: Achievable utilization study with K_{th} results.
- Expt. 5: Learning-based K_{th} prediction.

We first show routability assessments using the PROBE2.0 framework. Expt. 1 demonstrates evaluation of cell-level routability using our knight's tour-based assessment. Expts. 2 and 3 perform design-level routability assessments with various technology and design parameters. Next, Expt. 4 studies *achievable utilization* in relation to the K_{th} routability metric, across a range of generated standard-cell libraries. Last, Expt. 5 demonstrates how learning-based K_{th} prediction can mitigate potential logistical overheads of our framework.

Table 4.9. List of 39 standard cells per generated library.

Cell list	Size
Inverter (INV), Buffer (BUF)	X1, X2, X4, X8
2-input AND/OR/NAND/NOR (AND2/OR2/NAND2/NOR2)	X1, X2
3-input AND/OR/NAND/NOR (AND3/OR3/NAND3/NOR3)	X1, X2
4-input NAND/NOR (NAND4/NOR4)	X1, X2
2-1 AND-OR-Inverter (AOI21), 2-2 AND-OR-Inverter (AOI22)	X1, X2
2-1 OR-AND-Inverter (OAI21), 2-2 OR-AND-Inverter (OAI22)	X1, X2
2-input MUX/XOR (MUX2/XOR2), D flip-flop (DFF)	X1

Experimental Setup

We generate 39 cells (38 combinational and 1 sequential) for each standard-cell library that we study, as listed in Table 4.9. We name cells according to functionality, number of inputs, and size. For example, INV_X1 is an X1-sized inverter, and AND3_X2 is an X2-sized 3-input AND gate. Our SMT-based standard-cell layout generation tool is open-sourced in [169].

Our experiments are based on a commercial 14nm technology. We modify design rules of Mx (M1, M2 and M3) layers (and of vias above) in routing technology files to enable scaling to sub-7nm technologies. We draw FEOL and BEOL (Mx) layer parameters from [32] [119]. Layer parameters and rules for M4 and above are drawn from the 14nm technology. For example, we choose the 9M (9-layer) BEOL stack option from the 14nm technology, and modify rules for the Mx layers (M1, M2, M3). We use existing design rules for the upper layers (M4 to M9). Mx layers are constrained to use unidirectional and min-width routing for standard-cell layout generation and P&R. We also assume that M1 is unavailable for signal routing.

All design rules are described in LEF format and routing technology file format [172] for use by commercial P&R tools. SMT-generated standard-cell layouts are converted into LEF format. For cell-level routability assessments, we use a knight's tour with 50 by 50 instances, constructed using an open-sourced implementation of the Warnsdorff-rule algorithm [139]. For design-level assessments, we perform synthesis by using a commercial tool [171] and P&R by

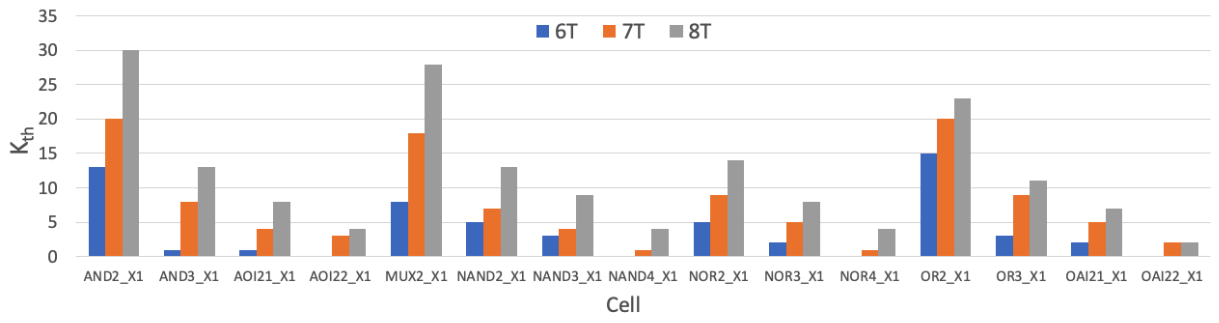


Figure 4.10. K_{th} results from knight's tour-based cell-level routability assessments across various track heights and cell masters. 15 cell masters from 6T, 7T and 8T standard-cell libraries are used in this experiment.

using two commercial P&R tools [144] [172].

We also use GNU parallel [152] to perform multiple evaluations in parallel. We assume that our designs with #DRCs less than the threshold finish within three hours. Setting the maximum runtime in GNU parallel reduces the total runtime of the framework. We use 500 DRCs as the DRC threshold to determine K_{th}. (The work of PROBE [64] used a threshold of 150 DRCs. We set the higher threshold of 500 DRCs since SMT-generated standard-cell layouts might not be as well-optimized for pin accessibility as mature libraries from industry.) Finally, we set default parameters as shown in Table 4.8. In all of our experiments, when not otherwise specified, we use these pre-defined default parameters.

Expt. 1: Cell-Level Assessment with a Knight's Tour

In this experiment, we perform knight's tour-based cell-level routability assessments. To show K_{th} of standard cells with various cell heights, we draw cells from 6T, 7T and 8T libraries. Among the library cells, we choose 15 combinational cells that have X1 size and more than one input pins.

Figure 4.10 shows K_{th} results from the cell-level routability assessment. From the results, we observe larger K_{th} as cell height increases. Also, by examining K_{th} per cell, we can see which cell types are routing-critical. For example, 4-input cells have relatively small values of K_{th} according to this experiment. Thus, at an early stage of technology development, cell-specific

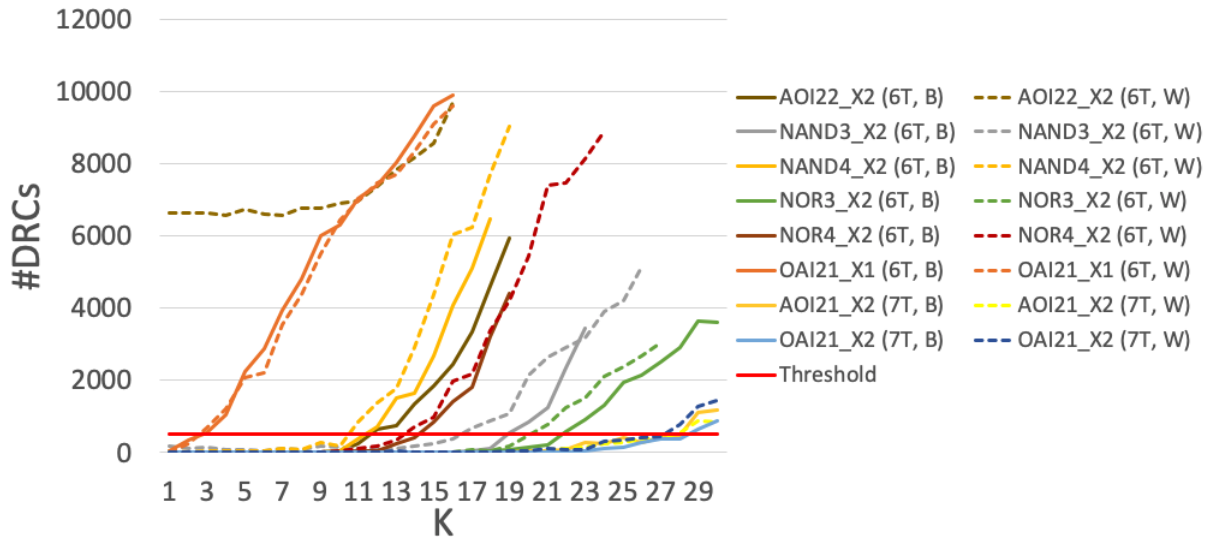


Figure 4.11. Cell-level routability assessments showing #DRCs versus K with different pin shape selection methods. RPA-Best (B) and RPA-Worst (W) pin shape selection methods are shown. Solid lines are for cells with RPA-Best, and dashed lines are for cells with RPA-Worst.

K_{th} is a promising indicator to determine which cells need to be improved in terms of routability.

We also show the benefit of our RPA-based pin selection. We generate standard-cell layouts using two pin shape selection methods, *RPA-Best* and *RPA-Worst*. RPA-Best denotes our proposed RPA-based pin shape selection which chooses the pin shape with the largest RPA value. To show that RPA-Best brings meaningful benefits, we also create standard-cell libraries by choosing a pin shape per pin that has *smallest* RPA value; this yields the RPA-Worst method. We study six cells from 6T libraries and two cells from 7T libraries. We perform cell-level routability assessments on RPA-Best and RPA-Worst versions of these standard cells.

Figure 4.11 plots the number of DRCs versus K, for the eight standard cells with different pin shape selection methods. Also, Table 4.10 shows the K_{th} values for each standard cell. Seven standard cells (out of eight cells) with RPA-Best show larger K_{th} than those with RPA-Worst. One cell (OAI_X2) shows the same K_{th} for both methods. We see that our RPA-based pin shape selection helps to improve the routability of standard cells.

Table 4.10. K_{th} results with RPA-Best and RPA-Worst pin shape selection methods. This table shows the K_{th} values corresponding to the data shown in Figure 4.11.

CH	**Cell**	K_{th}	
		RPA-Best	**RPA-Worst**
6T	AOI22_X2	11	0
	NAND3_X2	18	16
	NAND4_X2	11	10
	NOR3_X2	21	20
	NOR4_X2	14	13
	OAI21_X1	2	2
7T	AOI21_X2	28	27
	OAI21_X2	28	27

Expt. 2: Design-Level Assessment with Technology Parameters

In this experiment, we perform design-level routability experiments with various technology parameters. The goal is to show the routability impact of technology and standard-cell architecture options. Figure 4.12 shows K_{th} results that highlight the individual impacts of five technology parameters. We also experimentally confirm the ability of our framework to assess various design rule options. To study design rule choices, we define two scenarios, *DR1* and *DR2*. DR1 is a scenario which uses CPP = 54 and the EL (EUV-Loose) design rule set. DR2 uses CPP = 48 and the ET (EUV-Tight) design rule set. We choose these two design rule scenarios so that they are more likely to induce different standard-cell architectures.

From the results in Figure 4.12, we make five observations, as follows.
(1) As cell heights (or routing tracks) increase, the corresponding K_{th} values also increase.
(2) Standard cells with M1 PGpin (and the same RT) have better routability than those with BPR or M1+M2 PGpin. Also, standard cells with BPR and M1+M2 show similar K_{th} values. This reflects two phenomena. First, cells with BPR have one track less cell height than corresponding non-BPR cells. Second, cells with M1+M2 have worse routability than those with M1. This is because the M2 power and ground pins in standard cells with M1+M2 will block routing resources over the cells.

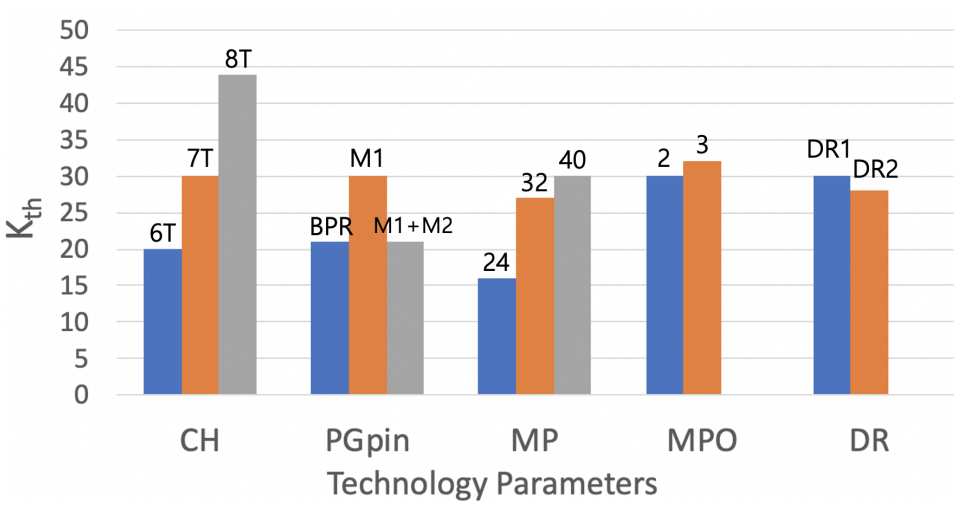


Figure 4.12. Design-level assessments with various technology parameters.

(3) Smaller pitches of Mx layers bring lower K_{th} values, despite smaller pitches nominally providing more routing resources (total number of wiring tracks) on Mx layers.

(4) Minimum pin openings of our standard-cell layouts have a clear impact on routability. Standard cells with 3 MPO show better routability than those with 2 MPO.

(5) For the DR cases as shown in Figure 4.12, DR2 has smaller K_{th} than DR1, implying that the tight design rule (DR-EOL in the scenario DR2) has a harmful impact on routability.

With respect to the fifth observation, we note that while DR2 brings smaller K_{th}, standard-cell layouts with 48nm CPP (DR2) have around 11% less area than layouts with 54nm CPP (DR1). We explore the relationship between area (utilization) and K_{th}. Finally, Table 4.11 gives additional details of K_{th} results for all the possible combinations with various Fin, MP, RT, PGpin and MPO in our experiments. (Data shown in Figure 4.12 is a subset of the table data.)

Expt. 3: Design-Level Assessment with Design Parameters

In this experiment, we study routability impacts of design parameter options. Figure 4.13 shows the K_{th} results from this study, which spans BEOL, PDN, Tool, Util and Design parameters. We make the following observations.

(1) According to the K_{th} values with the various BEOL stack options, 10M has better routability

Table 4.11. K_{th} results with various Fin, MP, RT, PGpin and MPO.

Fin	RT	MP	MPO	K_{th}		
				BPR	M1	M1+M2
2	4	24	2	2	11	4
			3	8	11	6
		32	2	10	15	9
			3	11	16	10
		40	2	10	20	13
			3	13	23	16
3	5	24	2	4	16	10
			3	6	19	11
		32	2	17	27	17
			3	19	28	18
		40	2	21	30	21
			3	24	32	24
3	6	24	2	17	28	19
			3	19	27	19
		32	2	27	44	25
			3	28	45	27
		40	2	32	44	34
			3	33	45	36

than 9M, and 11M has better routability than 10M. Also, 10M and 11M have better routability than 13M. As shown in Table 4.2, the routing resources of 10M and 11M are larger than those of 13M. This illustrates how overall routing capacity can worsen even if the number of metal layers is increased. However, the additional layers bring other benefits. For example, we may implement a denser, more robust PDN resulting in better design performance due to less resistivity of wide metal layers.

(2) As fewer layer resources are used for PDN, the results show better routability.

(3) The K_{th} results from Tool A dominate those of Tool B. In this sense, our framework also has the ability to evaluate placers and routers, as does the previous work of [64].

(4) As Util increases, the K_{th} values decrease since routing with denser placements is more difficult.

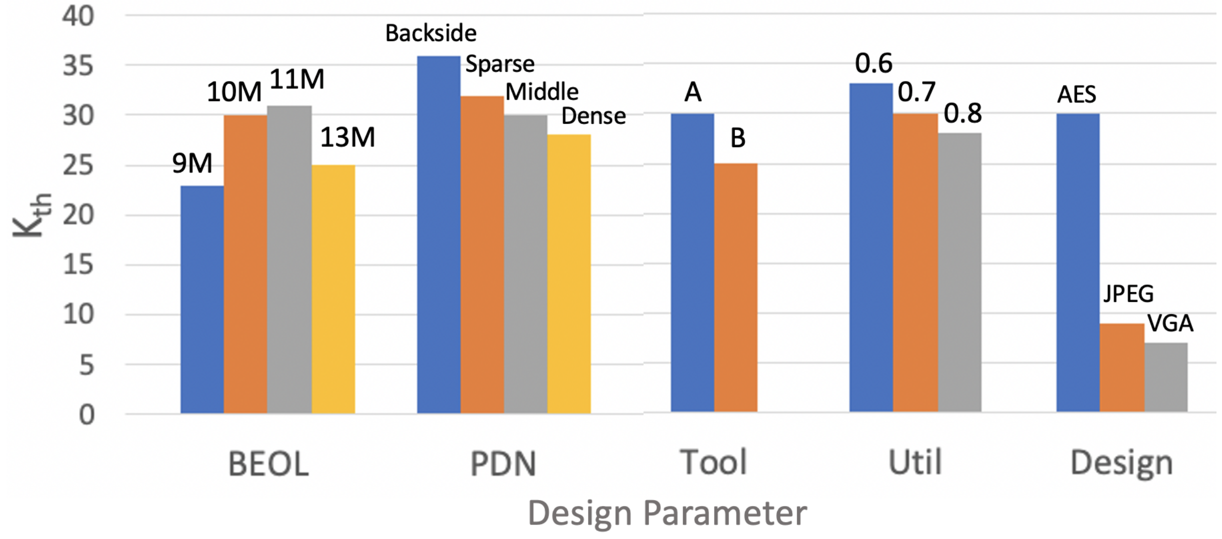


Figure 4.13. K_{th} results of design-level routability assessments with various design parameters.

(5) The base designs used in the design-level routability assessments also affect the results. In particular, designs with larger instance counts have smaller K_{th}. This is expected: for a given amount of "tangling" (K), and all else being equal, a larger design's placement is expected to have more routing #DRCs. ([64] gives a probabilistic analysis of routing hotspots and routing failure according to the number of instances.)

Expt. 4: Achievable Utilization versus K_{th}

In production IC design, maximizing the utilization of die area is a common objective toward achieving the best PPAC. Also, the maximum achievable utilization is a first-order indicator for routability, even as it captures the A, C dimensions of PPAC. In this experiment, we explore the relationship between achievable utilization (in design implementations using unbloated cells) and K_{th} (in design-level routability assessment using cell width-regularized libraries). Such a study could inform the future use of K_{th} to predict die utilization and area. We define *achievable utilization* as the highest utilization for which #DRCs is smaller than the pre-defined DRC threshold.

In our study of achievable utilization, we use two designs, AES and LDPC. For AES, we

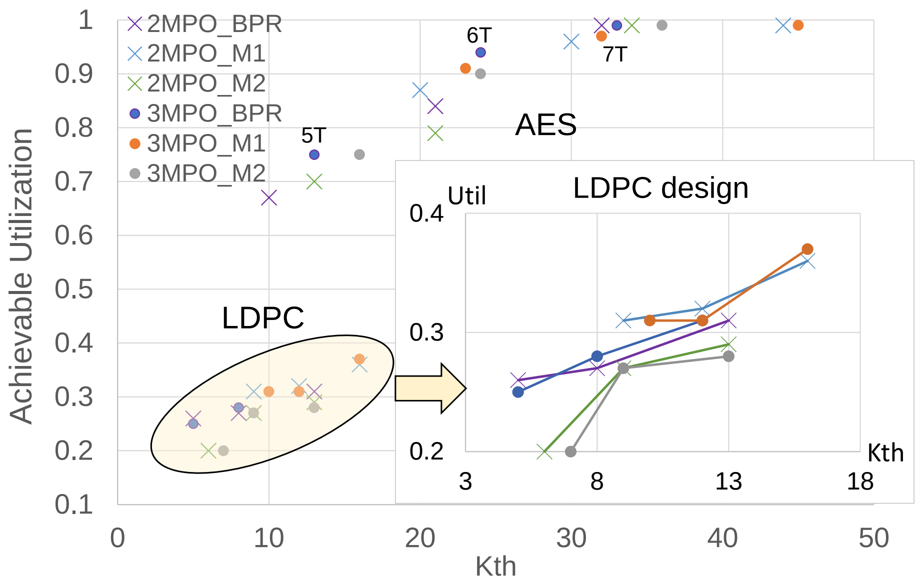


Figure 4.14. Comparison between K_{th} and *achievable utilization* for the AES and LDPC designs.

collect achievable utilizations from 0.60 to 0.99, with step size of 0.01. For LDPC, which is a notoriously difficult to route testcase, we collect achievable utilizations from 0.20 to 0.40, again with step size of 0.01. To extract K_{th}, we set Util as 0.7 for AES and 0.2 for LDPC. Figure 4.14 shows our K_{th} results and the corresponding achievable utilizations. The plot shows that K_{th} values and achievable utilizations are sensibly related for the given design parameters. The data also show how proper combination of technology and design parameter choices can potentially bring routing-friendliness and high utilization: large cell heights, large cell areas, sparse PDN, ample routing resources, and/or relaxed design rules all lead to routability with maximum utilization. Especially, in the enlarged view of LDPC design plot, we observe similar tendencies as seen in Figure 4.12. For example, the cells with M1 PGpin have better K_{th} compared to cells with M2 and BPR. Also, both K_{th} values and achievable utilizations gradually increase with cell heights. The cells with 2MPO and 3MPO have similar K_{th} values and achievable utilizations. This experiment hints that K_{th} could be useful in a predictor of design-specific achievable utilization. We leave this possibility for future work.

Expt. 5: Learning-Based K_{th} Prediction

We now show results from our learning-based K_{th} prediction. We compile a dataset of 2460 K_{th} values corresponding to a wide range of settings for the various features described in Table 4.4. We use the open-source AutoML package [154] to predict K_{th}. AutoML has a hyper-parameter tuning ability on various models including gradient boosting machine (GBM) [41], XGBoost [20], distributed random forest (DRF), extremely randomized tree (XRT) [42], deep learning (DL) and generalized linear model (GLM) [97]. AutoML also suggests combined models that outperform a single model.

We down-sample and up-sample our data since the data are imbalanced in terms of K_{th} distributions.[22] We use 60 as a target sample number to generate balanced data. We randomly partition our augmented (i.e., after sampling schemes are applied) dataset as 80% used for training and 20% used for testing. We perform all experiments using an Intel Xeon Gold 6148 2.40GHz server (80 threads) with 256GB RAM.

We use the 3.30.0.6 version of AutoML to train our models, with default input parameter settings for AutoML. The default settings of AutoML include *nfolds* = 5 for cross-validation, *leaderboard_frame* = testing set, and *sort_metric* = mean residual deviance to rank the trained regression models. Details of these settings are found at [154]. We set the training time limit as one hour with 80 threads. The suggested models from AutoML and the corresponding trained results are described in Table 4.12. Note that *Deviance* denotes mean residual deviance, *RMSE* denotes root mean squared error, and *MAE* denotes mean absolute error.

Figure 4.15 shows comparisons of *golden* K_{th} (i.e., ground truth data which comes from actual results obtained from our framework) versus *predicted* K_{th} using the model *StackedEnsemble_BestOfFamily*. The *StackedEnsemble_BestOfFamily* contains six models: GBM, XGBoost, DRF, XRT, DL and GLM. AutoML generates a stacked ensemble model with GLM

[22]When generated standard cells are not routing-friendly, routing for designs that use those cells might be infeasible. This causes the observed distribution of K_{th} values to be biased toward zero. The down-sampling and up-sampling schemes help us to avoid over-fitting in the prediction of K_{th}.

Table 4.12. Results on training data of Top-5 ML models by using AutoML. The three *GBM* models (*GBM_1*, *GBM_2* and *GBM_3*) have different configurations for hyperparameters. The training runtime is limited to one hour with 80 CPU threads.

Model	Deviance	RMSE	MAE
StackedEnsemble_BestOfFamily	0.733	0.857	0.353
XGBoost	0.865	0.930	0.300
GBM_1	0.896	0.947	0.374
GBM_2	0.947	0.973	0.350
GBM_3	0.950	0.975	0.440

Table 4.13. Results of DL modeling with AutoML. *DL_1h* (resp. *DL_24h*) denotes a DL model with a 1-hour (resp. 24-hour) runtime limit for training.

Model	Deviance	RMSE	MAE
DL_1h	6.275	2.505	1.789
DL_24h	1.469	1.212	0.615

as *meta_learner*, based on those six models. We show *max_error* and *avg_error* metrics, which are the maximum and average difference between golden and predicted K_{th}, respectively. Figure 4.15(c) shows a balanced error distribution for the testing set. To further understand our K_{th} prediction, we extract feature importance from the XGBoost model. We do this for XGBoost since *StackedEnsemble_BestOfFamily* consists of six models and does not support such a feature extraction. The extracted feature importance is visualized in Figure 4.15(d). We conclude that

Table 4.14. Model accuracy results with different amounts of training data (*TD*), i.e., 20, 40, 60 and 80% of the total data (2460 K_{th} values), with a one-hour limit and 80 threads. Testing data are fixed for an apples-to-apples comparison. *ME* and *AE* respectively denote maximum and average error of K_{th} prediction.

TD	ME	AE	Deviance	RMSE	MAE
20%	14.696	1.155	4.916	2.217	1.155
40%	8.131	0.610	1.702	1.304	0.610
60%	8.553	0.392	1.006	1.003	0.392
80%	8.459	0.353	0.733	0.857	0.353

K_{th} is mostly affected by Inst, CH and BEOL.

From the results in Table 4.12, we can infer that boosting ML models (XGBoost and GBM) outperform DL models in our experiment. (Put another way: no DL model made it into the AutoML Top-5 models.) However, given the recent intense interest in deep learning, we also explicitly study prediction with DL models. Table 4.13 shows the DL model results with AutoML, for the same training set as in Table 4.12. In this experiment, we train the DL models with two maximum runtimes for training. DL_1h denotes a DL model obtained with the same 1-hour training runtime limit used for Table 4.12. DL_24h denotes a DL model with 24-hour training runtime limit. Even with the longer training time, DL does not surpass the other ML models in Table 4.12. (The recent study [39] provides insights on boosting models outperforming DL models.)

