UNIVERSITY OF CALIFORNIA

Los Angeles

Co-optimization of Restrictive Patterning Technologies and Design

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical Engineering

by

Yasmine Ahmed Zaki Badr

2017

ABSTRACT OF THE DISSERTATION

Co-optimization of Restrictive Patterning Technologies and Design

by

Yasmine Ahmed Zaki Badr
Doctor of Philosophy in Electrical Engineering
University of California, Los Angeles, 2017
Professor Puneet Gupta, Chair

As the semiconductor industry strives to find novel technology scaling methods, the advanced technologies (sub-15nm) have become restrictive to the design. The restrictions implied by each technology affect the design metrics like area, delay, and power. Thus, the semiconductor industry has resorted to Design and Technology Co-Optimization (DTCO) in order to develop technologies whose characteristics are desirable from the perspective of design as well as the fabrication. This gave rise to the need for fast design-informed technology evaluation methods.

In the first part of this dissertation, we propose frameworks to address challenges of developing a new technology. The evaluation of technology impact on design is traditionally inferred from the evaluation of Design Rules (DRs). The traditional approach of evaluating DRs on the standard cell level is misleading for two reasons. First, a lot of designs are routing-limited and, hence, not every change in cell area results in a corresponding change in chip area. Second, a design rule change, which leads to a change in the delay, can affect chip area due to the buffering and gate sizing techniques required to meet timing requirements. Thus, we present *Chip-DRE*, a framework for **Chip**-scale systematic **E**valuation of **DR**s and their interaction with layouts, performance, margins and yield.

Due to sub-wavelength lithography, layouts can have low printability and, accordingly, low yield due to the existence of bad patterns even though they pass design rule checks. For that purpose we propose *Pattern-DRE*, which is a framework for **Pattern**-driven **DR** **E**valuation. This framework can be used by the foundries to guide them on the relative importance of patterns to the routability of standard cells and to evaluate candidate new technologies from the routability aspect.

One of the very attractive candidate new technologies is Directed Self-Assembly (DSA), because it depends on the natural multiplicative capabilities of block copolymers, which can increase the resolution and at the same time it is a relatively low-cost technology. However, DSA imposes unique constraints on the design. Therefore, in the second part of this dissertation, we focus on algorithms to enable Directed Self-Assembly (DSA).

DSA requires the use of another lithography technique in order to print templates that guide the self-assembly process. Therefore, optimizing a DSA-based process requires the choice of another patterning technique as well as optimizing the properties of the block copolymer used. For that purpose we propose DSA-Pathfind, a tool for technology pathfinding for DSA. DSA-Pathfind can be used to make choices including the number of exposures needed in printing the templates, the natural pitch of the BCP and the relevant design rules, for the objective of design-friendliness.

In order to enable the adoption of DSA in the industry, fast and chip-scalable heuristics are needed for DSA-grouping and Mask assignment for the hybrid DSA-MP process. We present an efficient heuristic algorithm for that purpose. Results show that the proposed heuristics are 39x-192x faster on the average, and result in 12%-32% more violations, in comparison to the optimal problem solution. Then, we propose a heuristic for hotspot-aware DSA grouping and MP decomposition.

Finally, we look at potential non-traditional technology scaling boosters. We propose the use of a buried layer interconnect as a scaling booster. Results show that it can save chip area by 9-13%, with negligible hit on performance. We also evaluated the possibility of relieving routing congestion using *supervia*, a double-height via between two non-adjacent metal layers without a landing pad on the intermediate layer.

The dissertation of Yasmine Ahmed Zaki Badr is approved.

Jane Chang

Lei He

Lieven Vandenberghe

Puneet Gupta, Committee Chair

University of California, Los Angeles

2017

To my parents . . .

TABLE OF CONTENTS

List of Figures

List of Tables

ACKNOWLEDGMENTS

I am very grateful to my mentor and advisor Professor Puneet Gupta, for helping create the researcher I am today. The countless hours and the huge effort he spent in our technical discussions and in guiding me have been crucial to this dissertation. I am thankful to my committee members: Professor Jane Chang, Professor Lei He and Professor Lieven Vandenberghe, for their constructive comments and feedback on my research.

I was very fortunate to be mentored by Dr. Andres Torres, during my Summer 2014 internship at Mentor Graphics in Wilsonville, Oregon. I am thankful to him for the enriching discussions we had, the knowledge I learnt form him and the interesting work we did. Some of the research in this dissertation was based on the work I started with Dr. Torres.

In Summer 2016, I interned in the Computational Imaging Technology group at Intel in Hillsboro, OR, where I worked on a very interesting and challenging research problem. I am grateful to to Dr. Bikram Baidya, who was my manager and I was also very lucky to get to work again with Dr. Abde Ali Kagalwalla.

I would also like to thank my collaborators: Professor Andrew Kahng, Professor Subramanian S. Iyer, Dr. Rani Ghaida, Mukul Gupta, Ko-wei Ma, Liheng Zhu, Hyein Lee, Kwangsoo Han and Saptadeep Pal.

Funding for my PhD studies came from SRC, IMPACT, IMPACT+, C-DEN and NSF and thus, I am thankful to these centers and foundations. In addition, our meetings with the affiliated folks from industry provided excellent collaboration opportunities and inspired us with stimulating research ideas.

My 5-year PhD journey would not have been the same without my labmates in NanoCAD, each with his/her colorful personality. I have learnt a lot from each of: Dr. Rani Ghaida, Dr. Abde Ali Kagalwalla, Dr. Liangzhen Lai, Dr. Shaodi Wang, Dr. Mark Gottscho, Weiche Wang, Saptadeep Pal and Irina Alam. I will miss our good times, NanoCAD folks.

PhD is an exciting but tough journey and it is impossible to go through it without the incredible support of friends. Thank you, my friends: Hanzada Salah, Hawraa Salami, Hoda ElSafty, Hoda Shagar, Ladan Soroudi, Dr. Maria Yefimova, Mariam Lotfy, Nada Abdalla, Salma El-

Malaki, Sara Azzam, Dr. Sara Mustafa, and my youngest friend Hannah Elarabawy.

Finally, none of my accomplishments in life could have been possible if it weren’t for my family’s emotional support. Thank you, Mom, Dad and Nesma!

Vita

2009	B.Sc., Computer Engineering, Cairo University, Cairo, Egypt.
2009–2012	Research and Teaching Assistant, Computer Engineering Department, Cairo University.
2010	Research Assistant, Microsoft Advanced Technology Labs, Cairo, Egypt.
2010–2012	Part-time Software Development Engineer, Mentor Graphics, Cairo, Egypt.
2012	M.Sc., Computer Engineering, Cairo University, Cairo, Egypt.
2012–2013	Graduate Division Fellowship, Electrical Engineering Department, UCLA.
2012	Graduate Dean's Scholar Award, UCLA.
2014	PhD Intern, Mentor Graphics, OR, U.S.
2015	PhD Intern, Intel, OR, U.S.

Publications

Y. Badr, A. Torres and P. Gupta, "Mask Assignment and DSA Grouping for DSA-MP Hybrid Lithography for Sub-7 nm Contact/Via Holes", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 36, No. 6, pp. 913–926, June 2017

Y. Badr, and P. Gupta, "Technology path-finding framework for directed-self assembly for via layers", *SPIE Journal of Micro/Nanolithography, MEMS and MOEMS (JM3)*, Vol. 16, No. 1, pp. 013505-1–013505-13, June 2017

Y. Badr, and P. Gupta, “Technology Path-finding for Directed Self-Assembly for Via Layers” *SPIE Advanced Lithography Symposium*, February 2017

L. Zhu, Y. Badr, S. Wang, S. Iyer, P. Gupta, “Assessing Benefits of a Buried Interconnect Layer in Digital Designs”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 36, No. 2, pp. 346–350, February 2017

Y. Badr, A. Torres and P. Gupta, “Mask Assignment and Synthesis of DSA-MP Hybrid Lithography for sub-7nm Contacts/Vias”, *Design Automation Conference (DAC)*, June 2015

Y. Badr, A. Torres and P. Gupta, “Incorporating DSA in multipatterning semiconductor manufacturing technologies,” *SPIE Advanced Lithography Symposium*, February 2015

Y. Badr, K.-w. Ma, and P. Gupta, “Layout pattern-driven design rule evaluation,” *SPIE Journal of Micro/Nanolithography, MEMS and MOEMS (JM3)*, 2014

R. Ghaida, Y. Badr, and P. Gupta, “Pattern-restricted design at 10nm and beyond,” *Proc. IEEE International Conference on Computer Design (ICCD)*, 2014 [Invited]

Y. Badr, K.-w. Ma, and P. Gupta, “Layout Pattern-driven Design Rule Evaluation,” *SPIE Advanced Lithography Symposium*, March 2014.

R. Ghaida, Y. Badr, M. Gupta, N. Jin, and P. Gupta, “Comprehensive Die-level Assessment of Design Rules and Layouts,” *Proc. Asia and South Pacific Design Automation Conference (ASPDAC)*, January, 2014.

CHAPTER 1

Introduction

Sustaining Moore's law, which predicted that number of transistors on chip would double every two years, has not come for free. Manufacturing technologies for smaller nodes have become disruptive to the design; i.e. enforcing a lot of restrictions on it. Accordingly, developing a new technology requires the co-optimization of the design and the process, in order to meet both the fabrication and design requirements.

In this chapter, we shall give a brief overview of the Photolithography process as well as some of the disruptive technologies that are candidates for the future smaller nodes. Then we will talk about the process of Design and Technology Co-Optimization (DTCO). The chapter will be concluded with an outline of the dissertation.

1.1 Photolithography

The Photolithography process is the process used to print a given layout onto the silicon wafer to create an Integrated Circuit (IC). To explain this process, suppose we want to pattern the oxide layer with a certain shape, the following steps take place and are illustrated in Figure 1.1.

(i) A mask is made to depict the required shape for the oxide layer. The mask is a glass sheet, covered with chrome in the areas desired to print on the oxide.

(ii) The oxide layer is covered with a material called Photoresist. It is a chemical substance whose properties change when exposed to light. Let's assume positive photoresist is being used, which means photoresist is exposed where we want the underlying material removed.

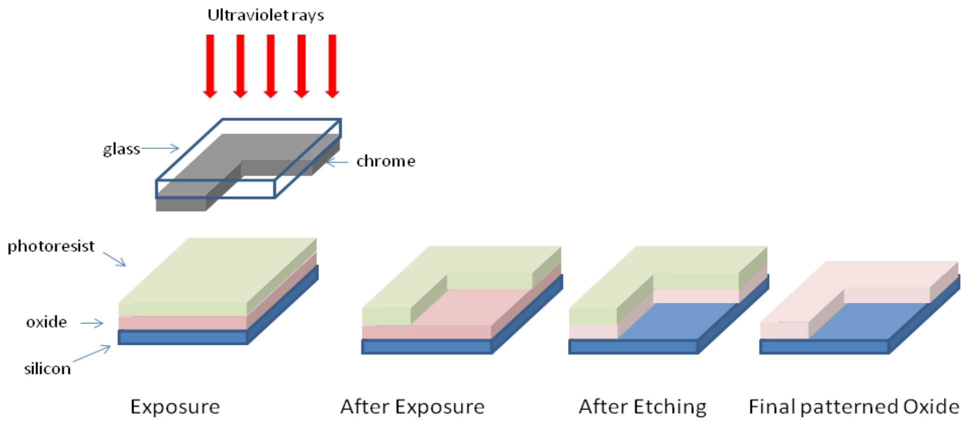


Figure 1.1: Photolithography Process

(iii) Ultraviolet rays are then projected through the mask, above the photoresist. Chemical properties of the photoresist will change in the areas that have been exposed to light, i.e. the areas that were not covered by chrome on the mask. These areas are removed, leaving only photoresist having the same shape as that on the mask (and required to print on the oxide).

(iv) The Etching process is done which removes the oxide material that is not covered by photoresist, leaving the oxide layer identical to the intended design.

(v) Finally the remaining photoresist material is removed.

The photolithography process imposes constraints on the minimum dimension of the polygons that can be printed. To print features with finer resolutions than what basic photolithography allows, novel manufacturing techniques are used as will be explained later.

The minimum feature size that can be patterned by the optical system used in photolithography is governed by Rayleigh criterion: $CD = k_1 * \lambda/\text{NA}$

- CD: Critical Dimension (smallest feature size)
- k_1: Process Difficulty Factor; its theoretical limit in a single exposure is 0.25
- λ: Wavelength of the ultraviolet light used; the currently used wavelength is 193nm. However Extreme Ultraviolet Lithography (EUV), which is not yet in production at the time of writing this dissertation, uses a wavelength of 13.5nm.

- NA: Numerical Aperture; maximum is 1.35 for water Immersion Lithography

Using the above values, the minimum CD is approximately 36 nm. However, using technologies like Multiple Patterning (MP) and Directed Self Assembly reduces the effective k_1 achieving smaller nodes. [FCC12]

1.2 Disruptive Patterning Technologies

In this section, we briefly explain two types of patterning technologies which impose restrictions on the design.

1.2.1 Multiple Patterning

Multiple patterning (MP) is the process of splitting a layout mask into multiple masks, and applying photolithography to each mask separately one after the other. The final printed layout will be the same as the original mask. However, by printing the target layout on multiple lithography steps, each mask used in each has larger spaces and less density. This way, we can print layouts of a smaller node (e.g 22 nm) while the manufacturing process can still use the tools of older nodes (e.g. 32 nm) and thus effectively multiplying layout density. Thus, the industry can still fulfill Moore's law.

Double Patterning (DP) is Multiple Patterning; when only two masks are used. A simple example for a type of DP called Litho Etch Litho Etch (LELE) can be seen in Figure 1.2. Notice how the combination of the two sparse masks on the right gives the original mask on the left. This allows for more (smaller) features on the same chip area.

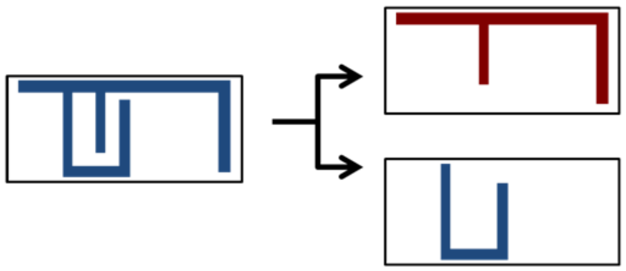


Figure 1.2: Example of mask splitting in LELE

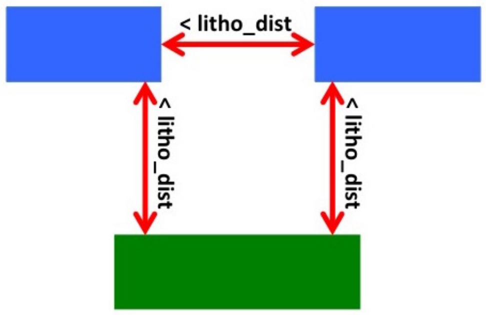


Figure 1.3: An LELE-incompliant pattern

1.2.1.1 Types of DP

There are two main types of Double Patterning:

(i) **Litho Etch Litho Etch (LELE)**

In this type of DP, a layout is split into two masks such that the mask features are distributed among them to relax the required pitch on each individual mask. The target layout is the result of ORing of both masks. Each of the two masks undergoes the normal single-exposure method that includes a photolithography step followed by etching, and hence the name. The splitting example of Figure 1.2 demonstrates how the layout is decomposed into two masks in LELE.

However, not all layouts are LELE-compliant. For example, Figure 1.3 can not be decomposed into two violation-free masks (i.e. with spacing values exceeding the minimum single exposure space (litho_dist)) , and accordingly this pattern is not LELE-manufacturable because it has a cycle between an odd number of polygons.

(ii) **Self Aligned Double Patterning (SADP)**

In SADP, the two masks are called mandrel and trim masks. In addition, a sidewall or spacer is deposited around the mandrel polygons. The mapping from the mandrel and trim masks to the final patterns is not as straight forward as LELE.[1] Instead the target layout is characterized by the boolean relationship: Target Layout = NOT {TRIM OR SPACER}. This means that any area covered by solid part in trim or by spacer will not

[1]This description is specific to a tone of SADP called Spacer Is Dielectric (SID), with using a cutting trim mask.

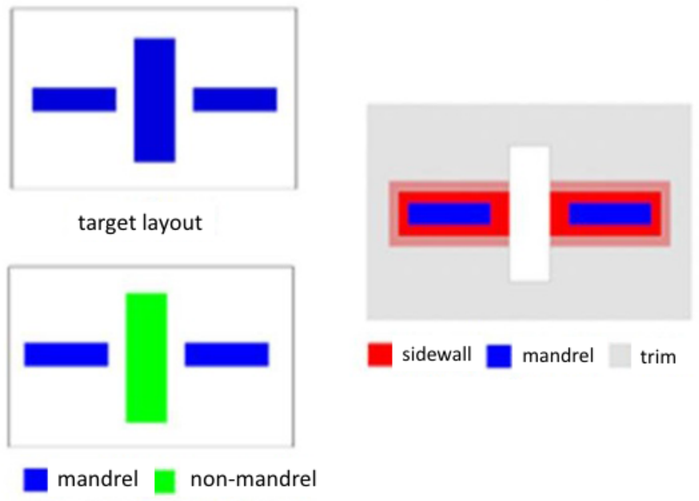


Figure 1.4: Example of mask splitting in SADP

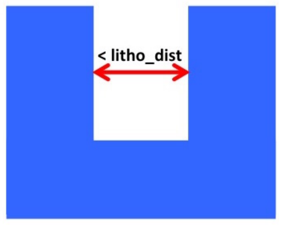


Figure 1.5: An SADP-incompliant pattern

print in the final layout. Figure 1.4 shows an example of SADP layout decomposition for a part of a layout.

Also for SADP, there are some incompliant patterns. A major reason for that is the fact that in SADP every polygon is either created by mandrel mask or trim mask. Unlike LELE where a polygon can be divided into multiple segments that are distributed among the two masks. This is because the spacer is deposited around the mandrel creating an empty space in the final layout; thus ”stitches” are not allowed. In addition, because the deposited spacer width is constant, a lot of spacing values in some pattern configurations are disallowed in SADP. Figure 1.5 shows a pattern which is SADP-incompliant (Note that this pattern could have been printed using a stitch in LELE.)

In this dissertation, whenever the term **DP** is used without mentioning the specific type (LELE or SADP), then the reference is to LELE.

1.2.2 Directed Self Assembly (DSA)

Self-Assembly is the phenomenon that occurs when block co-polymers composed of immiscible blocks phase-separate into organized structures [XPN09]. For example, a diblock co-polymer (with special composition) can self-assemble into periodic structures of one type of block into a matrix of the other. Lithographically-printed patterns (in a scheme called Graphoepitaxy) or chemically-treated surfaces (in a scheme called Chemoepitaxy) are used to *direct* the self-assembly process [Jar13]. The realizable assembled pitch depends on the characteristics of the used block co-polymer. The graphoepitaxy process for contact holes is shown in Figure 1.6, where trenches are lithographically printed first, then the surface is spin-coated with the block co-polymer (BCP) solution. Upon thermal annealing, the phase separation occurs, and with a particular BCP and surface treatment of substrate [KHS13] (not shown on the figure), we get cylinders of one block in a matrix of the other block.

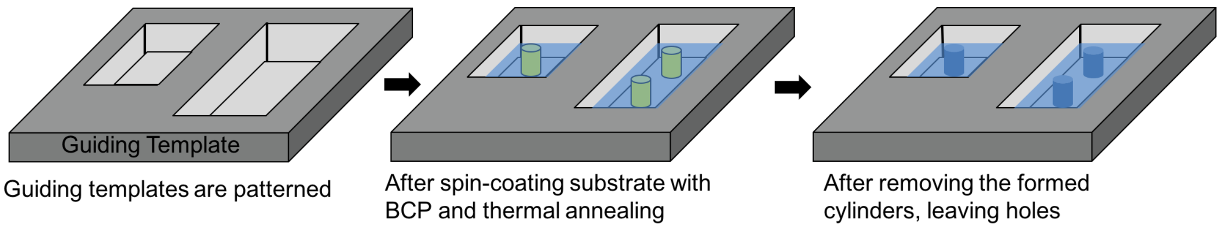

Guiding Template
Guiding templates are patterned
After spin-coating substrate with BCP and thermal annealing
After removing the formed cylinders, leaving holes

Figure 1.6: An example directed self- assembly process of a diblock co-polymer using Graphoepitaxy

1.3 Design Technology Co-Optimization (DTCO)

Historically for the simple large nodes, the fabrication houses produced Process Design Kits (PDKs) including the design rules and transistor models, and designers used these PDKs which were scaled down along the transition to smaller nodes. Around 90nm, PDKs started imposing restrictions on the design in order to achieve higher yield, and the designers used to optimize the design within the provided restrictions, so it was only one-way communication (from fab to designer) [YCS13]. With the current nodes being achieved by ultraviolet light of wavelength (193nm) much larger than the resolution which led to the use of disruptive technologies that have non-manufacturable patterns, and with additional constraints from the use of specific materials in order to meet device performance scaling [Nor11], and with the use of finfets and other new devices instead of planar CMOS; a lot of trade-offs and contradicting options have to be explored in the process definition phase. Through DTCO, the foundries can define the new node including the patterning technology, design rules, and device type[Nor11]. Moreover, area of standard cells is no longer an appropriate metric for a selection of technology options. Sometimes standard cells of larger area can lead to smaller chip area, due to better performance[GBG14, YCS13]. This will be tackled in length in Chapter 2.