As noted in Section 4.1.3, a primary benefit of learning-based K_{th} prediction (Figure 4.2(c)) is that it can save up to 90 hours of P&R tool runtime with each predicted value. To characterize the tradeoff of model accuracy versus training data overheads, we further conduct an experiment with different numbers of training data. Since obtaining a single K_{th} requires 30 P&R runs (averaging 0.4 hours each) in our experiment, obtaining a total of 2460 K_{th} values requires approximately 15.4 days of runtime. We have studied K_{th} prediction with various numbers of training data, and with fixed testing data. Table 4.14 shows results with different amounts of training data ($\sim$20%, $\sim$40%, $\sim$60% and $\sim$80% of the total dataset size). Generating 20% of the total data as training data reduces schedule overhead to approximately 3.1 days, and leads to 1.155 average K_{th} error in predicting the held-out 20% test data. As expected, more training data leads to better model accuracy (smaller errors) at the cost of schedule overhead.

4.1.5 Conclusion

We have proposed a novel framework to evaluate routability impacts of advanced-node scaling options, spanning across technology, design enablement, and design parameters. Several options for *scaling boosters* are assessed in terms of their routability impacts. Our framework is

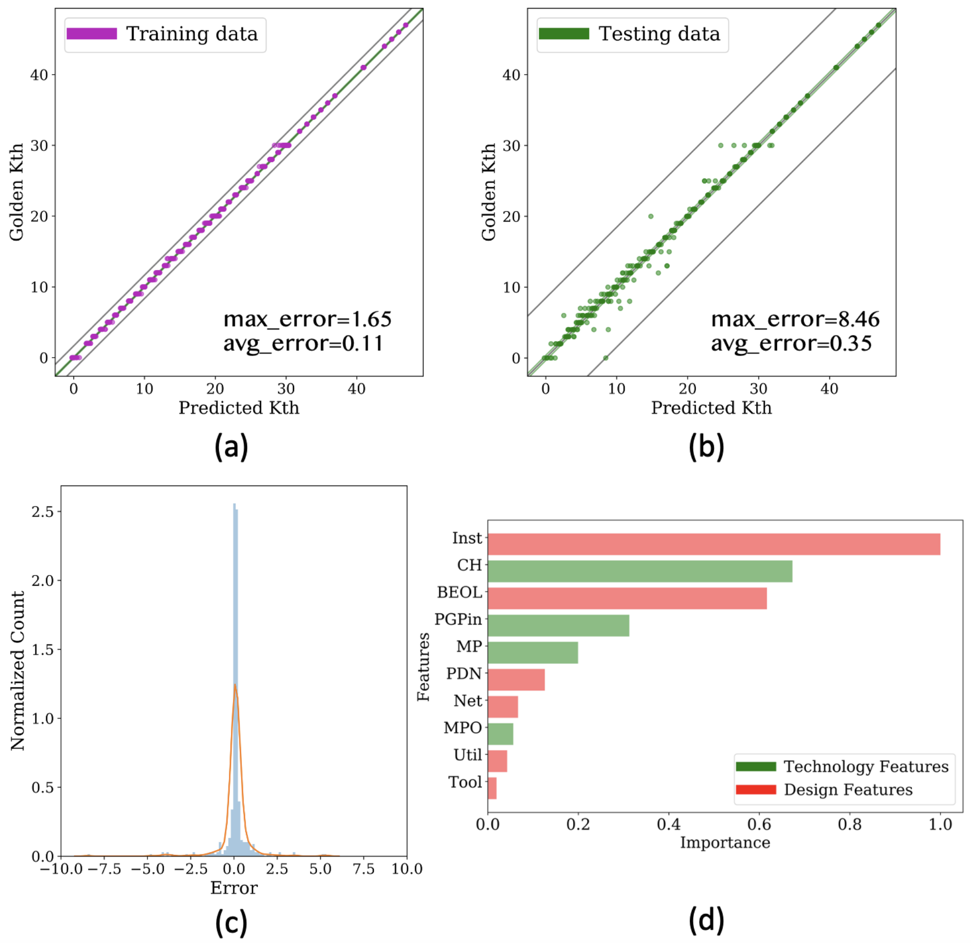


Figure 4.15. Comparison between golden K_{th} and predicted K_{th} and extracted feature importance; (a) training data, (b) testing data, (c) error distribution on testing data with kernel density estimation (KDE) plot and (d) extracted feature importance.

well-matched to the needs of DTCO, particularly for early stages of technology development. Crucially, our framework can reduce the timelines for design enablement and design implementation that limit today's DTCO methodologies. Also, our framework can flexibly support additional technology and design options.

We integrate a powerful SMT-based standard-cell layout generation capability. Optimal layout solutions are obtained via a unified constraint satisfaction formulation that spans technology and cell architecture parameters. We also build upon the previous routability assessment framework of [64].

Our new framework provides automatic generation of cell libraries and collaterals (LEF, Liberty, routing technology files) for commercial P&R tooling. We propose RPA-based and top-metal-only pin shape selection to improve routability of our generated standard-cell libraries. We validate these aspects of our methodology using a novel *cell-based* routability assessment with knight's tour-based topology generation. We also perform *design-based* routability assessment using open-source testcases, and show correlations of the K_{th} routability metric to achievable utilizations in P&R. Furthermore, we propose learning-based K_{th} prediction to reduce runtimes and required disk space, and to mitigate P&R tool license overheads. Finally, experimental studies confirm the capability of our framework to produce routability assessments across a large range of technology and design parameters.

Open directions for future research include the following.

- Our framework focuses on the Area and Cost dimensions of PPAC. This has value in early technology development, especially since density is the dominant driver for foundry technology and design enablement [123]. However, future work should broaden assessments to include power and performance evaluations, along lines presented in Section 4.3 below. Automation and/or prediction of library characterizations is a related challenge for future research.

- Our framework today covers a number of technology options, design rules and standard-cell architectures. However, extensions to support other technology options and/or design rules for advanced technologies may be desirable. For example, a target technology might require self-aligned double patterning (SADP) design rules and bidirectional routing. Also, new standard-cell architectures might be required, such as multi-height standard cells and rectilinear pin shapes.

- Extending our framework to incorporate an open-source P&R tool, such as [2] [176], may help to scale exploration and turnaround times beyond the limits of available commercial P&R tool licenses. In this context, learning to map routability and other PPAC-related assessments between implementation tool chains will be a valuable future contribution.

4.2 A Novel Framework for DTCO: Fast and Automatic Routability Assessment with Machine Learning for Sub-3nm Technology Options

Design-technology co-optimization (DTCO) is an essential, high-value element of technology enablement. However, DTCO has proved to be a very expensive process; large turnaround times and engineering efforts are needed to develop standard-cell libraries and perform comprehensive block implementation experiments. Turnaround time of DTCO at advanced nodes is weeks to months with hundreds of engineers. Meanwhile, standard cells have been optimized to aim for minimum cell area (cell height) for decades. However, cell-level area scaling does not necessarily result in block-level area scaling at advanced nodes. As standard-cell area is scaled down, local routing and pin accessibility of standard cells become significantly difficult. Various *scaling boosters*, such as buried power rails (BPR) [102], backside PDN and supervias, are proposed to improve power, performance, area and cost (PPAC). Scaling boosters also have an impact on routability. Due to significant routability impacts of standard-cell architectures and scaling boosters, block-level area must be estimated at an early stage of technology development.

Recently, DTCO methods for technology definition have been proposed in many aspects. [12] proposes a machine learning (ML)-based approach to find optimal combinations of design, technology and other ingredients for high-performance CPU designs. However, there is no automatic design enablement stage (i.e., standard-cell library generation), and the methodology requires four to six weeks of turnaround time. [133] proposes an ML-based modeling framework for devices. The framework can generate compact models of novel devices without prior knowledge. On the other hand, the work does not consider block-level evaluations.

In this work, we apply the PROBE2.0 framework [24] (Section 4.1 of this thesis) to evaluate various technology options and configurations for sub-3nm technology nodes. We generate 448 unique standard-cell architectures, combining cell height (*CH*, 80$\sim$150nm), contacted poly pitch (*CPP*, 39$\sim$57nm), metal pitch (*MP*, 16$\sim$30nm) and use/non-use of buried power rails

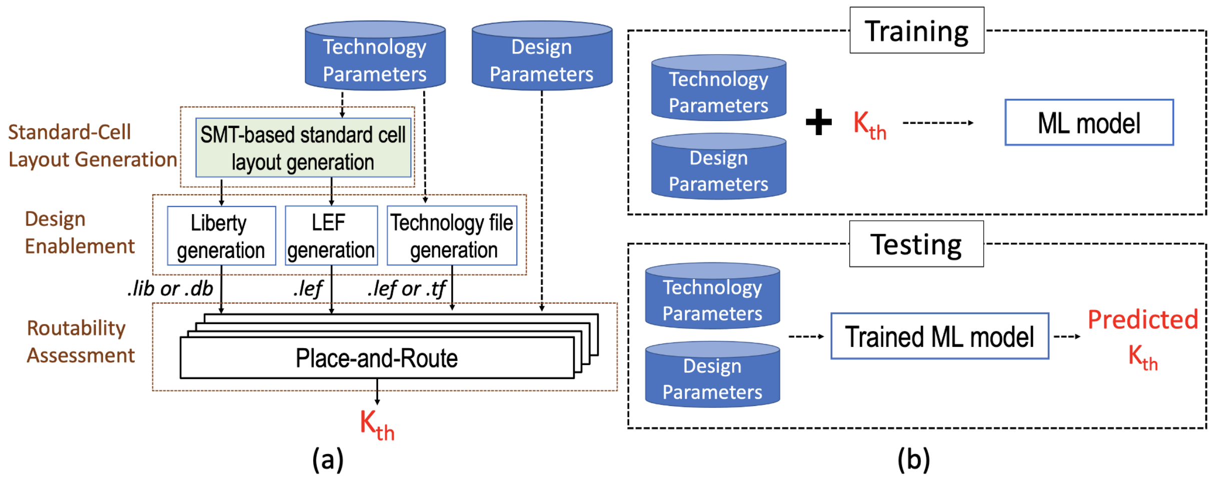


Figure 4.16. (a) Overall flow of PROBE2.0 [24] and (b) machine learning-based K_{th} prediction.

(BPR). We study achievable block-level area with automatically generated standard-cell libraries across four main axes: (i) available M2 routing tracks (RT) on standard cells, (ii) use of BPR, (iii) gear ratios for MP to CPP, and (iv) different available M2 RT with same CH. We also apply ML-assisted routability prediction to expedite assessment.

4.2.1 DTCO Framework and Configuration for Sub-3nm Nodes

In this section, we summarize PROBE2.0 [24], the framework used for routability assessment in DTCO, along with the configurations for sub-3nm standard-cell libraries that we study. As shown in Figure 4.16(a), the PROBE2.0 framework includes automatic standard-cell layout generation and design enablement generation, and (cell- and design-level) routability assessments. The basic idea of PROBE2.0 is to measure inherent routability, represented by K_{th} metric, through neighbor-swap operations applied to canonical placements. Figure 4.17 illustrates the neighbor-swap operation. For a given cell, we randomly choose a neighboring cell and swap the locations of the cell pair. Neighbor-swaps progressively increase routing difficulty by "tangling" the placement. We let K denote the number of neighbor-swap operations, normalized to the number of instances (standard-cells) in the design. As K increases, the number of post-routing design rule check violations (#DRCs) likely increases; we define K_{th} as the

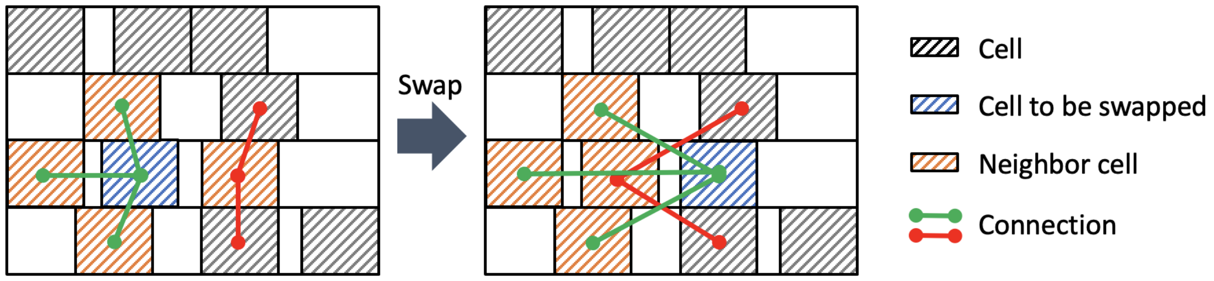


Figure 4.17. An example of neighbor-swap operation. The blue-colored cell is swapped with one of the red-colored neighboring cells.

maximum K for which #DRCs is less than a prescribed threshold. PROBE2.0 advances over the previous PROBE [64] with a satisfiability modulo theory (SMT)-based “automatic” standard-cell layout generation [26] [82] and an ML-based K_{th} prediction, as illustrated in Figure 4.16(b).

In this work, we generate nine standard-cell libraries to study routability and achievable block-level cell density at sub-3nm technology nodes. Table 4.16 shows sets of parameters for the nine libraries, as follows. (i) Lib{1,2,5,6} have 4 RT while Lib{3,4,7,8} have 5 RT. (ii) Lib{1,3,5,7,9} have BPR for power and ground pins while Lib{2,4,6,8} have M1 pins. (iii) Lib{1,2,3,4} have a 1:2 ratio for MP to CPP (20nm MP and 40nm CPP) and Lib{5,6,7,8} have a 2:3 ratio for MP to CPP (26nm MP and 39nm CPP). (iv) Lib{3,9} have 120nm CH but Lib3 has 5 RT and Lib9 has four tracks according to MP of Lib{3,9}. Figure 4.18 illustrates OAI21_X1 gates in the nine libraries.

Table 4.15. Technology and design parameters in this work. We use all the parameter types defined in [24].

Param. type	**Name**	**Option**	**Name**	**Option**
Technology	Fin	2, 3	PGpin	BPR, M1
	CPP	39, 40	CH	5, 6, 7
	MP	20, 24, 26	MPO	3
	RT	4, 5	DR	EUV-Loose
Design	BEOL	13M	Tool	Tool A
	PDN	Sparce	Util	0.6
	Design	Knight’s tour, AES, LDPC, JPEG, VGA		

Table 4.16. Technology parameters for the nine standard-cell libraries which we study in this work.

Name	Fin	CPP	MP	RT	PGpin	CH (T)	CH (nm)
Lib1	2	40	20	4	BPR	5T	100
Lib2	2	40	20	4	M1	6T	120
Lib3	3	40	20	5	BPR	6T	120
Lib4	3	40	20	5	M1	7T	140
Lib5	2	39	26	4	BPR	5T	130
Lib6	2	39	26	4	M1	6T	156
Lib7	3	39	26	5	BPR	6T	156
Lib8	3	39	26	5	M1	7T	182
Lib9	2	40	24	4	BPR	5T	120

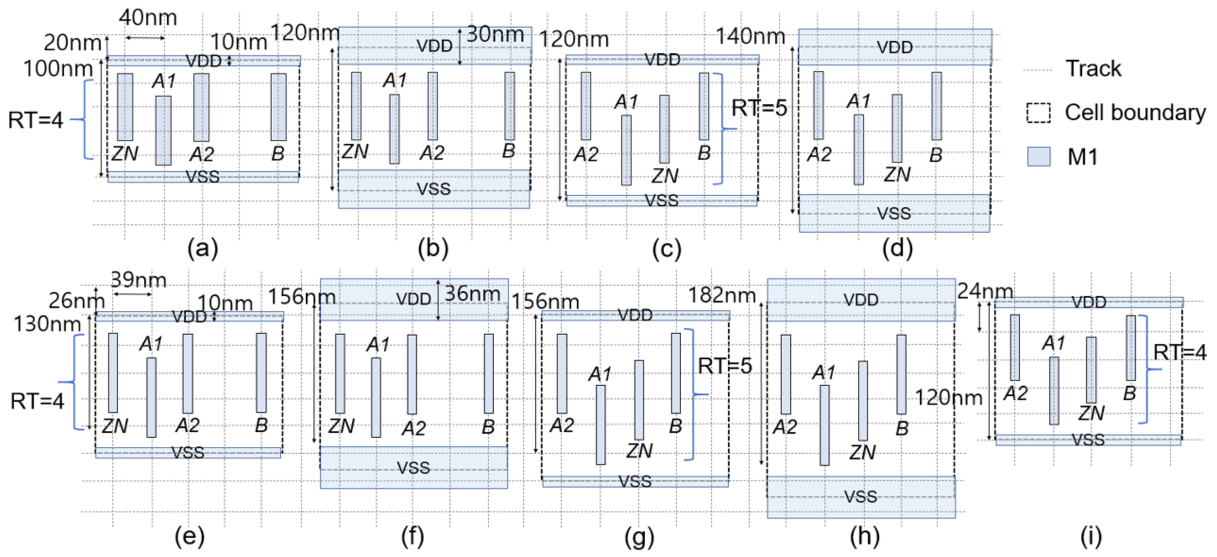


Figure 4.18. Standard-cell layout of the nine libraries in Table 4.16. OAI21_X1 of Lib1 to Lib9 are illustrated in (a) to (i), respectively. We show a LEF view (cell boundary and pin) of layout of OAI21_X1 per library.

4.2.2 Routability Assessment and Block-Level Area Case Study

We perform cell- and design-level routability assessments to evaluate generated standard cells. Cell-level assessment is used to assess inherent routability for each cell in a standard-cell library. Design-level assessment, i.e., at the level of an entire place-and-route block, is used to evaluate a set of standard cells in the same library. We also perform case study of cell-level and

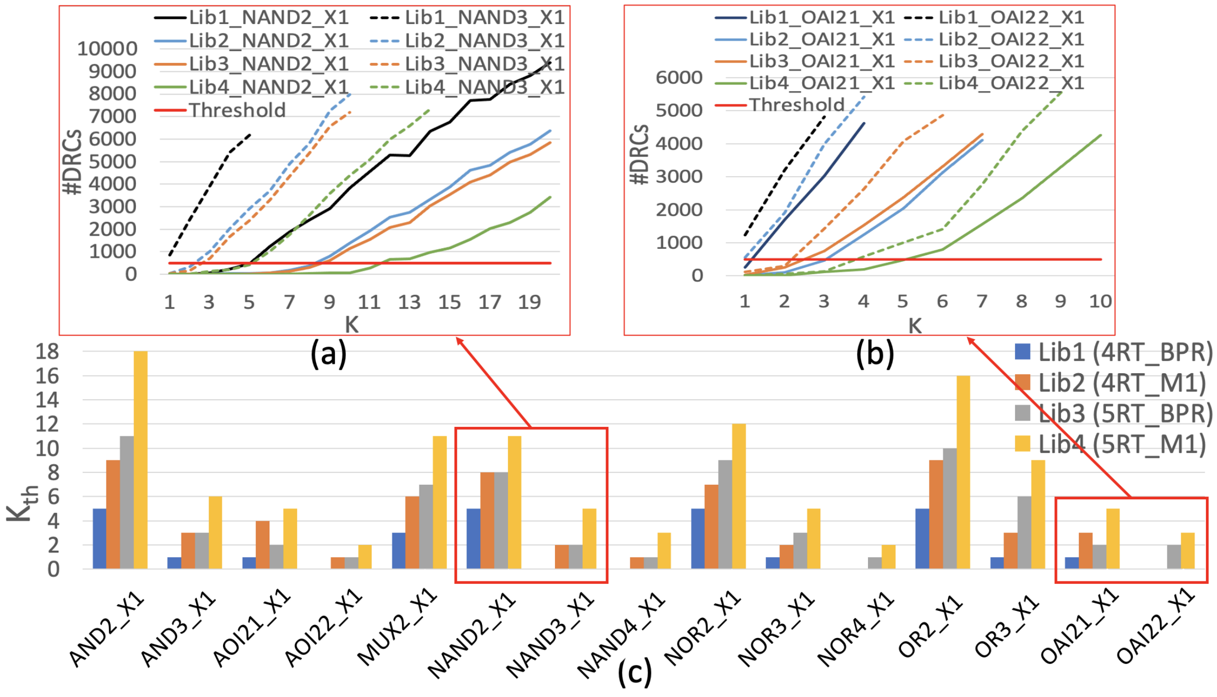


Figure 4.19. Results for cell-level routability assessments. (a) #DRC vs. K_{th} plot for NAND2_X1 and NAND3_X1. (b) #DRC vs. K_{th} plot for OAI21_X1 and OAI22_X1. (c) K_{th} values for 15 cell types per four standard-cell libraries.

achievable block-level area. Cell height of a standard-cell library is linearly related to cell-level area. However, cell-level area benefit does not always result in block-level area benefit, due to routability effects. Our case study shows how we can evaluate technology options with respect to block-level area density using the PROBE2.0 framework.

Figure 4.19 shows results for the *cell-level* routability assessments. We study 15 standard cells with size X1, in each of four libraries (Lib{1,2,3,4}). Results in Figure 4.19(c) show that standard cells with M1 have better routability than those with BPR, and standard cells with five tracks have better routability than those with four tracks. Further, standard cells with larger numbers of input pins have worse routability than those with smaller numbers of input pins. Figures 4.19(a) and (b) show #DRCs vs. K plots for NAND and OAI gates, respectively.

Figure 4.20 shows results for *design-level* routability assessments with four designs, AES, LDPC, JPEG and VGA. Figure 4.20(b) shows K_{th} metric per standard-cell library per design while Figure 4.20(a) shows #DRCs vs. K plots for the AES design. Figure 4.21 shows results

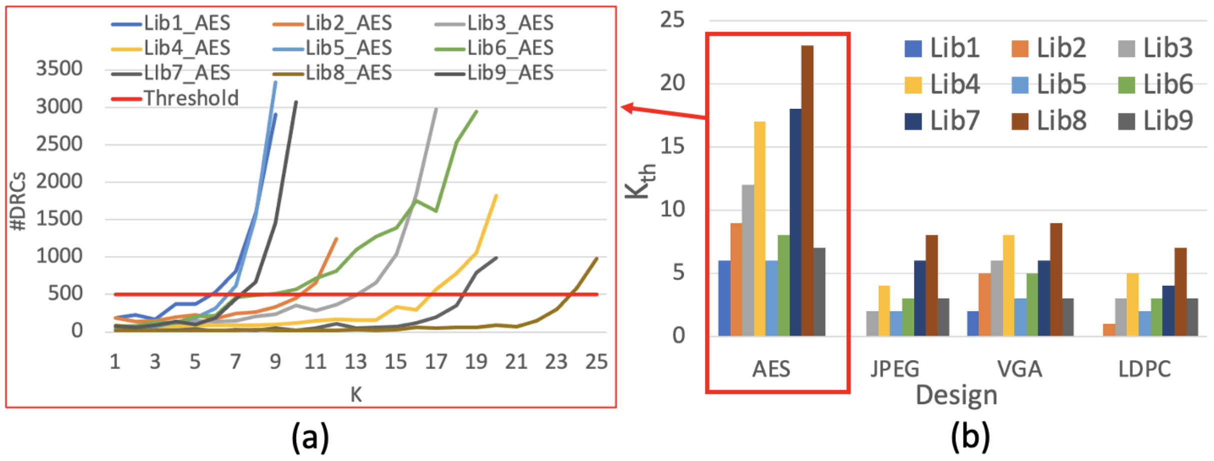


Figure 4.20. Design-level routability assessment for the nine standard-cell libraries. (a) #DRC vs. K_{th} plot for AES design. (b) K_{th} values for AES, JPEG and VGA designs. We use 0.6 Util for AES, JPEG and VGA and 0.15 Util for LDPC design.

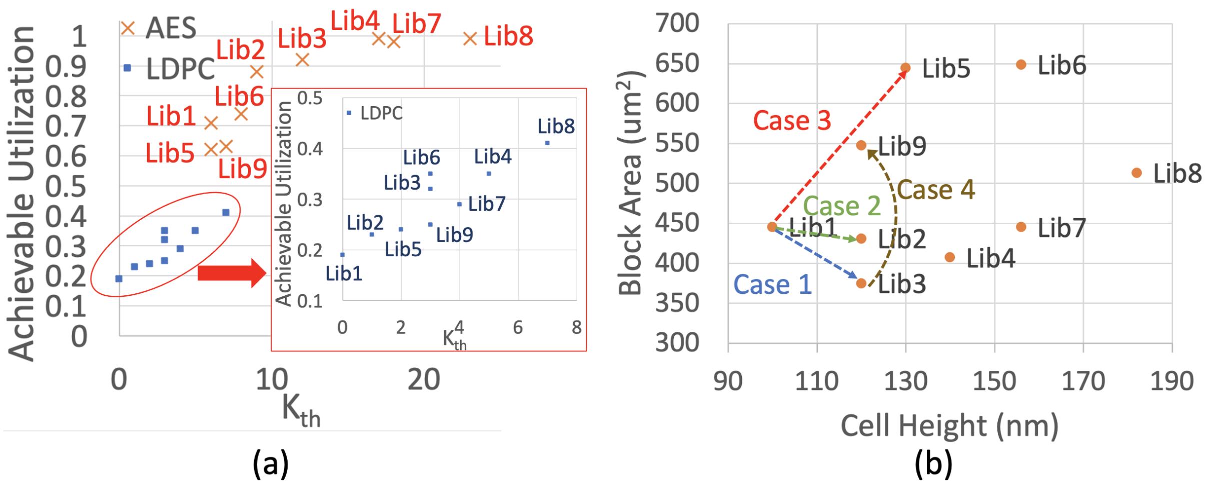


Figure 4.21. Achievable density and block-area case study. (a) Plot for K_{th} vs. achievable utilization for AES and LDPC designs. (b) Achievable block-area study with the nine libraries and AES design. The arrows show block-level area and cell height differences in the case study, as also shown in Table 4.17.

of our achievable utilization study for AES and LDPC designs. We observe strong positive correlation between K_{th} and achievable utilization.

In addition, we perform case study for block-level area benefits across four main axes of technology options.

- Case 1: Available M2 routing tracks (RT) (Lib1 vs. Lib3).

Table 4.17. Results for achievable block-level area case study, expressed as overhead of *Lib B* relative to *Lib A*, i.e., $(Area_B - Area_A)/Area_A \times 100\%$

Case	***Lib A***		***Lib B***		**Cell-level**	**Block-level**
	Name	**CH**	**Name**	**CH**	**area overhead**	**area overhead**
Case 1	Lib1	100nm	Lib3	120nm	**20%**	**-16%**
Case 2	Lib1	100nm	Lib2	120nm	**20%**	**-3%**
Case 3	Lib1	100nm	Lib5	130nm	**30%**	**45%**
Case 4	Lib3	120nm	Lib9	120nm	**0%**	**46%**

- Case 2: Use of buried power rails (Lib1 vs. Lib2).
- Case 3: Gear ratios for MP to CPP (Lib1 vs. Lib5).
- Case 4: Different RT with the same CH (Lib3 vs. Lib9).

In Case 1, Lib3 has 20% larger cell area than Lib1. However, based on the achievable utilization (0.71 and 0.92 for Lib1 and Lib3, respectively), the achievable block-level area with Lib1 is 16% larger than with Lib3. It is important to note that in this case, reduction of RT brings less cell-level area, but the block-level area actually increases. In Case 2, Lib2 has M1 PGpin while Lib1 has BPR, so that Lib2 also has 20% larger cell area than Lib1. Applying the same calculation as with Case 1, we obtain that the achievable area with Lib2 is 3% less than with Lib1. Thus, in Case 2, the cell-level area benefit from use of BPR is not reflected as a design-level area benefit. In Case 3, Lib1 has a 1:2 gear ratio for MP to CPP and Lib5 has a 2:3 gear ratio. Since Lib5 has 26nm MP, Lib5 has 30% more cell area than Lib1. But, block-level area of Lib5 is 45% larger than that of Lib1 since the respective Lib1 and Lib5 achievable utilizations are 0.71 and 0.62. Last, in Case 4, Lib9 has the same CH (120nm) as Lib3, but MP of Lib9 is 24nm instead of 20nm. Larger MP brings benefits of reduced resistance, but Lib3 incurs 46% area penalty at design-level. Table 4.17 summarizes our case study. As shown in Table 4.17 and Figure 4.21(b), block-level area no longer changes linearly with CH at advanced nodes. Even when CH increases, block-level area can decrease.

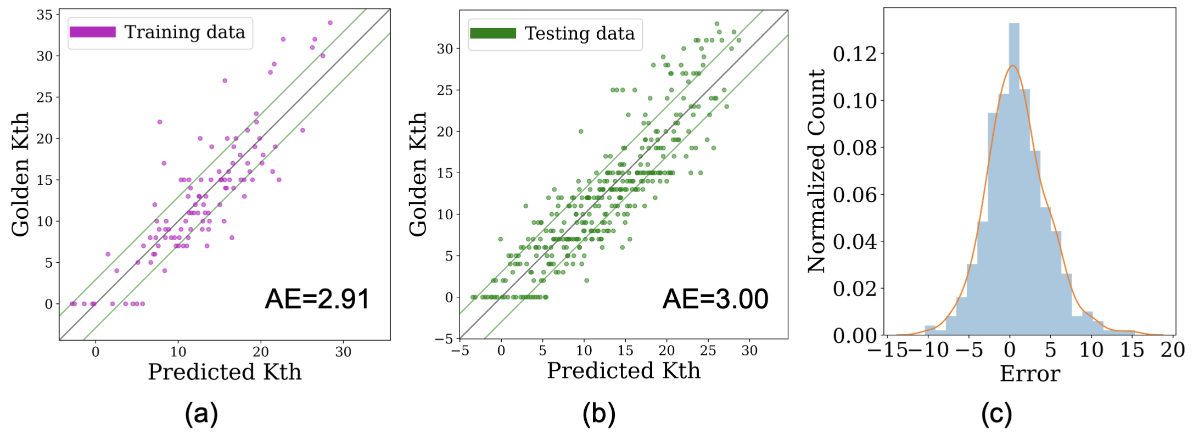


Figure 4.22. Comparison between golden K_{th} and predicted K_{th}. (a) Results for 128 training data sampled by LHS. (b) Results for 320 testing data. (c) Error distribution on the testing dataset with kernel density estimation (KDE) plot.

4.2.3 Machine Learning (ML)-Assisted Routability Assessment

PROBE2.0 [24] uses ML-based K_{th} prediction to reduce the number of place-and-route (P&R) implementation runs needed for accurate routability assessment. However, large training sets (e.g., 80% of libraries) are used. In this work, we apply Latin hypercube sampling (LHS) [95] to select an asymptotically more efficient training set for ML-based K_{th} prediction. In a case study, we create 448 standard-cell libraries as follows: (i) CPP = 39, 42, 45, 48, 51, 54, 57nm; (ii) MP = 16, 18, 20, 22, 24, 26, 28, 30nm for M1 and M2; (iii) 4 and 5 RT; (iv) BPR and M1 PGpin; and (v) M3 pitch:CPP ratios of 1:2 and 2:3. We choose training sets of 64, 128 and 256 data points (i.e., standard-cell library and K_{th}) by LHS and use all remaining (resp. 384, 320, 192) data points for model testing. Figure 4.22 shows comparisons for golden K_{th} values (which are extracted from P&R experiments) and predicted K_{th} values from our ML-based prediction. Figures 4.22(a) and (b) show the golden and predicted K_{th} comparisons when 128 and 320 data points are used for training and testing, respectively. Figure 4.22(c) shows the error distribution for the testing dataset. For the three LHS-based training set sizes (64, 128, 256) we obtain 3.07, 3.00, 2.90 average K_{th} prediction error in testing.

4.2.4 Conclusion

A machine learning (ML)-assisted design-technology co-optimization (DTCO) framework [24] is applied to study sub-3nm node cell and block-level DTCO. We perform cell- and design (block)-level routability assessments for >400 unique standard-cell architectures by varying technology options along axes of CPP, MP, RT, and use/non-use of BPR. The regime of CH <120nm with four available tracks shows increasing block-level area due to routing difficulty, indicating diminishing return on efforts to push ground rules. BPR improves block-level scaling with CH <120nm, but scaling benefits slow down with CH <100nm. A 1:2 gear ratio (MP:CPP) improves block-level area compared to a 2:3 ratio. The proposed DTCO method will extend to evaluate power and performance along with the cell- and block-level area density assessment presented in this study.

4.3 PROBE3.0: A Systematic Framework for Design-Technology Pathfinding with Improved Design Enablement

With the slowdown of Moore's-Law dimensional scaling, the concept of DTCO has emerged as a crucial approach at advanced technology nodes. DTCO has become the primary methodology for determining which technology and design configurations are suitable for mass production. As noted in Section 4.1, scaling boosters such as backside power delivery networks (BSPDN) and buried power rails (BPR) have gained importance in optimizing power, performance, area and cost (PPAC) in future technologies. Recognizing the growing significance of scaling boosters, Section 4.1 introduces a systematic framework for routability assessments, providing an exploration methodology for area and cost (AC) considerations. However, there is a substantial increase in the demand for comprehensive PPAC explorations, encompassing all aspects of PPAC.

Contributions of Our Work. Compared to the previous works PROBE1.0 [64] and PROBE2.0

[24], our new framework provides three main technical achievements.

(1) **We establish the first comprehensive end-to-end design and technology pathfinding framework.** [24] [64] focus on area and cost without considering power and performance. Thus, there is a significant discrepancy between [24] [64] and the actual DTCO process in the industry. In this work, we propose a more complete and systematic PROBE3.0 framework, which incorporates power and performance aspects for design-technology pathfinding at an early stage of technology development. PROBE3.0 enables fast and accurate PPAC evaluations by generating configurable PDKs, including standard-cell libraries.

(2) **We improve our designs for PPAC explorations.** Design is a critical factor for PPAC explorations, and artificially generated designs enable us to explore a wider solution space. We leverage [78] to generate artificial designs. To create more realistic artificial designs, we develop a machine learning (ML)-based parameter tuning flow built on [78] to find the best input parameters for generating such designs. Section 4.3.4 details our artificial design generation flow. Further, *cell width-regularization* is employed in [24] [64] to prevent illegal placements when swapping neighboring cells to assess the routability metric, K_{th}. We propose a *clustering-based cell width-regularization* to achieve more realistic utilization (and faster routability assessment) as described in Section 4.3.5.

(3) **We demonstrate the PPAC exploration of scaling boosters.** We incorporate scaling boosters (BSPDN and BPR) to support P&R and IR drop analysis flows within the framework, as detailed in Section 4.3.3. Our results show that incorporating BSPDN and BPR leads to a reduction in power consumption by up to 8% and area by up to 24% based on our predictive 3nm technology. The area reduction results are consistent with those reported in previous industry works [55] [104] [138] [141], which have demonstrated area reductions of 25% to 30% through the use of BSPDN and BPR techniques.

Due to limited access to advanced technology for academic research, we build our predictive 3nm technology, named *the PROBE3.0 technology*. To calibrate the technology, we refer to the International Roadmap for Devices and Structures (IRDS) [156], open-sourced PDKs,

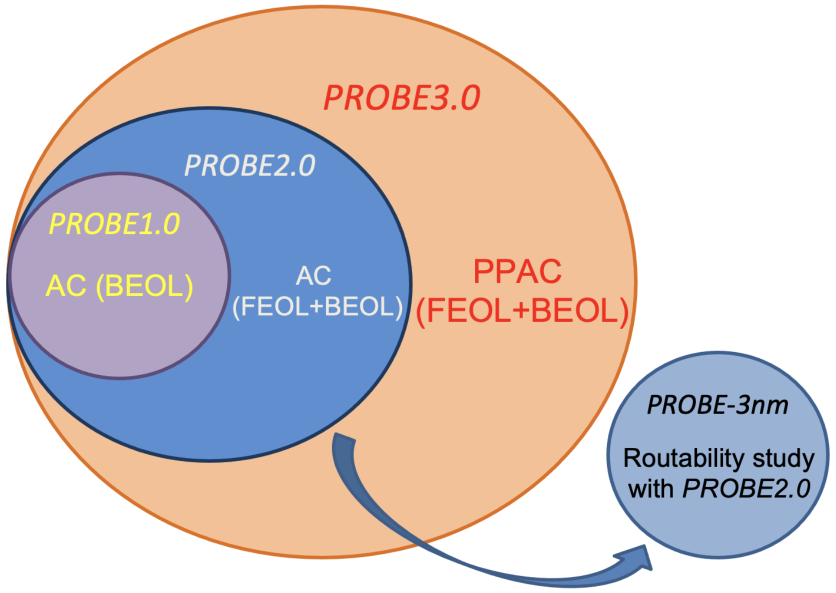


Figure 4.23. Scope of PROBE. *PROBE1.0* [64] and *PROBE2.0* [24] (Section 4.1 of this thesis) address AC given BEOL and FEOL/BEOL, respectively. *PROBE-3nm* [27] (Section 4.2 of this thesis) studies routability with sub-3nm configurations. PROBE3.0 provides true full-stack PPAC pathfinding.

and other publications [8] [29] [106] [150]. We open-source our work, including process design kits (PDKs), standard-cell libraries, and scripts for P&R and IR drop analysis; this is available in our GitHub repository [177]. In Section 4.3.2, we provide details on the automated PDKs and library generation flows, while in Section 4.3.6, we present three experiments to demonstrate the effectiveness of the PROBE3.0 framework for PPAC pathfinding.

4.3.1 Related Work

In this section, we divide the relevant previous works into the three categories of (i) advanced-technology research PDKs, (ii) design-technology co-optimization and (iii) scaling boosters, along with (iv) "PROBE" frameworks.

Advanced Technology Research PDKs. PDKs of advanced node technologies are highly confidential. Academic research can be blocked by limited access to relevant information. To unblock academic research, predictive advanced-node PDKs have been published. ASAP7 [29] is a predictive PDK for 7nm FinFET technology that includes standard cells which support commercial logic synthesis and P&R. FreePDK3 [106] [150] and FreePDK15 [8] are open-source PDKs for 3nm and 15nm technology. [77] proposes a 3nm predictive technology called NS3K with nanosheet FETs (NSFET). The authors of [77] also create 5nm FinFET and 3nm NSFET libraries to compare power, performance and area.

Design-Technology Co-Optimization. Previous DTCO works evaluate block-level PPAC and optimize design and technology simultaneously. [113] proposes UTOPIA to evaluate block-level PPAC with thermally limited performance, and to optimize device and technology parameters. [84] proposes a fast pathfinding DTCO flow for FinFET and complementary FET (CFET). [14] also proposes a fast and agile technology pathfinding platform with compact device models to accelerate the DTCO process. [63] describes power delivery network pathfinding for 3-D IC technology to study tradeoffs between IR drop and routability. [22] uses ML to predict sensitivities to changes for DTCO.

Scaling Boosters. Scaling boosters are used in advanced nodes to maximize benefit of new technology. BSPDN and BPR are among the most promising scaling boosters in sub-5nm nodes. [102] carries out a CPU implementation with BSPDN and BPR in their 3nm technology, demonstrating a reduction of up to 7X in worst IR drop. Similarly, [104] investigates BSPDN and BPR at sub-3nm nodes and finds that they can lead to a 30% reduction in area based on IR drop mitigation. [18] also explores the impact of BSPDN and BPR on design, concluding that their use can lead to a 43% reduction in area with 4X less IR drop. [55] studies BSPDN configurations with μTSVs, and observes 25% to 30% reduction in area using BSPDN and BPR. Additionally, [112] investigates BSPDN with nTSVs and μTSVs and finds that the average IR drop with BSPDN improves by 69% compared to traditional frontside PDN (FSPDN). Finally, [116] conducts holistic evaluations for BSPDN and BPR, demonstrating that FSPDN with BPR achieves a 25% lower on-chip IR drop, while BSPDN with BPR achieves an 85% lower on-chip IR drop with iso-performance and iso-area. In contrast to these previous DTCO works, here we propose a highly *configurable* framework that enables more efficient investigation of scaling boosters in advanced nodes.

“PROBE” Frameworks. Prior “PROBE” [24] [64] works propose systematic frameworks for assessing routability with different FEOL and BEOL configurations. Specifically, [64] begins with an easily-routable placement and increases the routing difficulty by random neighbor-swaps until the routing fails with greater than a threshold number of design rule violations (DRCs). The normalized number of swaps at which routing failure occurs, denoted by K_{th}, is a metric used to measure the inherent routability of the given parameters. On the other hand, [24] introduces an automatic standard-cell layout generation using satisfiability modulo theory (SMT) to support explorations of both FEOL and BEOL configurations. The authors of [24] also employ machine learning (ML)-based K_{th} prediction to expedite the DTCO pathfinding process. The work of [24] corresponds to Section 4.1 of this thesis. Additionally, [27] employs PROBE2.0 in a routability study with sub-3nm technology configurations; this work corresponds to Section 4.2 of this thesis.

4.3.2 Standard-Cell Library and PDK Generation

Expediting the DTCO process requires automation of the standard-cell library and PDK generation flows. Therefore, the PROBE2.0 framework [24] introduces standard-cell layout and PDK generation flows and utilizes them for routability assessments. In this work, we extend the PROBE2.0 framework to include proper electrical models of standard-cell libraries and interconnect layers for design-technology pathfinding. Additionally, we enhance the PDK generation flow to support advanced nodes. While the PROBE2.0 framework solely focuses on the physical layout of standard cells, the PROBE3.0 framework enables true full-stack PPAC pathfinding through automated, configurable standard-cell and PDK generation flows for advanced nodes. To demonstrate use of PROBE3.0 for advanced-node PPAC pathfinding, we use a technology that incorporates cutting-edge (3nm FinFET) technology predictions based on the works of [29] [156].

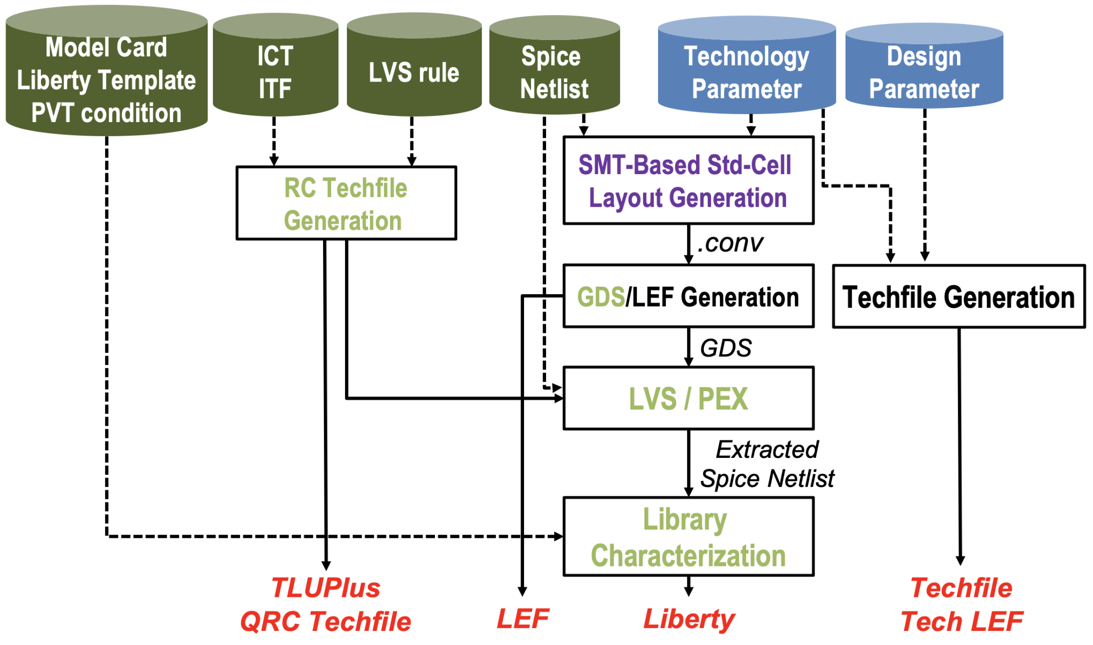


Figure 4.24. Automatic standard-cell library and PDK generation (*Design Enablement*) in the PROBE3.0 framework. In addition to technology and design parameters in the PROBE3.0 framework, other technology-related inputs are required.

Overall Flow

Figure 4.24 describes our overall flow of standard-cell and PDK generation. Technology and design parameters are defined as input parameters for the flow. Beyond these input parameters, there are additional inputs required to generate standard-cell libraries and PDKs, as follows: (i) SPICE model cards, (ii) Liberty template and PVT conditions, (iii) Interconnect technology files (ICT/ITF), (iv) LVS rule deck, and (v) SPICE netlists. Given the inputs, our SMT-based standard-cell layout generation and GDS/LEF generation are executed sequentially. Generation of timing and power models (Liberty) requires additional steps including LVS, parasitic extraction and library characterization flow. Aside from the standard-cell library generation, we also generate interconnect models from ICT/ITF, and P&R routing technology files from technology and design parameters. The PDK elements that we generate feed seamlessly into commercial logic synthesis and P&R tools. Further, to the best of our knowledge, ours is the first-ever work that is able to disseminate all associated EDA tool scripts for research purposes.

Table 4.18. Layer definition in the PROBE3.0 technology.

Layer	Name	Description
FEOL	WELL	N-Well
	FIN	Fin.
	GATE	Poly (gate).
	GCUT	Gate cut.
	ACTIVE	Active area for fin definition.
	NSELECT	N-implant.
	PSELECT	P-implant.
	CA	Contact (via) between LIG/LISD and M0.
	LIG	Gate interconnect layer.
	LISD	Source-drain interconnect layer.
	SDT	Source-drain trench (ACTIVE to LIG/LISD).
	BOUNDARY	Boundary layer for P&R.
BEOL	M0-M13	Metal layers.
	V0-V12	Via layers.

PROBE3.0 Technology

We build our own predictive 3nm technology node, called *the PROBE3.0 technology*. Based on [29], we define our FEOL and BEOL layers as described in Table 4.18. We assume that all BEOL layers are unidirectional routing layers. Hence, we first change M1 to a unidirectional routing layer with vertical preferred direction, since the work of [29] has a bidirectional M1 routing layer. We add an M0 layer with horizontal preferred direction below the modified M1 layer, and we add contact layers V0 and CA which respectively connect between M1 and M0, and between gate/source-drain and M0.

Also, electrical features of technologies are critical to explore "PP" aspects. Therefore, parasitic extractions of standard cells and BEOL metal stacks are important steps. To extract parasitic elements, interconnect technology files are required to use commercial RC extraction, P&R and IR drop analysis tools. In this work, we use commercial tools [147] [168] [173] for extractions, and each tool has its own technology file format.[23] Interconnect technology

[23]The file formats for each tool are unique. The MIPT file format is for *Siemens Calibre* [168] for extraction, and is converted to an RC rule file for standard-cell layout extractions. On the other hand, the ICT and ITF file formats are for *Cadence* and *Synopsys* extraction tools, respectively. We convert ICT to QRC techfile, and ITF to TLUPlus file, to enable P&R tools and IR drop analysis.

Table 4.19. Key features of the PROBE3.0 technology.

Layer	Feature	Value
FEOL	Fin pitch	24nm
	Fin width	6nm
	Gate pitch (CPP)	45nm
	Gate width	16nm
	Standard-cell height	100 / 120 / 144nm
	Dielectric constant	3.9
BEOL	Aspect ratio (width/thickness)	1.5
	Power/ground pin width (M0)	36nm
	M0/M2/M3 pitch	24nm
	M1 pitch	30nm
	M4-M11 pitch	64nm
	M12-M13 / BM1-BM2 pitch	720nm
	V0-V3 via resistance	50ohm/via
	V4-V11 via resistance	5ohm/via
	V12 via resistance	0.06294ohm/via
	Dielectric constant	2.5-3

files include layer structures of technology and electrical parameters, such as thickness, width, resistivity, dielectric constant and via resistance. We refer to the values of physical features in the 3nm FinFET technology of [156], such as fin pitch, fin width, gate pitch, gate width, metal pitch and aspect ratio. We also refer to [156] for the values for electrical parameters such as via resistance and dielectric constant. Table 4.19 describes key features of the PROBE3.0 technology.

Improved Standard-Cell Library Generation

We generate standard-cell libraries via several steps illustrated in Figure 4.24: (i) SMT-based standard-cell layout generation, (ii) generation of GDS and LEF files, (iii) LVS and PEX flow, and (iv) library characterization flow.

SMT-Based Standard-Cell Layout Generation. In recent technology nodes, standard-cell architectures use a variety of pitch values for different layers in order to optimize power, performance, area and cost (PPAC). To accommodate this, PROBE3.0 improves the SMT-based layout

Table 4.20. List of 41 standard cells per generated library.

Cell list	Size
Inverter (INV), Buffer (BUF)	X1, X2, X4, X8
2-input AND/OR/NAND/NOR (AND2/OR2/NAND2/NOR2)	X1, X2
3-input AND/OR/NAND/NOR (AND3/OR3/NAND3/NOR3)	X1, X2
4-input NAND/NOR (NAND4/NOR4)	X1, X2
2-1 AND-OR-Inverter (AOI21), 2-2 AND-OR-Inverter (AOI22)	X1, X2
2-1 OR-AND-Inverter (OAI21), 2-2 OR-AND-Inverter (OAI22)	X1, X2
D flip-flop (DFFHQN), D flip-flop with reset (DFFRNQ)	X1
2-input MUX/XOR (MUX2/XOR2), Latch (LHQ)	X1

generation used in PROBE2.0 to support non-unit gear ratios for M1 pitch (M1P) and contacted poly pitch (CPP).

Our standard-cell layouts are generated using SPICE netlists, technology and design parameters from [24]. However, in PROBE3.0 we change two key parameters: metal pitch (MP) and power delivery network (PDN). Instead of using MP, we define parameters for pitch values of each layer. Since M0, M1 and M2 layers are used for standard-cell layouts, we define M0P, M1P and M2P as pitches of M0, M1 and M2 layers, respectively. Table 4.25 shows four layouts of AND2_X1 cells with four parameter settings. The four standard-cell libraries (*Lib1*, *Lib2*, *Lib3* and *Lib4*) along with their corresponding parameter sets are used for our experiments in Section 4.3.6. For our PPAC exploration, we generate 41 standard cells for each standard-cell library as shown in Table 4.20.

GDS/LEF Generation and LVS/PEX Flow. While [24] only supports LEF generation for P&R, PROBE3.0 generates standard-cell layouts in both GDS and LEF formats. The GDS files are used to extract parasitics from standard-cell layouts and check LVS between layouts and schematics. We use *Calibre* [168] to check LVS and generate extracted netlists for standard cells with intra-cell RC parasitics. Scripts for GDS/LEF generation and LVS/PEX flows are open-sourced in [177].