To enable the scaling, the design and technology have to be co-optimized. The DTCO process aims at getting the design metrics involved in the defining the new technology from the beginning [Nor11]. The process starts with designers providing a set of circuit designs using rough estimate of technology and patterning capabilities [Nor11]. These layouts are then used back and forth between the technology optimization and design change until a solution which is manufacturable and meets design objectives is reached.

One can imagine how lengthy this process would be; an iterative, non-systematic process that requires manual design at least for the standard cell design. Yet, a new technology is expected every two years according to Moore's law. Thus frameworks are strongly required to automate the process of getting design-informed feedback to the technology developers, and eliminate -as much as possible- the manual work involved in the process, achieving a timely transition to the new nodes.

1.4 Dissertation Outline

This dissertation first presents frameworks to automate the Design and Technology Co-Optimization (DTCO) phase that occurs before a new technology node comes into production. Then, the dissertation focuses on a particular disruptive technology **Directed Self Assembly (DSA)** and presents algorithms to enable DSA. Finally, we propose two technology scaling boosters.

The rest of the dissertation is organized as follows:

Part I: Frameworks to evaluate new technology nodes

Chapter 2 - Comprehensive Die-Level Assessment of Design Rules and Layouts: In this chapter, we propose a framework which performs design rule evaluation on the chip-level, as opposed to the standard cell level.

Chapter 3 - Layout Pattern-Driven Design Rule Evaluation: In this chapter we present a framework which can evaluate the effect of forbidding specific patterns on the routability of the standard cells. In addition, this framework can be used to compare different disruptive patterning technologies.

Part II: Computational Methods to enable DSA

Chapter 4 - A Technology Path-finding Framework for DSA for Via Layers: A DSA-based technology requires a complementary lithography technique in order to print the guiding templates. Therefore, the optimization of a DSA technology requires optimizing the dimensions of the block copolymer and making choices for the complementary lithography technique. Therefore, so many combinations of choices exist. In this chapter we propose an optimal framework for path-finding for DSA for via layers. The framework is used to make choices such as the number of masks used, the type of the complementary lithography, dimensions of the block copolymer, in order to have a design-friendly technology.

Chapter 5 - Mask Assignment and Synthesis of DSA-MP Hybrid Lithography for sub-7nm Contacts/Vias: The integration of DSA and MP is one of the candidate patterning technologies for the sub-7nm nodes. In this chapter, we propose a heuristic algorithm for performing the mask assignment along with the DSA grouping, assuming 193nm Lithography is used to pattern the guiding templates.

Chapter 6 - Hotspot-aware DSA Grouping and Mask Assignment: In this chapter we present an algorithm for hotspot-aware DSA grouping and MP decomposition. Hotspots can be due to lithographic imperfections or intrinsic self-assembly defects.

Part III: Technology Scaling Boosters

Chapter 7 - Assessing Benefits of a Buried Interconnect Layer in Digital Designs: We propose the use of a buried interconnect layer in the standard cells, in this chapter. Result show that buried interconnect reduces routing congestion and saves up to 13% in chip area.

Chapter 8 - Supervia: Relieving Routing Congestion using Double-height Vias: This chapter evaluates the density scaling benefit of using double-height vias in logic layouts.

Chapter 9 - Conclusion In this chapter, the conclusion of the research performed for this dissertation is summarized.

Part I

Frameworks to evaluate new technology nodes

CHAPTER 2

Comprehensive Die-Level Assessment of Design Rules and Layouts

Co-development of design rules and layout methodologies is the key to successful adoption of a technology. In this work, we propose Chip-level Design Rule Evaluator (ChipDRE), the first framework for systematic evaluation of design rules and their interaction with layouts, performance, margins and yield at the chip scale (as opposed to standard cell-level). A "good chips per wafer" metric is used to unify area, performance, variability and yield. The framework uses a generated virtual standard-cell library coupled with a mix of physical design, semi-empirical, and machine-learning-based models to estimate area and delay at the chip level. The result is a unified design-quality estimate that can be computed fast enough to allow using ChipDRE to optimize a large number of complex design rules. For instance, a study of well-to-active spacing rule reveals a non-monotone dependence of rule value to chip area (although the dependence to cell area is monotone) due to delay changes coming from well-proximity effect.[1]

2.1 Prior Work

The evaluation of technology impact on design is traditionally inferred from the evaluation of Design Rules (DRs), which are the biggest design-relevant quality metric for a technology. Unfortunately, even after decades of existence, DR evaluation is largely unsystematic and empirical in nature; it relies on limited and small-scale experiments and manufacturing tests and much on speculations based on technologists/designers experience with previous technology generations [CGK04, ZCY08, DCY09, CBP09]. The work in [DMY11] presents a flow for the

[1]The material in this chapter is based on the published work [GBG14].

optimization of double-patterning design rules. The method consists of an optimization loop in which rules are modified, standard-cell layouts are generated, and printability is analyzed. Although this approach, like [DCY09, CBP09], may be suited for exploring rules from a pure printability perspective, it does not examine the electrical effects of rules. Moreover, because actual layout generation and printability analysis are excessively time-consuming, exploring a wide range of rules and rule combinations is impractical with these approaches.

More recently, the work of [GG12] offers a framework for evaluating design rules, at early stages of technology development, through fast layout-topology generation of standard-cell layouts and estimation of variability and manufacturability using first-order models. This work has two major limitations. First, the evaluation was performed at the cell-level, which may lead to false conclusions because most designs are routing-limited and, hence, *not every change in cell area results in a corresponding change in chip area*. Second, delay was not evaluated but it is well-known that *delay-change can affect chip area* due to techniques like buffering and gate sizing required to meet timing requirements.

2.2 Contribution of this work

In this work, we propose Chip-level Design Rule Evaluator (ChipDRE), the first framework for systematic evaluation of design rules and their interaction with layouts, performance, margins and yield at the chip scale. ChipDRE uses a "good chips per wafer" (GCPW) metric to unify area, performance, variability and functional yield. It uses a generated virtual standard-cell library coupled with a mix of physical design and semi-empirical models to estimate area, delay and yield at the chip level. To predict the design-rule/layout impact on delay and delay variability, ChipDRE employs a Static Timing Analysis model to estimate cell-delay and a neural network-based model to predict delay-margin dependent area penalty. Chip-level area is estimated from cell area – including the delay-margin area penalty – and a cell-area to chip-area model that is calibrated using actual Synthesis, Place and Route (SPR) data. Finally, GCPW is calculated taking into consideration a chip-level functional yield estimate. The result is a unified design-quality estimate that can be computed fast enough to allow using ChipDRE to optimize a large number of complex design rules and achieve "true" design/technology co-optimization.

We make the following contributions.

- We offer ChipDRE, the first framework for collective evaluation of design rules, layout styles, and library architectures *at the chip scale*. ChipDRE is designed to be *used for design/technology co-optimization* and supports state-of-the-art technologies including FinFETs and Local Interconnects (LI). It aims at making rule generation and optimization easier and much faster. Rather than exploring the entire search space of design rules manually or with conventional compute-expensive methods, the framework can be used to quickly eliminate poor rule choices.

- We develop a cell-delay estimator and a neural network-based model to project the impact of cell-delay change on the overall chip area.

- We propose a cell-area to chip-area model to project how cell area translates into chip area.

- We evaluate the rule impact on delay and report the evaluation in terms of GCPW unifying area, performance, variability and functional yield metrics. This comprehensive evaluation allows studying interesting trade-offs that occur at the chip level like the one between variability, performance and area.

- We perform evaluation studies of major design rules at advanced nodes (some FinFET-specific) including: gate to local-interconnect spacing, gate-to-well edge spacing and fin pitch.

The remaining chapter is organized as follows. Section 2.3 gives an overview of our approach. Sections 2.4 elaborates the cell-delay estimation and the virtual standard-cell layout generation including I/O pin-access estimation and supporting FinFET and local-interconnect technologies. The cell-area to chip-area model is described in Section 2.6, while the model to predict delay-margin dependent area penalty is described in Section 2.5. Section 2.7 presents the results of a number of evaluation studies at 45nm technology node using ChipDRE. Finally, Section 2.8 gives a brief summary of the chapter and some directions for future research.

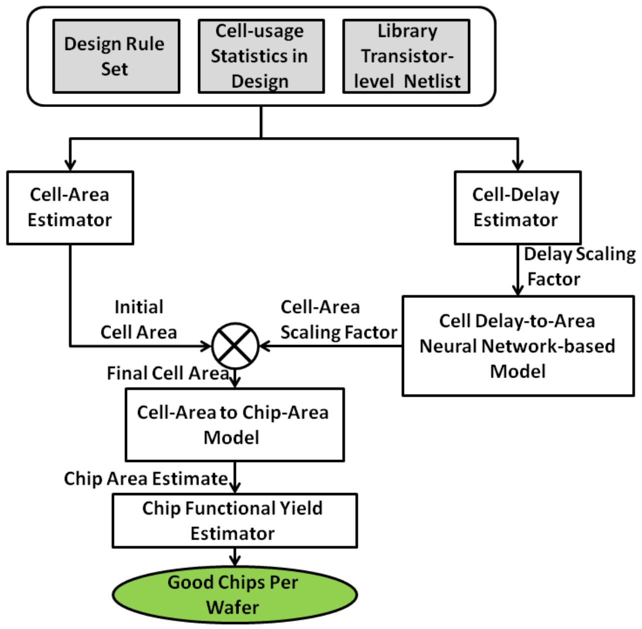


Figure 2.1: Overview of ChipDRE and its main components.

2.3 Overview and Standard-Cell Layout Estimation

In this section, we give an overview of ChipDRE and briefly describe its components. We also present our approach for cell-layout estimation.

2.3.1 Overview

An overview of ChipDRE is depicted in Figure 2.1. The framework takes the following inputs: transistor-level netlists (SPICE) of cells, rules and their values, estimates of process control (e.g., overlay error distribution), and cell-usage statistics of the design to evaluate the rules on. In ChipDRE, only the values of rules under evaluation are modified while all others remain unchanged. This modified set of rules is then used to estimate the cell-layout and perform the design-level evaluation.

Concisely, the first stage of ChipDRE is to estimate the cell layout/area and cell delay for a given set of rules. If the cell delay changes in comparison with the delay obtained using a base set of rules, the cell-delay change is converted into a delay-scaling factor which is used to scale the timing characteristics of the standard-cell library (in Liberty file format). A neural network-based model is then used to estimate the impact of cell-delay change on the design overall cell area (Figure 2.2 manifests the significance of this impact). The model essentially predicts gate-sizing and buffer-insertion to meet the timing requirements with the new cell-delay characteristics. In the second stage of ChipDRE, another semi-empirical model – fitted to SPR data – is used to predict how the cell area translates into chip area. The final stage of ChipDRE, chip-level functional yield is estimated and a unified design-quality metric, number of "good chips per wafer" (GCPW), is calculated.

2.3.2 Cell-area Estimation

The cell-area estimator is based on the virtual-cell generator from [GG12]. This generator[2] accurately estimates cell area ($< 1\%$ error [GG12]) through fast generation of front-end-of-line (FEOL) layers and congestion-based estimation of wiring area. In this work, we extend the cell-layout estimator of [GG12] to enable its application at the chip level and using state-of-the-art technologies (e.g., FinFETs).

For chip-level evaluation, we generate I/O pin segments and the physical specifications of the technology and standard-cells (in Library Exchange Format or LEF). In studies presented in this work, pin segments are kept at minimum possible dimensions while meeting the minimum area design rule. We first sort vertical pins from left to right and horizontal pins from bottom to top. We then assign pins sequentially to the closest available track without creating DR violations. It is worth noting that we allow three pin configurations: (1) all pins on M1, (2) all pins on M2, and (3) pins on either M1 or M2 layers. In case of (3), a pin will be assigned to M1 by default and moved to M2 if doing so helps resolving M1 congestion in the cell (see Figure 2.3 for an example). In all our experiments, we use pin configuration (3).

FinFET technology with local-interconnect layers will be standard across the industry at advanced nodes (22nm and below [McG]). Hence, to enable rule-evaluation at advanced nodes, we extend the layout-generation of front-end-of-line layers to include additional local-interconnect and FinFET-specific layers. The additional layers are: CA, CB, and fin-layer. CA is the vertical

[2]Publicly available at nanocad.ee.ucla.edu.

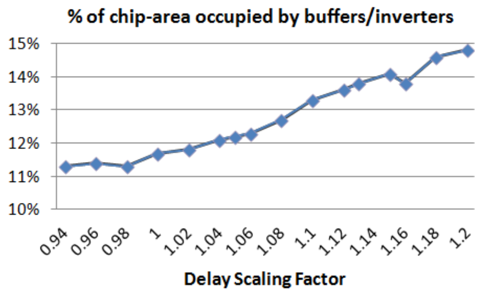


Figure 2.2: Empirical data from placement-aware synthesis commercial tool manifesting the impact of cell delay on the percentage of chip area that is occupied by buffers/inverters.

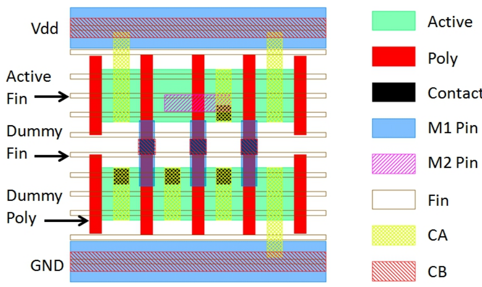


Figure 2.3: Example layout for OAI21_X1 cell generated by ChipDRE with FinFETs, local interconnects (i.e., CA and CB layers), and DR violation-free I/O pin segments.

local-interconnect layer and is used to connect the fins of the same FET together[3], primarily to make contact from the contact-layer to the fins. CA can also be optionally used to make power/ground connection to the FETs (when a local-interconnect power rail exists). CB is the horizontal local-interconnect layer and is used to make contact from the contact-layer to Poly and to make short Poly-to-Poly connections when possible. The fin layer constitutes the actual FinFETs, referred to as active fins, and dummy fins, which are necessary to conform the fin layer to a grid and ensure printability. The fin grid needs to be in accordance with the cell-height so that it is maintained after cell-placement in the design. This constraint makes finding a valid configuration of fin count and pitch in active regions (P/N networks) as well as top, bottom, and center overhead regions complex. Given a range of allowed fin pitches, we run an exhaustive search to find a working configuration with maximum number of total active fins in one column and the smallest active fin pitch. To improve the chances to reach a better solution, we optionally allow the dummy fin pitch in top/bottom/center overhead regions to differ and allow the cell top/bottom edges to coincide either with the center of the fin (as in Figure 2.3) or with the center of the dummy fin-to-fin spacing.

[3]Note this is optional when the source/drain is not contacted

To migrate a planar FET-based netlist to a FinFET-based netlist, we employ the following model to determine the number of fins for every transistor:

$$n = \lceil \frac{W}{\alpha \times F_H} \rceil, \tag{2.3.1}$$

where W is the transistor width specified in the planar-based netlist, F_H is the fin-height, and α is a planar-to-finFET width translation parameter[4]. The rounding up of number of fins in Equation 2.3.1 is done to ensure the minimum transistor performance is preserved after the migration.

2.4 Variability-Aware Cell-Delay Estimation

A crucial aspect of Design Rule Evaluation is the assessment of the impact of the DRs on performance. To characterize a digital chip-level delay, it is required to model the delay for each standard cell. First-order delay models are employed in order to have a fast and approximate delay estimation.

2.4.1 Cell Delay Model

To characterize the cell rise or fall delay, the cell is considered as a sequence of stages and the delays of these stages are then added up. For each stage, all paths connecting the output to the power supply (Vdd or ground) are enumerated. An RC tree is constructed for each path and Elmore delay [Elm48] is applied to compute the path delay [RCN04]. The worst case pull up and pull down delays are determined for each stage. Identical paths (paths that switch simultaneously) are considered as parallel resistances and their capacitances are added up.

[4]We use $\alpha = 2$ in our fin-pitch experiment like [MMJ11]. A higher value of α can be used to take into account the contribution of the top gate as well as the triangular profile of FinFETs.

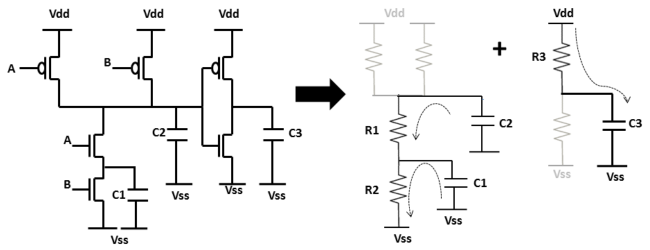


Figure 2.4: Estimation of low-to-high propagation delay for AND gate, equivalent RC tree and charge/discharge paths. It consists of two stages, the pull-down network for the NAND gate followed by the pull-up network of the inverter. Using Elmore delay and adding up delay stages, the propagation delay for the cell rise is estimated as: t_{pLH}=R_1 C_1+(R_1+ R_2) C_2+R_3 C_3. C_1, C_2 and C_3 include all the gate and diffusions capacitances connected at each of the 3 nets.

2.4.2 Transistor Model

We apply an RC approximation for each transistor where the capacitance model [RCN04] considers the gate capacitance (including channel and overlap capacitances) as well as diffusion capacitances, and accounts for Miller effect. The MOS switch model in [RCN04] is used to estimate the equivalent resistance R_{on} of the transistor.

To model delay variability and consider the worst case delay, we use the current variability estimates from [GG12] which primarily models layout-dependent, lithography-induced variations in drive current. Variability is computed as 3σ change in current which is subtracted from the nominal current value before calculating resistance. As an example, we illustrate the pull-up of an AND gate in Figure 2.4.

2.4.3 Verification and Results

For verification, we used NCX [NCX] with HSPICE [HSP12] to generate the liberty file for some standard cells from Nangate Standard Cell Library [Nana]. The worst cases for cell rise and cell fall were compared to the values reported by ChipDRE delay estimator, using the same load capacitance.

Gate Length Scaling Experiments. For these experiments, the gate length rule was scaled by 10%, and the scaling factor of the ChipDRE-estimated delay (i.e. the ratio between delay at the scaled gate length to the delay at the original length) was compared to the scaling factor ob-

Table 2.1: Verification of Delay model using gate-length scaling experiments by comparing the ChipDRE-estimated delay scaling factor to the scaling factor from Spice

	Pull-up			**Pull-down**		
Cell	**ChipDRE**	**Spice**	**Abs Error (%)**	**ChipDRE**	**Spice**	**Abs Error (%)**
INV_X32	1.10	1.09	0.9	1.10	1.07	3
NAND2_X1	1.10	1.10	0.3	1.10	1.07	3
INV_X1	1.08	1.09	1.1	1.08	1.07	1.1
AND2_X4	1.10	1.09	0.8	1.10	1.09	1.2
OAI21_X2	1.10	1.09	0.7	1.10	1.07	2.6
AOI211_X1	1.10	1.09	0.5	1.10	1.08	2.2
OAI33_X1	1.10	1.10	0.3	1.10	1.08	2.1
AND2_X2	1.10	1.09	0.8	1.10	1.08	1.6
Average	1.1	1.09	0.7	1.1	1.08	2

Table 2.2: Verification of Delay model using Well-Proximity Effect (WPE) experiment by comparing the ChipDRE-estimated delay scaling factor to the scaling factor from Spice

	Pull-up			**Pull-down**		
Cell	**ChipDRE**	**Spice**	**Abs Error(%)**	**ChipDRE**	**Spice**	**Abs Error(%)**
INV_X32	0.96	0.96	0.6	0.96	0.97	0.8
NAND2_X1	0.76	0.78	2.4	0.85	0.88	2.8
INV_X1	0.76	0.78	2.1	0.79	0.84	5.4
AND2_X4	0.93	0.92	1.5	0.89	0.84	6.6
OAI21_X2	0.93	0.93	0.1	0.93	0.94	1.0
AOI211_X1	0.89	0.89	0.3	0.85	0.85	0.5
OAI33_X1	0.89	0.89	0.8	0.85	0.88	3.7
AND2_X2	0.89	1.00	11.0	0.87	0.94	7.3
OR2_X2	0.87	0.88	1.4	0.88	0.88	0.2
Average	0.88	0.89	2.4	0.88	0.89	3.5

tained by our spice simulation setup. Table 2.1 lists the scaling factors obtained from ChipDRE and spice, as well as the magnitude of the error which does not exceed 3%.