Library Characterization Flow. We perform library characterization to generate standard-cell

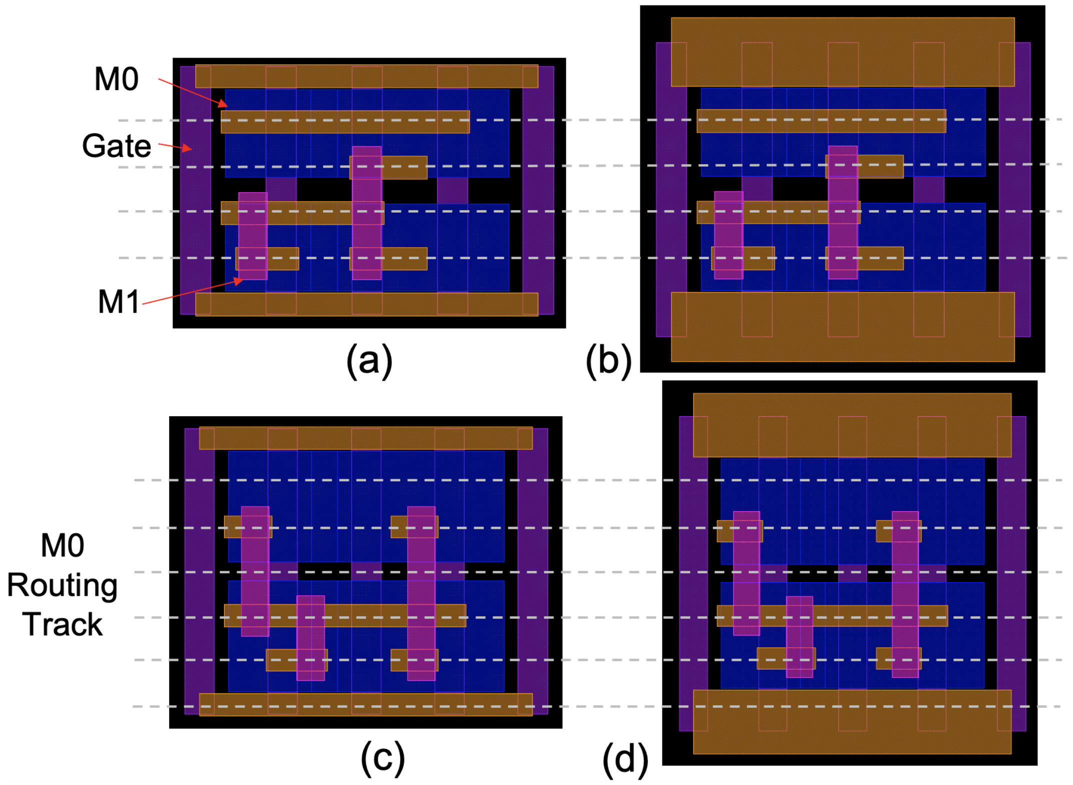


Figure 4.25. Example standard cells (AND2_X1). The cells are generated by our standard-cell layout generation with the following parameters (Fin, RT, PGpin, CH): (a) *Lib1* (2Fin, 4RT, BPR, 5T), (b) *Lib2* (2Fin, 4RT, M0, 6T), (c) *Lib3* (3Fin, 5RT, BPR, 6T) and (d) *Lib4* (3Fin, 5RT, M0, 7T).

libraries in the Liberty format. The inputs to the flow are model cards for FinFET devices, Liberty template including PVT conditions, and interconnect technology files. We use model cards from [150]. For the Liberty template, we define the PVT conditions, and the capacitance and transition time indices of (7×7) tables for electrical models (delay, output transition time, and power). We use 5, 10, 20, 40, 80, 160, and 320ps as the transition time indices. For the input capacitance, we obtain the input pin capacitance C_{inv} of an X1 inverter, then multiply this value by predefined multipliers, 2, 4, 8, 16, 24, 32, and 64. For characterization, we use the PVT corner (TT, 0.7V, 25°C).

4.3.3 Power Delivery Network

We study PDN scaling boosters to showcase the DTCO and pathfinding capability of PROBE3.0. There are two key challenges of traditional PDNs at advanced technologies:

- *High resistance of BEOL* [89]: Elevated resistance in BEOL layers exacerbates IR drop issues, necessitating denser PDN topologies.
- *Routing overheads (routability)* [114]: PDN occupies routing resources that are shared with signal and clock distribution. The routability and area density impact of PDN becomes more severe with denser PDN at advanced nodes.

To overcome these challenges, multiple foundries have begun implementing backside power delivery networks (BSPDN) and buried power rails (BPR) as scaling boosters in their sub-5nm technologies. We use these scaling boosters, BSPDN and BPR, to demonstrate use of PROBE3.0. We establish four options for PDN parameter in the PROBE3.0 framework: (i) Frontside PDN without BPR (P_{FS}); (ii) Frontside PDN with BPR (P_{FB}); (iii) Backside PDN without BPR (P_{BS}); and (iv) Backside PDN with BPR (P_{BB}). Figure 4.26 illustrates the four PDN configurations in the PROBE3.0 framework.

Frontside and Backside Power Delivery Network

We have defined realistic structures for both frontside power delivery networks (FSPDN) and backside power delivery networks (BSPDN), and enabled IR drop analysis within our framework. Table 4.21 shows the configurations for FSPDN and BSPDN. Since BEOL layers with smaller pitches (e.g., 24nm-pitch layer) have high resistance, we add power stripes for every layer. While the work of [24] has multiple options for FSPDN, the PROBE3.0 framework has only one PDN structure for FSPDN. Instead, we add other options such as P_{FB}, P_{BS} and P_{BB}. Furthermore, while the *Backside* option in [24] assumes no PDN at the frontside for the BSPDN option, we add power stripes at the backside for BSPDN in the PROBE3.0 framework to enable IR drop analysis for BSPDN.

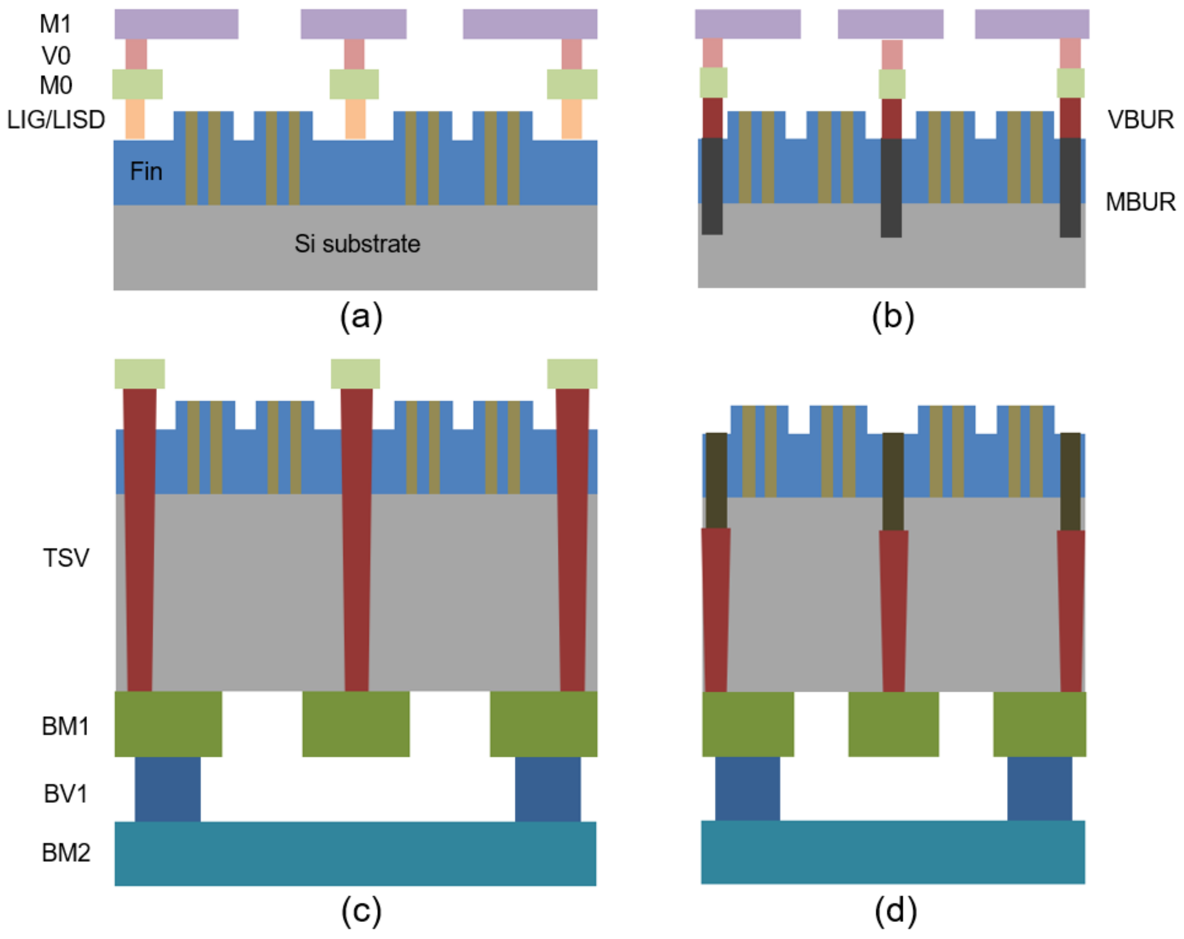


Figure 4.26. Cross-section view of four PDN options in the PROBE3.0 framework: (a) Frontside PDN (P_{FS}), (b) Frontside PDN with BPR (P_{FB}), (c) Backside PDN (P_{BS}) and (d) Backside PDN with BPR (P_{BB}).

Figures 4.26(a) and (c) respectively show cross-section views of P_{FS} and P_{BS} options. The P_{FS} option has M0 power and ground pins for standard cells, which connect to power stripes at the frontside of the die. The P_{BS} option uses the same M0 power and ground pins for standard cells but connects to power stripes at the backside of the die. For the P_{BS} option, we employ two backside metal layers (BM1 and BM2) and one via layer (BV1) between the backside metal layers. The layer characteristics (width, pitch and spacing) are identical to the top two layers (M12 and M13) of FSPDN. Additionally, the M0 pins of standard cells and BSPDN are connected using Through-Silicon Vias (TSVs). We assume nano-TSVs with 90nm [112] width for the P_{BS} option, and 1:10 width-to-height aspect ratio. For the P_{BS} option, TSV insertions necessitate

Table 4.21. PDN configurations for FSPDN and BSPDN. A pair of power (VDD) and ground (VSS) stripes are placed every pitch, while maintaining the spacing between VDD and VSS. *Density* denotes the percentage of routing tracks occupied by PDN.

PDN	**Layer**	**Pitch** (μm)	**Width** (μm)	**Spacing** (μm)	**Density** (%)
FSPDN	M3	1.08	0.012	0.508	4
	M4	1.152	0.032	0.544	11
	M5-M11	5.0	1.0	1.5	20
	M12-M13	4.32	1.8	0.36	100
BSPDN	BM1-BM2	4.32	1.8	0.36	100

reserved spaces in front-end-of-line (FEOL) layers, including keepout margins surrounding the TSVs. To accommodate this, we insert *power tap cells* prior to standard-cell placement.

Frontside and Backside PDN with Buried Power Rail

In advanced nodes, power rails on BEOL metal layers can be "buried" into FEOL levels with shallow-trench isolation (STI). Using deep trench and creating space between devices lowers the resistance of power rails. In addition to the resistance benefits, standard-cell height (area) can be further reduced with deep and narrow widths of power and ground pins. Figures 4.26(b) and (d) respectively show cross-section views of FSPDN with BPR (P_{FB}) and BSPDN with BPR (P_{BB}) options. In the case of P_{FB}, connections between FSPDN and BPR are made through nano-TSVs with the same 90nm width as in the P_{BS} option (but, with 1:7 aspect ratio). These nano-TSVs also necessitate insertion of reserved spaces.

Power Tap Cell Insertion

Although use of BSPDN and BPR can reduce area and mitigate IR drop problems, connecting frontside layers to BSPDN and/or BPR remains a critical challenge. To establish "tap" connections from frontside metals to BPR, or from backside to frontside metals, space must be reserved on device layers – e.g., [102] proposes power tap cells for the connection between BPR to MINT (M0) layers. More frequent "taps" will mitigate IR drop problems, but occupy

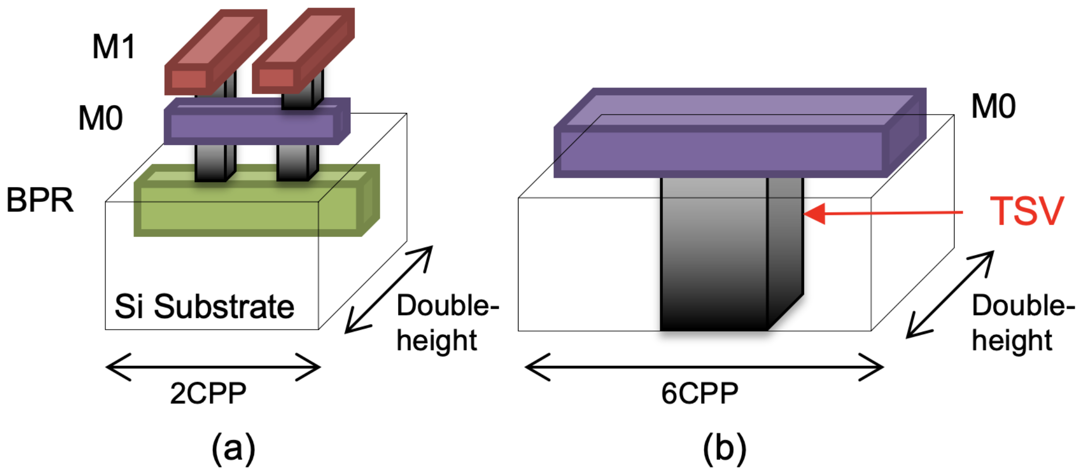


Figure 4.27. Power tap cells for (a) P_{FB} and (b) P_{BS}.

more placement area. In PROBE3.0, we define two types of power tap cells for the P_{FB} and P_{BS} options. Tap cells for P_{FB} connect BPR to M1, and tap cells for P_{BS} connect BM1 to M0. By contrast, P_{FS} and P_{BB} do not require power tap cells.

Power Tap Cell Structure. Figure 4.27(a) shows a structure of power tap cells for P_{FB}. Double-height power tap cells for P_{FB} have 2CPP cell width. The connection between BPR and M0 is through a $1{\times}2$ via array, and the two M1 metals are aligned with M1 vertical routing tracks. There are also two types of power tap cells for P_{FB} according to starting power and ground pins: power/ground pins on the double-height power tap cells are ordered as Power-Ground-Power (VDD-VSS-VDD) or Ground-Power-Ground (VSS-VDD-VSS). On the other hand, Figure 4.27(b) shows a structure of power tap cells for P_{BS}. While power tap cells for P_{FB} have 2CPP width, double-height power tap cells for P_{BS} have 6CPP width due to the $\sim$90nm width of nano-TSVs [112]. We also assume a 50nm keepout spacing around nano-TSVs. Similar to power tap cells for P_{FB}, there are two types of double-height power tap cells for P_{FB}, Power-Ground-Power and Ground-Power-Ground.

Power Tap Cell Insertion Scheme. Power tap cell insertion affects routability and IR drop, and hence affect PPAC of designs. In this work, we define five tap cell insertion pitches and two power tap insertion schemes, as follows.

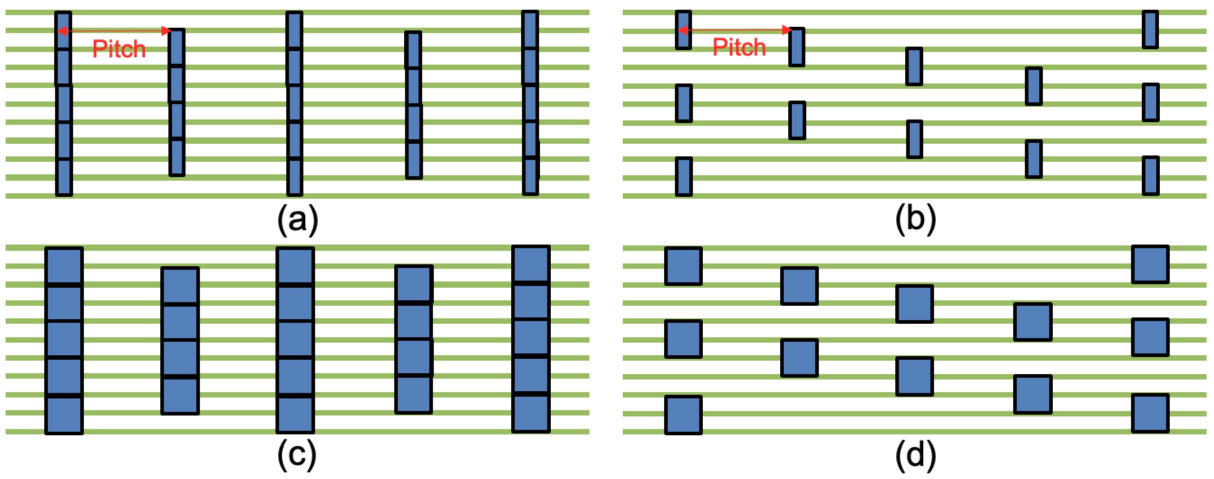


Figure 4.28. Four power tap cell insertion results: (a) Power tap cells for P_{FB} (2CPP width) with *Column*; (b) power tap cells for P_{FB} with *Staggered*; (c) power tap cells for P_{BS} (6CPP width) with *Column*; and (d) power tap cells for P_{BS} with *Staggered*.

- I_{pitch}: 24, 32, 48, 96 and 128CPP
- I_{scheme}: *Column* and *Staggered*

I_{pitch} and I_{scheme} denote tap cell insertion pitch and tap cell insertion scheme, respectively. Tap cell insertion scheme *Column* places double-height power tap cells on every two placement rows with the given tap cell pitch. Conversely, tap cell insertion scheme *Staggered* places double-height power tap cells on every four placement rows with the given tap cell pitch. Figure 4.28 shows four power tap cell insertion results for P_{FB} and P_{BS} with *Column* and *Staggered* insertion schemes.

IR Drop Analysis Flow

We develop two IR drop analysis flows for FSPDN and BSPDN. Figure 4.29(a) presents our IR drop analysis flow for FSPDN. After P&R, we generate DEF and SPEF files for routed designs using a commercial P&R tool to perform standalone vectorless dynamic IR drop analysis. Additionally, an interconnect technology file (QRC techfile) is needed for RC extraction as input for the IR drop analysis flow. In contrast, Figure 4.29(b) depicts our IR drop analysis flow for BSPDN. After P&R, we only create a SPEF file from routed designs. We then remove all routed

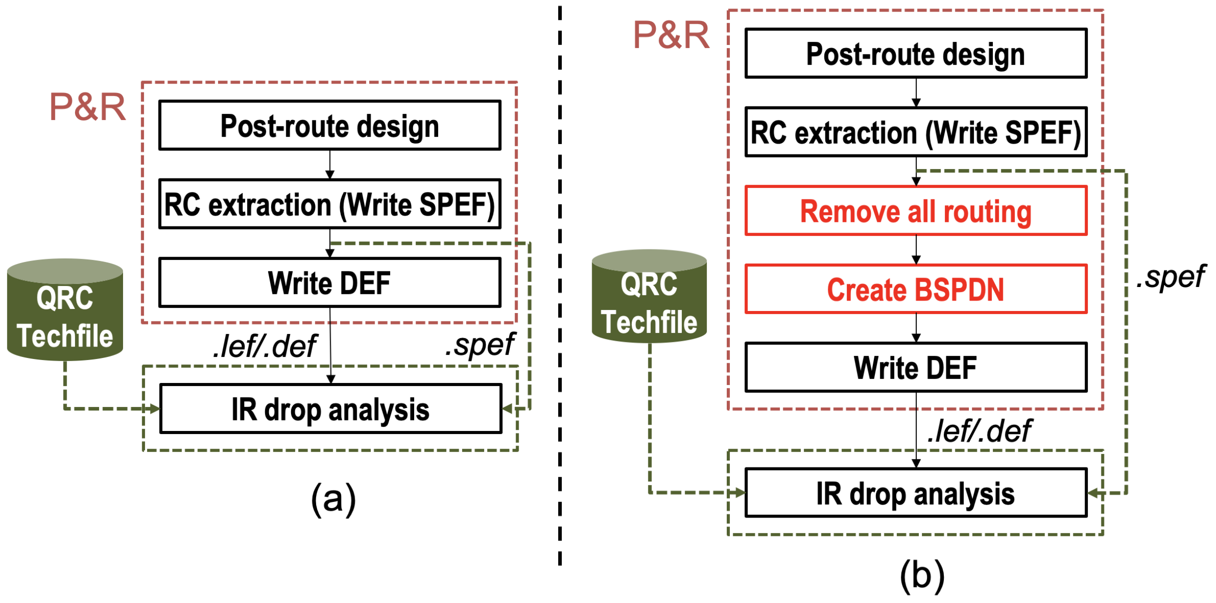


Figure 4.29. IR drop analysis flow for (a) FSPDN and (b) BSPDN. For the IR drop flow for BSPDN, we delete all the signal and clock routing after P&R and build power stripes for BSPDN.

signals and clocks from the P&R database and construct new power stripes for BSPDN. Since the standalone IR drop analysis tool obtains power stripe information from a DEF file, we generate a DEF file after creating power stripes on the backside. There are two backside metal layers, BM1 and BM2. When creating PDN on backside metal layers, we consider M1 as BM1 and M2 as BM2, respectively. For RC extraction with BSPDN, the QRC techfile must be scaled for backside metals since we assume BM1 and BM2 have the same pitches as M12 and M13. Full details are visible in open-source scripts at [177].

4.3.4 Enhanced Artificial Designs for PPAC Exploration

The use of specific real designs in DTCO and PPAC exploration can bring risk of biases and incorrect decisions regarding technology configurations (e.g., cell architecture or BEOL stack). To avoid such biases, the PROBE1.0 [64] bases its routability assessment on a mesh-like netlist topology, and PROBE2.0 [24] similarly uses a knight's tour-based topology. However, these artificial topologies have two main limitations as we bring "PP" aspects of PPAC into the picture. First, they are highly regular and cannot capture a wide range of circuit types.

Table 4.22. Definition of topological parameters in ANG [78] [140].

Parameter	Definition
N_{inst} (T_1)	Number of instances.
N_{prim} (T_2)	Number of primary inputs/outputs.
D_{avg} (T_3)	Average net degree. The net degree of a net is the number of terminals of the net.
B_{avg} (T_4)	Average size of net bounding box. The placed (or routed) layout is divided into a bin grid where each bin contains $\sqrt{N_{inst}}$ instances.
T_{avg} (T_5)	Average depth of timing paths. The depth of a given timing endpoint is the maximum number of stages in any fanin combinational path of that endpoint. T_{avg} is the average of all endpoint depths.
S_{ratio} (T_6)	Ratio of the number of sequential cells to the total number of cells. S_{ratio} equals to number of sequential cells over total number of instances.

Second, they do not mimic timing and power properties of real netlists, as they target routability assessment without regard to timing path structure.

PROBE3.0 overcomes these limitations by generating artificial but realistic netlists with the *Artificial Netlist Generator* (ANG) of [78] [140], for use in PPAC studies. We use the six topological parameters of ANG (see Table 4.22) to generate and explore circuits with various sizes, interconnect complexity, routed wirelengths and timing. Moreover, we apply machine learning (AutoML) to improve the match of generated artificial netlists to targeted (real) netlists.

Comparison of ANG and Real Designs

In this section, we study four real designs from OpenCores [161] and the corresponding artificial netlists generated by ANG [78]. Each design is taken through commercial logic synthesis and P&R tools [171] [172] in the PROBE3.0 technology, to obtain a final-routed layout. For AES, JPEG, LDPC and VGA, we respectively use target clock periods of 0.2ns, 0.2ns, 0.6ns and 0.2ns, and utilizations of 0.7, 0.7, 0.2 and 0.7. We then extract the six topological parameters from the routed designs and use these parameters to generate artificial netlists with ANG.

We introduce a *Score* metric to quantify similarity between artificial and real netlists, as defined in Equation (4.3).

Table 4.23. Topological parameters for real netlists from OpenCores [161] and artificial netlists generated by [78]. Design names followed by $*$ indicate ANG-generated artificial netlists.

Design	**Parameters**						**Score**
	N_{inst}	N_{prim}	D_{avg}	B_{avg}	T_{avg}	S_{ratio}	
AES	12318	394	3.28	0.55	7.98	0.04	-
JPEG	70031	47	3.09	0.21	10.36	0.07	-
LDPC	77379	4102	2.85	1.00	12.94	0.03	-
VGA	60921	185	3.71	0.42	8.25	0.28	-
AES*	10371	394	3.28	0.79	5.19	0.13	8.53
JPEG*	63185	47	3.16	0.70	6.97	0.15	12.03
LDPC*	58699	4106	3.10	0.78	6.96	0.13	14.8
VGA*	64412	188	3.32	0.26	6.39	0.25	2.8

$$Score = \Pi_{i=1}^{N} \max\left(\frac{T_i^{target}}{T_i^{out}}, \frac{T_i^{out}}{T_i^{target}}\right) \tag{4.3}$$

where: T_i^{target} $= T_i$ in target parameter set

T_i^{out} $= T_i$ of output parameter set

N $=$ Number of parameters ($N = 6$)

In Equation (4.3), target and output parameters are elements T_i^{target} and T_i^{out} of the target and output parameter sets. For each parameter, we calculate the discrepancy (ratio) between target and output values. The *Score* value is the product of these ratios. Ideally, if output parameters are exactly the same as target parameters, *Score* is 1. Larger values of *Score* indicate greater discrepancy between ANG-generated netlists and the target netlists.

Table 4.23 shows the input parameters, extracted parameters and *Score* metric in our comparison of real and artificial designs. The causes of discrepancy are complex, e.g., [78] has steps that heuristically adjust average depths of timing paths T_{avg} and the ratio of sequential cells S_{ratio}. Also, performing P&R will change the number of instances N_{inst}, the average net degree D_{avg}, and the routing which determines B_{avg}. Hence, it is difficult to identify the input parameterization of ANG that will yield artificial netlists whose post-route properties match those of (target) real netlists. We use machine learning to address this challenge.