Well-Proximity Effect (WPE) Experiment. To model the Well-Proximity effect, BSIM [BSI] model for WPE impact on threshold voltage and mobility was used. Values of the model's parameters were computed as in [Cou]. The gate-to-well distance value in the BSIM model was scaled down by 10%, and the corresponding delay values were computed. The ratios of cell delay with scaled gate-to-well distance to original cell delay were compared to the equivalent ratios obtained using Spice [HSP12] simulation. Table 2.2 shows the comparison between the ratios obtained by ChipDRE to those obtained by Spice and the corresponding error which is below 7.3%.

2.4.4 Liberty Delay File Generation

For the baseline set of design rules, we assume a Liberty file [5]. To generate the liberty file for virtual standard-cell library corresponding to the set of rules under evaluation, the worst-case pull-up and pull-down delays for the gates are computed as explained in section 2.4.1. This is also done for the baseline set of design rules to create a reference gate delay (computed by ChipDRE). The ratios between the gate delays in the case of design rules under evaluation and those of the baseline design rules are used to scale the baseline liberty file to obtain an estimated Liberty file for the design rules under evaluation. For sequential elements, their hold and setup times are left unchanged (same as baseline liberty file), and their clock to output delay is scaled by the same scaling factor as inverter. The entire flow of generating layouts, estimating delays and generating the Liberty file within ChipDRE takes less than 49 minutes for a 100 cell library as opposed to commercial library characterization tools which take several CPU days.

2.5 Delay-to-Area Modeling

One of the major issues ChipDRE addresses which typical cell-based design rule optimization approaches suffer from is the effect of timing optimization - during physical synthesis - on area. Physical synthesis tools use several optimization techniques to meet timing constraints at the minimum possible area, like gate sizing, buffer insertion and logic restructuring. Thus, as delay of standard cell increases, we can expect an increase in the resultant chip area. Previous work [JKS09] has experimentally characterized the impact of timing guardband reduction on some metrics of the circuit by running synthesis, place and route for several scaled libraries. However, this is impractically slow to explore design rule choices. Moreover, the work of [KM02] has demonstrated that little noise can have huge effect on place-and-route solution quality; this makes using a model-based estimate even more attractive.

Modeling these optimization techniques analytically is complicated with a tremendous num-

[5] This could be a characterized or scaled version from a previous technology node. The absolute values of delays in the Liberty file are not very important for ChipDRE as we are more interested in relative delay changes with rule changes.

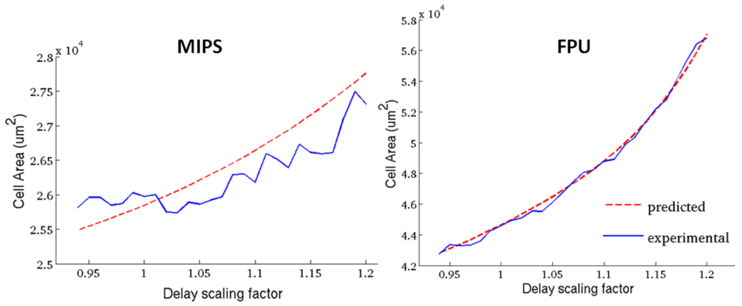


Figure 2.5: NN testing on MIPS design (a blind test case) and on FPU (used in training). This network has been trained resulting in a training mean square error of 8.16x10^4

ber of degrees of freedom. Thus we use a machine learning technique to predict the cell-area scaling factor (ratio between the cell area of the design at some delay scaling factor to the baseline cell area of the same design) as the standard cells delay scales (due to a change in DRs).

A neural network has been trained using data from physically-aware synthesis performed using [RCP][6]. To train the neural network (one hidden layer with 6 nodes), the following features have been used: number of instances on critical path, average fanout, average interconnect length, average delay and area of gates on critical path, utilization, timing constraint, ratio between area of critical paths to the total cell area and the delay scaling factor. Those features have been selected because they affect the amount of buffering and gate sizing performed by the tool to meet the timing constraints. We assumed that there is no change to the back end rules and only the front end rules are undergoing change and evaluation. Otherwise, other features need to be added like capacitance and resistance of the used metal layers per unit length.

The network was trained – using Matlab Neural Network Toolbox – on 27 delay scaling factors (each time the liberty file being scaled) from 9 test cases; 3 from [ope] and 6 from ISCAS85 benchmarks. Upon testing the network on MIPS design from [ope] (not used in training), the neural network was able to predict the cell-area scaling factor – used to calculate cell area – and rule out tool noise as shown in Figure 2.5. The figure also shows the performance of the neural network on one of the training test cases, the FPU design (from [ope]).

[6]Physically-aware synthesis, which performs placement to estimate interconnect delay, has been used since it takes less time than the complete time-consuming place and route and yet produces estimates that are accurate enough for our purpose.

2.6 Chip Area and Yield Modeling

2.6.1 Minimum Routable Area

Minimum Routable Area (MRA) of a design requires the estimation of maximum utilization at which the number of DR violations cease to be zero. This implies that for finding MRA multiple Place & Route (P&R) runs are required, making the whole process time consuming (detailed routing being the main culprit). For instance, an experiment to estimate MRA of AES ($\sim$10K gate design) using binary search took 14 hours (as shown in column 6 of Table 2.3). Such excessive runtime makes chip-level evaluation of multiple design rules impractical.

Thus, we propose a new methodology, Area Estimation using Global Routing (AEGR) that estimates MRA using global routing congestion estimates. Global routing congestion estimates require the estimation of wiring demand and wiring supply on each of the global routing cell – called G-cell – which represents a fixed number of available routing tracks in each layer. If wiring demand exceeds supply, the detailed routing is unlikely to implement a design rule correct wire pattern. Congestion in an arbitrary G-cell is given by

$$C = \frac{\text{routing demand (d)}}{\text{routing supply (s)}}. \tag{2.6.1}$$

SPR tools cannot resolve all instances of congestion and for very high congestion values, the tool might not find enough unused G-cells to successfully route the design. Hence we propose that there exists a threshold on congestion beyond which tool cannot successfully route the design. Based on this we define a metric, $m(u)$, in the following manner

$$m(u) = \alpha \times C_{peak}(u) + \beta \times C_{avg}(u), \tag{2.6.2}$$

where C_{avg} is the average congestion over all G-cells and C_{peak} is the maximum congestion over all G-cells , and α and β are the tool dependent parameters. The utilization u_{max} for which m(u_{max}) is 1 is classified as the maximum utilization of the design.

To further refine the estimation of maximum utilization, we run detailed routing in the range $[0.9u_{max}, 1.1u_{max}]$ to get two utilization values where number of DR violations is greater than zero. Then linear extrapolation is done using these two points to estimate the utilization value where number of DR violations is equal to zero. This estimated utilization value is termed as the maximum utilization value. Using this methodology substantial runtime improvement was achieved as we show later in this section.

2.6.2 Model Formulation

Although AEGR gives substantial improvement in runtime, it still requires running Place & Route (P&R) for all the designs and large number of FEOL design rules (increasing with every new technology node). Also, tool noise leads to problems in optimization. To overcome these problems, we model chip area as a function of total cell area thereby skipping P&R to the maximum possible extent. Our proposed model in differential form is given in Equation (2.6.3). Here y is the chip area and x is the total cell area. $\frac{x}{y}$ is the utilization of the design. In the proposed model, as the utilization increases or equivalently white space decreases, change in chip area is more sensitive to any change in cell area. The final analytical equation is given in Equation (2.6.4).

$$\frac{dy}{dx} = k1 - k2 \times (y/x). \tag{2.6.3}$$

After solving Equation (2.6.3), we get

$$y = \frac{k1}{k2+1} \times x + \left(y0 - (\frac{k1}{k2+1}) \times x0 \right) \times (\frac{x}{x0})^{-k2}. \tag{2.6.4}$$

There are four unknowns in the model viz. $k1$, $k2$, $x0$ and $y0$. $y0$ can be thought of as the routing limited chip area. $x0$ can be thought of as any unutilized whitespace area[7] when the chip area is $y0$. $x0$ depends on the cell routability which in turn is dependent on the pin access and congestion within the cell [TLA10]. Larger congestion implies router needs to drop more vias outside the cells to make connections with the cell instance pins, effectively decreasing any unutilized whitespace and hence decreasing $x0$.

[7]chip area minus the area required by the router to make connection with the cell instance pins using M1 layer.

Table 2.3: Runtime comparison between Area Estimation using Global Routing (AEGR) method and actual P&R for finding the minimum routable area.

Design	Routing Layers	AEGR Util.	P&R Util.	Runtime in mins (AEGR)	Runtime in mins (P&R)	Runtime Reduction
MIPS	3	0.83	0.83	97	322	3.3x
MIPS	4	0.97	0.97	23	145	6.3x
JPEG	3	0.93	0.93	345	892	2.6x
AES	3	0.44	0.47	57	1267	22x
AES	4	0.76	0.76	110	842	7.6x
AES	5	0.85	0.84	52.4	141	2.7x
FPU	3	0.91	0.90	52	261	5x
NOVA	4	0.88	0.88	296	519	1.8x

To find $k1$ and $k2$ we apply the following boundary conditions

$$k1 - k2 = 1, \tag{2.6.5}$$

$$k1 - k2 \times \frac{y0}{x0} = 0. \tag{2.6.6}$$

Equation (2.6.5) is based on the fact that for very high utilization values, change in chip area is roughly equal to change in total cell area. This implies that as $u \to 1$, $\frac{dy}{dx} \to 1$. Hence the boundary condition follows from Equation (2.6.3). Similarly from the other extreme, for any total cell area less than $x0$, chip area is routing limited and is equal to $y0$. Hence, Equation (2.6.6) follows from Equation (2.6.3). Based on these boundary conditions, model coefficients and final analytical equation are given by

$$k1 = \frac{y0}{y0 - x0}, \tag{2.6.7}$$

$$k2 = \frac{x0}{y0 - x0}, \tag{2.6.8}$$

$$y = x + (y0 - x0) \times \left(\frac{x0}{x}\right)^{\frac{x0}{y0-x0}} \text{ for } x > x0, \tag{2.6.9}$$

$$y = y0 \text{ for } x <= x0. \tag{2.6.10}$$

Since $y0$ and $x0$ are design dependent parameters, we estimate them by actual P&R runs for each design under consideration. $x0$ and $y0$ need to be estimated only once for a given back-end interconnect stack and library architecture. This gives substantial improvement in runtime making it possible to simultaneously evaluate large number of design rules.

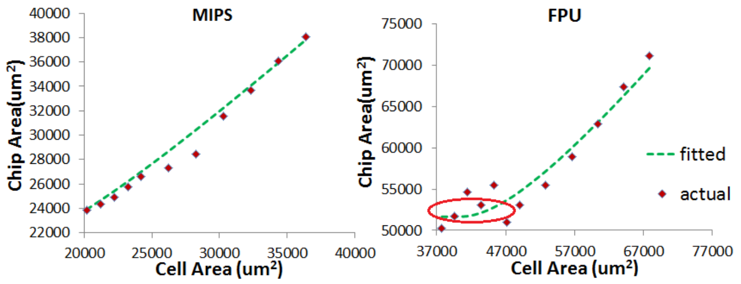


Figure 2.6: Plots showing MIPS and FPU chip-area vs. cell-area results obtained from actual P&R runs and those estimated using our analytical predictive model. Notice the circled region on FPU which exhibits a flat relationship between cell area and chip area. FPU is more routing-limited than MIPS.

Table 2.4: Values of $x0$ and $y0$ for various designs (see plots of Figure 2.6).

Design Name	**x0**(um^2)	**y0**(um^2)
MIPS	12526	20437
FPU	30950	36760

Our experiments to validate our methodology were performed on 5 designs from [ope], synthesized using Nangate Open Cell-Library [Nana], and FreePDK open-source process [Fre]. First, data for actual P&R were created for all the designs using cadence encounter, with router objective function as ”minimize congestion”, and for varying number of routing layers. Based on these runs α and β (in Equation (2.6.2)) were estimated to be $\frac{1}{3}$, i.e. the coefficients were estimated such that the metric agrees with the routability of designs confirmed by P&R runs. Runtime comparison between AEGR and actual P&R methods for MRA estimation is given in Table 2.3. For actual P&R, maximum utilization was found using binary search algorithm.

To evaluate the area model, area of various cells was increased in the LEF file to closely imitate cell-area change due to FEOL design rule changes. However, the pin shapes and pin positions were not modified. Chip area was then estimated using AEGR for every increase in total cell area and the proposed model was fitted on the resulting data. The plots are shown in Figure 2.6 and values of $x0$ and $y0$ are shown in Table 2.4.

2.6.3 Functional Yield Modeling and GCPW Calculation

Functional yield at the cell-level is computed similarly to [GG12]. It includes three yield-loss sources: overlay error (i.e. misalignment between layers) coupled with lithographic line-end shortening (a.k.a. pull-back), contact-hole failure, and random particle defects. The yield at the cell level is extended to the chip level using the well-known negative binomial model [8]. GCPW can then be calculated as the ratio of $\frac{wafer\_area}{chip\_area} \times yield$.

2.7 Experimental Results

As examples, we study three interesting rules in ChipDRE: (1) well-to-active spacing rule which affects number of transistor folds (hence area and delay variability) as well as threshold voltage and mobility of transistors (hence delay); (2) local-interconnect to gate spacing rule which affects capacitances as well as area; and (3) fin-pitch rule for a candidate FinFET technology. We observe that simple cell-based estimates (as is the state-of-the-art) to assess rule quality can be misleading highlighting the importance of the ChipDRE framework [9].

2.7.1 Well-to-active Spacing Rule Exploration

ChipDRE was used to perform a study of the well-to-active spacing rule, which impacts cell delay as well as cell area. The rule values that were chosen are 140nm, 185nm, 200nm and 210nm with 140nm as the baseline value. SPR data were generated for MIPS design using the ChipDRE-generated LEF and LIB files for each spacing rule with timing optimization done at both placement and post-routing stages while keeping the congestion effort "high". The clock period was chosen such that minimum positive slack was achieved for the baseline case. The maximum possible cell-utilization with no DR violations and a positive timing slack is used to compute the chip area. Chip-area comparison between actual results from SPR and estimation from the proposed ChipDRE flow is given in Table 2.5. The table also shows the GCPW metric for the design rule[10]. This study shows that a well-to-active spacing rule of value of 185nm results in the best number of GCPW even though it does not achieve the minimum cell area.

[8] Yield loss in routing-layers will be addressed in future work.

[9] We use 45nm rules from a publicly available pdk [Fre] to perform example studies which could be performed for future technology nodes.

[10] Note that for calculation of yield and GCPW, we assume the final design area is actually n copies of the indicated area (analogous to multiple cores), where n was selected to make the final design area roughly 100 mm^2 at the baseline design rule value.

Table 2.5: Chip area comparison between golden SPR and model based prediction on MIPS. The runtime for ChipDRE is just the cell estimation time: 49 minutes for a 100 cell library. Golden flow uses ChipDRE-generated libraries with commercial tools for physical design with the AEGR method proposed in this chapter. "est" is the value estimated by ChipDRE.

Well-to-active spacing [nm]	Run-time (SPR) [mins]	Cell Area (est.) [um^2]	Chip Area (est.) [um^2]	Chip Area (SPR) [um^2]	Error in %	GCPW (est.)
140	118	28171	30364	30130	0.8	667
185	356	28171	29709	29460	0.8	681
200	240	32527	33008	33913	-2.7	612
210	207	32554	32787	33554	-2.3	616

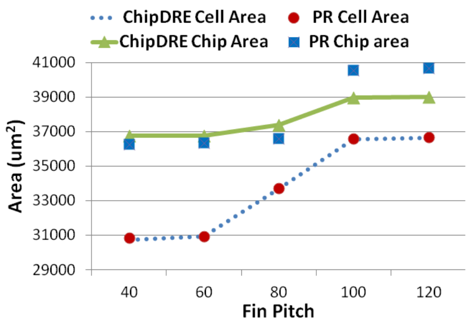


Figure 2.7: Plots for cell/chip area of FPU design as a function of fin pitch.

Table 2.5 results show that ChipDRE predictions are in strong agreement with the full SPR based flow and match the trends well. Interestingly, the dependence of GCPW and chip area on the rule value are *non-monotone*. This is primarily due to improved delay when well-to-active spacing is increased and despite the fact that the cell area monotonically increases as the rule value increases.

2.7.2 FinFET Fin-Pitch Study

Fin pitch value is a technology parameter that has a strong impact on the layout density. Although fin pitch is usually defined by process and technology constraints, exploring the design implications of this rule can help process developers decide which patterning technology to adopt (e.g. Self Aligned Double Patterning vs Directed Self Assembly). We use this fin pitch exploration as an example study to highlight the difference between chip-level and cell-level assessment of DRs. Hence, we use our framework to evaluate the impact of fin pitch on cell/chip area [11]. The impact of fin pitch on delay was ignored in this experiment since its impact on parasitic capacitances was not modeled in this work. Fin pitch was varied from 60nm to

[11] We realize there is no finfets in a 45nm process, but the study is performed for demonstration purposes.

120nm in steps of 20nm and for each value standard cell layouts were generated. Based on the standard cell usage of FPU design, total cell area was computed. The cell area was then plugged into cell-area to chip-area model' and chip area was computed. This has been verified against PR runs, and the maximum error in the model predictions was found to be 5%. Figure 2.7 shows the chip area and cell area variations as the fin pitch is varied, both from ChipDRE and PR experiments. The figure shows that for a fin pitch of 60nm through 80nm, the cell area is steeply increasing with a very slight change in chip area, which emphasizes the importance of chip-level evaluation as opposed to cell-level evaluation. It is also observed that the fin pitch can be increased from 40nm to 60nm with a negligible impact on cell area. GCPW trends are similar to chip-area trends in this case.

2.7.3 LI-to-gate spacing

Local interconnect is used in modern technologies to relieve congestion on local metal layers. One of the primary purposes is to make the power and ground rail connections from corresponding active areas in the devices. These connections replace contacts and metal. Unfortunately, these long contacts also increase capacitive coupling between gate and the local interconnect resulting in increased C_{gs}. To complicate matters further, increased spacing between gate and local interconnect can cause increase in the active area resulting in increased diffusion capacitance as well. We model both these effects in ChipDRE for the planar process and explore this spacing rule. Figure 2.8 shows the effect of changing the LI-to-gate spacing on the chip area (with GCPW trends being similar). In this case, the cell-area increase due to rule-value increase dominates the potential area reduction coming from delay improvement brought by a reduced gate-to-LI coupling capacitance (unlike the well-to-active rule experiment which showed a stronger delay impact).

2.8 Conclusions

We presented ChipDRE, the first framework for *fast*, *early* and *systematic* collective evaluation of design rules, layout styles, and library architectures at the chip-scale. The framework makes

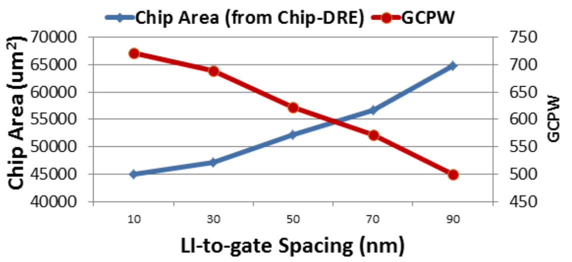


Figure 2.8: LI-to-gate design rule evaluation and effect on chip area for FPU.

rule definition and optimization easier, efficient, and much more systematic. Rather than exploring the entire search space of design rules manually or with conventional compute-expensive methods, the framework can be used to quickly eliminate poor rule and technology choices. By using fast layout-estimation methods coupled with semi-empirical and neural network-based models for cell-area/cell-delay impact and trade-offs at the chip-level, the ChipDRE framework unifies area, performance, variability, and yield a "good chips per wafer" metric. To show potential applications of ChipDRE, we use it to perform evaluation studies of debatable rules for state-of-the-art technologies, including FinFETs and local-interconnects, at the chip-scale. For instance *a study of well-to-active spacing rule reveals a non-monotone dependence of rule value to chip area* (although the cell-area relationship is monotone) due to delay changes coming from well-proximity effect.

CHAPTER 3

Layout Pattern-Driven Design Rule Evaluation

With the use of sub-wavelength photolithography, some layouts can have low printability and, accordingly, low yield due to the existence of bad patterns, even though they pass design rule checks. A reasonable approach is to select some of the candidate bad patterns as “forbidden”. These are the ones with high yield-impact or low routability-impact, and these are to be prohibited in the design phase. The rest of the candidate bad patterns may be fixed in the post-route stage, in a best-effort manner. The process developers need to optimize the process to be friendly to the patterns of high routability-impact. Hence, an evaluation method is required early in the process, to assess the impact of forbidding layout patterns on routability. In this work, we propose Pattern-driven Design Rule Evaluation (Pattern-DRE), which can be used to evaluate the importance of patterns for the routability of the standard cells and, accordingly, select the set of bad patterns to forbid in the design. The framework can also be used to compare restrictive patterning technologies (e.g. LELE, SADP, SAQP, SAOP). Given a set of design rules and a set of forbidden patterns, Pattern-DRE generates a set of virtual standard cells, then it finds the possible routing options for each cell, without using any of the forbidden patterns. Finally, it reports the routability metrics. We present a few studies that illustrate the use cases of the framework. The first study compares LELE to SADP, by using a set of forbidden patterns that are allowed by LELE but not by SADP. Another study compares LELE to EUVL from the routability aspect, by prohibiting patterns that have LELE native conflicts. In addition we present a study that investigates the effect of placing the active area of the transistors close to the P/N interface instead of close to the power rails. [1]

[1]The material in this chapter is based on the published work [BMG14].

3.1 Motivation and Overview

As the semiconductor industry continues to use sub-wavelength photolithography, new printability problems arise. Some layouts can pass Design Rule Check (DRC), but will have “bad patterns”; patterns that have low printability. There are two extreme candidate solutions to the bad patterns problem. The first solution is to handle that problem in the design stage by prohibiting all candidate bad patterns from appearing in the design. However, disallowing all those patterns can make the standard cell routability very hard, and this in return can lead to a tremendous increase in the standard cell area. An alternative, but also extreme solution is to allow all bad patterns in the design phase, and then later, after routing, try to legalize the layout in order to eliminate those bad patterns. Yet, at this stage, it may be too late to fix all those patterns. Thus, a hybrid approach is recommended where a set of “forbidden patterns” is disallowed in the design phase. Then later, after routing, try to fix the remaining bad patterns, in a best-effort manner. As a result, we need to answer the question of which patterns to select as “forbidden patterns”. A forbidden pattern needs to have high yield-impact or low routability-impact. High yield-impact patterns can be identified by lithography simulation. Low routability-impact patterns are those that, if forbidden in the design stage, the routability of the standard cells and the design will not be drastically hurt. In other words, we can still route the design even with those patterns being forbidden.

Another problem that is similar to the bad patterns problem is the emergence of restrictive patterning technologies like Double Patterning (LELE and SADP), Triple Patterning, Quadruple Patterning, and beyond. Each of those restrictive patterning technologies has some non-manufacturable patterns. An essential question arises for foundries; which technology to adopt for the next node.

Thus, an evaluation method is required early in the process to assess the effect of prohibiting some forbidden patterns on the routability. In this work, we propose Pattern-driven Design Rule Evaluation (Pattern-DRE), which can be used to assess the sensitivity of the standard cell routability to the patterns and design rules and can be used to compare restrictive patterning technologies from the point of view of standard cell routability. It can also be used to count the occurrence of the undesired patterns as the design rules change. In addition, the framework can also be used to guide the process development on the relative importance of the various patterns, and accordingly indicate from a design perspective, the patterns that the process needs to be optimized for. A high level overview of the framework is shown in Figure 3.1, where the framework uses a set of design rules, candidate forbidden patterns, and the transistor-level netlists of the standard cells and then reports routability metrics as output.

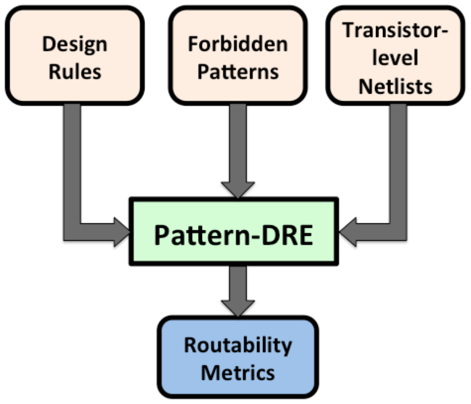


Figure 3.1: Overview of Pattern-DRE framework

3.2 Prior work

To the authors' knowledge, this is the first attempt to systematically evaluate patterns along with design rules, and study the sensitivity of routability to the patterns. In our previous work[GG12], we proposed DRE; a framework for systematic evaluation of design rules, layout styles, and library architectures. However, DRE was not pattern-aware, and hence we propose Pattern-DRE in this work. Since the use-context of our framework is related to handling bad patterns (a.k.a "hotspots") and comparing patterning technologies, we discuss the work that has tackled both topics here.

Several works have addressed the problem of hotspot or pattern-aware design, i.e. using a correct-by-construction approach. The methodologies in ref. [JPR06] and ref. [JRL10] apply template-based correct-by-construction design. These methods are very promising since they can be used to achieve micro and macro levels of granularity. However they require very high effort in the design of the templates, whether done manually or automatically by RTL synthesis or other template library creation methods. Thus, these methods may not be appropriate for technology exploration phase which requires evaluating a lot of alternatives in a fast and automated fashion. Ref. [GG12] used conservative rules to have correct-by-construction standard cell layouts that are compatible with LELE and SADP, and performed a comparison study. However, the conservative rules used can waste a lot of area, and this can skew the results of the comparison between the patterning technologies. In ref. [CYB08], a lithography-aware router was proposed, which used a printability metric to guide the router. Another approach was developed in ref. [DGY11], which used routing path prediction, along with lithography simulation and a hotspot prediction kernel to construct lithography-friendly routes.

As opposed to the correct-by-construction techniques, a lot of work focused on the detection and correction of those bad patterns after the design stage. Several works [YLJ13, DYG12, DTP11, WPM11, DWG09, GMM09, KPX08a, MGM08] used various techniques of Machine Learning or Fuzzy Pattern Matching to identify the hotspots in the design. Ref. [DYR07] and ref. [JHJ12] suggested a flow that integrates a pattern checker, a pattern fixer, and a router; such that the router completes its job, and then pattern check and fix are performed if needed, and tentatively some routes are redone. Ref. [MYP05] also used rip-up and re-route to build a router that is RET (Resolution-Enhancement Techniques) - aware. Similar to the post-design hotspot detection, ref. [DC13] proposed using –in addition to the design rule check– a pattern matcher to detect the short range patterns that are incompatible with double patterning and apply fixes to them.

To explore design rules for Multiple Patterning technologies, ref. [GSK12] used Machine Learning techniques to predict the number of conflicts, which can be used to compare several sets of design rules. Ref. [DMY11] suggested optimizing the design rules for double patterning technologies in an iterative flow, where in each iteration: test layouts are generated and decomposed, lithography simulation is performed, impact on the design is analyzed and accordingly the design rules are optimized.

A lot of work focused on developing multiple patterning-aware routers, like LELE-aware routing [GM10, YLP09, AW12], Triple Patterning-aware routing [MZW12], SADP and SAQP-aware routing [GP12, KIN13].

However, none of these works offered a pattern-centric design rule evaluation method, and this is the main contribution of our work. The rest of this chapter is organized as follows: section 3.3 explains the flow of the framework and breaks down each module used into detail, and it shows how Pattern-DRE can be used to make decisions about forbidden patterns. In section 3.4, we show how we validated Pattern-DRE, and then we illustrate some studies that have been performed using Pattern-DRE. Finally, we present the conclusions and future work in section 3.5.

3.3 Pattern-DRE Flow

In this section, we explain the flow of the Pattern-DRE framework, which is illustrated in Figure 3.2. The input to Pattern-DRE is the set of design rules, transistor-level netlist for the standard cell, and a set of forbidden patterns. Pattern-DRE generates a virtual standard cell library and studies the possible routing options for each cell, while avoiding the given forbidden patterns. After generating the front-end layers, the cell may not be routable. In such a case the standard cell is generated in a different way and the routing is re-attempted, until it becomes routable or we reach a certain number of trials (further details are provided in section 3.3.1). Routability metrics are reported by the framework at the end. In addition, the count of all occurring patterns are reported. In the following subsections, the details of each block in the flow will be explained.

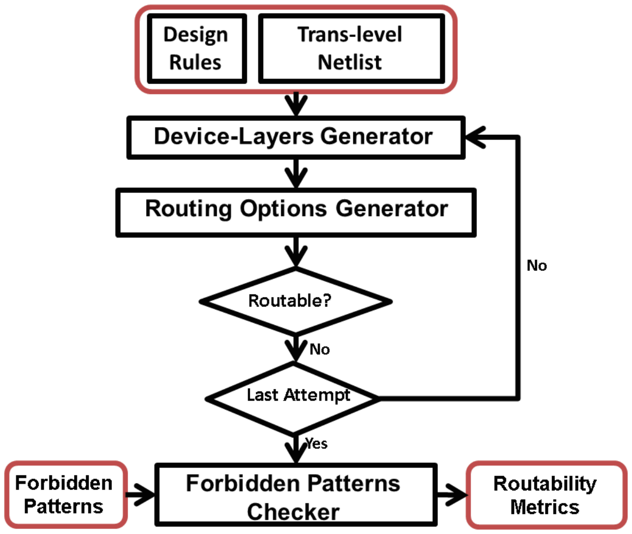


Figure 3.2: Flow of Pattern-DRE

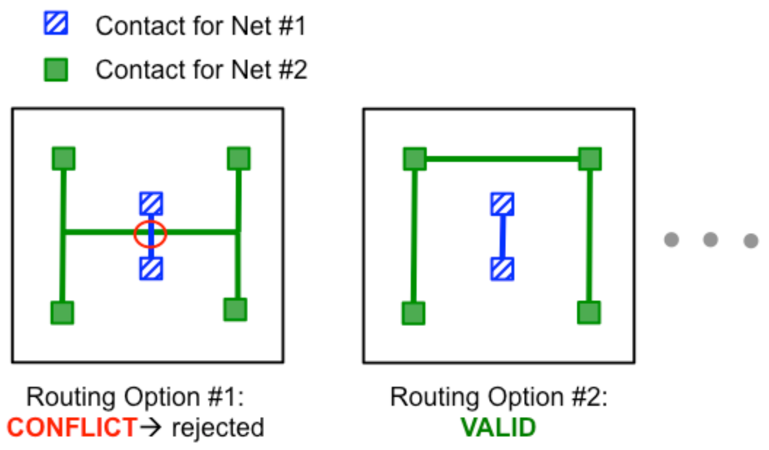


Figure 3.3: An invalid routing option (on the left) because of a conflict between the routes of the nets and a valid routing option (on the right)

3.3.1 Device-Layers Generator

Pattern-DRE first generates the essential device layers for the given standard cells. This includes building the required transistors based on the given design rules and transistor-level netlists for the cells. We use the device-layers generator of DRE[GG12]. As part of the device-layers generation, the contact locations forming the nets are generated[2]. The nets along with their contact locations are used as inputs to the next module.

As shown in figure 3.2, the Device-Layers Generator can be invoked again for a few iterations if the cell is found unroutable. In such a case, the abutment of transistors is done in a different way. The Chaining step [GG12] then chooses a sub-optimal solution with respect to area, in order to give another chance for the routability of the cell.

3.3.2 Routing Options Generator

This module mimics a router and tries to find possible ways in which the nets of each cell can be routed.

Given all the nets in the cell, the routing options generator generates a list of candidate

[2]A change has been performed to the gate contact locations generated by DRE, such that the gate contacts are not all generated on the same horizontal level; instead connected gates have contacts that are aligned at the same y-location for ease of routing, but unconnected gates can have their poly contacts at different y-locations. This improves the routability of the cells.

wiring solutions for each cell. Instead of routing with a specific topology, we try to enumerate all possible routing options under a single trunk Steiner tree [Sou81] topology type. The wiring solutions for each net are generated as presented in Ref. [GSK12]. Starting with each net, the bounding box is determined according to the contact locations inside that net. If the width or height of net bounding box lies below a certain threshold, then we expand the bounding box by a few tracks, in order to allow detours for the net[3]. In addition, if the bounding box is too skewed in a certain direction, then having a single trunk steiner tree trunk along the short direction will lead to unnecessarily long wire length, as shown in Ref. [GSK12]. The possible wiring solutions [4] for each net are constructed by placing the tree trunk at each of the tracks within the bounding box, and then, constructing perpendicular branches from the trunk to reach out to each contact. With all the wiring solutions for each net, we need to construct complete routing options for each standard cell. Not all combinations will form valid routing options for the cell, because some routes from different nets can cross/intersect. An example showing a possible conflict between routes of two different nets is shown on the left in Figure 3.3 and another example showing a valid routing option is shown on the right. After we discuss the pattern representation that we use, we will illustrate how the check for conflicts, between wiring solutions of nets, is performed.

Layout and Tile/ Pattern Representation

The layout is represented as a 2D matrix of tiles. Each tile has two representations: segment representation and node representation. The same representation is used for the tiles in the layout and the patterns, except that the tile has fixed size (2x2 tracks), while the size of the patterns is specified as an input[5]. All wiring is assumed to be on-track and with a uniform width.

[3]In our experiments we used a threshold of one track and we expanded the bounding box to three tracks in such a case.

[4]To avoid confusion, when we mention "wiring solution", we are referring to a way to route the net, but when we say "routing option" we are pointing to one way to route all the nets in the cell (i.e. a set of wiring solutions; one for each net).

[5]Currently the maximum allowed pattern size is 5x5 tracks.

- Segment representation: Intersection of wiring tracks break themselves into segments, and the segment representation encodes the presence/absence of a wire between the tracks in the opposite direction. The rows and columns are then serialized as a binary string, and the equivalent decimal number is used as the pattern segment representation. An example of the segment representation of a tile/pattern is shown in Figure 3.4 on the left, where the rows are read first from left to right followed by columns from bottom to top (first segment occupies least significant bit). Then the equivalent number formed by the binary string is used as the segment representation for the tile. The segment representation is required because it uniquely identifies the pattern.

- Node Representation: A node is the intersection of a vertical and a horizontal track. So the node representation encodes whether or not each node is occupied (A node is occupied if any of its neighboring segments is occupied.). An example for the node presentation is shown in Figure 3.4, on the right. The node representation is required for the conflict detection, which will be explained shortly.

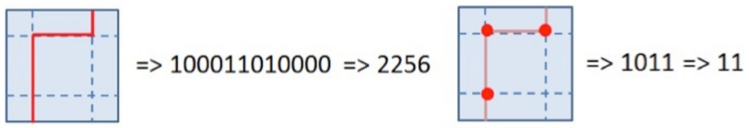


Figure 3.4: Segment and Node Representations for Tile/Pattern. On the left is the Segment Representation; columns and rows are read off into a binary string (100011010000), then the equivalent decimal number (2256) is used as the segment representation. On the right is the Node Representation; nodes are serialized as a binary string (1011), then the decimal equivalent (11) is used as the node representation.

Conflict Checker

Two wiring solutions for two different nets are conflicting if their segments overlap or cross. These cases can be checked by doing AND operation between the node representations of the routing options in each tile. If the result of the AND operation is non-zero for any tile, then there is a conflict. The reason for doing the conflict check on the node representation is that some conflict cases can not be detected on the segment representation. For example a vertical and horizontal wire will not have any common segments but will have common nodes. This is the case illustrated in Figure 3.5.

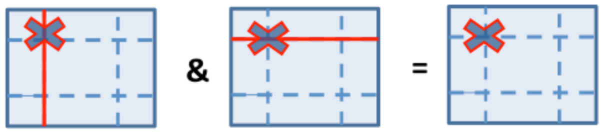


Figure 3.5: Checking conflicts between routing options of different nets by ANDing Node Representations

Minimum Number of Unroutable Nets

In some cases the routing options generator may fail to find a conflict-free routing option for the cell. In such a case, it reports the routing option with minimum number of unroutable nets. This problem is formulated as an Integer Linear Program (ILP), and is shown in equation 3.3.1.

n_i is a binary variable which is assigned to true if cell is unroutable. r_{jq} is a binary variable representing whether the qth wiring solution for the jth net is selected. Let C be the set of pairs of conflicting wiring solutions belonging to different nets. The objective is to minimize the number of nets whose wiring solutions are conflicting with wiring solutions of other nets in the chosen routing option for the cell. The first set of constraints guarantees that if two conflicting wiring solutions (for two different nets) are in the selected routing option, then one of the two nets is selected as unroutable. A constraint is generated for every pair of conflicting routing options. The second set of constraints guarantees that for every net, exactly one wiring solution is chosen. Thus the program has to choose the routing options in a way that minimizes the number of unroutable nets.

$$\begin{array}{lllllllll} minimize & \sum\limits_i n_i & & & & & & & \\ subject\,to & r_{jq} & + & r_{kp} & - & n_j & - & n_k \leq 1 & \forall (r_j, r_k) \in C \\ & \sum_q & r_{iq} & & & & & = 1 & \forall i \end{array} \tag{3.3.1}$$

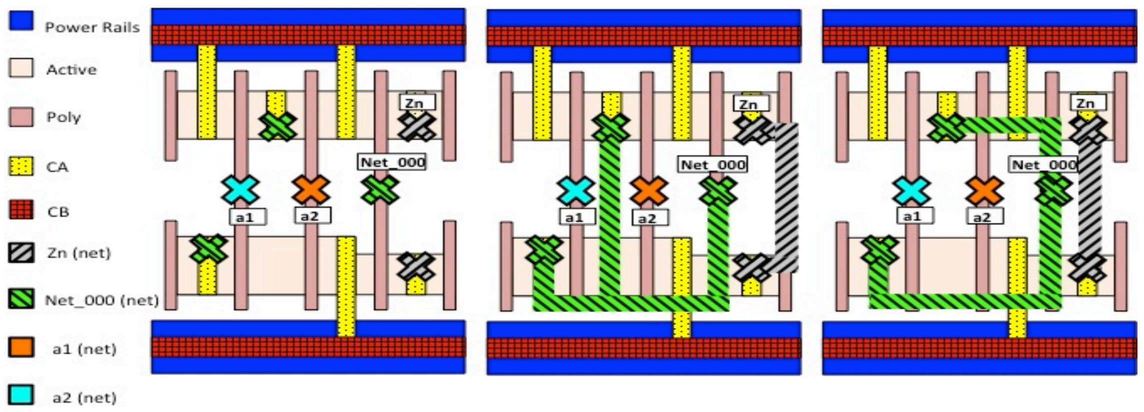


Figure 3.6: Sample output of Pattern-DRE on AND2_X1 cell. On the left is the cell generated by Pattern-DRE before the routing options are generated. Cross markers are placed to show the contact locations to be connected. There are four nets in this cell: a1 (in light blue) is a single-contact net, a2 (in orange) is a single-contact net, Net_000 (in green) has 3 contacts and Zn (in gray) has 2 contacts. Only two routing options are shown (middle and right figures).

Sample Output

The sample output of the routing options generator is shown in Figure 3.6 where the figure on the left shows the AND2_X1 cell without any of the generated routing options and two of the routing options are shown on the right.