Machine Learning-Based ANG Parameter Tuning

We improve the realism of generated artificial netlists with ML-based parameter tuning for ANG. Figure 4.30(a) shows the training flow in the parameter tuning. First, to generate training data, we sweep the six ANG input parameters to generate 21,600 combinations of input parameters, as described in Table 4.24. Second, we use ANG with these input parameter combinations to generate artificial gate-level netlists. Third, we perform P&R with the (21,600) artificial netlists and extract the output parameters. The extracted output parameters are used as output labels for the ML model training. We use the open-source H2O AutoML package [154] (version 3.30.0.6) to predict the output parameters; the *StackedEnsemble_AllModels* model consistently returns the best model. The model training is a one-time overhead which took 4 hours using an Intel Xeon Gold 6148 2.40GHz server. Executing commercial P&R required just over 7 days in our academic lab setting, and is again a one-time overhead.[24]

Figure 4.30(b) shows our inference flow. First, we define ranges around the target parameter and sweep the parameters to generate multiple combinations of input parameters as candidates, which are shown in Table 4.24. Second, we use our trained model to predict the output parameters from each input parameter combination. Note that although there are 12.3M combinations as specified in the rightmost two columns of Table 4.24, this step requires less than 10 minutes on an Intel Xeon Gold 6148 2.40GHz server.[25] Third, we calculate a predicted *Score* per each input parameter combination, and then choose the parameter combination with lowest predicted *Score*. Finally, we use ANG and the chosen parameter combination to generate an

[24]The average P&R runtime on our 21,600 ANG netlists is 0.4 hours on an Intel Xeon Gold 6148 2.40GHz server. The data generation used 50 concurrently-running licenses of the P&R tool, with each job running single-threaded. (21,600 $\times$ 0.4 / 50 / 24 $\sim=$ 7.2 days. With multi-threaded runs, we estimate that data generation would have taken 3 to 4 days.)

[25]$11 \times 11 \times 21 \times 21 \times 11 \times 21$ = 12,326,391. We apply simple filtering based on lower and upper bounds, to avoid parameter values for which ANG does not work properly. Specifically, parameter values are restricted to be within: $0 < B_{avg} \leq 1.0$; $0 < S_{ratio} \leq 1.0$; $1 < D_{avg} < 2.6$; and $3 < T_{avg}$. For example, the AES testcase then has $\sim$3M input parameter combinations, and predicting output parameters for all of these takes 441 seconds of runtime.

Table 4.24. Parameter sets for training and testing. We train our ML model with ANG input parameters and post-P&R output parameters. The total number of datapoints is $4 \times 6 \times 6 \times 5 \times 5 \times 6 = 21600$. Testing is performed in the ranges around given target parameters, according to the step sizes.

Parameter	**Training value**	**Testing value**	
		Range	**Step**
N_{inst} (T_1^{in})	10000, 20000, 40000, 80000	$T_1^{target} \pm 500$	100
N_{prim} (T_2^{in})	100, 200, 500, 1000, 2000, 4000	$T_2^{target} \pm 5$	1
D_{avg} (T_3^{in})	1.8, 2.0, 2.2, 2.4, 2.6	$T_3^{target} \pm 0.2$	0.02
B_{avg} (T_4^{in})	0.70, 0.75, 0.80, 0.85, 0.90, 0.95	$T_4^{target} \pm 0.2$	0.02
T_{avg} (T_5^{in})	6, 8, 10, 12, 14, 16	$T_5^{target} \pm 10$	2
S_{ratio} (T_6^{in})	0.2, 0.4, 0.6, 0.8, 1.0	$T_6^{target} \pm 0.2$	0.02

artificial netlist for P&R and PPAC explorations.

Table 4.25 shows the benefit from ML-based ANG parameter tuning. Columns 2-5 show parameters from real netlists, which we use as target parameters. The trained ML model and the inference flow produce the tuned parameters for ANG shown in Columns 6-9 of the table, and corresponding results are shown in Columns 10-13. The average *Score* decreases from to 4.89 from the original value of 8.87 for ANG without ML-based parameter tuning (Table 4.23).

The ML-enabled improvement of realism in ANG netlists can be seen using t-SNE visualization [91] from P&R results. We perform P&R for the four real designs by sweeping initial utilization from 0.6 to 0.8 with a 0.01 step size, and target clock period from 0.15 to 0.25ns with a 0.01ns step size; this results in 21 $\times$ 11 = 231 P&R runs. (For LDPC, we sweep utilization from 0.1 to 0.3 with a 0.01 step size, and clock period from 0.55 to 0.65ns with a 0.01ns step size.) We then perform P&R for artificial netlists with and without our parameter tuning flow, with 0.7 utilization (0.2 for LDPC) and 0.2ns (0.6ns LDPC) target clock period. Figure 4.31 shows t-SNE visualization[26] of the real and artificial designs. The 231 real datapoints per design form well-defined clusters. In Figure 4.31(a), the datapoints of the artificial AES

[26]For t-SNE visualization, we collect ten features from P&R results: Number of instances, number of nets, number of primary input/output pins, average fanout, number of sequential cells, wirelength, area, number of design rule violations, worst negative slack, total negative slack, and number of failing endpoints.

Table 4.25. Topological parameters for target, input and output netlists. The design names followed by $**$ indicate ANG-generated artificial netlists with ML-based ANG parameter tuning.

Parameter	**Parameters of target netlists**				**ANG input parameters**				**Parameters from artificial netlists**			
	AES	**JPEG**	**LDPC**	**VGA**	**AES**	**JPEG**	**LDPC**	**VGA**	**AES****	**JPEG****	**LDPC****	**VGA****
N_{inst}	12318	70031	77379	60921	12718	69531	76979	60421	10200	64296	64796	65113
N_{prim}	394	47	4102	185	390	42	4106	199	394	46	4110	202
D_{avg}	3.28	3.09	2.85	3.71	3.40	3.10	3.03	3.53	3.26	3.13	3.18	3.30
B_{avg}	0.55	0.21	1.00	0.42	0.49	0.31	1.98	0.28	0.72	0.21	0.73	0.36
T_{avg}	7.98	10.36	12.94	8.25	13.98	18.36	20.94	12.25	8.01	9.29	11.64	8.54
S_{ratio}	0.04	0.07	0.03	0.28	0.01	0.27	0.01	0.16	0.11	0.20	0.13	0.16
$Score$	-	-	-	-	-	-	-	-	4.39	3.59	2.77	8.81

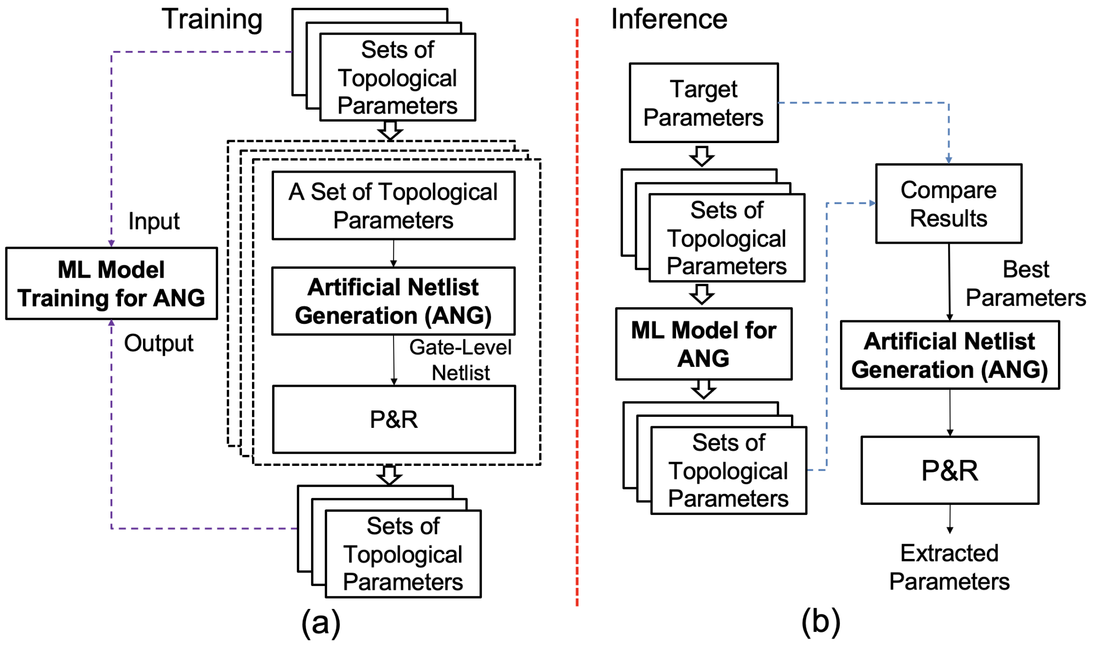

Training
Sets of Topological Parameters
A Set of Topological Parameters
Input
ML Model Training for ANG
Artificial Netlist Generation (ANG)
Output
Gate-Level Netlist
P&R
Sets of Topological Parameters
(a)
Inference
Target Parameters
Sets of Topological Parameters
Compare Results
Best Parameters
ML Model for ANG
Artificial Netlist Generation (ANG)
Sets of Topological Parameters
P&R
Extracted Parameters
(b)

Figure 4.30. ML-based parameter tuning for ANG.

and JPEG designs are located in the corresponding designs' clusters. However, the artificial LDPC and VGA designs are not close to the corresponding clusters of real designs. By contrast, Figure 4.31(b) shows that with our ML-based ANG parameter tuning, datapoints of all four artificial designs are located within the corresponding clusters of real designs. This suggests that the ML-based ANG parameter tuning helps create artificial netlists that better match targeted design parameters – including parameters that are relevant to PPAC exploration.

4.3.5 Cell Width-Regularized Placements for More Realistic Routability Assessment

Recall that in the PROBE approach, routability ("AC") is evaluated using the K-threshold (K_{th}) metric [64]. That is, given a placed netlist, routing difficulty is gradually increased by iteratively swapping random pairs of neighboring instances. The cell-swaps progressively "tangle" the placement until it becomes unroutable ($>$ 500 DRCs post-detailed routing). The number of swaps K – expressed as a multiple of the instance count – at which routing fails is the K_{th} metric. Larger K_{th} implies greater routing capacity or intrinsic routability.

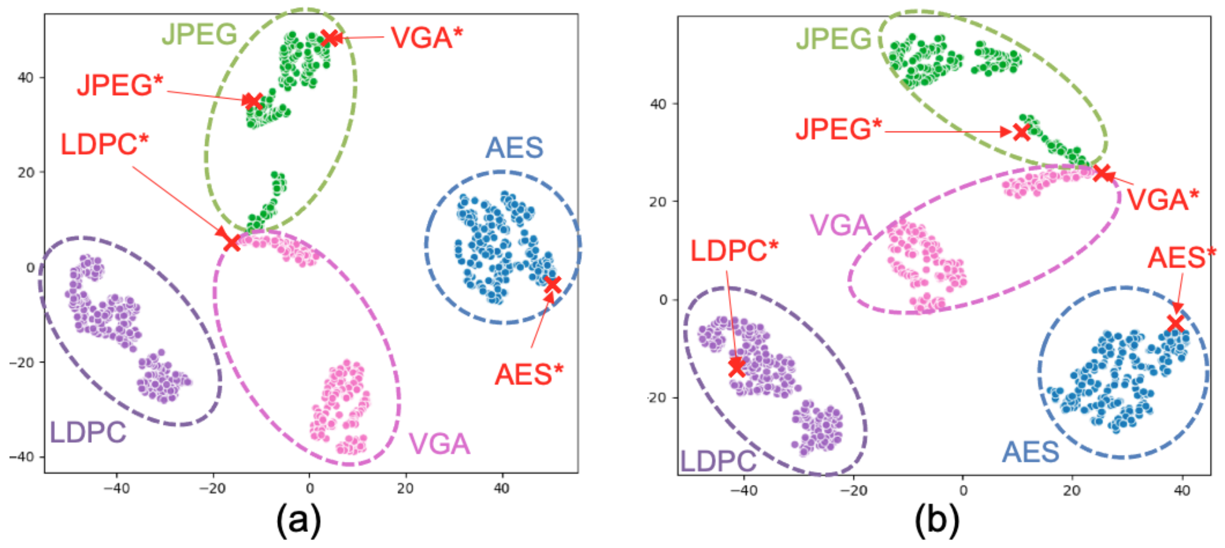


Figure 4.31. Comparison between real and artificial designs by t-SNE [91]. (a) t-SNE visualization for real and artificial (ANG, design names followed by $*$) designs **without** our parameter tuning flow. (b) Real and artificial (ANG) designs **with** our ML-based parameter tuning flow.

Both PROBE1.0 [64] and PROBE2.0 [24] enable the study of real netlists through the concept of a *cell width-regularized* placement. In this approach, combinational cells are inflated (by LEF modification) to match the maximum width among all the combinational cells in the cell library. This process, called *cell width-regularization*, prevents illegal placements (i.e., cell overlaps due to varying widths) from arising due to neighbor-swaps during K_{th} evaluation. Unfortunately, while cell width-regularization permits real designs to be placed and then tangled by random neighbor-swaps, it also forces low utilizations that harm the realism of the study. (Moreover, high whitespace leads to high K_{th} values that require more P&R runs to determine.)

We now describe a *clustering-based cell width-regularization* methodology that generates placements with realistic utilizations, based on real designs. Our experiments in Section 4.3.6 show that clustering-based cell width-regularization obtains the same K_{th} rank-ordering of design enablement, with less P&R expense, than the previous cell width-regularization approach.

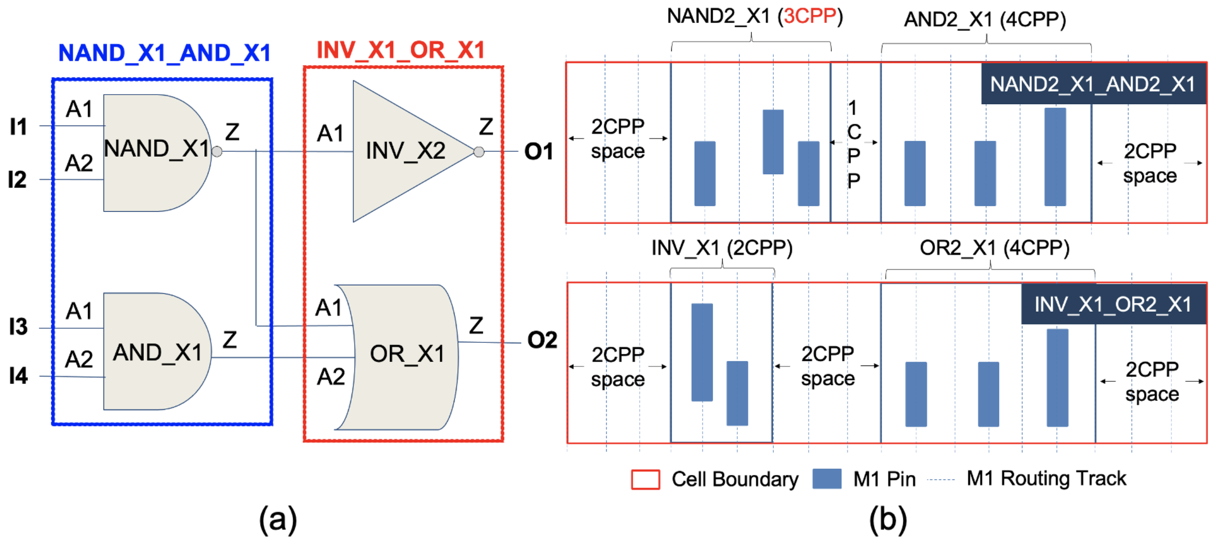


Figure 4.32. Two example clustered cells NAND_X1_AND_X1 and INV_X1_OR_X1, in clustering-based cell width-regularization. (a) Schematic view, and (b) physical layout view assuming *Lib2*. Here, the maximum clustered cell width w_{max} is 12CPP.

Clustering-Based Cell Width-Regularization

We propose *clustering-based cell width-regularization* using bottom-up hypergraph clustering, as detailed in Algorithm 7. In the following, we refer to standard cells of the original netlist as $cells_{orig}$, and (clustered) cells of the clustered netlist as $cells_{clustered}$.

Clustered Hypergraph Creation. For a given design, we first obtain a netlist hypergraph using OpenDB [162]. We perform *cell width-regularized clustering*, where cells $cells_{orig}$ (vertices) in the original netlist hypergraph are clustered such that *clustered cell width*[27] does not exceed w_{max}, the maximum cell width in the library. The inputs to cell width-regularized clustering are (i) a hypergraph $H(V,E,W)$ with vertices V, hyperedges E and cell widths W, (ii) the maximum cell width, w_{max}, and (iii) a limit on number of clustering iterations, N_{iter}.[28] The output is a clustered hypergraph (H_{out}). We use *First-Choice* (*FC*) clustering [74] and refer to our clustering method as *cell width-regularized clustering with FC*, or *CWR-FC*.

[27]Given vertex set V with cell widths W, clustering vertices $v_i, v_j \in V$ yields a clustered cell with width $W[v_i] + W[v_j]$.

[28]In our experiments, we set $N_{iter} = 20$. However, the cell width-regularized clustering is strongly constrained by w_{max}, and we observe on our testcases that clustering stops after ~ 3 iterations.

Algorithm 7. Cell width-regularization by clustering.

```
Inputs: Hypergraph $H(V,E,W)$, Maximum cell width $w_{max}$, Number of iterations $N_{iter}$
Outputs:, Clustered hypergraph $H_{out}(V_{out},E_{out},W_{out})$
 1: $N_{cluster} \leftarrow |V|$
 2: Hypergraph at iteration 0, $H_0(V_0,E_0,W_0) \leftarrow H(V,E,W)$
 3: for $k \leftarrow 0$; $k < N_{iter}$ ; $k++$ do
 4:     $V_{ordered} \leftarrow$ Sorted $V_k$ in increasing order of $W_k$
 5:     $visited[v] \leftarrow false \ \forall v \in V_k$
 6:     Cluster assignments, $cmap[v] \leftarrow v \ \forall v \in V_k$
 7:     Clustered cell widths, $W_{k+1} \leftarrow W_k$
 8:     for $v_i \in V_{ordered}$ do
 9:         if $visited[v_i] == true$ or $v_i$ is a sequential cell then
10:             continue
11:         end if
12:         $V_{neighbor} \leftarrow$ Find adjacent vertices of $v_i$
13:         Best cluster score, $\phi_{best} \leftarrow 0$; Best cluster candidate, $v_{best} \leftarrow -1$
14:         for $v_j$ in $V_{neighbor}$ do
15:             if $W_k[v_i] + W_{k+1}[cmap[v_j]] \leq w_{max}$ then
16:                 $\phi(v_i,v_j) \leftarrow \frac{\sum_{v_i \in e, v_j \in e} \frac{weight_e}{|e|-1}}{W_k[v_i]+W_{k+1}[cmap[v_j]]}$   // Cluster Score
17:                 if $\phi(v_i,v_j) > \phi_{best}$ then $v_{best} \leftarrow v_j$
18:                 end if
19:             end if
20:         end for
21:         if $v_{best} == -1$ then
22:             $W_{k+1}[v_i] \leftarrow W_k[v_i]$
23:             $visited[v_i] \leftarrow true$
24:         else
25:             $cmap[v_i] \leftarrow cmap[v_{best}]$
26:             $W_{k+1}[v_{best}] \leftarrow W_k[v_i] + W_{k+1}[cmap[v_{best}]]$
27:             $visited[v_i] \leftarrow true$; $visited[v_{best}] \leftarrow true$
28:             $N_{cluster} \leftarrow N_{cluster} - 1$
29:         end if
30:     end for
31:     if $N_{cluster} == |V_{k-1}|$ then
32:         break
33:     else
34:         $H_{k+1}(V_{k+1},E_{k+1},W_{k+1}) \leftarrow$ Build clustered hypergraph using $cmap$
35:     end if
36: end for
37: $H_c \leftarrow$ Clustered hypergraph generated at last iteration
38: $H_{out} \leftarrow$ Best-fit bin packing on $H_c$
39: Return $H_{out}$
```

CWR-FC first sorts vertices in increasing order of cell widths (Line 4) and initializes *cluster assignments* (Line 6). The cluster assignment *cmap* is the mapping of vertices to clusters (V_k to V_{k+1}). Clustered cell widths W_{k+1} are initialized in Line 7. Next, vertices are traversed in order to perform pairwise clustering; note that only combinational cells are considered for clustering (Line 8). For each vertex v_i that is traversed, we find its neighbors v_j in the hypergraph (Line 11). Each v_j is considered only if it does not violate the w_{max} limit (Line 14); a cluster score $\phi(v_i, v_j)$ is calculated in Line 15. In the cluster score, $weight_e$ is the weight of hyperedge e and $W_k[v_i]$ is the width of vertex v_i. The numerator aims to cluster vertices that are strongly connected (i.e., share many hyperedges) while the denominator promotes clusters of similar widths. If all neighboring vertices v_j violate the threshold width constraint, then no new clusters are formed (Lines 17-19). Otherwise, the vertex with the highest cluster score is selected, and a new cluster is created (Lines 21-24). After all vertices are visited, we construct the clustered hypergraph and proceed with subsequent iterations (Line 28). If no further clustering is feasible, the process terminates (Line 26).

Note that *CWR-FC* clusters vertices that are adjacent to each other in the hypergraph. However, if all pairings of vertices selected for clustering violate the w_{max} width constraint, the algorithm can stall (Line 25). To address this issue and improve the uniformity of cluster contents, we perform best-fit bin-packing [60] with bins having capacity w_{max} (Line 30).[29] Finally, the output is the clustered hypergraph H_{out}.

Clustered Netlist Creation. We convert the clustered hypergraph H_{out} into Verilog using OpenDB. Then, to run P&R we require a new LEF file that captures the cluster assignments from cell width-regularized clustering. I.e., we require a new netlist over the clusters, $cells_{clustered}$.

Figure 4.32(a) provides a schematic view of two clustered cells, NAND_X1_AND_X1 and INV_X1_OR_X1. These correspond to two clusters of original cells: NAND_X1 and AND_X1, and INV_X1 and OR_X1. Figure 4.32(b) shows how the clustered cells are composed from

[29]The choice of best-fit is motivated by its simplicity and intuitiveness. Best-fit also enjoys a better approximation ratio compared to first-fit or next-fit alternatives [60].

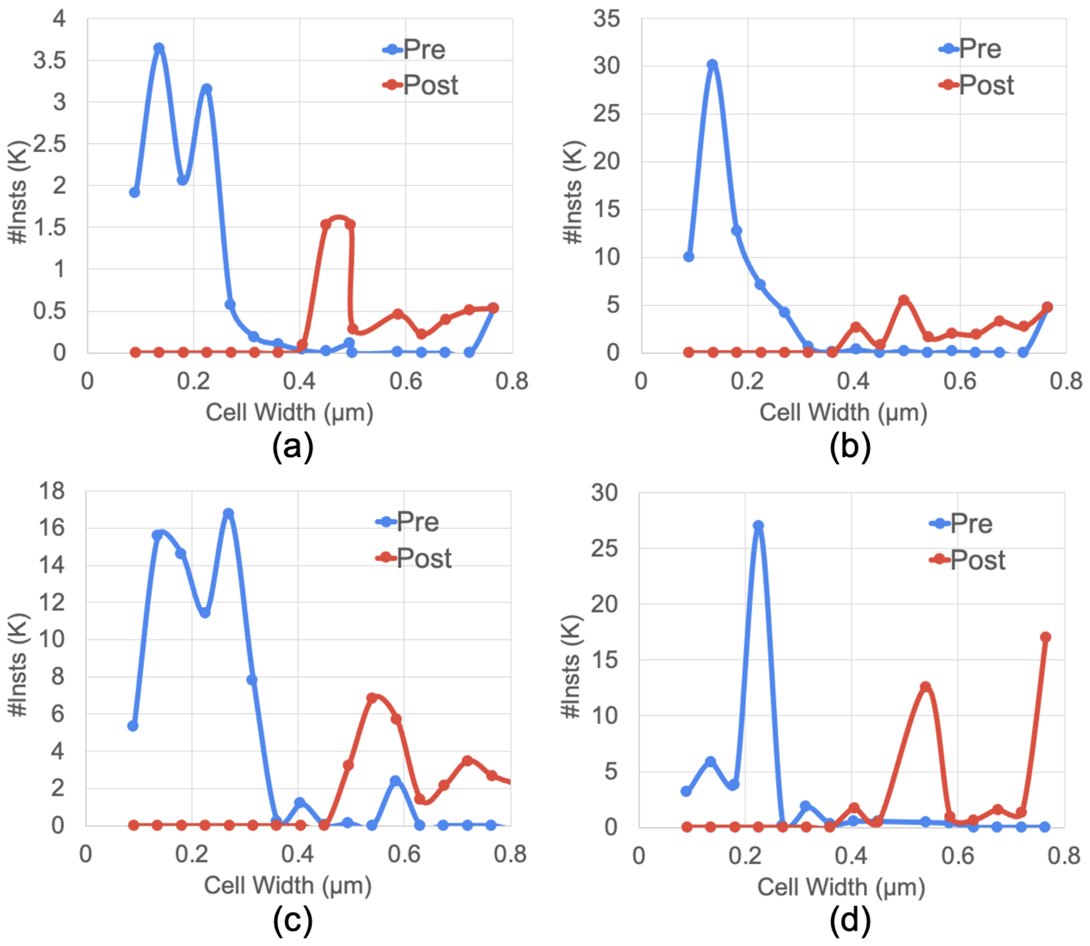


Figure 4.33. Cell width distributions pre-clustering (i.e., original netlist) and post-clustering (i.e., by *CWR-FC*) for (a) AES, (b) JPEG, (c) LDPC and (d) VGA.

original standard-cell layouts. In this case, a non-integer gear ratio between M1P (30nm) and CPP (45nm) forces cells in $cells_{clustered}$ to be positioned at even CPP sites, to avoid M1 pin misalignment. In the first cluster, NAND_X1 width (3CPP) is an odd number of CPPs, necessitating addition of 1CPP padding between the two cells. In the second cluster, the total cell width is less than w_{max}, so whitespace is included along with the clustered original cells. We distribute whitespace uniformly, (i) at the sides of $cells_{clustered}$ and (ii) between consecutive cells in each cluster, as illustrated in Figure 4.32(b). During this whitespace allocation, we first allocate whitespace at junctions (between consecutive original cells) where no extra padding has been previously allocated.

Performance of Clustered Cell Width-Regularization

We now document advantages of our proposed clustered cell width-regularization, i.e., more realistic utilization in P&R blocks, and realistic topological and wirelength characteristics of P&R outcomes.

Comparison to Previous Cell Width-Regularization. Figure 4.33 compares cell width distributions for instances in the clustered netlist and instances in the original netlist. The blue lines show the distribution of cell widths in the original netlist, where smaller cell widths predominate. The red lines indicate that *CWR-FC* increases the prevalence of cells with larger widths through creation of the merged $cells_{clustered}$. The larger amount of actual cell widths in $cells_{clustered}$ leads to smaller amounts of added whitespace needed to regularize cell widths.

As anticipated, *clustered* cell width-regularization significantly reduces whitespace in the placed designs. With *Lib2* and FSPDN for P&R, placing cell width-regularized instances used in PROBE2.0 at 90% density achieves actual utilizations of 0.21, 0.21 and 0.40 for AES, JPEG and VGA, respectively. For LDPC, placing cell width-regularized instances at 30% density achieves actual utilization of 0.08. By contrast, with clustered cell width-regularization, we achieve actual utilizations of 0.71, 0.74 and 0.71 for AES, JPEG and VGA, respectively. For LDPC, we achieve actual utilization of 0.23. In this way, our new methodology enables K_{th} evaluation by iterated neighbor-swapping while preserving realistic placement utilizations.

Topological and Wirelength Comparisons to Real Designs. We have confirmed additional similarities between between clustered cell width-regularized netlists and the original real designs. Table 4.26 compares characteristics of our clustering-based cell width-regularized netlists and placements ([C]), versus analogous characteristics of real netlists and placements ([A]). We also implement another plausible clustering methodology, which is to induce clusters from a placement of the original design ([B]). In [B], clusters from the placement are induced by (i) traversing combinational cells left-to-right in each standard cell row, and (ii) clustering maximal contiguous sets of cells without exceeding W_{max}.

Table 4.26. Comparison of the width-regularized clustered netlist produced by *CWR-FC* ([C]) with the original flat netlist ([A]) and a width-regularized clustered netlist induced from a placement of the flat netlist ([B]).

Stage	Design	#Insts	Area (μm^2)	Util	WL (μm)	Avg. FO
[A]	AES	12318	426.254	0.83	30849	2.32
	JPEG	70031	2781.981	0.73	112605	2.15
	LDPC	77379	6250.563	0.43	567630	1.85
	VGA	60921	4238.205	0.76	208845	2.71
[B]	AES	4275	426.254	0.83	32632	1.96
	JPEG	23281	2781.981	0.73	111241	1.86
	LDPC	42383	6250.563	0.43	585923	1.43
	VGA	40084	4238.205	0.76	189612	2.14
[C]	AES	4661	426.254	0.83	32679	2.08
	JPEG	25961	2781.981	0.73	143693	1.76
	LDPC	30636	6250.563	0.43	637417	1.29
	VGA	32768	4238.205	0.76	220915	2.02

We run P&R using *Lib2* and P_{FS} for PDN, maintaining the same core area and utilization. Clustering decreases the number of instances and average fanouts for [B] and [C], relative to [A]. However, wirelengths exhibit no significant changes. The similarities between [A], [B] and [C] suggest that our *CWR-FC* methodology can preserve netlist properties relevant to P&R outcomes, with more realistic placement utilizations.

4.3.6 Experimental Setup and Results

We extensively study the design-technology pathfinding capability of the PROBE3.0 framework using the PROBE3.0 technology. In this section, we report three main experiments. Expts. 1 and 2 show PROBE3.0's capability to assess PPAC trends and tradeoffs, using real and artificial designs respectively. Expt. 3 performs assessments of routability and achievable utilization.

In Expts. 1 and 2, we analyze four tradeoffs. (i) We present *Performance-Power* plots that quantify tradeoffs between performance (maximum frequency) and power. (ii) We present *Performance-Area* plots to quantify the tradeoffs between performance and area. (iii) To address

PP aspects, we use the *Energy-Delay Product* (EDP) [84] as a single metric for power and performance. *EDP-Area* plots depict tradeoffs between performance/power and area. (iv) We present *IR drop-Area* plots to demonstrate tradeoffs between IR drop and area. We also compare results obtained using artificial designs with those obtained using real designs. Expt. 3 assesses routability and achievable utilization using our clustering-based cell width-regularized placements.

Experimental Setup

Based on the definition of technology and design parameters in [24], we define ten technology parameters and eight design parameters as the input parameters for the PROBE3.0 framework. Table 4.27 describes the definitions of these parameters and the options used in our experiments. Also, we use commercial tools for PDK generation, logic synthesis, P&R, and IR drop analysis. We use open-source tools for GDT-to-GDS translation [151] and SMT solver [178]. Table 4.28 summarizes the tools and versions that we use in our experiments.

Criteria for Valid Result. In our experiments, for given Design, PDN and technology parameters, we perform logic synthesis, P&R and IR drop analysis with multiple sets of parameters including I_{pitch}, I_{scheme}, $Util$ and $Clkp$. We use 24, 32, 48, 96 and 128CPP for I_{pitch}, and *Column* and *Staggered* for I_{scheme}. For Util, we use values ranging from 0.70 to 0.94 with a step size of 0.02, and for Clkp, we use values ranging from 0.12 to 0.24ns with a step size of 0.02ns. Importantly, after the implementation and the analysis steps, we filter out results that are deemed invalid – in that they are likely to fail signoff criteria even with additional human engineering efforts.

To be precise, a "valid" result must satisfy three conditions: (i) the worst negative slack is larger than -50ps; (ii) the number of post-route DRCs is less than 500; and (iii) the 99.7 percentile of the effective instance voltage is greater than 80% of the operating voltage (V_{op}). To assess (iii), we use a commercial IR drop analysis tool [148] to measure vectorless dynamic IR drop, and calculate the effective instance voltage as $V_{op} - V_{drop}$ per each instance, where V_{op} is an operating voltage (0.7V) and V_{drop} is the worst voltage drop per instance. We take the 99.7

Table 4.27. Technology and design parameters in our experiments.

Type	Parameter	Description	Option
Technology	Fin	The number of fins for devices of standard cells.	2, 3
	CPP	Contacted poly pitch for standard cells in nm.	45
	M0P	M0 (horizontal) layer pitch in nm.	24
	M1P	M1 (vertical) layer pitch in nm.	30
	M2P	M2 (horizontal) layer pitch in nm.	24
	RT	The number of available M0 routing tracks in standard cells.	4, 5
	PGpin	Power/ground pin layer for standard cells.	BPR, M0
	CH	Cell height of standard cells, expressed as a multiple of M0P. When the cell height is 120nm and M0P is 24nm, the cell height (CH) is 5. The cell height value is calculated as RT + 2 for M0 PGpin and RT + 1 for BPR PGpin.	5, 6, 7
	MPO	The number of minimum pin openings (access points).	2
	DR	Design rules. We define the same grid-based design rules, minimum area rule (*DR-MAR*), end-of-line spacing rule (*DR-EOL*) and via spacing rule (*DR-VR*) as [24]. We use the *EUV-tight* (ET) design rule set, which includes *DR-MAR* = 1, *DR-EOL* = 2 and *DR-VR* = 1.	*EUV-Tight*
Design	BEOL	Metal stack options. The 14M metal option which contains 14 metal layers (M0 to M13). We define 1.2X, 2.6X, 3.2X and 30X layer pitches based on 24nm as the 1X pitch.	14M
	PDN	Power delivery network options.	P_{FS}, P_{FB}, P_{BS}, P_{BB}
	I_{pitch}	Power tap cell pitch in CPP.	24, 32, 48, 96, 128
	I_{scheme}	Power tap cell insertion scheme.	*Column*, *Staggered*
	Tool	Commercial P&R tools.	*Synopsys IC Compiler II*
	Util	Initial placement utilization.	0.70 to 0.94 with 0.02 step size
	Design	Designs studied in our experiments: four open-source designs from OpenCores [161] and artificial netlists generated by ANG with our ML-based parameter tuning.	AES, JPEG, LDPC, VGA
	Clkp	Target clock period for logic synthesis and P&R. Clkp values reflect maximum achievable frequencies of the designs.	0.12 to 0.24ns with 0.02ns step size

Table 4.28. Tools and versions in our experiments.

Purpose	Tool	Version	Ref.
Format conversion	*GDT-to-GDS translator*	4.0.4	[151]
IR drop analysis	*Cadence Voltus*	19.1	[148]
Library characterization	*Cadence Liberate*	16.1	[145]
Logic synthesis	*Cadence Genus*	21.1	[143]
	Synopsys Design Compiler	R-2020.09	[171]
LVS	*Siemens Calibre*	2017.4_19	[168]
P&R	*Synopsys IC Compiler-II*	R-2020.09	[172]
PEX	*Cadence QRC Extraction*	19.1	[147]
	Synopsys StarRC	O-2018.06	[173]
	Siemens Calibre	2017.4_19	[168]
SMT solver	*Z3*	4.8.5	[34] [178]

percentile of effective instance voltage as representative of IR drop for the post-P&R result, as it is within three standard deviations from the mean per the empirical rule [137].

Expt. 1: PPAC Exploration with Real Designs

Performance versus Power. We first present PPAC explorations that show tradeoffs between performance and power. (We assume that area is proportional to cost, since chip area is closely related to cost.) In this study, we show results for JPEG with four standard-cell libraries (*Lib1-4*). Also, we use four PDN structures, P_{FS}, P_{FB}, P_{BS} and P_{BB}, and measure improvements due to scaling boosters relative to the traditional frontside PDN (P_{FS}).

Figure 4.34(a) gives *Performance-Power* plots that show tradeoffs between performance and power for JPEG, and improvements from the traditional FSPDN. We calculate the maximum achievable frequency (f_{max}) as $1/(Clkp - WNS)$ where $Clkp$ is the target clock period and WNS is the worst negative slack. Also, we add up leakage and dynamic power to obtain the total power. To measure the improvement from P_{FS}, we compare the second-largest value (on the x-axis) attained with each PDN configuration. From the result, we make two main observations. (i) Power consumption with P_{BS} and P_{BB} decreases by 7 to 8%, compared to P_{FS} with the same performance. (ii) Power consumption with P_{FB} is similar to P_{FS}, with the same performance. We observe power reductions from use of scaling boosters, BSPDN and BPR. However, use of BPR without BSPDN does not reduce power consumption.

Performance versus Area. Figure 4.34(b) shows *Performance-Area* tradeoffs for JPEG. We make two main observations. (i) Area with P_{FB}, P_{BS} and P_{BB} decreases by up to 8%, 5% and 24%, respectively, as compared to P_{FS}, while maintaining the same level of performance. (ii) We find that use of scaling boosters results in area reductions across all four standard-cell libraries. The area reduction results obtained using the PROBE3.0 framework are consistent with previous industry works [55] [104] [138] [141], which show that use of BSPDN and BPR techniques can result in area reductions of 25% to 30%.

Energy-Delay Product (EDP) versus Area. Given the tradeoffs among PPAC criteria, a simpler metric is useful to comprehend multiple aspects simultaneously. The *Energy-Delay Product* (EDP) is adopted by, e.g., [84] as a single-value metric that captures both power efficiency and maximum achievable frequency (performance). EDP is calculated as $P \times {f_{max}}^2$, where P denotes power consumption and f_{max} denotes maximum achievable frequency. Lower EDP means more energy-efficient operations for the chip. Since we address power, performance and area (cost), we draw *EDP-Area* plots to show PPAC tradeoffs of various PDN structures. We again use four standard-cell libraries (*Lib1-4*).

From Figure 4.34(c), we derive four key observations. (i) For 4RT (*Lib1* and *Lib2*), EDP with P_{FB}, P_{BS} and P_{BB} decreases by 0.2, 0.2 and 0.4 $mW \cdot ns^2$, respectively, compared to P_{FS} with the same area. (ii) For 5RT (*Lib3* and *Lib4*), EDP with P_{BB} decreases by 0.3 $mW \cdot ns^2$, compared to P_{FS} with the same area. (iii) For 5RT, EDP with P_{FB} shows no improvements, and EDP with P_{BS} increases by 0.1 $mW \cdot ns^2$, as compared to P_{FS} with the same area. (iv) Use of P_{BB} better optimizes area than other PDN structures with the same EDP.

Supply Voltage (IR) Drop versus Area. With recent advanced technologies and designs, denser PDN structures are required due to large resistance seen in tight-pitch BEOL metal layers. The denser PDN structures bring added routability challenges which critically impact area density. In light of this, we measure IR drop and area from valid runs, and plot *IR drop-Area* tradeoffs

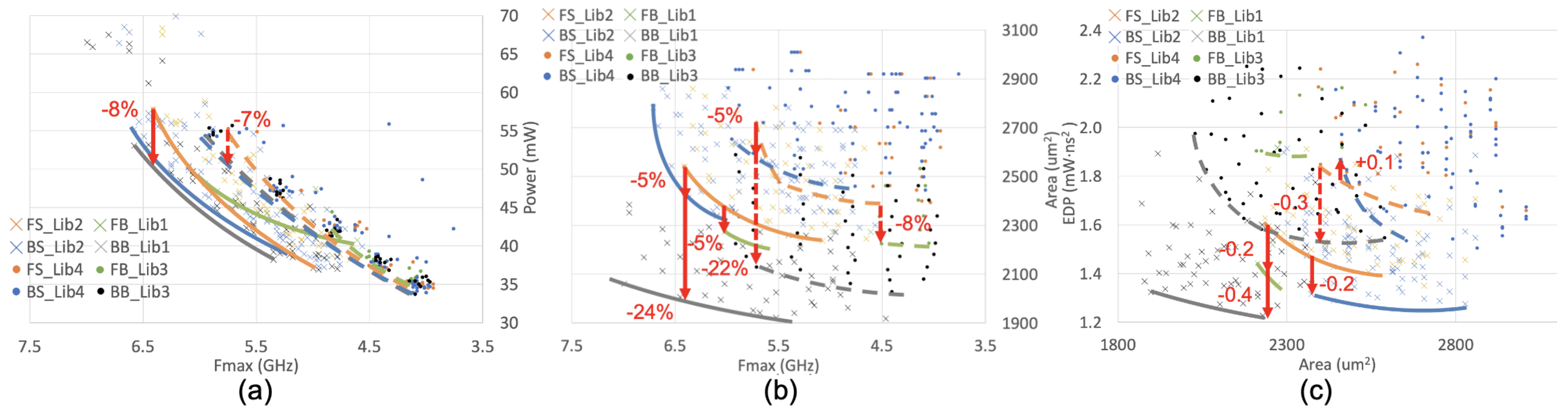


Figure 4.34. PPAC tradeoffs for JPEG with four standard-cell libraries (*Lib1-4*) and four PDN structures. We measure improvements relative to traditional PDN (P_{FS}), to show the benefits of BSPDN and BPR: (a) *Performance-Power*, (b) *Performance-Area* and (c) *Energy-Delay Product-Area*.

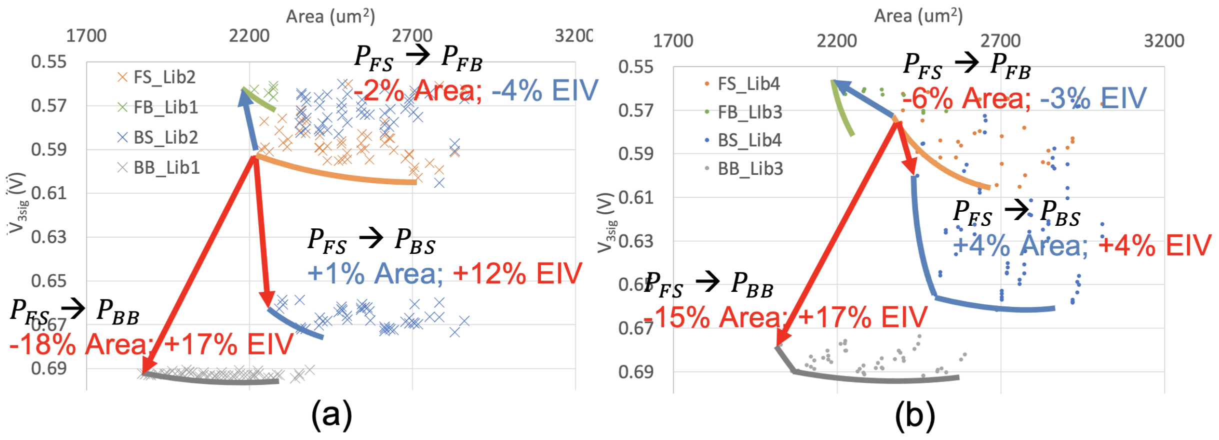


Figure 4.35. *IR drop-Area* plots for JPEG with four standard-cell libraries (*Lib1*, *Lib2*, *Lib3* and *Lib4*). (a) JPEG with 4RT (*Lib1/2*). (b) JPEG with 5RT (*Lib3/4*).

in Figure 4.35. In the plots, we compare the points with the minimum area for each PDN configuration in terms of area and 99.7 percentile (three-sigma) of effective instance voltage (EIV). Note that larger effective instance voltage means better IR drop mitigation. Figures 4.35(a) and (b) show *IR drop-Area* tradeoffs for JPEG with 4RT (*Lib1* and *Lib2*) and 5RT (*Lib3* and *Lib4*), respectively. From the results, we make four main observations. (i) Area with P_{FB} decreases by 2 to 6% compared to P_{FS}, while the effective instance voltage (EIV) increases by 3 to 4%. (ii) Area with P_{BS} increases by 1 to 4% compared to P_{FS}, while EIV decreases by 4 to 12%. (iii) Area with P_{BB} decreases by 15 to 18% compared to P_{FS}, while EIV decreases by 17%. (iv) We observe that there is IR drop mitigation from use of backside PDN, while use of BPR (P_{FB}) worsens IR drop. This implies that more power tap cells will need to be inserted to mitigate IR drop. However, the area overhead of power tap cells will degrade the IR drop quality achieved by use of BPR.

Expt. 2: PPAC Exploration with Artificial Design

Our second main experiment uses the *artificial* JPEG design generated by ANG using our ML-based parameter tuning. We conduct the same studies as in Expt. 1 and analyze the results.

Performance versus Power. Figure 4.36(a) shows the tradeoffs between performance and power with the artificial JPEG design. From the result, we make three main observations. (i)

Power consumption with P_{BS} and P_{BB} decreases by 6 to 14%, compared to P_{FS} with the same performance. (ii) Power consumption with P_{FB} is similar to P_{FS} with the same performance. (iii) Results with the artificial JPEG show up to 7% differences, but with similar trends, compared to the results obtained with the real JPEG design.

Performance versus Area. Figure 4.36(b) shows tradeoffs between performance and area with the artificial JPEG design. We make three main observations. (i) Area with P_{FB} and P_{BB} decreases up to 14% and 21% compared to P_{FS} with the same performance. (ii) Area with P_{BS} increases by 0% to 3% compared to P_{FS} with the same performance. This area penalty is caused by power tap cell insertion for P_{BS}. (iii) We observe that the results with the artificial JPEG show up to 9% differences, but with similar trends, compared to the results obtained with the real JPEG design. However, area for P_{BS} shows opposite trends to what we observe with the real design, although the discrepancy is not too large.

Energy-Delay Product (EDP) versus Area. From Figure 4.36(c), we make three main observations. (i) For 4RT (*Lib1* and *Lib2*), EDP with P_{BB} decreases by 0.5 $mW \cdot ns^2$, compared to P_{FS} with the same area. However, EDP with P_{FB} and P_{BS} shows no improvements. (ii) For 5RT (*Lib3* and *Lib4*), EDP with P_{FB} and P_{BB} decreases by 0.6 and 0.9 $mW \cdot ns^2$, compared to P_{FS} with the same area. However, EDP with P_{BS} shows no improvements. (iii) We observe that results with the artificial JPEG show similar trends as results obtained with the real JPEG design.

Supply Voltage (IR) Drop versus Area. Figures 4.37(a) and (b) show tradeoffs between IR drop and area for the artificial JPEG design with 4RT (*Lib1* and *Lib2*) and 5RT (*Lib3* and *Lib4*), respectively. We make four main observations. (i) Area with P_{FB} decreases by 9 to 14%, compared to P_{FS}, while the effective instance voltage (EIV) increases by 1 to 6%. (ii) Area with P_{BS} increases by 2%, compared to P_{FS}, while EIV decreases by 2 to 3%. (iii) Area with P_{BB} decreases by 14 to 18%, compared to P_{FS}, while EIV decreases by 6 to 11%. (iv) We observe that results with the artificial JPEG show similar trends as results obtained with the real JPEG design, and that discrepancies are reasonably small.

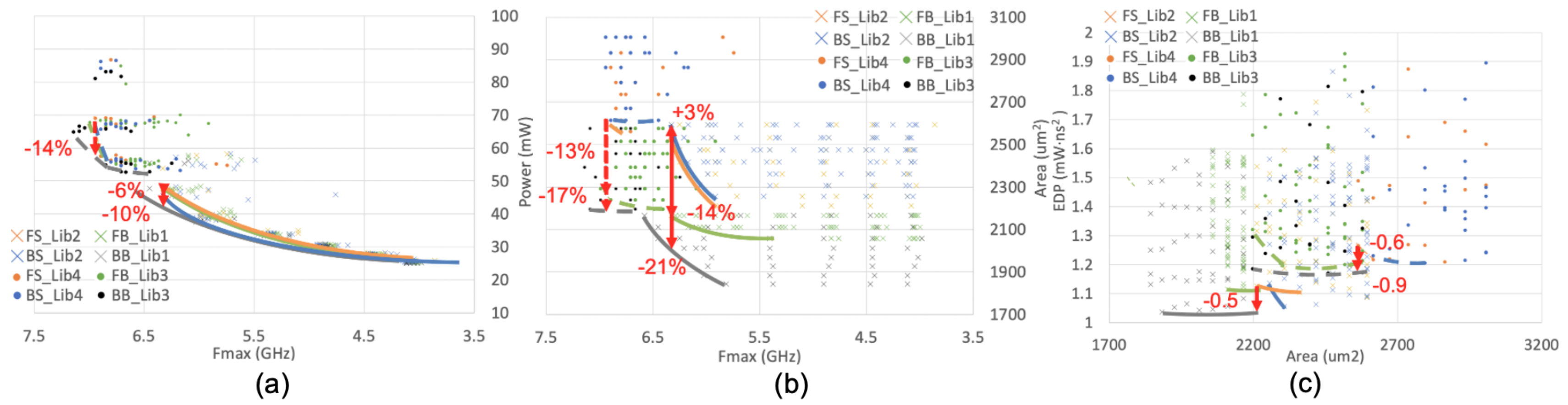


Figure 4.36. PPAC tradeoffs for “artificial” JPEG with four standard-cell libraries (*Lib1*, *Lib2*, *Lib3* and *Lib4*). (a) *Performance-Power*. (b) *Performance-Area*. (c) *Energy-Delay Product-Area*.

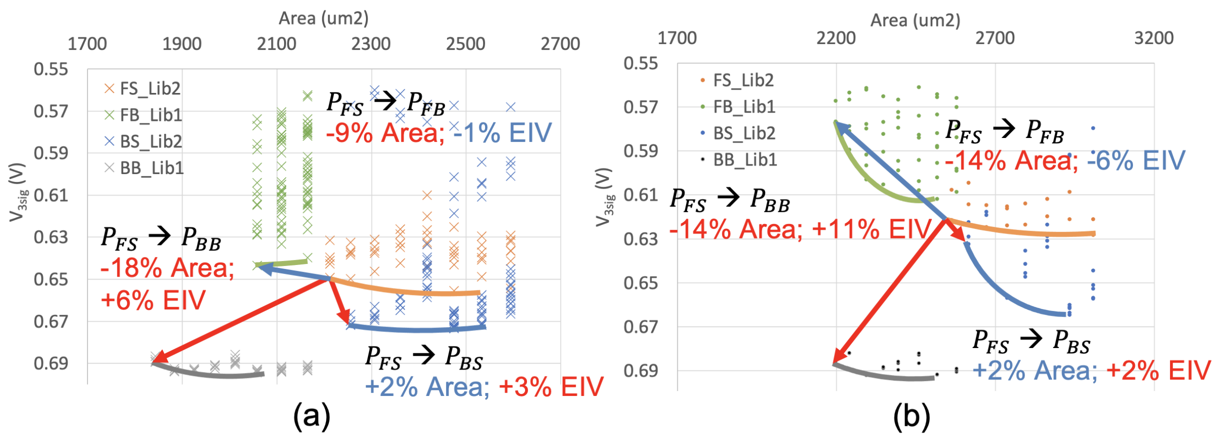


Figure 4.37. *IR drop-Area* plots for "artificial" JPEG with four standard-cell libraries (*Lib1*, *Lib2*, *Lib3* and *Lib4*). (a) JPEG with 4RT (*Lib1/2*). (b) JPEG with 5RT (*Lib3/4*).

Expt. 3: Routability Assessment and Achievable Utilization

Our third main experiment measures K_{th} using our *clustering-based cell width-regularized placements*, and explores the relationship between K_{th} and achievable utilization. We note that the previous work of [24] introduced *Achievable Utilization* as the maximum utilization for which the number of DRCs is less than a predefined threshold of 500 DRCs. Here, we include all three criteria for a valid result, and define *Achievable Utilization* as the maximum utilization among all valid runs seen.