3.3.3 Forbidden Patterns Checker

The generated routing options are checked against the given set of forbidden patterns. A window is slid over the layout with a track granularity, and the pattern of required size is formed starting at each row and column combination. The tracks in the pattern are serialized and represented, as shown in Figure 3.4, in order to do an easy and fast comparison with the input forbidden patterns. If the routing option contains any of these patterns, then it is discarded. For example, the routing option shown in Figure 3.7 on the right will be discarded if the pattern in on the left of Figure 3.7 is forbidden. It is worth mentioning that Pattern-DRE is not sensitive to the existence of forbidden patterns that can be formed across borders of two adjacent cells as a result of placement. However, it can be easily extended to handle these patterns by considering cells in pairs where each pair simulates the side-by-side placement of two cells, assuming the thick power rails can do a shielding effect between different rows. More complex abutments can also be handled in a similar way at the cost of runtime. [6]

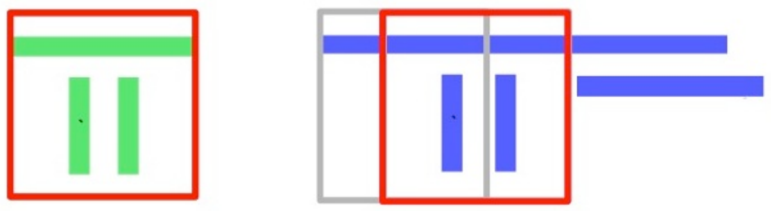


Figure 3.7: Checking a Routing Option for Forbidden Patterns. A forbidden pattern is shown on the left. On the right, we show a snippet of the routing option where the forbidden pattern is matched. The routing option is drawn in blue, while the gray and red boxes are two sliding windows

3.3.4 Routability Metrics

The output of the framework is the routability metrics. The framework reports the **number of routable cells**, the **total number of routing options** and the "**distance from a routable library**". The first two metrics are indicative of the ease of routing the cell without the forbidden patterns. The reason why we need the number of routing options may not be obvious. While two sets of design rules can produce the same number of routable cells, they actually may not have the same routability-impact. So if prohibiting a set of patterns drastically affects the number of routing options and only leaves a few options, then this means that the set of forbidden patterns has high routability-impact. As a result of eliminating a large number of routing options, there is low chance of a post-route fix for other patterns, that were not forbidden in the design stage.

For example if we want to compare the two sets of forbidden patterns: set A and set B, then we run Pattern-DRE twice, once for each set. If set A, as a set of forbidden patterns, leads to fewer number of routable cells, then set A has a higher routability-impact. If both of them have same number of routable cells but set A leads to fewer number of routing options, then this means that it has higher routability -impact than set B. The same method can be used to study sensitivity of routability to specific patterns, where set A and set B will only differ by two patterns (one in set A exchanged by the other in set B). Pattern-DRE also reports the distance

[6] For patterns that are forbidden due to multiple patterning requirements, boundary conditions can be enforced as proposed in [LPG11] so that no coloring conflicts are formed after placement. However we tend to avoid tailoring Pattern-DRE to any particular technology, keeping it as generic as possible, while the input set of forbidden patterns impose the specific requirements of the technology.

from a routable library: the smallest-cardinality subset of forbidden patterns that, if allowed, will lead to a routable library. This can be used to give higher priority to certain patterns during process development (i.e. the process can be optimized in order to be friendly to these patterns in order to have a routable library). This metric is calculated from the cells which are unroutable due to the existence of forbidden patterns, not because of conflicts between the nets. To calculate this metric, we keep track of the sets of forbidden patterns that have occurred in routing options in each cell, and then the combinations of these sets from different cells are analyzed in order to find the smallest subset of forbidden patterns that if allowed will lead to routable cells. In addition, Pattern-DRE also reports the non-zero **number of times each pattern occurs in the layout**. For cells that are unroutable because of conflicts between nets (not because of forbidden patterns), Pattern-DRE reports the minimum number of unroutable nets.

Area and routability are inter-dependent; therefore when the two scenarios under comparison are only different in the set of forbidden patterns applies, we compare routability at the same cell area in order to have a fair comparison based on routability. So when we're comparing two scenarios (one of them considered baseline scenario), we restrict the iterations of the device-layer generation such that the area of the standard cells would not be larger than the area in the baseline scenario, and if the cell is not routable within this limited number of iterations, it is considered unroutable.

Pattern-DRE does not study the performance impact -if any- due to changed parasitics. However since the area is preserved/matched between the scenarios under comparison, the performance impact is expected to be small.

3.4 Experiments

In this section, we first explain how the framework has been validated, then we present a few studies to give examples of how the Pattern-DRE framework can be used.

3.4.1 Validation

To validate the framework, several benchmarking comparisons were performed. First the generated standard cells were compared, in terms of area, to cells of Nangate open standard cell library[Nanb], using same design rule values.

The average error in area between standard cells generated by DRE and those of Nangate was 2%. To validate the routing options generator, the average wirelength of the cell routing options was compared to the wirelength of a rectilinear Steiner minimal tree routing algorithm [CW08]. On the average, our routing options generator produced 12% higher wirelength, but it was 44x faster. For the pattern occurrences, we found that the patterns that occupied 82.4% of Metal1 layer in Nangate layouts took up 81.5% of the Pattern-DRE Metal1 layer. Also the cosine similarity between the pattern counts vectors from Pattern-DRE and Nangate was 0.86. To calculate the cosine similarity, we counted the number of times each pattern occurs in the Nangate cells[7]. From Pattern-DRE, we calculated the average number of pattern occurrences per routing option for each cell, and summed that average count for all routable cells. Then we calculated the cosine similarity between the pattern counts vectors from Nangate and Pattern-DRE.

These validation attempts show that the Pattern-DRE estimates are good enough in comparison to actual layouts. In addition, Pattern-DRE is very fast, in comparison to other evaluation methods involving manual design/tweaks that would require weeks; Pattern-DRE can process 92 cells in 45 hours for a maximum of 7 front-end layer generation iterations to find a routable solution. Without finding the minimum number of unroutable nets, the 92 cells can be processed in 17 hours. If we use only one iteration (and don't re-generate the transistors in a different way if the cell is unroutable), and if we disable the minimum number of unroutable nets calculation, then 92 cells are solved in 40 minutes with 11.5% less routable cells.

[7]We only used the Pattern-DRE-routable cells in the validation.

3.4.2 Litho-Etch-Litho-Etch (LELE) vs. Self Aligned Double Patterning (SADP)

In this experiment we performed a simple comparison between Litho-Etch-Litho-Etch (LELE) and Self-Aligned Double Patterning (SADP) . It is known that SADP has less susceptibility to overlay error than LELE [XDZ13]. To make better use of the overlay advantage of SADP, we assume that the process does not allow the formation of the side of the polygons using trim mask (which is subject to overlay error). Thus if we assume that the distance between the corner of the stitched polygon and the tip on the same mask exceeds the minimum spacing, then any small odd cycle between two tips and a side like the one shown in Figure 3.8 can be resolved in LELE by introducing a stitch, but it can not be resolved in SADP where stitches are prohibited.

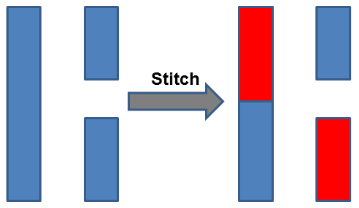


Figure 3.8: An Odd cycle that can be resolved in LELE (if distance between the corner to corner after stitch is greater than the spacing rule) by introducing a stitch, but can't be resolved in SADP which does not allow stitches.

To perform this experiment, we generated a list of 258 forbidden patterns. Each pattern has two columns and two rows and needs a stitch to be resolved. Examples of such patterns are shown in Figure 3.9. All these patterns are SADP-incompliant but LELE-compliant. Thus we conducted the SADP experiment using those patterns as forbidden ones, but the LELE experiment is done without any forbidden patterns.

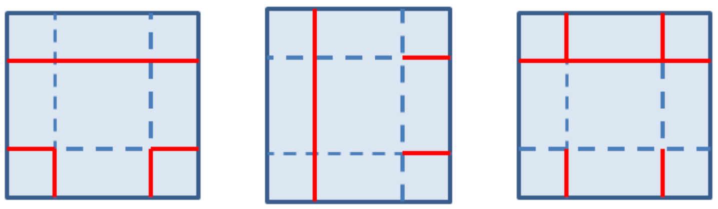


Figure 3.9: Three samples of the 258 forbidden patterns used to conduct the LELE vs. SADP Experiment. Each of these patterns requires a stitch, so these patterns are assumed to be SADP-incompliant but LELE-compliant.

The experiment was performed with 22nm rules and planar CMOS transistors. In the results, we focused on the 78 routable cells with the given design rules (out of the input 92 cells). The baseline scenario for this experiment is the LELE case which doesn't have any forbidden patterns. Results are shown in Table 3.1. The reported columns are:

- Routable cells: number of cells which have one or more routing options
- Routing Options: total number of routing options for all cells
- Difference in routing options: the difference (as a percentage) between the number of routing options in the current scenario and in the baseline scenario

	Routable Cells	Routing Options	Decrease in Routing Options	Distance from Routable Library
SADP	77	2766	17.1%	1
LELE	78	3338	0%	0

Table 3.1: Comparison results of SADP (i.e. when the 2 tips and a side odd cycles are prohibited) and LELE (when those odd cycles are not prohibited)

In some cases, as explained in section 3.3, Pattern-DRE may attempt a different device-layers design in order to find a routing solution, which in turn can affect the area. Thus to have a fair comparison based on routability, we restricted the iterations of Pattern-DRE so that the cells generated for the SADP scenario have the same areas as those of the LELE scenario. According to the experiment results and with this selection of forbidden patterns, we sacrifice 1.3% of the routable cells and 17.1% of the routing options for the sake of the overlay advantage of SADP. The minimum distance to a routable library was one forbidden pattern, i.e. if we remove one pattern the 1.3% of the cells which are unroutable will be routable. It is worth mentioning that we present these studies as examples to demonstrate how to use the framework and the objective is not the actual comparison of these processes. Accordingly, we emphasize that in order to make a proper decision, it is required to enumerate all SADP-incompliant patterns that are compliant to LELE, and use them as forbidden patterns, then enumerate all LELE-incompliant patterns that are compliant to SADP (if any), and compare the results of these two scenarios. Instead of exhaustive pattern enumeration, the generation of those patterns can be performed by a Monte Carlo method to produce an enormous number of patterns at random

and LELE and SADP decomposition are to be performed on those patterns. Then patterns that are LELE compliant and not SADP compliant are to be used as forbidden patterns for SADP, and vice versa. In our case, we didn't have access to a commercial SADP decomposer, so we selected those patterns in the way explained above, for demonstration purposes. However in the next section, we show how we generate the forbidden patterns in a more precise manner.
Note that PatternDRE, as a framework, is not aware of Multiple Patterning. However the applied forbidden patterns impose the restrictions of the patterning scheme that is used. For example, when it is desired to test SADP, the forbidden patterns should be the patterns that are not allowed by SADP like the patterns that require stitches.[8]

3.4.3 Litho Etch Litho Etch (LELE) vs. Extreme Ultraviolet Lithography(EUVL)

In this section, we perform a comparison between Litho Etch Litho Etch (LELE) and Extreme Ultraviolet Lithography(EUVL). In this experiment we used patterns of size 4x4; i.e. patterns of larger range than the ones used in sections 3.4.2 and 3.4.4. By using these longer range patterns, we can detect more decomposition conflicts that cannot be represented in smaller patterns. We assume EUVL does not have any forbidden patterns. For LELE, we first ran Pattern-DRE without any forbidden patterns to get all the patterns that were used in the routing options. Then we ran a commercial LELE decomposer [cal31] on those patterns, and used the patterns with unsuccessful decomposition as forbidden patterns, and re-ran Pattern-DRE. After applying the forbidden patterns, it is possible to have new patterns that were not generated before due to exploring different front-end options and possible expansion of the bounding box of the nets to find a routing solution. To check that the patterns used in case of LELE are all LELE-compliant, we ran the commercial decomposer again on the final patterns, and none of them had LELE decomposition conflicts. Results in table 3.2 show that by using LELE instead of the unconstrained EUV, we sacrifice routability of 7.8% of the cells, and 56.9% of the routing options, at the same cell area.[9] The minimum distance to a routable library is 16 forbidden

[8]Foundries are encouraged to download the framework from `http://nanocad.ee.ucla.edu/Main/DownloadForm` and try it with their own patterns and rules.

[9]Note that the results for LELE in this experiment are different from those in Table 3.1 because here we use 4x4 patterns that are incompliant to LELE, as explained above.

patterns, i.e. if we remove these 16 patterns, we reclaim the routability of 7.8% of the cells which are unroutable.

	Routable Cells	Routing Options	Decrease in Routing Options	Distance from Routable Library
LELE	72	1440	56.9 %	16
EUVL	78	3338	0%	0

Table 3.2: Comparison results of LELE and EUVL

3.4.4 Diffusion Location

In this study, we investigate the effect of the location of the diffusion within the cells, assuming an SADP process. This experiment is different from the previous ones in the sense that we don't compare two different process, but we use Pattern-DRE to compare two front-end choices for the same process (SADP), and we use the same SADP forbidden patterns used in section 3.4.2. We study two options for the diffusion location; diffusion being as close as possible to the P/N interface vs. as close as possible to the power rails. Placing the diffusion close to the P/N interface allows a larger value of Line End Extension (LEE), which is more robust against overlay error but is expected to lead to less routability since the diffusion contacts will be spread out in a smaller area and in the vicinity of the poly contacts. In addition, changing the location of the diffusion can affect stress especially in the presence of tensile and compressive nitride liners, which in turn affects performance [JCS08]. Results in table 3.3 show that by locating the diffusion close to the P/N interface instead of close to the power rails, we lose 5.1% of the routable cells, and 68.9% of the routing options. Note that in this experiment the two scenarios under comparison have different device-layer designs so it is not possible to force the same area, but the areas turned out to be very similar.

3.4.5 Area vs. Routability

In a lot of cases it's possible to gain more routability by increasing the cell area. In this experiment we varied the maximum number of chaining iterations (see section 3.3.1), and checked the number of routable cells (out of 78 total routable cells) as well as the total cell area. We

Diffusion location	Routable Cells	Routing Options	Decrease in Routing Options	Total Cell Area (um^2)
Close to power rails	78	2772	0	39.7
Close to P/N interface	74	861	68.9	39.6

Table 3.3: Comparison between routability of cells with diffusion placed close to power rails and close to P/N interface

used the same LELE setup used in section 3.4.3. The result is plotted in Figure 3.10. The results are interesting, since they show that with a very little increase in area, we can get great routability benefits (˜ 21.8% routability improvement).

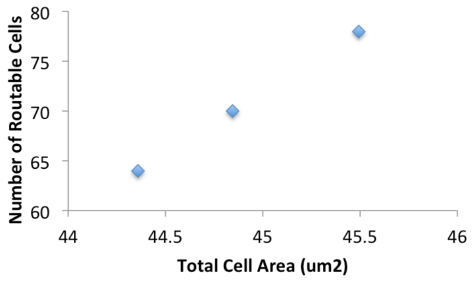


Figure 3.10: Total cell area vs. number of routable cells

3.5 Conclusion

In this chapter, we introduced Pattern-DRE, a pattern-aware design rule evaluator. Pattern-DRE can be used to optimize pattern-based design rules, identify important patterns which need to be focused on, by patterning technology. It can also be used to compare restrictive patterning technologies. As examples, we used Pattern-DRE to evaluate SADP vs. LELE, and LELE vs EUVL. It was also used to evaluate the choice of the diffusion location within the cell; being close to the power rails or close to the P/N interface. Our ongoing work considers a method for pattern-based design rule evaluation for back-end layers.

Part II

Computational methods to enable Directed Self Assembly

CHAPTER 4

A Technology Path-finding Framework for Directed-Self Assembly (DSA) for Via Layers

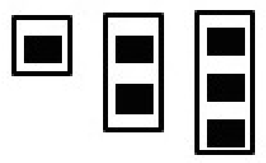
Directed Self Assembly (DSA) is a very promising patterning technology for the sub-7nm technology nodes, especially for via/contact layers. In the Graphoepitaxy type of DSA, a complementary lithography technique is used to print the guiding templates, where the Block Copolymer (BCP) phase-separates into regular structures. Accordingly, the design-friendliness of a DSA-based technology is affected by several factors: the complementary lithography technique, the legal guiding templates, the number of masks/exposures used to print the templates, the related design rules, the forbidden patterns (hotspots) and the characteristics of the BCP. Thus, foundries have a huge number of choices to make for a future DSA-based technology, affecting the design-friendliness and the cost of the technology. In this chapter, we propose a framework for DSA technology path-finding, for via layers, to be used by the foundry as part of Design and Technology Co-optimization (DTCO). The framework optimally evaluates a DSA-based technology where an arbitrary lithography technique is used to print the guiding templates, possibly using many masks/exposures and provides a design-friendliness metric. In addition, if the evaluated technology is not design-friendly, the framework computes the minimum-cost technology change that makes the technology design-friendly. The framework is used to evaluate technologies like DSA+193nm Immersion (193i) Lithography, DSA+Extreme Ultraviolet (EUV) and DSA+ 193i Self-Aligned Double Patterning. For example, one study showed that one mask of EUV in a DSA+EUV technology can replace three masks of 193i in a DSA+193i technology.

[1]

[1]The material in this chapter is based on the published work [BG17].

4.1 Introduction

A DSA technology for via layers is characterized by a lot of factors. First, a complementary lithography technique is needed in graphoepitaxy to print the guiding templates. The candidate complementary lithography techniques include 193nm Immersion Lithography (193i), Extreme Ultraviolet (EUV) Lithography, E-beam Direct Write, Self-Aligned Double Patterning (SADP) as well as possibly any other emergent technology. The choice of the complementary lithography technique will determine the legal guiding templates. The legal templates, along with the BCP properties, determine the legal DSA groups, where a **DSA group** is a set of vias that are to be patterned in the same guiding template. For example if EUV or E-beam is used, then the templates for more complicated DSA groups (e.g. L-shaped groups) may be printed, while if 193i is used then only collinear groups are allowed, as shown in Figure 4.1, due to the higher lithography variations in the case of 193i which leads to higher defectivity in self-assembly [BDG15]. The BCP properties also determine the allowed contact/via pitches that can be manufactured by self-assembly. Moreover, the guiding templates may be patterned using several masks/exposures (Multiple Patterning). Finally, if the foundry has a database of hotspots (forbidden patterns), it is required to prohibit DSA groups that will lead to templates causing any of the hotspots. These factors need to be evaluated during the technology path-finding of future nodes using DSA.


(a) Examples of DSA groups allowed in both 193i and EUV

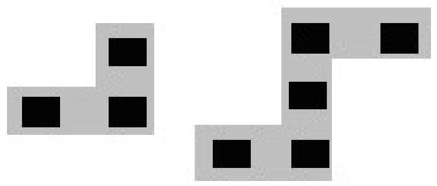

(b) Examples of DSA groups allowed in EUV but not in 193i

Figure 4.1: Examples showing that the lithography technique used to print DSA guiding templates affects the allowed DSA groups

We propose a DSA technology exploration framework for via layers. The input to the framework is the specifications of the technology to be explored and a benchmark layout. The framework evaluates the technology, from the point of view of design compliance and provides a design friendliness metric. If the technology is not design-friendly, then the framework computes the minimum-cost change to the technology that would make it compliant to the provided benchmark design. The objective of this framework is to be used by the foundry for Design and Technology Co-optimization (DTCO) in order to develop a new technology node, and not for processing large full chip-layouts for technologies already in production.

The contribution of this work can be summarized as follows:

(i) To the best of our knowledge, this is the first **optimal** and **general** framework to be proposed for evaluation of **any** DSA-based technology (using any complementary lithography technique), that can have multiple masks/exposures to print the guiding templates. [2]

(ii) Our framework manifests correct-by-construction methods to avoid DSA templates that create technology-specific hotspots.

(iii) If the evaluated technology is not design-friendly, the framework computes the minimum-cost technology change that makes the technology design-friendly.

(iv) Several novel technologies are evaluated using the proposed framework, including DSA+EUV, DSA+SADP and DSA+E-beam.

The rest of the chapter is organized as follows: Section 4.2 discusses the related work in literature. Section 4.3 presents an overview of the framework. Section 4.4 breaks down the framework into a sequence of stages, and describes them in detail. The ILP formulation for path-finding is presented in Section 4.5. Section 4.6 describes the minimum-cost technology change computation. In Section 4.7, we show case studies that have been performed using the framework, followed by conclusion and future work in Section 4.8.

4.2 Prior Work

The need for the co-optimization of BCP, design and lithography in order to find a design-friendly technology with lithography-friendly guiding templates, using the minimum number of

[2]By optimal evaluation, we mean that given that the assumptions employed in modeling the problem are reasonable and justifiable, the mathematical formulation is solved using an optimal solver, and not using heuristics.

mask/exposures- is emphasized by Ma et al. [MWW16]. However, their work focuses on the selection of the BCP and the guiding template dimensions to maximize the robustness in self-assembly, and they do not offer methods for optimizing the technology for design-friendliness. There is a lot of research targeting the optimization and verification of the guiding templates in order to generate the required self-assembled shapes. Approaches in this category have used combinations of simulation and mathematical models as in the work of Ma et al.[MLT15], machine learning as in the work of Xiao et al.[XDW15], level-set based algorithm with Self-Consistent Field Theory (SCFT) in the work of Ouaknin et al. [OLD16] in addition to experimental studies performed by Gharbi et al. [GTA14]. Our framework is not to be used for the purpose of optimizing the templates for the robustness of the self-assembly process, but it is used to determine the DSA groups that are important for the design; generating the actual guiding template shapes is not within the scope of this framework.