Figure 4.38 shows experimental results for K_{th} and achievable utilization. We conduct our experiment with artificial JPEG and four cell width-regularized libraries (*Lib1-4*). From the plots, we make two observations. (i) We compare the results with 2Fin/4RT standard-cell libraries (*Lib1/2*) to those with 3Fin/5RT standard-cell libraries (*Lib3/4*). The data show that a larger number of M0 routing tracks brings better routability. (ii) Compared to P_{FS}, P_{FB} and P_{BB}, the plots for P_{BS} are skewed to the right for each design, showing better routability than the other PDN configurations. We observe that the routability improvement of P_{BS} comes from regularly-placed power tap cells: the power tap cell placement eases routing congestion caused by high cell and/or pin density.

Finally, we compare the K_{th} results obtained with the previous *cell width-regularized*

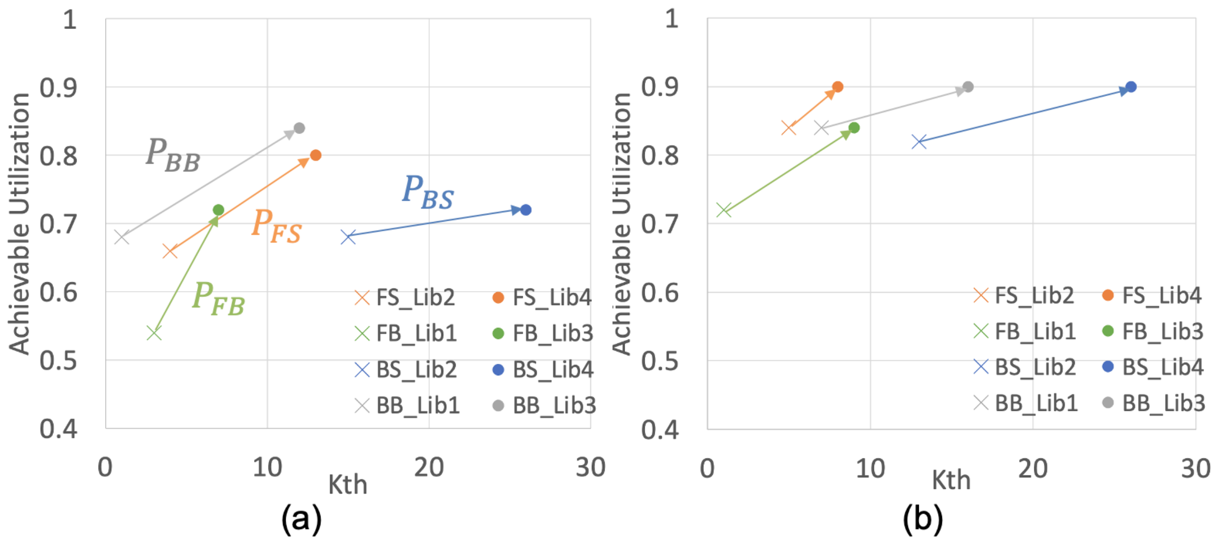


Figure 4.38. K_{th} and achievable utilization for (a) AES and (b) JPEG, with various libraries and power delivery methodologies.

placements used in the PROBE2.0 work ([A]) and *clustering-based cell width-regularized placements* obtained using the *CWR-FC* algorithm ([C]). We perform routability assessments as summarized in Table 4.29. We rank-order K_{th} across the eight combinations of four PDN and two RT with the JPEG design. The main observation from this comparison is that the ordering of enablement based on K_{th} is the same for both placements, even as the area utilization of the *clustering-based cell width-regularized placements* is closer to the initial utilization (0.6). We conclude that our *clustering-based cell width-regularized placement* methodology successfully provides more realistic placements without disrupting the K_{th}-based rank-ordering of enablement. Moreover, the generally smaller K_{th} values seen in the rightmost two columns of Table 4.29 imply fewer P&R trials needed to evaluate the K_{th} metric.

4.3.7 Conclusion

We have presented PROBE3.0, a systematic and configurable framework for "full-stack" PPAC exploration and pathfinding in advanced technology nodes. We introduce automated PDK and standard-cell library generation, along with enablement of scaling boosters in a predictive 3nm technology. Our work is permissively open-sourced in GitHub [177], and includes open-

Table 4.29. K_{th} comparison for the JPEG design with *cell width-regularized placements* ([A]) and *clustering-based cell width-regularized placements* ([C]). Util denotes real utilization with 0.6 initial utilization.

Rank	PDN	RT	**Library**	[A]		[C]	
				K_{th}	Util	K_{th}	Util
1	P_{FB}	4	*Lib1*	6	0.14	3	0.50
2	P_{FS}	4	*Lib2*	9	0.14	5	0.49
3	P_{BB}	4	*Lib1*	12	0.14	7	0.50
4	P_{FS}	5	*Lib4*	15	0.14	8	0.50
5	P_{FB}	5	*Lib3*	16	0.14	9	0.50
6	P_{BS}	4	*Lib2*	17	0.14	13	0.49
7	P_{BB}	5	*Lib3*	18	0.14	16	0.50
8	P_{BS}	5	*Lib4*	23	0.14	26	0.50

sourceable PDKs and EDA tool scripts that incorporate power and performance considerations into the framework.

We employ artificial netlist generation with a machine learning-based parameter tuning to mimic properties of arbitrary real designs. Along with a new *CWR-FC* clustering-based width-regularized netlist and placement methodology, this enables PPAC exploration of a much wider space of technology, design enablement, and design options. From our experiments, we find that the use of backside power delivery network (BSPDN) and buried power rails (BPR) can lead to up to 8% reduction in power consumption and up to 24% reduction in area using our predictive 3nm technology. These results from PROBE3.0 closely match previous works [55] [104] [138] [141] which estimated 25% to 30% area reduction from use of BSPDN and BPR.

Ongoing and future directions include the following. (i) Improving the software architecture of PROBE3.0 will make it more accessible and flexible for users to pursue their own PPAC explorations. Supporting the addition of user-defined variables can help capture and study variant technology and design assumptions. (ii) To improve robustness of the framework, and its usefulness as a "proxy" in real-world advanced technology development and DTCO, improved device models, parasitic extraction models, signoff corner definitions, relevant design examples, etc. will be beneficial. It will also be necessary to add generation of DRC rule decks for commercial tools. (iii) While our scripts for commercial tools are shared publicly in our GitHub repository, using these tools still requires valid licenses. Incorporation of open-source tools into the PROBE3.0 framework can potentially lead to highly-scaled deployments, shorter turnaround times, and improved utility to a broader audience.

4.4 Acknowledgments

Chapter 4 contains a reprint of Chung-Kuan Cheng, Andrew B. Kahng, Hayoung Kim, Minsoo Kim, Daeyeal Lee, Dongwon Park and Mingyu Woo, “PROBE2.0: A Systematic Framework for Routability Assessment from Technology to Design in Advanced Nodes”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and System* 41(5), 2022; Chidi Chidambaram, Andrew B. Kahng, Minsoo Kim, Giri Nallapati, S. C. Song and Mingyu Woo, “A Novel Framework for DTCO: Fast and Automatic Routability Assessment with Machine Learning for Sub-3nm Technology Options”, *Proc. IEEE Symposium on VLSI Technology*, 2021; and Suhyeong Choi, Jinwook Jung, Andrew B. Kahng, Minsoo Kim, Chul-Hong Park, Bodhisatta Pramanik and Dooseok Yoon, “PROBE3.0: A Systematic Framework for Design-Technology Pathfinding with Improved Design Enablement”, in submission to *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023. The dissertation author is a main contributor to, and a primary author of, each of these papers.

I would like to thank my coauthors, Professor Chung-Kuan Cheng, Dr. Chidi Chidambaram, Suhyeong Choi, Dr. Jinwook Jung, Professor Andrew B. Kahng, Hayoung Kim, Dr. Daeyeal Lee, Dr. Giri Nallapati, Dr. Chul-Hong Park, Dr. Dongwon Park, Bodhisatta Pramanik, Dr. S. C. Song, Mingyu Woo and Dooseok Yoon.

Chapter 5

Conclusion

This thesis presents robust physical design and design-technology co-optimization methodologies at advanced VLSI technology. The proposed methodologies address significant challenges posed by (i) aggressive PPAC optimization for various applications in modern IC designs, (ii) local layout effect and high resistance of advanced nodes, and (iii) limitations of the current DTCO process.

Chapter 2 presents two general physical design methodologies. First, we have proposed a new *k*-active dynamic power minimization problem that arises in clock distribution for memory-dominant IC designs. We formulate an ILP for the *k*-active dynamic power minimization problem, and the ILP minimizes the weighted sum of the total costs (wirelengths) and the maximum power across all sets of *k* active sinks. Our experimental results give new insights into the tradeoff for maximum *k*-(consecutive)-active dynamic power and wirelengths of subtrees as well as skew and total wirelengths. Second, we have described *CoRe-ECO*, a novel concurrent refinement framework for the automated ECO flow. We propose simultaneous and perturbation-minimized refinements of placement and routing to remove the remaining DRCs after the ECO stage. By ripping up and refining a local window of the entire layout design, *CoRe-ECO* is capable of achieving a DRC-clean layout solution. We have demonstrated that our framework successfully resolves an average of 58.6% (range: 33.3% to 100.0%) of remaining post-ECO route DRCs, across a range of benchmark circuits with various cell architectures, with no adverse effect on total routed wirelength (average of 0.003% reduction).

Chapter 3 presents two technology-aware physical design methodologies. First, we study the 2^{nd} DB effect, which is one of the critical LLEs for physical design in sub-10nm technologies, along with the mixed-DB design that can have both SDB and DDB cells. We propose new heuristics to recover leakage power increments caused by the 2^{nd} DB effect, including the use of 2^{nd} DB-aware relocation, gate sizing, Vt swapping and DB swapping while satisfying placement constraints. Our work achieves up to 80% leakage recovery on average using the proposed flow. Second, we have presented novel dynamic programming-based single- and double-row detailed placement optimizations with metaheuristics to maximize power staples. We perform extensive studies on the scalability and sensitivity to parameters, impact of the benefit table, as well as balance of power and ground staples. Compared to the traditional post-placement flow, we achieve up to 13.2% (10mV) reduction in IR drop, with almost no WNS degradation compared to a pre-placement flow.

Chapter 4 presents three design-technology co-optimization works. First, we have proposed a novel framework to evaluate routability impacts of advanced-node scaling options, spanning across technology, design enablement and design parameters. Crucially, our framework can reduce the timelines for design enablement and design implementation that limit today's DTCO methodologies. Also, our framework can flexibly support additional technology and design options. We integrate a powerful SMT-based standard-cell layout generation capability. Optimal layout solutions are obtained via a unified constraint satisfaction formulation that spans technology and cell architecture parameters. We validate these aspects of our methodology using a novel *cell-based* routability assessment with knight's tour-based topology generation. We also perform *design-based* routability assessment using open-source testcases, and show correlations of the K_{th} routability metric to achievable utilization in P&R. Furthermore, we propose learning-based K_{th} prediction to reduce runtime and required disk space, and to mitigate P&R tool license overheads. The experimental studies confirm the capability of our framework to produce routability assessments across a large range of technology and design parameters. Second, the

ML-assisted DTCO framework is applied to study sub-3nm node cell and block-level DTCO. The regime of CH $<$120nm with four available tracks shows increasing block-level area due to routing difficulty, indicating diminishing return on efforts to push ground rules. The proposed DTCO method will extend to evaluate power and performance along with the cell- and block-level area density assessment presented in this study. Last, we propose a systematic framework to explore PPAC tradeoffs with advanced technology nodes, by enabling scaling boosters and improved design enablements including standard-cell libraries. We enable our predictive 3nm technology and the corresponding PDKs. Also, we propose an ML-based parameter tuning flow and *clustering-based cell width-regularized placements* to generate realistic artificial designs. We experimentally demonstrate 24% area reduction through use of BSPDN and BPR. Our results are comparable to the previous works from industry which report 25% to 30% area reduction from use of BSPDN and BPR.

Bibliography

[1] A. Agnesina, K. Chang and S. K. Lim, “VLSI Placement Parameter Optimization Using Deep Reinforcement Learning”, *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2020, pp. 1-9.

[2] T. Ajayi, V. A. Chhabria, M. Fogaça, S. Hashemi, A. Hosny, A. B. Kahng, M. Kim, J. Lee, U. Mallappa, M. Neseem, G. Pradipta, S. Reda, M. Saligane, S. S. Sapatnekar, C. Sechen, M. Shalan, W. Swartz, L. Wang, Z. Wang, M. Woo and B. Xu, “Toward an Open-Source Digital Flow: First Learnings from the OpenROAD Project”, *Proc. ACM/IEEE Design Automation Conference*, 2019, pp. 76:1-76:4.

[3] Y. P. Aneja, “An Integer Linear Programming Approach to the Steiner Problem in Graphs”, *Networks* 10(2) (1980), pp. 167-178.

[4] C. Auth, A. Aliyarukunju, M. Asoro, D. Bergstrom, V. Bhagwat, J. Birdsall, N. Bisnik, M. Buehler, V. Chikarmane, G. Ding, Q. Fu, H. Gomez, W. Han, D. Hanken, M. Haran, M. Hattendorf, R. Heussner, H. Hiramatsu, B. Ho, S. Jaloviar, I. Jin, S. Joshi, S. Kirby, S. Kosaraju, H. Kothari, G. Leatherman, K. Lee, J. Leib, A. Madhavan, K. Marla, H. Meyer, T. Mule, C. Parker, S. Parthasarathy, C. Pelto, L. Pipes, I. Post, M. Prince, A. Rahman, S. Rajamani, A. Saha, J. Dacuna Santos, M. Sharma, V. Sharma, J. Shin, P. Sinha, P. Smith, M. Sprinkle, A. St. Amour, C. Stuas, R. Suri, D. Towner, A. Tripathi, A. Tura, C. Ward and A. Yeoh, “A 10nm High Performance and Low-Power CMOS Technology Featuring 3^{rd} Generation FinFET Transistors, Self-Aligned Quad Patterning, Contact Over Active Gate and Cobalt Local Interconnects”, *Proc. IEEE International Electron Devices Meeting*, 2017, pp. 29.1.1-29.1.4.

[5] H. B. Bakoglu and J. D. Meindl, “Optimal Interconnect Circuits for VLSI”, *IEEE Transactions on Electron Devices* ED-32(5) (1985), pp. 903–909.

[6] R. Berthelon, F. Andrieu, E. Josse, R. Bingert, O. Weber, E. Serret, A. Aurand, S. Delmedico, V. Farys, C. Bernicot, E. Bechet, E. Bernard, T. Poiroux, D. Rideau, P. Scheer, E. Baylac, P. Perreau, M.A. Jaud, J. Lacord, E. Petitprez, A. Pofelski, S. Ortolland, P. Sardin, D. Dutartre, A. Claverie, M. Vinet, J.C. Marin and M. Haond, “Design / Technology Co-optimization of Strain-induced Layout Effects in 14nm UTBB-FDSOI CMOS: Enablement and Assessment of Continuous-RX Designs”, *Proc. IEEE Symposium on VLSI Technology*, 2016, pp. 1-2.

[7] P. Besser, "BEOL Interconnect Innovations for Improving Performance", *Proc. NCCAVS Joint Users Group Technical Symposium*, 2017.

[8] K. Bhanushali and W. R. Davis, "FreePDK15: An Open-Source Predictive Process Design Kit for 15nm FinFET Technology", *Proc. International Symposium on Physical Design*, 2015, pp. 165–170.

[9] D. Bhattacharya, V. Boppana, R. Roy and J. Roy, "Method for Automated Design of Integrated Circuits with Targeted Quality Objectives Using Dynamically Generated Building Blocks", *US Patent*, US7225423B2, 2007.

[10] K. Boese and A. B. Kahng, "Zero-Skew Clock Routing Trees With Minimum Wirelength", *Proc. IEEE International ASIC Conference and Exhibit*, 1992, pp. 1.1.1 - 1.1.5.

[11] J. L. Burns and J. A. Feldman, "C5M-A Control-Logic Layout Synthesis System for High-Performance Microprocessors", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 17(1) (1998), pp. 14–23.

[12] A. Ceyhan, J. Quijas, S. Jain, H.-Y. Liu, W. E. Gifford and S. Chakravarty, "Machine Learning-Enhanced Multi-Dimensional Co-Optimization of Sub-10nm Technology Node Options", *Proc. IEEE International Electron Devices Meeting*, 2019, pp. 36.6.1-36.6.4.

[13] W.-T. J. Chan, Y. Du, A. B. Kahng, S. Nath and K. Samadi, "BEOL Stack-Aware Routability Prediction from Placement Using Data Mining Techniques", *Proc. IEEE International Conference on Computer Design*, 2016, pp. 41-48.

[14] D. Chanemougame, J. Smith, P. Gutwin, B. Byrns and L. Liebmann, "Agile Pathfinding Technology Prototyping: the Hunt for Directional Correctness", *Proc. International Conference on Simulation of Semiconductor Processes and Devices*, 2020, pp. 301-305.

[15] T.-H. Chao, Y.-C. Hsu, J.-M. Ho, K. D. Boese and A. B. Kahng, "Zero Skew Clock Routing With Minimum Wirelength", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39(11) (1992), pp. 799-814.

[16] W.-C. Chao and W.-K. Mak, "Low-Power Gated and Buffered Clock Network Construction", *ACM Transactions on Design Automation of Electronic Systems* 13(1) (2008), pp. 1-20.

[17] B. Chava, J. Ryckaert, L. Mattii, S. M. Y. Sherazi, P. Debacker, A. Spessot and D. Verkest, "DTCO Exploration for Efficient Standard Cell Power Rails", *Proc. SPIE 10588, Design-Process-Technology Co-optimization for Manufacturability XII*, 105880B, 2018, pp. 1-6.

[18] B. Chava, K. A. Shaik, A. Jourdain, S. Guissi, P. Weckx, J. Ryckaert, G. Van Der Plaas, A. Spessot, E. Beyne and A. Mocuta, "Backside Power Delivery as a Scaling Knob for Future Systems", *Proc. SPIE 10962, Design-Process-Technology Co-optimization for Manufacturability XIII*, 1096205, 2019, pp. 1-6.

[19] G. Chen and E. F. Y. Young, “Dim Sum: Light Clock Tree by Small Diameter Sum”, *Proc. Design, Automation & Test in Europe Conference & Exhibition*, 2019, pp. 174-179.

[20] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System”, *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[21] L. Chen, C. Huang, Y. Chang and H. Chen, “A Learning-Based Methodology for Routability Prediction in Placement”, *Proc. International Symposium on VLSI Design, Automation and Test*, 2018, pp. 1-4.

[22] C.-K. Cheng, C.-T. Ho, C. Holtz and B. Lin, “Design and System Technology Co-Optimization Sensitivity Prediction for VLSI Technology Development Using Machine Learning”, *Proc. ACM/IEEE International Workshop on System-Level Interconnect Prediction*. 2021, pp. 1-8.

[23] C.-K. Cheng, C.-T. Ho, D. Lee and D. Park, “A Routability-Driven Complimentary-FET (CFET) Standard Cell Synthesis Framework Using SMT”, *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2020, pp. 1-8.

[24] C.-K. Cheng, A. B. Kahng, H. Kim, M. Kim, D. Lee, D. Park and M. Woo, “PROBE2.0: A Systematic Framework for Routability Assessment from Technology to Design in Advanced Nodes”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41(5) (2022), pp. 1495-1508.

[25] C.-K. Cheng, A. B. Kahng, I. Kang and L. Wang, “RePlAce: Advancing Solution Quality and Routability Validation in Global Placement”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38(9) (2019), pp. 1717-1730.

[26] C.-K. Cheng, D. Lee and D. Park, “Standard-Cell Scaling Framework with Guaranteed Pin-Accessibility”, *Proc. IEEE International Symposium on Circuits and Systems*, 2020, pp. 1-5.

[27] C. Chidambaram, A. B. Kahng, M. Kim, G. Nallapati, S. C. Song and M. Woo, “A Novel Framework for DTCO: Fast and Automatic Routability Assessment with Machine Learning for Sub-3nm Technology Options”, *Proc. IEEE Symposium on VLSI Technology*, 2021, pp. 1-2.

[28] C. Chu and Y.-C. Wong, “FLUTE: Fast Lookup Table Based Rectilinear Steiner Minimal Tree Algorithm for VLSI Design”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27(1) (2008), pp. 70-83.

[29] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy and G. Yeric, “ASAP7: A 7-nm FinFET Predictive Process Design Kit”, *Microelectronics Journal* 53 (2016), pp. 105–115.

[30] J. Cong, A. B. Kahng, C. K. Koh and C.-W. A. Tsao, "Bounded-Skew Clock and Steiner Routing", *ACM Transactions on Design Automation of Electronic Systems* 3(3) (1998), pp. 341-388.

[31] J. M. Cohn, J. Venuto, I. L. Wemple and P. S. Zuchowski, "Method for Supply Voltage Drop Analysis During Placement Phase of Chip Design", *US Patent*, US6523154B2, 2000.

[32] P. Cremer, S. Hougardy, J. Schneider and J. Silvanus, "Automatic Cell Layout in the 7nm Era", *Proc. International Symposium on Physical Design*, 2017, pp. 99-106.

[33] W. M. Dai, T. Asano and E. S. Kuh, "Routing Region Definition and Ordering Scheme for Building-Block Layout", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 4(3) (1985), pp. 189-197.

[34] L. De Moura and N. Bjørner, "Z3: An Efficient SMT Solver", *Proc. International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2008, pp. 337–340.

[35] Y. Ding, C. Chu and W.-K. Mak, "Pin Accessibility-Driven Detailed Placement Refinement", *Proc. International Symposium on Physical Design*, 2017, pp. 133–140.

[36] M. Donno, E. Macii and L. Mazzoni, "Power-Aware Clock Tree Planning", *Proc. International Symposium on Physical Design*, 2004, pp. 138-147.

[37] Y. Du and M. D. F. Wong, "Optimization of Standard Cell Based Detailed Placement for 16nm FinFET Process", *Proc. Design, Automation & Test in Europe Conference & Exhibition*, 2014, pp. 1-6.

[38] M. Edahiro, "A Clustering-Based Optimization Algorithm in Zero-Skew Routings", *Proc. ACM/IEEE Design Automation Conference*, 1993, pp. 612-616.

[39] M. Fernández-Delgado, M. S. Sirsat, E. Cernadas, S. Alawadi, S. Barro and M. Febrero-Bande, "An Extensive Experimental Survey of Regression Methods", *Neural Networks* 111 (2019), pp. 11-34.

[40] G. Flach, T. Reimann, G. Posser, M. Johann and R. Reis, "Effective Method for Simultaneous Gate Sizing and Vth Assignment Using Lagrangian Relaxation", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 33(4) (2014), pp. 546-557.

[41] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine", *Annals of Statistics* 29(5) (2001), pp. 1189-1232.

[42] P. Geurts, D. Ernst and L. Wehenkel, "Extremely Randomized Trees", *Machine Learning* 63(1) (2006), pp. 3-42.

[43] B. Giraud, O. Thomas, A. Amara, A. Vladimirescu and M. Belleville, *SRAM Circuit Design*, Dordrecht, Springer, 2009.

[44] P. Gupta, A. B. Kahng, P. Sharma and D. Sylvester, "Gate-Length Biasing for Runtime Leakage Control", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 25(8) (2006), pp. 1475-1485.

[45] M. Guruswamy, R. L. Maziasz, D. Dulitz, S. Raman, V. Chiluvuri, A. Fernandez and L. G. Jones, "Cellerity: A Fully Automatic Layout Synthesis System for Standard Cell Libraries", *Proc. Design Automation Conference*, 1997, pp. 327-332.

[46] N. Ha, Y.-D. Kim, B.-H. Lee, H.-O. Kim, K. Jeong and J. Kim, "System and Method of Designing Integrated Circuit by Considering Local Layout Effect", *US Patent*, US20180032658A1, 2018.

[47] L. Hagen, A. B. Kahng, F. Kurdahi and C. Ramachandran, "On the Intrinsic Rent Parameter and New Spectra-Based Methods for Wireability Estimation", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 13(1) (1994), pp. 27-37.

[48] C. Han, K. Han, A. B. Kahng, H. Lee, L. Wang and B. Xu, "Optimal Multi-Row Detailed Placement for Yield and Model-Hardware Correlation Improvements in Sub-10nm VLSI", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2017, pp. 667-674.

[49] K. Han, A. B. Kahng and H. Lee, "Evaluation of BEOL Design Rule Impacts Using an Optimal ILP-Based Detailed Router", *Proc. ACM/EDAC/IEEE Design Automation Conference*, 2015, pp. 1-6.

[50] K. Han, A. B. Kahng and H. Lee, "Scalable Detailed Placement Legalization for Complex Sub-14nm Constraints", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2015, pp. 867-873.

[51] K. Han, A. B. Kahng, C. Moyes and A. Zelikovsky, "A Study of Optimal Cost-Skew Trade-off and Remaining Suboptimality in Interconnect Tree Constructions", *Proc. ACM/IEEE International Workshop on System-Level Interconnect Prediction*, 2018, pp. 2:1-2:8.

[52] C. Han, A. B. Kahng, L. Wang and B. Xu, "Enhanced Optimal Multi-Row Detailed Placement for Neighbor Diffusion Effect Mitigation in Sub-10nm VLSI", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38(9) (2019), pp. 1703-1716.

[53] M. Hanan, "On Steiner's Problem with Rectilinear Distance", *SIAM Journal on Applied Mathematics* 14(2), pp. 255–265.

[54] X. He, Y. Wang, Y. Guo and E. F. Y. Young, "Ripple 2.0: Improved Movement of Cells in Routability-Driven Placement", *ACM Transactions on Design and Automation of Electronic Systems* 22(1) (2016), pp. 10:1-10:26.

[55] M. O. Hossen, B. Chava, G. Van der Plas, E. Beyne and M. S. Bakir, "Power Delivery Network (PDN) Modeling for Backside-PDN Configurations With Buried Power Rails and μ TSVs", *IEEE Transactions on Electron Devices* 67(1) (2020), pp. 11-17.