Another category of research aims at achieving DSA-friendly design. For example, the design of a DSA-compliant contact layer in standard cells has been studied by Du et al. [DGW13], assuming Single Patterning of the guiding templates. Yi et al. [YBT15] show a design strategy (no automated design methods) for standard cell design based on the requirements of DSA technology, so it can not be used to choose a design-friendly technology. DSA-aware routing has been addressed by Du et al. [DXW14]. Shim et al. [SCS15] proposed a method for perturbing the placement of standard cells, in order to decrease the DSA defect probability. In addition, the traditional idea of dummy via insertion has been revived in the works of Fang et al. [FHL15] and Ou et al. [OYP16], with the new objective of DSA-compliance. The work of Lin et al. [LC16a, LC16b] develops a cut redistribution algorithm to be able to print cuts in gridded layouts using DSA. The work of Wang et al. [WLZ14] can find non-DSA-friendly configurations by finding the configurations which result in defective self-assembly through simulation.

The third category of research develops algorithms for hybrid technologies involving DSA, DSA+EUV and DSA+Multiple Patterning (MP) for 193i. Several works [BTG15a, BTG15b, KY16, XLW16] perform DSA-aware mask assignment for DSA+193i technology. In addition, Ou et al. [OYP16] solve the same problem while adding redundant vias, while Lin et al. [LC16b] add cut redistribution. Karageorgos et al. [KRT16] solve the same problem with a variable number of masks, but can only run on a cluster of 15 vias at most, using exhaustive enumeration of grouping and mask assignment options, then they extend the work [KRG16], to employ heuristics for bigger clusters of vias, potentially sacrificing optimality. Gronheid et al. [GBD16] use experimental work to show advantages of using DSA+EUV. None of the works on hybrid DSA technologies offers the capability of modeling different and arbitrary DSA technologies optimally on a macro layout.

4.3 Overview of the Framework

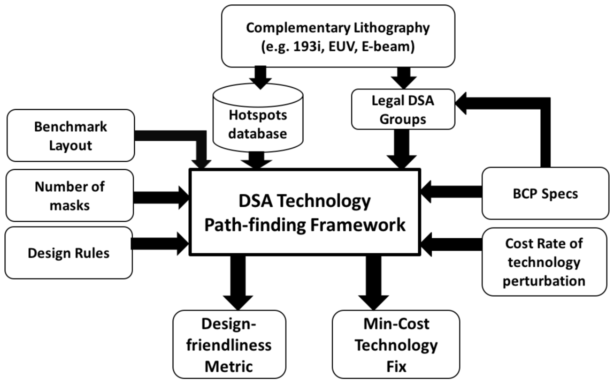

Complementary Lithography (e.g. 193i, EUV, E-beam)
Hotspots database
Legal DSA Groups
Benchmark Layout
Number of masks
Design Rules
DSA Technology Path-finding Framework
BCP Specs
Cost Rate of technology perturbation
Design-friendliness Metric
Min-Cost Technology Fix

Figure 4.2: Overview of the Hybrid DSA Technology Exploration Framework

The overview of the framework is shown in Figure 4.2. The framework takes as input the specifications of the technology under evaluation, which are the following:

BCP Specifications. These are the minimum pitch to which the BCP can be compressed (min_dsa_pitch), the maximum pitch to which the BCP can be stretched (max_dsa_pitch)[3], assembled holes dimension (via_width) and the maximum allowed number of vias per DSA group (max_g). The max_g constraint exists because earlier research has shown that smaller DSA group sizes can lead to more robust self-assembly [MTF14].

Number of Masks. This is the number of masks/exposures used to print the guiding templates, in case of Multiple Patterning.

Design Rules. These include $min\_pitch\_same\_mask$ which is the minimum allowed pitch on a mask, and $min\_pitch\_diff\_mask$ which is the minimum allowed pitch between any two guid-

[3]Note that the natural pitch (L_0) of the BCP lies between min_dsa_pitch and max_dsa_pitch.

ing templates even if they are assigned to different masks. [4] Unidirectionality of the masks can also be enforced dictating that the shapes on each mask should all be in the same direction (vertical/horizontal).

Specifications of the Legal DSA Groups. These are the properties of the allowed DSA groups. Properties include manhattan only, collinear only, and equidistant vias only (i.e. pairwise distances between neighboring vias in a DSA group should be identical).

Hotspots database. These are the patterns that are forbidden by the technology under evaluation. More details are provided in Section 4.4.5.

Cost Rate of Technology Perturbation These are the costs of changing the technology. These costs are used by the framework to compute the minimum-cost perturbation to the technology that can make it design-compliant (more details are in Section 4.6).

The definitions of the input parameters are summarized in Table 4.1.

Table 4.1: Definition of Input Parameters

Parameter	**Definition**
$min_dsa_$pitch	minimum pitch to which the BCP can be compressed
$max_dsa_$pitch	maximum pitch to which the BCP can be stretched
max_g	maximum allowed number of vias per DSA group
via_width	width of the via hole
$min\_pitch\_same\_mask$	minimum allowed pitch on a mask
$min\_pitch\_diff\_mask$	minimum allowed pitch between any two guiding templates even if they are assigned to different masks

The output of the framework is a design-friendliness metric for the technology, which is the number of violations on the used benchmark. In addition, the framework shows the resulting DSA groups for each mask/exposure.

To the best of our knowledge, this is the first DTCO framework that can be used to optimally evaluate any DSA-based technology.[5]

[4]If the technology under evaluation assumes that self-assembly is done one time only after all Litho Etch steps (i.e. $(Litho - Etch)^x + DSA$), then then the $min\_pitch\_diff\_mask$ rule should be satisfied even if the shapes are assigned to different masks because of overlay error. However, if the assumed process performs self-assembly after each Litho-Etch step (i.e. $(Litho - Etch - DSA - Etch)^x$), then this rule is not needed and should be set to zero.

[5]The framework is available for download at http://nanocad.ee.ucla.edu/Main/DownloadForm

4.4 Components of the DSA Path-finding Framework

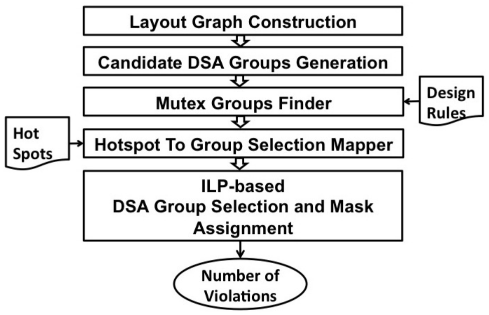

Layout Graph Construction
Candidate DSA Groups Generation
Mutex Groups Finder
Design Rules
Hot Spots
Hotspot To Group Selection Mapper
ILP-based DSA Group Selection and Mask Assignment
Number of Violations

Figure 4.3: Flow of the DSA Path-finding Framework

The flow of the proposed framework is shown in Figure 4.3. First, the candidate DSA groups are generated. Then the pairs of groups which can not co-exist are determined. After that, the group combinations which will result in a forbidden pattern if assigned to the same mask are found. Finally, the output of the previous steps is used to formulate and solve an Integer Linear Program (ILP), that simultaneously performs the group selection and assigns the selected DSA groups including singletons to the masks (a Singleton is a DSA group containing one via only).

4.4.1 Layout Graph Construction

Given the via layer, a graph is constructed such that a graph node is created for each via and a graph edge exists between any two vias whose center-to-center distance is less than $min\_pitch\_same\_mask$. This step runs in $O(n)$, where n is the number of vias.

4.4.2 Candidate Group Generation

All the candidate legal grouping options are generated in this step, starting at each graph node. This is performed on two stages:

(i) **Finding Grouping Options**: Starting at a particular graph node, the layout graph is used to find all the possible strongly connected subgraphs, that contain this node and with number of nodes less than or equal to max_g. This is done by a custom graph traversal algorithm, which saves such subgraphs. This traversal truncates the search from each subgraph as soon as it contains max_g nodes. Practically the number of strongly connected subgraphs is not very large due to the constraint that for each node, we only enumerate the subgraphs having number of nodes less than or equal to max_g. Assuming max_g has a small value, which is usually the case due to DSA yield issues[MTF14], this process runs in $O(n)$, where n is the number of vias. The analysis of this complexity is as follows. Assume the average branching factor (number of neighbors of a node) of the graph is b. The desired maximum number of nodes in the strongly connected subgraph is max_g. Then we enumerate the strongly connected subgraphs by finding all paths of depth max_g or smaller, starting at each node in the graph. So starting at a particular node, the number of these paths is $O(b^0 + b^1 + b^2 + \cdots + b^{max\_G})$=$O(b^{max\_G+1})$. Since we enumerate these paths starting at each node, then the total number of strongly connected subgraphs is $O(n * b^{max\_G+1})$. In the case of our via graphs, b is the number of vias within $min\_pitch\_same\_mask$ from a particular via, and this number is usually small; and with max_G usually being a very small number ($\sim$ 3), the complexity becomes $O(n)$.

(ii) **Finding Candidate Groups**: Not all the grouping options are valid DSA groups. Thus, a technology-specific grouping checker is run on each grouping option, in order to disqualify the non-compliant ones. Grouping checkers are explained in Section 4.4.3.

4.4.3 Grouping Checkers

The specific requirements of the technology are modeled in the technology-specific grouping checker used by the framework. Some common checkers are provided like the collinear grouping checker typically used for 193i and the more flexible grouping checker which is used for EUV experiments. Other options are also provided like requiring all neighboring vias inside the same group to be equidistant. Two examples of grouping checkers are presented next.

193i Grouping Checker In the 193i experiments, a manhattan and collinear grouping checker is used. Given a grouping option represented as a set of vias, the checker considers a group legal if all centers of the vias are vertically or horizontally aligned[6], and the center-to-center distance between every two neighboring vias in the group is within the allowed BCP pitch range.

[6]All vias must be square shapes with a dimension equal to the assembled hole diameter.

EUV Grouping Checker In the EUV experiments, it has been assumed that the legal group can be any non-self-intersecting chain of vias. The following groups are illegal:

(i) Groups whose graphs, similar to the graph explained in Section 4.4.1, have a cycle. This is because such groups will require donut-shape templates in order to confine the self assembly process, and such templates have been assumed difficult to print.

(ii) Groups whose graphs have T-shapes or Fork structures, since it has been assumed that the self-assembly in such a configuration has high defectivity due to the existence of many corners leading to lithography variation and lack of strong confinement [BDG15] (unless high-NA EUV is in use, then such restriction can be alleviated.)

In addition, the distance between every two neighboring vias must satisfy the BCP pitch range. Figure 4.4 shows some examples of EUV groups that are legal in green, others that are illegal in red and a non-manhattan group which can be determined legal or illegal according to the the used knob allowing or disallowing non-manhattan neighborhood.

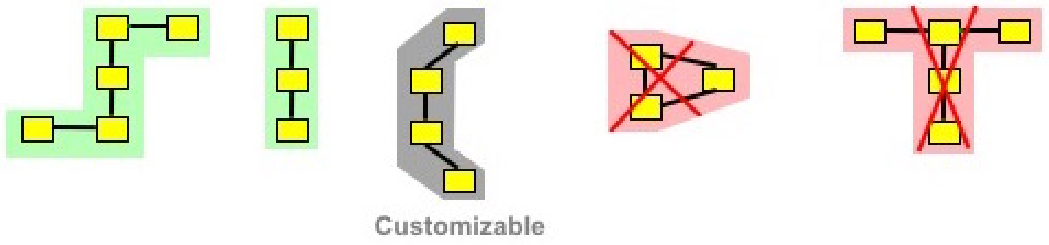


Figure 4.4: Examples of Legal and Illegal groups according to our EUV grouping checker. A line between two vias means that distance between their centers is less than $min\_pitch\_same\_mask$

4.4.4 Mutually Exclusive (Mutex) Groups Finder

A set of two or more DSA groups may not be allowed to co-exist even though each of them is a legal DSA group. This can happen in the following cases:

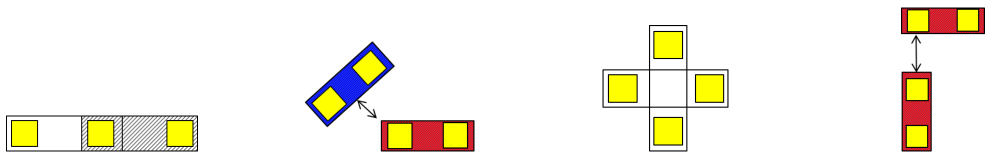
MUTEX Case 1. A set of groups have one (or more) common via(s). Out of all the candidate groups involving a certain via, only one group can be selected. An example is shown in Figure 4.5a.

MUTEX Case 2. Distance between two groups is smaller than ($min\_pitch\_diff\_mask$ - via_width). Thus only one of these groups can be selected (See the definition of the used rules in Section 4.3). An example is shown in Figure 4.5b. Note that if one of the two groups in MUTEX case 2 is a singleton (i.e. non-grouped via), the non-singleton group is removed from the grouping options. This is because the non-singleton group will result in a design rule violation, regardless which mask it gets assigned to; even though the input via layer is DRC-clean.

MUTEX Case 3. Two groups overlap geometrically. However some processes may allow the groups to overlap if they are assigned to different masks[7]. Thus the input knobs of the framework can disable this case. An example is shown in Figure 4.5c.

MUTEX Case 4. Distance between two groups is smaller than ($min\_pitch\_same\_mask$ - via_width). The two groups can be selected only if they are assigned to different masks. An example is shown in Figure 4.5d.

To find the mutex groups belonging to cases 2, 3 and 4 described above, the neighborhood of each via is examined in order to find such pairs of groups. For a Quad-tree implementation of Region Query, finding mutex groups runs in $O(n^{1.5})$. Mutex groups are fed into the ILP formulation (Section 4.5.3).


(a) MUTEX Case 1 (b) MUTEX Case 2 (c) MUTEX Case 3 (d) MUTEX Case 4
Figure 4.5: Examples showing the four MUTEX groups cases

4.4.5 Hot Spot to Group Selection Mapper

In addition to the mutex groups described in Section 4.4.4, some groups can not be assigned to the same mask because they will cause hotspots, even though they satisfy the design rules. A hotspot can be one of the following:

[7]This can be done if self-assembly is done for each mask then the assembled holes are transferred to a hard mask

(i) Lithographic hotspot. This is a low-yield pattern, which is likely to cause a printing failure. [DTP11].

(ii) Complex design rule. In advanced nodes, foundries had to introduce a lot of complex 2D and conditional rules. These rules can require pattern-based representation [DYR07].

(iii) Forbidden pattern due to using a restrictive patterning technology like Self-Aligned Multiple Patterning.

Thus in order to have a correct evaluation for the technology, the generated DSA templates must be hotspot-free. Moreover, by using forbidden patterns, the framework can be used to model and evaluate any new technology with unusual pattern-based requirements.

We use the same pattern representation proposed by Badr et al. [BMG14], where the *segment* representation is used to encode the groups of size two or bigger and the *node* representation is used to encode the singletons. Both representations are needed for every hotspot. For example, Figure 4.6 shows an example of a 2x2 hotspot and its group and singleton representation, where the *nodes* and *segments* are written as a binary string and stored as the equivalent number. Only gridded layouts can have hotspots, in this framework.

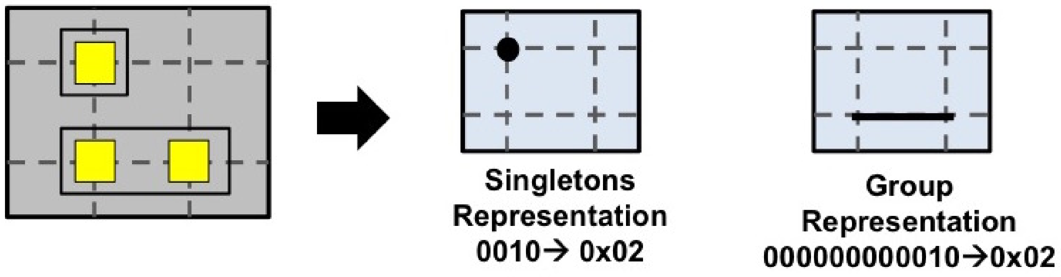


Figure 4.6: A hotspot and its corresponding representation

A hotspot on a mask is defined by a list of *segments* that are occupied by DSA groups, a list of *segments* that are empty, a list of *nodes* that are filled due to singletons, and a list of *nodes* that are empty (i.e. no singletons exist at the *node* location).

The framework performs the grouping and mask assignment such that none of the hotspots occurs in any window in the **mask**. This is done by scanning all non-empty windows, and generating the forbidden combinations of groups. This is performed in $O(k * n)$, where k is the number of hotspots and n is the number of vias. For each hotspot and for each non-empty layout window, the following sets of groups are defined:

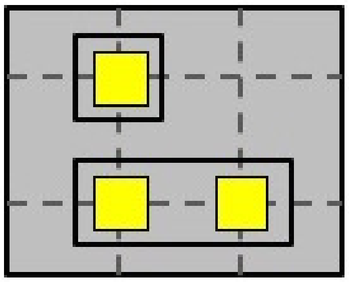
On Groups: Groups that need to exist in order to form the hotspot. For every filled *segment* in the hotspot pattern, one group which spans the *segment* must be *on*. Since one *segment* can be filled by one of several candidate groups, there can be several sets of *On Groups* for a certain window and for a certain hotspot.

Off Groups: Groups that need to be absent in order to create the hotspot. For every empty *segment* in the pattern, all the groups which span it must be *off*. In addition, for every occupied *node*, all the candidate groups of size bigger than one for the via at this location (in the window) must be off (i.e. the via at this location (if any) in the window must exist as a singleton).

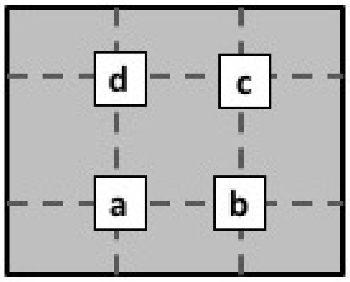
Absent Singletons: Vias that need to be absent in order to form the hotspot. For every empty *node*; the via existing at this location (if any) in the window must not exist as a singleton.

The forbidden group combinations will then be used in the ILP.

For example, for the hypothetical hotspot shown in Figure 4.7a to exist in the layout window shown in Figure 4.7b, there is only one set of **On Groups** in this case and it is $\{g_{\{a,b\}}\}$, the set of **Off Groups** is $\{g_{\{b,c\}}, g_{\{c,d\}}, g_{\{a,d\}}\}$, and the set of **Absent Singletons** is {c}.



(a) Hotspot



(b) Layout Window

Figure 4.7: Example showing the different sets of groups generated for one hypothetical hotspot and one layout window. In this example, **On Groups**: $\{g_{\{a,b\}}\}$, **Off Groups**: $\{g_{\{b,c\}}, g_{\{c,d\}}, g_{\{a,d\}}\}$, **Absent Singletons**: {c}.

4.5 Path-finding Solution using ILP

An ILP is used to do the group selection and mask assignment for the selected groups, simultaneously. [8]

The used constraints are derived from the input to the framework described in Section 4.3, the candidate DSA groups as explained in Section 4.4.2, the mutex groups as described in Section 4.4.4 as well as the forbidden group combinations due to hotspots as explained in Section 4.4.5.
A **conflict** exists between two vias if their center-to-center distance is less than $min\_pitch\_same\_mask$, they are assigned to the same mask and are not in the same DSA group.

The variables and notation used are explained in Table 4.2.

Although the ILP works for 1 mask (Single Patterning (SP)), 2 masks (Double Patterning (DP)), 3 masks (Triple Patterning (TP)) or 4 masks (Quadruple Patterning (QP)) or any higher power of two, the mathematical formulation is presented assuming four masks for the sake of simplicity of the notation.[9]

4.5.1 Objective Function

The objective function in Equation (4.5.1) aims at minimizing the number of conflicts. As explained earlier, a conflict exists between two vias if there is graph edge between them and they are not in the same DSA group.

Note that the objective function does not differentiate between two solutions of different group selections, as long as both solutions have the same number of conflicts. However, the cost function can be modified to add a weighted sum for the groups, and to add a weight to the sum of conflicts. The weight for each group should be inversely proportional to the expected

[8] In case of evaluating a single exposure technology, same formulation is used but there will be no mask or similarity variables, so the result is only the group selection.

[9] In case of TP, other constraints are added in order to prohibit the unused mask bit combinations, similar to the work of Yu et al. [YYZ11] and Badr et al. [BTG15b]. Similarly, in order to support the Quintuple, Sextuple Patterning or other number of masks which is not a whole power of two, constraints are needed to prevent the unused mask bit combinations.

Table 4.2: Notation used in ILP Formulation

c_{ij}	Variable indicating if i^{th} and j^{th} vias are in conflict
m_i^b	b^{th} bit of mask variable of i^{th} via
s_{ij}^b	Similarity variable indicating if b^{th} bit in mask of i^{th} via is identical to the b^{th} bit in mask of j^{th} via
GEs	Set of graph edges in the layout graph
$\mathbf{P}_k$	k^{th} set of vias which can form a legal group
K	Number of candidate groups
$g_{\mathbf{I}}$	Flag indicating if the vias in set **I** are grouped. Variable only exists if the vias in set **I** form a candidate group and if $\lvert\mathbf{I}\rvert \geq 2$
$dir(\mathbf{P_k})$	Orientation of the candidate DSA group formed of $\mathbf{P_k}$. Value is 'v' if vertical; 'h' if horizontal; 'o' if non-collinear.
$\mathbf{E_m}$	m^{th} set of mutex DSA groups
M	Number of sets of mutex DSA groups of MUTEX Cases 1-3
N	Number of sets of mutex DSA groups of MUTEX Case 4
Notation for the Hot Spot Prevention Constraints	
$\mathbf{ONG}_{wq}^h$	q^{th} Set of ON Groups for the h^{th} hotspot, for the w^{th} layout window
$\mathbf{OFFG}_w^h$	Set of OFF Groups for the h^{th} hotspot, for the w^{th} layout window
$\mathbf{AS}_w^h$	Set of Absent Singletons for the h^{th} hotspot, for the w^{th} layout window
$\mathbf{GPS}_w^h$	Groups of Present Singletons for the h^{th} hotspot, for the w^{th} layout window
n_{wq}^h	Index of an arbitrary via in an arbitrary group in $\mathbf{ONG}_{wq}^h$
f_w^h	Index of an arbitrary via in an arbitrary group in $\mathbf{OFFG}_w^h$
$\mathbf{N}_{wq}^{hy}$	y^{th} group in $\mathbf{ONG}_{wq}^h$
$\mathbf{F}_w^{hy}$	y^{th} group in $\mathbf{OFFG}_w^h$

yield of the group, and the sum of the weights of the chosen groups must still be lower than the weight of one conflict. However, with the absence of a succinct but accurate yield model, the notion of optimality of the solution will be suspect, and thus we avoid the weighted groups idea in the cost function.

$$minimize \sum_i \sum_j c_{ij} \tag{4.5.1}$$

4.5.2 Constraints between Vias

Constraints are added to assert the conflict variable between two vias if there is a graph edge between them, they are assigned to the same mask and none of the grouping options including both vias is asserted, as shown in Equation (4.5.2). To represent the problem in **linear** constraints, binary variables are used to encode the mask number, like the work of Yu et al. [YYZ11] The constraints in Equations (4.5.3) are used to assert the similarity variable between any two vias if they are assigned to the same mask, i.e. they force the similarity variable to be the output of XNOR between the two corresponding mask bits. For example, for a pair of vias i and j, if they are both assigned to mask 3, then $m_i^1 = m_j^1 = 1$ and $m_i^2 = m_j^2 = 1$. Accordingly, the similarity variables s_{ij}^1 and s_{ij}^2 are both set to 1 due to Equations (4.5.3a-4.5.3h). In addition, the constraints in Equations (4.5.4) are added in order to only allow the selection of the DSA group if all the involved vias are assigned to the same mask.

$$s_{ij}^1 + s_{ij}^2 - \sum_{\substack{k\in[1..K] \\ \{i,j\}\subseteq \mathbf{P}_k}} g_{\mathbf{P}_k} \le c_{ij} + 1 \quad \forall (i,j) \in \mathbf{GEs} \tag{4.5.2}$$

$$s_{ij}^1 \ge 1 - m_i^1 - m_j^1 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3a}$$

$$s_{ij}^1 \le 1 - m_i^1 + m_j^1 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3b}$$

$$s_{ij}^1 \le 1 + m_i^1 - m_j^1 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3c}$$

$$s_{ij}^1 \ge -1 + m_i^1 + m_j^1 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3d}$$

$$s_{ij}^2 \ge 1 - m_i^2 - m_j^2 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3e}$$

$$s_{ij}^2 \le 1 - m_i^2 + m_j^2 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3f}$$

$$s_{ij}^2 \le 1 + m_i^2 - m_j^2 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3g}$$

$$s_{ij}^2 \ge -1 + m_i^2 + m_j^2 \qquad \forall (i,j) \in \mathbf{GEs} \tag{4.5.3h}$$

$$s_{ij}^1 \ge g_{\mathbf{P}_k} \; \forall \{i,j,k | (i,j) \in \mathbf{GEs}, \{i,j\} \subseteq \mathbf{P}_k, k \in [1..K]\} \tag{4.5.4a}$$

$$s_{ij}^2 \ge g_{\mathbf{P}_k} \; \forall \{i,j,k | (i,j) \in \mathbf{GEs}, \{i,j\} \subseteq \mathbf{P}_k, k \in [1..K]\} \tag{4.5.4b}$$

4.5.3 Mutual Exclusive Group Constraints

As explained in Section 4.4.4, some groups can not co-exist due to MUTEX cases 1-4.

Constraints for MUTEX Cases 1-3

For MUTEX cases 1-3, constraints in Equation (4.5.5) are generated to prohibit the selection of more than one group from each set of MUTEX groups.

$$\sum_{g \in \mathbf{E_i}} g \leq 1 \quad \forall i \in [1..M] \tag{4.5.5}$$

Constraints for MUTEX Case 4

For MUTEX case 4, two mutually exclusive groups can co-exist only if they are assigned to different masks. The constraints in Equations (4.5.6) and (4.5.7) prevent every pair of mutex groups of case 4 from being assigned to the same mask if they are both selected, in the case of the two groups being non-singletons and in the case of one of the two groups being a singleton, respectively.

$$\begin{aligned} &g_A + g_B + s^1_{xy} + s^2_{xy} \leq 3 \quad \forall n \in [1..N] \\ &s.t. \quad \mathbf{E}_n = \{\mathbf{A}, \mathbf{B}\}, |\mathbf{A}| \geq 2, |\mathbf{B}| \geq 2, x \in \mathbf{A}, y \in \mathbf{B} \end{aligned} \tag{4.5.6}$$

$$\begin{aligned} &g_{\mathbf{A}} + s^1_{xy} + s^2_{xy} \leq 2 \quad \forall n \in [1..N] \\ &s.t. \quad \mathbf{E}_n = \{\mathbf{A}, \mathbf{B}\}, |\mathbf{A}| \geq 2, x \in \mathbf{A}, \mathbf{B} = \{y\} \end{aligned} \tag{4.5.7}$$

4.5.4 Hotspot Prevention Constraints

Constraints are added in order to prevent the existence of guiding templates which create hotspots. As explained in Section 4.4.5, for each hotspot pattern and a layout window, there is one or more sets of On Groups, a set of Off Groups, and a set of Absent Singletons. Thus the constraints in Equations (4.5.8) are generated in order to prevent at least one of the required conditions for a hotspot from occurring on any mask. That is; at least one of the On Groups is not selected or is not on the same mask as the rest; or one of the Off Groups is selected and assigned to the same mask or one of the Absent Singletons is present on the same mask. The similarity variables between the vias are used along with the grouping variables (see Table 4.2) to enforce that.

$$\sum_{\mathbf{A}\in\mathbf{ONG}^h_{wq}} g_{\mathbf{A}} + \sum_{\substack{x\in\mathbf{AS}^h_w \\ x\neq(n^h_{wq})}} (1-\prod_{k=1}^{k=2} s^k_{x(n^h_{wq})}) + \sum_{\substack{y=1 \\ i\in\mathbf{N}^{hy}_{wq} \\ j\in\mathbf{N}^{h(y+1)}_{wq} \\ i\neq j}}^{y=\left|\mathbf{ONG}^h_{wq}\right|-1} \prod_{k=1}^{k=2} s^k_{ij} + \sum_{\substack{y=1 \\ i\in\mathbf{F}^{hy}_w \\ i\neq j}}^{y=\left|\mathbf{OFFG}^h_w\right|} [1-\prod_{k=1}^{k=2} s^k_{i(n^h_{wq})}] \leq 2(\left|\mathbf{ONG}^h_{wq}\right| + \left|\mathbf{OFFG}^h_w\right|) + \left|\mathbf{AS}^h_w\right| + \left|\mathbf{GPS}^h_w\right| - 2 \quad \forall q,h,w \tag{4.5.8}$$

4.5.5 Unidirectional Group Constraints

Unidirectional layers have become favorable in advanced nodes using 193i in order to make the most benefit of polarized illumination and Off-axis Illumination [JRL10]. The framework provides the option to force the formed groups on each mask to follow a certain orientation (vertical or horizontal). For unidirectional masks with QP, two masks are vertical and the other two masks are horizontal; for TP, one mask is vertical and the other two are horizontal or vice-versa; finally for DP, one mask is vertical and the other is horizontal. For QP, the constraints in Equations (4.5.9) force each vertical group to be assigned to mask 1 or mask 2 if the group is selected, and each horizontal group to be assigned to mask 3 or mask 4 if the group is selected. A vertical group is a group of two or more vias which are aligned on the same Y-axis whereas a horizontal group is a group of two or more vias which are aligned on the same X-axis. Singletons are not constrained to any direction because the template for a singleton is likely to have aspect ratio of 1:1 [GTA14].

$$g_{\mathbf{P}_k} + m^1_i \leq 1 \quad \forall \{k,i | k\in[1..K], i\in\mathbf{P}_k, dir(\mathbf{P}_k) = \text{`}v'\} \tag{4.5.9a}$$

$$g_{\mathbf{P}_k} - m^1_i \leq 0 \quad \forall \{k,i | k\in[1..K], i\in\mathbf{P}_k, dir(\mathbf{P}_k) = \text{`}h'\} \tag{4.5.9b}$$

4.5.6 Parallelization

It is required to solve the ILP in parallel in order to reduce runtime without sacrificing optimality. Thus, the **connected components** [HT73] of the graph are determined, and the ILP for each connected component is constructed and solved independently. Multiple threads are used to solve the ILPs for the components.

4.6 Minimum-Cost Technology Fix

If the number of violations is not zero, meaning that the technology is not design-friendly, we propose to find the minimum change to the technology rules to ensure design compliance. Note that we also allow some design rule value changes which may require design fixes as well. Any DSA-aware design flows that exist are captured in the design benchmarks used. The technology parameters that are allowed to change are the following:

Decreasing $min\_pitch\_same\_mask$ which means allowing a smaller distance on the same mask and may translate to additional resolution enhancement cost. A cost rate of changing this value by 1nm is one of the inputs to the framework.

Decreasing $min\_pitch\_diff\_mask$ which means allowing a smaller distance between any two DSA groups on different masks and may translate to additional costs in overlay control and etch.

Increasing max_dsa_pitch which indicates that the BCP can be stretched to a longer distance and translates to costs in resolution enhancement of templates and masks to get better confinement or BCP optimization.

For the above three parameters, cost is expressed per nm change.

Increasing max_g, allowing larger DSA group sizes, which translates to BCP optimization and/or better confinement via enhanced resolution in printing templates.

Removing a specific hotspot. Each hotspot in the input hotspots database can be removed, which can be achieved by forcing a design change or a patterning/OPC change.

Removing the unidirectionality constraint on the DSA groups of each mask which again can require more aggressive, expensive OPC or incur yield loss.

Allowing DSA groups on different masks to overlap. The constraint that prevents the existence of two geometrically-overlapping DSA groups on two different masks can be relaxed. This comes at the cost of needing multiple self-assembly steps in the process instead of one.

Using an alternative grouping checker. The alternative grouping checker that allows grouping of any set of vias is similar to the EUV grouping checker explained in Section 4.4.2, with non-manhattan groups allowed. This would usually mean using a different, more expensive template patterning scheme.

An ILP is formulated in order to find the minimum-cost change to make the technology design-friendly.

4.6.1 Technology Change ILP Formulation

This ILP is a variant of the one explained in Section 4.5. The added notation used in the ILP is shown in Table 4.3, showing the technology change variables as well as their associated costs. The technology change variables are similar to the idea of Elastic Programming [Chi07] variables, except that in conventional Elastic Programming, a different elastic variable is added to each constraint that may need to be relaxed; but here the same technology change variable is added to all the constraints representing conflicts that will be resolved by the technology change.

The objective function is to minimize the cost of the selected technology changes, as shown in Equation (4.6.1).

$$\begin{aligned} minimize \sum_d cost\_dm_d * e\_dm_d + \sum_d cost\_sm_d * e_{sm_d} \\ + \sum_i cost\_h_i * e\_h_i + cost\_u * e_u + cost\_ov * e\_ov \\ + \sum_x \sum_d cost\_gc^d_x * e\_gc^d_x \\ + \sum_x \sum_d cost\_agc^d_x * e\_agc^d_x \end{aligned} \tag{4.6.1}$$

The constraints in Section 4.5 have been modified in order to add the technology modifi-

Table 4.3: Notation used for Minimum-Cost Technology Change ILP

e_sm_d	Variable indicating if $min\_pitch\_same\_mask$ is relaxed to d
e_dm_d	Variable indicating if $min\_pitch\_diff\_mask$ is relaxed to d
e_h_i	Variable indicating if the min-cost technology change involves removing the i^{th} hotspot
e_u	Variable indicating if the min-cost technology change involves removing the uni-directionality constraint
e_ov	Variable indicating if the overlap between templates on different masks is part of the min-cost technology change
$e\_gc_x^d$	Variable indicating if the min-cost technology change involves a relaxed grouping condition allowed by the original grouping checker under evaluation, $max\_g = x$ and $max\_dsa\_pitch = d$
$e\_agc_x^d$	Variable indicating if the min-cost technology change involves a relaxed grouping condition allowed by the alternative grouping checker, $max\_g = x$ and $max\_dsa\_pitch = d$
$\mathbf{GC}(g_{\mathbf{I}})$	Set of grouping conditions ($e\_gc_x^d$ and $e\_agc_x^d$) allowing the grouping of set of vias **I**
$cost\_sm_d$	Cost of e_sm_d which is the input cost rate of changing $min\_pitch\_same\_mask$ multiplied by $(d - min\_pitch\_same\_mask)$ Default value of cost rate is 2 per nm.
$cost\_dm_d$	Cost of e_dm_d which is the input cost rate of changing $min\_pitch\_diff\_mask$ multiplied by $(d - min\_pitch\_diff\_mask)$ Default value of cost rate is 2 per nm.
$cost\_h_i$	Cost of e_h_i (input). Default value is 1.
$cost\_u$	Cost of e_u (input). Default value is 1
$cost\_ov$	Cost of e_ov (input). Default value is 4.
$cost\_gc_x^d$	Cost of $e\_gc_x^d$ which is the input cost rate of changing max_dsa_pitch multiplied by $(d - max\_dsa\_pitch)$
$cost\_agc_x^d$	Cost of $e\_agc_x^d$ which is the cost of the alternative checker added to the product of the input cost rate of changing max_dsa_pitch and $(d - max\_dsa\_pitch)$. Default value of cost of the alternative checker is 10.
$dist_{ij}$	distance between centers of vias i and j

cations. Constraints of Equation (4.5.2) have been updated as shown in Equation (4.6.2) where the technology change variables that would solve the conflict, represented by the constraint, are added. The conflict can be resolved by using a $min\_pitch\_same\_mask$ smaller than $dist_{ij}$, or using a bigger max_dsa_pitch or using the alternative grouping checker with same or bigger max_dsa_pitch. The conflict removal due to grouping is reflected in the addition of groups, which means that K has increased. Moreover the conflict variables (c_{ij}) no longer exist, which means that the solution is not allowed to have any conflict/violation.

$$s^1_{ij} + s^2_{ij} - \sum_{\substack{k\in[1..K]\\ \{i,j\}\subseteq \mathbf{P}_k}} g_{\mathbf{P}_k} \leq 1 + \sum_{d=1}^{d=dist_{ij}} e\_sm_d \quad \forall (i,j) \in \mathbf{GEs} \tag{4.6.2}$$

A constraint has been added to make sure that at most one grouping condition is selected as shown in Equation (4.6.3).

$$\sum_x \sum_d e\_gc^d_x + \sum_x \sum_d e\_agc^d_x \leq 1 \tag{4.6.3}$$

Another constraint has been added to pick a grouping condition variable enabling a certain group of vias, as shown in Equation (4.6.4).

$$\sum_{e\in \mathbf{P_k}(g_\mathbf{I})} e \geq g_{\mathbf{P}_k} \ \forall \{k \mid k \in [1..K]\} \tag{4.6.4}$$

The constraints for MUTEX case 2 and 4 in Equation (4.5.5) have been relaxed by adding the enabling change variables e_sm_d and e_dm_d respectively on the right hand side (rhs) of the constraint. Similarly, the constraints for MUTEX case 3 in Equation (4.5.5) have been relaxed by adding the variable that allows overlap (e_ov) between the groups on different mask on the rhs of the constraint. Similarly, the hotspot constraints in Equation (4.5.8) have been relaxed by adding the corresponding hotspot removal variable e_h_i on the rhs.

4.7 Case Studies and Results

In this section, we present several exploration studies which have been done for DSA-based technologies using the proposed framework. The explored complementary lithography techniques include combinations of 193i, EUV, SADP and E-beam. It is worth noting that these experiments are only examples to illustrate the usage of the framework. However, the output of the framework will strongly depend on the used parameters, thus the changing the parameters will lead to different conclusions about the technology.

The framework is implemented in C++, using Open Access for layout manipulation. IBM CPLEX was used to solve the ILP. The experiments were run on a computing cluster, with a maximum of 4 threads on four cores and total of 80G of virtual memory. The benchmarks were synthesized, placed and routed using a projected 7nm library from a leading IP provider, then the layouts were scaled down to 5nm layouts. After scaling, the via dimension is 15nm. All the experiments are either performed on the V1 layer or the V3 layer. The number of vias on these two layers in the used benchmarks is shown in Table 4.4.

The parameters used with different lithography techniques are shown in Table 4.5.

Table 4.4: Number of vias in test cases

Test case	Number of Vias on V1	Number of Vias on V3
AES	98896	14360
MIPS	86939	7274
USB	99366	7346

Table 4.5: Parameters used in studies

Parameter	**Lithography**	**Value**
min_dsa_pitch	all except SADP	27nm [GTA14]
max_dsa_pitch	all except SADP	51nm
min_dsa_pitch	SADP	48nm
max_dsa_pitch	SADP	50nm
max_g	193i	3
max_g	SADP	2
max_g	EUV	7
$min\_pitch\_same\_mask$	193i	90nm
$min\_pitch\_same\_mask$	EUV	40nm
$min\_pitch\_diff\_mask$	193i	25nm
$min\_pitch\_diff\_mask$	EUV	22nm

4.7.1 DSA+ EUV SP vs. DSA+193i TP

In this study, we explore the feasibility of using EUV with 1 mask only to replace three masks in a 193i process to print the guiding templates for V1 layer. As shown in Table 4.5, a relatively big maximum group size was used (max_g=7); while in 193i a smaller group size was used (max_g=3) because the higher resolution of EUV can be used to achieve strongly confining templates having peanut shapes [GBD16], which result in less placement error [MTF14]. For EUV, two scenarios are compared: one where only manhattan DSA groups are allowed and one where non-manhattan groups are allowed as well.

The results of the experiment, in Table 4.6, show that EUV with non-manhattan DSA groups can replace three masks of 193i since it has only one violation on one benchmark which occurred because of an off-grid via. However, EUV with manhattan groups only can not. Our result for EUV with non-manhattan groups agrees with the claim and empirical observation by Gronheid et al. [GBD16] that DSA+EUV SP can be used to pattern via layer in 5nm node.

Table 4.6: Number of violations with SP EUV with manhattan DSA groups only, SP EUV with manhattan and non-manhattan DSA groups, 193i TP. Number of Violations in case of 193i DP is also shown.