[56] M.-K. Hsu, N. Katta, H. Y.-H. Lin, K. T.-H. Lin, K. H. Tam and K. C.-H. Wang, "Design and Manufacturing Process Co-Optimization in Nano-Technology", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2014, pp. 574-581.

[57] J. Hu, A. B. Kahng, S. Kang, M.-C. Kim and I. L. Markov, "Sensitivity-Guided Metaheuristics for Accurate Discrete Gate Sizing", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2012, pp. 233-239.

[58] J.-H. Huang, A. B. Kahng and C.-W. A. Tsao, "On the Bounded-Skew Clock and Steiner Routing Problems", *Proc. Design Automation Conference*, 1995, pp. 508-513.

[59] K. Jo, S. Ahn, J. Do, T. Song, T. Kim and K. Choi, "Design Rule Evaluation Framework Using Automatic Cell Layout Generator for Design Technology Co-Optimization", *IEEE Transactions on Very Large Scale Integration Systems* 27(8) (2019), pp. 1933-1946.

[60] D. Johnson, "Near-Optimal Bin Packing Algorithms", *Ph.D. Thesis*, Massachusetts Institute of Technology, 1973.

[61] A. B. Kahng, J. Dodrill and P. Gupta, "Designing in Advanced Technologies: A Quick Review of Approaches", Tutorial 4, *ACM/ESDA/IEEE Design Automation Conference*, 2018.

[62] A. B. Kahng and H. Lee, "Minimum Implant Area-Aware Gate Sizing and Placement", *Proc. Great Lakes Symposium on Very Large Scale Integration*, 2014, pp. 57-62.

[63] A. B. Kahng, S. Kang, S. Kim and B. Xu, "Enhanced Power Delivery Pathfinding for Emerging 3D Integration Technology", *IEEE Transactions on Very Large Scale Integration Systems* 29(4) (2021), pp. 591-604.

[64] A. Kahng, A. B. Kahng, H. Lee and J. Li, "PROBE: Placement, Routing, Back-End-of-Line Measurement Utility", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37(7) (2018), pp. 1459-1472.

[65] A. B. Kahng, S. Kang, H. Lee, I. L. Markov and P. Thapar, "High-Performance Gate Sizing with a Signoff Timer", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2013, pp. 450-457.

[66] A. B. Kahng, J. Kuang, W.-H. Liu and B. Xu, "In-Route Pin Access-Driven Placement Refinement for Improved Detailed Routing Convergence", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 41(3) (2022), pp. 784-788.

[67] A. B. Kahng, B. Liu and Q. Wang, "Supply Voltage Degradation Aware Analytical Placement", *Proc. International Conference on Computer Design*, 2005, pp. 437-443.

[68] A. B. Kahng, I. L. Markov and S. Reda, "On Legalization of Row-Based Placements", *Proc. ACM Great Lakes Symposium on Very Large Scale Integration*, 2004, pp. 214-219.

[69] A. B. Kahng and C.-W. A. Tsao, “Planar-DME: A Single-Layer Zero-Skew Clock Tree Router”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 15(1) (1996), pp. 8-19.

[70] A. B. Kahng and C.-W. A. Tsao, “Practical Bounded-Skew Clock Routing”, *Journal of VLSI Signal Processing* 16 (1997), pp. 199-215.

[71] A. B. Kahng and Q. Wang, “A Faster Implementation of APlace”, *Proc. International Symposium on Physical Design*, 2006, pp. 218-220.

[72] A. B. Kahng, L. Wang and B. Xu, “The Tao of PAO: Anatomy of a Pin Access Oracle for Detailed Routing”, *Proc. ACM/IEEE Design Automation Conference*, 2020, pp. 1–6.

[73] I. Kang, D. Park, C. Han and C.-K. Cheng, “Fast and Precise Routability Analysis with Conditional Design Rules”, *Proc. ACM/IEEE International Workshop on System-Level Interconnect Prediction*, 2018, pp. 1-4.

[74] G. Karypis, R. Aggarwal, V. Kumar and S. Shekhar, “Multilevel Hypergraph Partitioning: Applications in VLSI Domain”, *IEEE Transactions on Very Large Scale Integration Systems* 7(1) (1999), pp. 69-79.

[75] M.-C. Kim, J. Hu, D. J. Lee and I. L. Markov, “A SimPLR Method for Routability-Driven Placement”, *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2011, pp. 67-73.

[76] S. Kim, K. Jo and T. Kim, “Boosting Pin Accessibility through Cell Layout Topology Diversification”, *Proc. Asia and South Pacific Design Automation Conference*, 2021, pp. 1–6.

[77] T. Kim, J. Jeong, S. Woo, J. Yang, H. Kim, A. Nam, C. Lee, J. Seo, M. Kim, S. Ryu, Y. Oh and T. Song, “NS3K: A 3-nm Nanosheet FET Standard Cell Library Development and Its Impact”, *IEEE Transactions on Very Large Scale Integration Systems* (2022), pp. 1-14.

[78] D. Kim, S.-Y. Lee, K. Min and S. Kang, “Construction of Realistic Place-and-Route Benchmarks for Machine Learning Applications”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2022), doi:10.1109/TCAD.2022.3209530.

[79] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T.-Y. Liu, “LightGBM: A Highly Efficient Gradient Boosting Decision Tree”, *Proc. Advances in Neural Information Processing Systems*, 2017, pp. 3146-3154.

[80] E. D. Kurniawan, H. Yang, C.-C. Lin and Y.-C. Wu, “Effect of Fin Shape of Tapered FinFETs on the Device Performance in 5-nm Node CMOS Technology”, *Microelectronics Reliability* 83 (2018), pp. 254-259.

[81] B. Landman and R. Russo, “On a Pin Versus Block Relationship for Partitions of Logic Graphs”, *IEEE Transactions on Computers* C-20(12) (1971), pp. 1469-1479.

[82] D. Lee, D. Park, C.-T. Ho, I. Kang, H. Kim, S. Gao, B. Lin and C.-K. Cheng, "SP&R: SMT-based Simultaneous Place-&-Route for Standard Cell Synthesis of Advanced Nodes", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 40(10) (2021), pp. 2142-2155.

[83] Y.-L. Li, S.-T. Lin, S. Nishizawa, H.-Y. Su, M.-J. Fong, O. Chen and H. Onodera, "NCTU-cell: A DDA-Aware Cell Library Generator for FinFET Structure with Implicitly Adjustable Grid Map", *Proc. ACM/IEEE Design Automation Conference*, 2019, pp. 1-6.

[84] L. Liebmann, D. Chanemougame, P. Churchill, J. Cobb, C.-T. Ho, V. Moroz and J. Smith, "DTCO Acceleration to Fight Scaling Stagnation", *Proc. SPIE 11328, Design-Process-Technology Co-optimization for Manufacturability XIV*, 2020, 113280C, pp. 1-15.

[85] L. Liebmann, V. Gerousis, P. Gutwin, X. Zhu and J. Petykiewicz, "Exploiting Regularity: Breakthroughs in Sub-7nm Place-and-Route", *Proc. SPIE 10148, Design-Process-Technology Co-optimization for Manufacturability XI*, 2017, pp. 1-10.

[86] T. Lin and C. Chu, "POLAR 2.0: An Effective Routability-Driven Placer", *Proc. ACM/EDAC/IEEE Design Automation Conference*, 2014, pp. 1-6.

[87] Y. Lin, B. Yu, X. Xu, J.-R. Gao, N. Viswanathan, W.-H. Liu, Z. Li, C. J. Alpert and D. Z. Pan, "MrDP: Multiple-row Detailed Placement of Heterogeneous-sized Cells for Advanced Nodes", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2016, pp. 7:1-7:8.

[88] W.-H. Liu, C.-K. Koh and Y.-L. Li, "Optimization of Placement Solutions for Routability", *Proc. ACM/EDAC/IEEE Design Automation Conference*, 2013, pp. 1-9.

[89] L.-C. Lu, "Physical Design Challenges and Innovations to Meet Power, Speed, and Area Scaling Trend", *Proc. ACM International Symposium on Physical Design*, 2017, p. 63.

[90] T. Luo, D. Newmark and D. Z. Pan, "Total Power Optimization Combining Placement, Sizing and Multi-Vt Through Slack Distribution Management", *Proc. Asia and South Pacific Design Automation Conference*, 2008, pp. 352-357.

[91] L. Maaten and G. Hinton, "Visualizing Data Using t-SNE", *Journal of Machine Learning Research* 9 (2008), pp. 2579-2605.

[92] P. Majumder, B. Kumthekar, N. R. Shah, J. Mowchenko, P. A. Chavda, Y. Kojima, H. Yoshida and V. Boppana, "Method of IC Design Optimization via Creation of Design-Specific Cells from Post-Layout Patterns", *US Patent*, US7941776B2, 2011.

[93] W. K. Mak, W. S. Kuo, S. H. Zhang, S. I. Lei and C. Chu, "Minimum Implant Area-Aware Placement and Threshold Voltage Refinement", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36(7) (2017), pp. 1103-1112.

[94] L. Mattii, D. M. Milojevic, P. Debacker, Y. Sherazi, M. Berekovic and P. Raghavan, "IR-Drop Aware Design & Technology Co-Optimization for N5 Node with Different Device and Cell Height Options", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2017, pp. 89-94.

[95] M. D. McKay, W. J. Conover and R. J. Beckman, "A Comparison of Three Methods for Selection Values of Input Variables in the Analysis of Output from a Computer Code", *Technometrics* 21(2) (1979), pp. 239–245.

[96] C. W. Moon, P. Gupta, P. J. Donehue and A. B. Kahng, "Method of Designing a Digital Circuit by Correlating Different Static Timing Analyzers", *US Patent*, US7823098B1, 2010.

[97] J. A. Nelder and R. W. M. Wedderburn, "Generalized Linear Models", *Journal of the Royal Statistical Society Series A (Statistics in Society)* 135(3) (1972), pp. 370-384.

[98] J. Oh, I. Pyo and M. Pedram, "Constructing Lower and Upper Bounded Delay Routing Trees Using Linear Programming", *Proc. Design Automation Conference*, 1996, pp. 401-404.

[99] D. Park, I. Kang, Y. Kim, S. Gao, B. Lin and C.-K. Cheng, "ROAD: Routability Analysis and Diagnosis Framework Based on SAT Techniques", *Proc. International Symposium on Physical Design*, 2019, pp. 65-72.

[100] D. Park, D. Lee, I. Kang, C. Holtz, S. Gao, B. Lin and C.-K. Cheng, "Grid-Based Framework for Routability Analysis and Diagnosis with Conditional Design Rules", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39(12) (2020), pp. 5097-5110.

[101] G. Posser, V. Mishra, P. Jain, R. Reis and S. S. Sapatnekar, "A Systematic Approach for Analyzing and Optimizing Cell-Internal Signal Electromigration", *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2014, pp. 486-491.

[102] D. Prasad, S. S. T. Nibhanapudi, S. Das, O. Zografos, B. Chehab, S. Sarkar, R. Baert, A. Robinson, A. Gupta, A. Spessot, P. Debacker, D. Verkest, J. Kulkarni, B. Cline and S. Sinha, "Buried Power Rails and Back-side Power Grids: Arm® CPU Power Delivery Network Design Beyond 5nm", *Proc. IEEE International Electron Devices Meeting*, 2019, pp. 19.1.1-19.1.4.

[103] X. Qiu and M. Marek-Sadowska, "Can Pin Access Limit the Footprint Scaling?", *Proc. Design Automation Conference*, 2012, pp. 1100–1106.

[104] J. Ryckaert, A. Gupta, A. Jourdain, B. Chava, G. Van der Plas, D. Verkest and E. Beyne, "Extending the Roadmap Beyond 3nm through System Scaling Boosters: A Case Study on Buried Power Rail and Backside Power Delivery", *Proc. IEEE Electron Devices Technology and Manufacturing Conference*, 2019, pp. 50-52.

[105] N. Ryzhenko, S. Burns, A. Sorokin and M. Talalay, “Pin Access-Driven Design Rule Clean and DFM Optimized Routing of Standard Cells under Boolean Constraints”, *Proc. International Symposium on Physical Design*, 2019, pp. 41–47.

[106] S. Sadangi, “FreePDK3: A Novel PDK for Physical Verification at the 3nm Node”, *M.S. Thesis*, Computer Engineering, North Carolina State University, 2021.

[107] R. Sengupta, J. G. Hong and M. Rodder, “Semiconductor Device Having Buried Power Rail”, *US Patent*, US9570395, 2017.

[108] J. Seo, J. Jung, S. Kim and Y. Shin, “Pin Accessibility-Driven Cell Layout Redesign and Placement Optimization”, *Proc. ACM/EDAC/IEEE Design Automation Conference*, 2017, pp. 54:1-54:6.

[109] S. M. Y. Sherazi, J. K. Chae, P. Debacker, L. Mattii, P. Raghavan, V. Gerousis, D. Verkest, A. Mocuta, R. H. Kim, A. Spessot and J. Ryckaert, “Track Height Reduction for Standard-Cell in Below 5nm Node: How Low Can You Go?”, *Proc. SPIE 10588, Design-Process-Technology Co-optimization for Manufacturability XII*, 1058809, 2018, pp. 1058809-1:1058809-13.

[110] S. M. Y. Sherazi, M. Cupak, P. Weckx, O. Zografos, D. Jang, P. Debacker, D. Verkest, A. Mocuta, R. H. Kim, A. Spessot and J. Ryckaert, “Standard-Cell Design Architecture Options Below 5nm Node: The Ultimate Scaling of FinFET and Nanosheet”, *Proc. SPIE 10962, Design-Process-Technology Co-optimization for Manufacturability XIII*, 1096202, 2019, pp. 1096202:1-1096202:15.

[111] S. M. Y. Sherazi, C. Jha, D. Rodopoulos, P. Debacker, B. Chava, L. Mattii, M. G. Bardon, P. Schuddinck, P. Raghavan, V. Gerousis, A. Spessot, D. Verkest, A. Mocuta, R. H. Kim and J. Ryckaert, “Low Track Height Standard Cell Design in iN7 Using Scaling Boosters”, *Proc. SPIE 10148, Design-Process-Technology Co-optimization for Manufacturability XI*, 101480Y, 2017, pp. 101480Y:1-101480Y:8.

[112] G. Sisto, B. Chehab, B. Genneret, R. Baert, R. Chen, P. Weckx, J. Ryckaert, R. Chou, G. Van der Plas, E. Beyne and D. Milojevic, “IR-Drop Analysis of Hybrid Bonded 3D-ICs with Backside Power Delivery and μ- & n-TSVs”, *Proc. IEEE International Interconnect Technology Conference*, 2021, pp. 1-3.

[113] S. C. Song, J. Xu, D. Yang, K. Rim, P. Feng, J. Bao, J. Zhu, J. Wang, G. Nallapati, M. Badaroglu, P. Narayanasetti, B. Bucki, J. Fischer and G. Yeap, “Unified Technology Optimization Platform using Integrated Analysis (UTOPIA) for Holistic Technology, Design and System Co-Optimization at $<=$ 7nm Nodes”, *Proc. IEEE Symposium on VLSI Circuits*, 2016, pp. 1-2.

[114] H. Su, J. Hu, S. S. Sapatnekar and S. R. Nassif, “Congestion-Driven Codesign of Power and Signal Networks”, *Proc. Design Automation Conference*, 2002, pp. 64-69.

[115] S. Sutanthavibul, E. Shragowitz and J. B. Rosen, “An Analytical Approach to Floorplan Design and Optimization”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 10(6) (1991), pp. 761–769.

[116] S. S. Teja Nibhanupudi, D. Prasad, S. Das, O. Zografos, A. Robinson, A. Gupta, A. Spessot, P. Debacker, D. Verkest, J. Ryckaert, G. Hellings, J. Myers, B. Cline and J. P. Kulkarni, “A Holistic Evaluation of Buried Power Rails and Back-Side Power for Sub-5nm Technology Nodes”, *IEEE Transactions on Electron Devices* 69(8), 2022, pp. 4453-4459.

[117] I. Tseng, Z. C. Lee, V. Tripathi, C. M. T. Yip, Z. Chen and J. Ong, “A System for Standard Cell Routability Checking and Placement Routability Improvements”, *Proc. IEEE Asia Pacific Conference on Circuits and Systems*, 2019, pp. 125-128.

[118] P. Van Cleeff, S. Hougardy, J. Silvanus and T. Werner, “BonnCell: Automatic Cell Layout in the 7nm Era”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39(10) (2020), pp. 2872-2885.

[119] V. Vashishtha, M. Vangala and L. T. Clark, “ASAP7 Predictive Design Kit Development and Cell Design Technology Co-optimization”, *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2017, pp. 992-998.

[120] A. Vittal and M. Marek-Sadowska, “Low-Power Buffered Clock Tree Design”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 16(9) (1997), pp. 965-975.

[121] D. M. Warme, P. Winter and M. Zachariasen, “Exact Algorithms for Plane Steiner Tree Problems: A Computational Study”, *Advances in Steiner Trees* (D. Z. Du, J. M. Smith and J. H. Rubinstein, eds.), Kluwer Academic Publishers, 2000, pp. 81-116.

[122] L. Wei, K. Roy and C.-K. Koh, “Power Minimization by Simultaneous Dual-Vth Assignment and Gate-Sizing”, *Proc. IEEE Custom Integrated Circuits Conference*, 2000, pp. 413-416.

[123] H.-S. P. Wong, “Semiconductor Technology: A System Perspective”, *Opening Keynote*, *ACM/IEEE Design Automation Conference*, 2020. http://www2.dac.com/events/eventdetails.aspx?id=295-151

[124] Z. Xie, Y.-H. Huang, G.-Q. Fang, H. Ren, S.-Y. Fang, Y. Chen and J. Hu, “RouteNet: Routability Prediction for Mixed-Size Designs Using Convolutional Neural Network”, *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2018, pp. 1-8.

[125] R. Xie, K.-Y. Lim, M. G. Sung and R. R.-H. Kim, “Methods of Forming Single and Double Diffusion Breaks on Integrated Circuit Products Comprised of FinFET Devices and the Resulting Products”, *US Patent*, US9412616, 2016.

[126] X. Xu, B. Cline, G. Yeric, B. Yu and D. Z. Pan, “Self-Aligned Double Patterning Aware Pin Access and Standard Cell Layout Co-Optimization”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 34(5) (2015), pp. 699–712.

[127] X. Xu, Y. Lin, V. Livramento and D. Z. Pan, “Concurrent Pin Access Optimization for Unidirectional Routing”, *Proc. ACM/EDAC/IEEE Design Automation Conference*, 2017, pp. 1-6.

[128] S. Yang, Y. Liu, M. Cai, J. Bao, P. Feng, X. Chen, L. Ge, J. Yuan, J. Choi, P. Liu, Y. Suh, H. Wang, J. Deng, Y. Gao, J. Yang, X.-Y. Wang, D. Yang, J. Zhu, P. Penzes, S.C. Song, C. Park, S. Kim, J. Kim, S. Kang, E. Terzioglu, K. Rim and PR. C. Chidambaram, “10nm High Performance Mobile SoC Design and Technology Co-Developed for Performance, Power, and Area Scaling”, *Proc. Symposium on VLSI Technology*, 2017, pp. T70-T71.

[129] G. Yeric, “Circuit Application Requirements”, *Proc. IEEE International Electron Devices Meeting*, Short Course, 2014.

[130] T.-C. Yu, S.-Y. Fang, H.-S. Chiu, K.-S. Hu, P. H.-Y. Tai, C. C.-F. Shen and H. Sheng, “Pin Accessibility Prediction and Optimization with Deep Learning-Based Pin Pattern Recognition”, *Proc. ACM/IEEE Design Automation Conference*, 2019, pp. 1–6.

[131] C.-Y. Yu, Y.-T. Hou, C.-M. Fu, W.-H. Chen and W.-Y. Lo, “Apparatus and Method for Mitigating Dynamic IR Voltage Drop and Electromigration Affects”, *US Patent*, US9768119B2, 2017.

[132] A. Z. Zelikovsky and I. I. Mandoiu, “Practical Approximation Algorithms for Zero- and Bounded-Skew Trees”, *SIAM Journal Discrete Mathematics* 15(1) (2002), pp. 97-111.

[133] Z. Zhang, R. Wang, C. Chen, Q. Huang, Y. Wang, C. Hu, D. Wu, J. Wang and R. Huang, “New-Generation Design-Technology Co-Optimization (DTCO): Machine-Learning Assisted Modeling Framework”, *Proc. Silicon Nanoelectronics Workshop*, 2019, pp. 1-2.

[134] P. Zhao, S. M. Pandey, E. Banghart, X. He, R. Asra, V. Mahajan, H. Zhang, B. Zhu, K. Yamada, L. Cao, P. Balasubramaniam, M. Joshi, M. Eller, F. Benistant and S. Samavedam, “Influence of Stress Induced CT Local Layout Effect (LLE) on 14nm FinFET”, *Proc. Symposium on VLSI Technology*, 2017, pp. T228-T229.

[135] Q. Zhou, X. Wang, Z. Qi, Z. Chen, Q. Zhou and Y. Cai, “An Accurate Detailed Routing Routability Prediction Model in Placement”, *Proc. Asia Symposium on Quality Electronic Design*, 2015, pp. 119-122.

[136] A. M. Ziesemer and R. A. da Luz Reis, “Simultaneous Two-Dimensional Cell Layout Compaction Using MILP with ASTRAN”, *Proc. IEEE Computer Society Annual Symposium on VLSI*, 2014, pp. 350-355.

[137] 68-95-99.7 Rule. https://en.wikipedia.org/wiki/68-95-99.7_rule

[138] Applied Materials Logic Master Class. https://ir.appliedmaterials.com/static-files/acba6be3-4778-41eb-9183-5c8e52884dea

[139] An Open-Source Knight’s Tour Solver by Warnsdorff-Rule Algorithm. https://github.com/NiloofarShahbaz/knight-tour-warnsdorff

[140] Artificial Netlist Generator. https://github.com/daeyeon22/artificial_netlist_generator

[141] D. O'Laughlin, "Backside Power Delivery and Bold Bets at Intel", June 2022. https://www.fabricatedknowledge.com/p/backside-power-delivery-and-bold

[142] Cadence Design Systems, Inc., Sr Software Engineering Group Director, *Personal Communication*, 2019.

[143] Cadence Genus User Guide. http://www.cadence.com

[144] Cadence Innovus User Guide. http://www.cadence.com

[145] Cadence Liberate User Guide. http://www.cadence.com

[146] Cadence PVS User Guide. http://www.cadence.com

[147] Cadence QRC Extraction User Guide. http://www.cadence.com

[148] Cadence Voltus IC Power Integrity Solution User Guide. http://www.cadence.com

[149] Design Technology Team, Samsung Electronics Co. Ltd., *Personal Communication*, May 2018.

[150] FreePDK3 Predictive Process Design Kit. https://github.com/ncsu-eda/FreePDK3

[151] GDT to GDS Format Translator. https://sourceforge.net/projects/gds2

[152] O. Tange, GNU Parallel 2018. https://doi.org/10.5281/zenodo.1146014

[153] Gurobi Optimizer Reference Manual. http://www.gurobi.com

[154] H2O AutoML. https://www.h2o.ai

[155] IBM ILOG CPLEX. http://www.ilog.com/products/cplex

[156] IEEE International Roadmap for Devices and Systems (IRDS) 2020 Edition. https://irds.ieee.org/editions/2020

[157] i.MX RT600 Crossover MCU with Arm Cortex-M33 and DSP Cores Data Sheet. https://www.nxp.com/docs/en/data-sheet/DS-RT600.pdf

[158] LEF/DEF Reference 5.8. http://www.si2.org/openeda.si2.org/projects/lefdefnew

[159] Liberty Library Format. http://www.opensourceliberty.org

[160] lowRISC. https://www.lowrisc.org

[161] OpenCores: Open Source IP-Cores. http://www.opencores.org

[162] OpenDB. https://github.com/The-OpenROAD-Project/OpenDB

[163] OpenMP Architecture Review Board, "OpenMP Application Program Interface, Version 4.0".

[164] OpenSTA: Static Timing Analyzer. https://github.com/abk-openroad/OpenSTA

[165] K. Jeong, A. B. Kahng and H. Yao, RentCon: Rent Parameter Evaluation Using Different Methods. https://vlsicad.ucsd.edu/WLD/index.html

[166] Scaling The Lowly SRAM. https://semiengineering.com/scaling-the-lowly-sram

[167] Si2 OpenAccess. http://www.si2.org/?page=69

[168] Siemens EDA Calibre User Guide. https://eda.sw.siemens.com/en-US/ic/calibre-design

[169] C.-K. Cheng, D. Lee and D. Park, SMT-Based Standard-Cell Layout Generator for PROBE2.0. https://github.com/ckchengucsd/SMT-based-STDCELL-Layout-Generator-for-PROBE2.0

[170] SweRV RISC-V Core 1.1 from Western Digital. https://github.com/westerndigitalcorporation/swerv_eh1

[171] Synopsys Design Compiler User Guide. http://www.synopsys.com

[172] Synopsys IC Compiler II User Guide. http://www.synopsys.com

[173] Synopsys StarRC User Guide. http://www.synopsys.com

[174] Synopsys DTCO Flow: Technology Development. https://www.synopsys.com/silicon/resources/articles/dtco-flow.html

[175] The iPhone XS & XS Max Review: Unveiling the Silicon Secrets. https://www.anandtech.com/show/13392/the-iphone-xs-xs-max-review-unveiling-the-silicon-secrets/2

[176] The OpenROAD Project GitHub Repository. https://github.com/The-OpenROAD-Project

[177] The PROBE3.0 Framework. https://github.com/ABKGroup/PROBE3.0

[178] Z3 SMT Solver. https://github.com/Z3Prover/z3