Testcase	DSA+EUV SP man.		DSA+EUV SP non-man		DSA +193i TP	DSA+193i DP
	Viol.	Runtime(m)	Viol.	Runtime(m)	Viol.	Viol.
aes	134	5.9	0	6.2	0	5930
mips	186	10.6	1	5.12	0	3476
usb	152	7.76	0	6.97	0	3977

4.7.2 DSA+ 193i TP + Unidirectional templates vs. DSA + 193i TP + Bidirectional templates

As explained in Section 4.5.5, restricting the shapes on a mask to a certain direction can be beneficial to the process optimization. In this experiment, we evaluate the design-friendliness penalty of forcing the all the groups on each mask to be unidirectional, using TP where two masks are horizontal and one mask is vertical. The uni-directionality of the mask shapes did not sacrifice design-friendliness as shown in Table 4.7, which also shows the number of candidate

DSA groups resulting from Section 4.4.2 and the number of selected DSA groups having more than one via in the design. Results show that the number of DSA groups has decreased, leading to more singletons (a guiding template printing one via only), which is expected since the undirectionality constraint has limited some groups. The number of candidate groups has not changed because the unidirectionality constraint has an effect only when the ILP is solved.

Table 4.7: Bidirectional DSA templates vs. Unidirectional templates on each mask (2 horizontal masks and 1 vertical) vs. on V1, using DSA+193i TP: Number of violations, Number of candidate groups and number of selected groups

Testcase	DSA+193i TP Bidirectional				DSA+193i TP Unidirectional			
	Viol.	Num Cand. Groups	Num Groups.	Runtime(m)	Viol.	Num Cand. Groups	Num Groups	Runtime(m)
aes	0	54885	7432	2.8	0	54885	5819	2.9
mips	0	49587	5691	2.18	0	49587	4816	2.68
usb	0	56038	6666	3.3	0	56038	5618	3.4

4.7.3 DSA+ E-beam + 193i

Hybrid lithography involving E-beam has already been studied in several works [YLZ15, TZX14]. In this experiment we consider a hybrid lithography process where the guiding templates for DSA are printed using 193i lithography. Then the templates which violate the $min\_pitch\_same\_mask$ are printed using e-beam, with the hope that the number of violations would be small enough such that the throughput is still not too low. The percentage of the vias which are in conflict and require their templates to be printed using e-beam is shown in Table 4.8. Assuming a threshold of 10%, it is clear that E-beam can likely save one mask exposure.

4.7.4 DSA+193i SADP

In this experiment, we study the feasibility of using SADP (using 193i) to pattern the templates for DSA. We use the SADP decomposition method used by Xu et al. [XCY15], where tracks are alternated between between mandrel and non-mandrel and the trim is used to create the vertical edges, which are the line ends. We use the SADP-friendly design rules used by Xu et al. [XCY15], which have been adapted from the work of Luk-Pat et al. [LMP12]. These design rules are translated into pattern-based rules (like hotspots). This experiment is run on

Table 4.8: DSA+193i+E-beam:Percentage of shapes to print with E-beam with 193i and different number of masks

Testcase	DSA +193i SP +E-beam		DSA +193i DP +E-beam		DSA +193i TP +E-beam	
	% of Ebeam	Runtime(s)	% of Ebeam	Runtime(s)	% of Ebeam	Runtime(s)
aes	92%	2.2	9%	2.5	0%	2.8
mips	88%	2.1	7%	2.5	0%	2.18
usb	89%	3.3	8%	3.2	0%	3.288

the V3 layer for 7nm layouts (without scaling to 5nm). SADP is modeled as follows: the framework is run with one mask only. The first SADP rule (*OnTrackSpace*) has been enforced by setting $min\_pitch\_same\_mask$ to 59nm, thus there is no need for pattern-based enforcing of the *OnTrackSpace* rule. However the other three rules are enforced by representing the possible design rule violation as a forbidden pattern (hotspot), to be avoided. We used the following rule values [XCY15]: s_r=50nm, w_r=50nm, w_e=5nm, w_sp=40nm. No minimum area rule was enforced. The forbidden patterns used to model the *OffTrackOverlap* rule are shown in Figure 4.8 and those used to model the *OffTrackSpace* rule are shown in Figure 4.9. The patterns required to enforce the *OffTrackOffset* happen to be already included among the patterns of *OffTrackOverlap* rule. All the patterns are input to the framework in the format described by Badr et al. [BMG14] and shown in Figure 4.6.

Since SADP is more appropriate for regular layouts, we assume that the printed templates are all squares (for singletons) or rectangles (for groups of size two). Thus, we assume the templates do not have peanut shapes and this can increase the placement error of self-assembled holes [MTF14]. To compensate, we only allowed groups of size two maximum, and we restricted the range of the self-assembly pitch to 2 nm: 48nm-50nm. The guiding templates were designed as ellipses by Gharbi et al. [GTA14] with an aspect ratio of 2:1 to print groups of size two, and with an aspect ratio of 1:1 to print singletons (dimension of square template for a singleton was assumed to be 40nm). We adopt the same aspect ratio, but our templates are rectangles.

The results are shown in Table 4.9. We compare safe SADP, where trim edges can only print

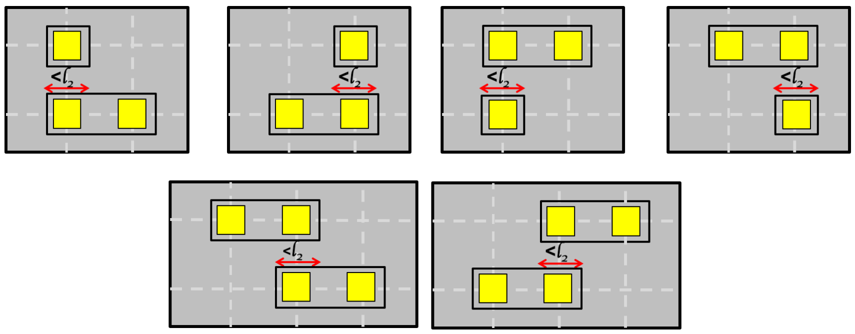


Figure 4.8: The forbidden patterns used to model SADP-friendly rule *OffTrackOverlap*, defined by Xu et al.[XCY15] l_2 is the value of the *OffTrackOverlap* rule

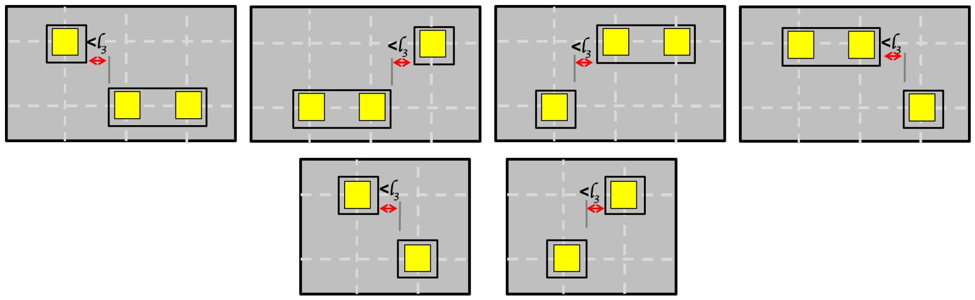


Figure 4.9: The forbidden patterns used to model SADP-friendly rule *OffTrackSpace*, defined by Xu et al.[XCY15]. l_3 is the value of the *OffTrackSpace* rule

the vertical edges of the shapes and thus trim edges always lie in the middle of the sidewall; sensitive SADP, where trim edges are allowed to coincide with the spacer edge to have more relaxed design rules, eliminating the need for the *OffTrackSpace* rule; and Single Patterning (SP). The overlay-sensitive SADP has a few violations, while safe SADP is not appropriate for patterning the templates in this scenario.

Table 4.9: Number of violations with overlay-safe SADP, overlay-sensitive SADP and SP on V3 layer

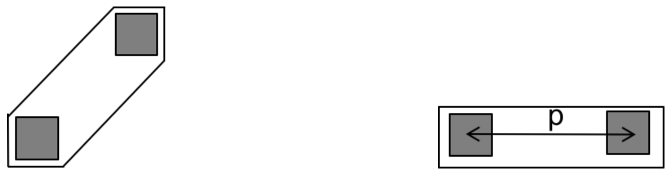
Testcase	DSA +SADP safe		DSA +SADP sensitive		DSA +193i SP	
	Viol.	Runtime(s)	Viol.	Runtime(s)	Viol.	Runtime(s)
aes	1169	12	17	28	1618	12
mips	342	13	9	22	468	9
usb	350	13	5	13	452	8

4.7.5 Minimum-cost Technology Fix Experiments

As explained in Section 4.6, if the technology under evaluation is not friendly to the design, then the framework finds the minimum-cost change to the technology to remove all the violations. The computation of the technology change is run using one thread with maximum virtual memory of 16G. We don't use graph decomposition methods to solve it, because the technology change is global across the whole benchmark even if there are disconnected subgraphs of vias. The runtimes for the technology change finder ranges between 30 seconds to 4 hours on our benchmarks. We used the default cost parameter values mentioned in Table 4.3.

4.7.5.1 DSA+193i DP

The use of DSA+193i DP was shown to be insufficient to pattern the guiding templates of V1 layer, as shown in Table 4.6. The benchmarks have pairs of vias that can not be grouped because the two vias form an inclined DSA group (Figure 4.10a) which is considered illegal by the used 193i grouping checker and the distance between the two vias is greater than the max_dsa_pitch (Figure 4.10b). In such cases DSA was unable to resolve the violation through grouping. However, these were not the only reasons of failure, the minimum pitch is also too constrained for the dimensions and the configurations in the 5nm layouts leading to DP decomposition errors. A snippet of the result of decomposition and grouping for DSA+193i DP is shown in Figure 4.11.



(a) Illegal grouping configuration (b) Distance p $< max\_dsa\_pitch$

Figure 4.10: Two reasons where DSA did not help remove the violations in DSA +193i DP

The computed technology fix is is to change max_dsa_pitch to 60nm (instead of 51nm) and change $min\_pitch\_same\_mask$ to 78nm (instead of 90nm). The cost of the technology change is 41. Changing the cost parameters can result in different solutions.

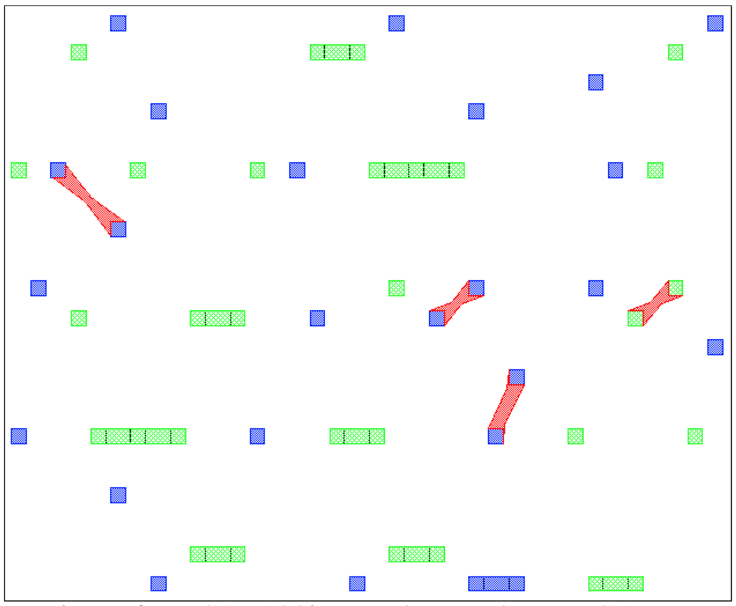


Figure 4.11: Layout snippet for DSA+193i DP. Blue markers: DSA groups on Mask 1, Green markers: DSA Groups on Mask 2. Red markers: Mask Violations.

4.7.5.2 DSA+ SP EUV using Manhattan DSA groups Only

The results of using DSA+SP EUV with manhattan DSA groups only were shown in Table 4.6. The computed technology fix is to decrease the $min\_pitch\_same\_mask$ to 35nm (instead of 40nm), at a cost of 9. Using the alternative grouping checker which allows non-manhattan groups would have removed the violations, but at a higher cost of 10.

4.7.5.3 DSA+ SADP on V3

In the scenario of using DSA+safe SADP to print the V3 layer (Table 4.9), the suggested technology fix is to remove 9 out of 12 forbidden patterns representing the DSA rules. While for DSA+sensitive SADP, the suggested technology fix is to remove 3 out of 6 forbidden patterns. Interpreting this as a technology fix would mean violating SADP constraints. However, in this case, the technology fix is rather interpreted as removing the indicated patterns from the design, so the only way to be able to use DSA+SADP and benefit from the overlay advantages of SADP, is to have a correct by construction DSA+SADP-aware design.

4.8 Conclusion

We have proposed a framework that can be used for path-finding for the hybrid DSA technologies in which a complementary lithography technique that is possibly multi-patterned is used to print the guiding templates. Given, the choice of the allowed groups, number of masks, design and mask rules, characteristics of block copolymer and hotspots, the framework reports design-friendliness on the provided benchmarks. The framework is generic in the sense that it can be used to evaluate any type of hybrid DSA technology. Several case studies have been shown, including studies where the complementary lithography technique is 193nm immersion lithography, EUV, SADP and E-beam.

CHAPTER 5

Mask Assignment and DSA Grouping for DSA-MP Hybrid Lithography for sub-7nm Contact/Via Holes

Directed Self Assembly (DSA) is a very promising candidate for the sub-7nm technology nodes. To print such small dimensions, Multiple Patterning (MP) is likely to be used to print the guiding templates for DSA. Therefore algorithms are required to perform the DSA grouping at the same time as the mask assignment. In this work, we present an optimal Integer Linear Program (ILP) to solve this problem for two schemes of hybrid DSA-MP process. Scalable heuristic algorithms are also proposed to solve the same problem.[1]

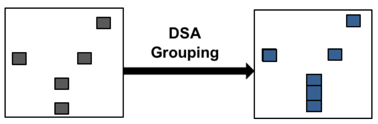
5.1 Introduction

In continuous search for new technologies to enable the sub-7nm nodes, Directed Self Assembly (DSA) has presented itself as a strong candidate, especially with the continuous delay of Extreme Ultraviolet Lithography (EUVL). Even if EUVL gets into production, there are far more challenges with the transition to high Numerical Aperture (high-NA) EUVL which will be needed for sub-11nm resolution, making the partnership of EUVL with Multiple Patterning (MP) an alternative option [KWH14]. Thus, Multiple Patterning is expected to enable several sub-7nm nodes. With the cost being the main drawback of MP and with DSA having native frequency multiplication properties, substituting one mask in an MP process with DSA is a tempting cost reduction[MTF14]. In addition, DSA has been reported to possess significant rectification capability in Critical Dimension Uniformity (CDU) and Edge Roughness for contacts[SYS13]. DSA has been successfully demonstrated for contact holes (for e.g.,

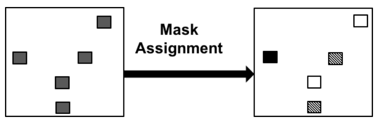
[1]The material in this chapter is based on the published work [BTG17].

[CBB10]) and lamellae (for e.g., [WRG13]). Since DSA is capable of printing dense nano features of roughly uniform dimensions [RKD08], it is a very good fit for contact and via layers.

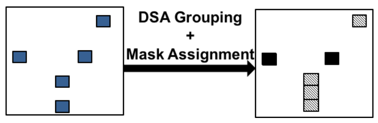
In this work, we focus on the hybrid DSA-MP process for contact/via holes and study the problem of simultaneous MP decomposition and DSA grouping. DSA Grouping is the task of assigning contacts into groups such that each group is self assembled in the same guiding template (Figure 5.1a), while MP decomposition is determining the mask for every polygon (Figure 5.1b). The guiding templates for DSA are assumed to be printed using 193nm Immersion Lithography (193i). Simultaneous DSA Grouping and Mask Assignment required for a DSA-MP process are shown in Figure 5.1c, and are the objective of the algorithms in this chapter. It is required to determine the DSA groups and choose the mask for each group. Details about the hybrid DSA-MP process are presented in Section 5.2.



(a) Grouping in DSA, resulting in four groups: one doublet and three singletons



(b) Mask Assignment in Triple Patterning.



(c) Simultaneous DSA Grouping and Mask Assignment in a Hybrid DSA-MP process, using Double Patterning.

Figure 5.1: Grouping required for DSA process, Mask Assignment required for MP process and Simultaneous DSA Grouping and Mask Assignment required for a DSA-MP process.

Our contribution in this work is summarized as follows:

- An optimal Integer Linear Programming (ILP) formulation is presented to perform the mask assignment and DSA grouping simultaneously, for a hybrid DSA-MP process in which all masks apply self-assembly. Heuristic algorithms are also proposed and are benchmarked against the ILP solution.

- An optimal ILP formulation is proposed to solve the mask assignment and DSA-grouping for a hybrid process in which not all the masks apply self-assembly. Heuristics are also presented to solve the same problem.

5.1.1 Multiple Patterning (MP)

The semiconductor industry has managed to scale to the 22nm node and beyond using MP [PLY15], where pitch multiplication is achieved by using multiple masks to print one layer. There are two types of Multiple Patterning: the first one is based on N repetitions of Litho-Etch $(LE)^N$ where N is the number of masks, and the second is Self Aligned Multiple Patterning [PLY15]. A lot of work has been done for mask decomposition for both types of MP, for example: [KPX08b, YYZ11, ZYC14, ZDW11]. In this work, we assume a hybrid DSA-MP process which uses $(LE)^N$ along with DSA.

5.1.2 DSA Capabilities

The BCP has a natural pitch L_0, to which it assembles, if not strongly guided by templates. To create a hole array with a pitch different from the natural pitch of the block copolymer, strong confinement is needed in the templates [YBT13]. Small templates achieve strong lateral confinement for the block copolymer leading to more precise control of the self assembly process [CBB10, BYB11]. In addition, smaller defects density can be obtained with smaller size of templates [BRB07, KCH11, SDS04]. Accordingly templates should be designed such that a very small number of contacts are created per template [MTF14]. In addition, previous research has reported that using peanut-shaped templates with a very narrow neck between every pair of contacts can lead to less placement error [MTF14]. However well-modulated peanut shapes are hard to print in 193i photolithography, therefore it is preferred to have the pitch of grouped contacts close to the natural pitch of the copolymer and to avoid 2D groups altogether [MTF14]. Diagonal groups (which have larger pitch than L_0) are also not desired because they need very strong confinement which can only be achieved by very complicated peanut-shape guiding templates which are also difficult to print in 193i photolithography [DGW13].

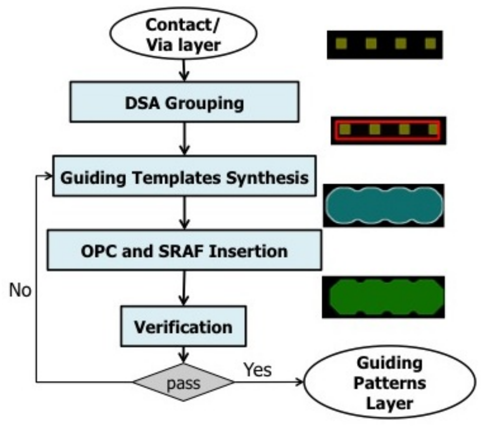


Figure 5.2: CAD Flow for a Single-Exposure DSA Process

5.1.3 CAD Flow for DSA

In a DSA process which prints the guiding templates in a single exposure, the guiding templates need to be determined based on the given contact/via layer. Figure 5.2 shows a typical flow that is used to design the guiding templates. First, the DSA grouping algorithm determines which contacts are to be assembled using the same guiding template and hence the DSA groups are generated. The grouping algorithm has to consider the lithography-driven spacing constraints which the guiding templates need to satisfy. For each DSA group, a guiding template is synthesized; the synthesis process attempts to reverse-engineer the self assembly process in order to come up with the correct templates. The templates then undergo the classical optical treatment like OPC and SRAF insertion to enhance the resolution. Finally verification is performed, to compare the expected assembled contacts (based on the synthesized templates) to the target contacts.

However in a technology that has multiple exposures, the DSA grouping method has to be coupled with the mask assignment method. In [BTG15a], it has been shown that cascading the traditional DSA grouping method with the Multiple Patterning Decomposer, which are both unaware of the hybrid nature of the process, produces poor results.

In this chapter we study the optimal formulation and heuristic algorithms that can solve the DSA grouping and Mask assignment problem for two different schemes of the hybrid DSA-MP process.

The rest of the chapter is organized as follows: section 5.2 describes the two schemes of the hybrid DSA-MP process assumed in this work and explains the rules of the hybrid process. In section 5.3, we introduce the graph structure used in our algorithms. Sections 5.4 and 5.5 present the optimal and the heuristic algorithms proposed for the two schemes of the hybrid DSA-MP process. Section 5.6 shows and analyses the results. Finally, conclusions and future work are presented in section 5.7.

5.2 Hybrid DSA-MP Process

5.2.1 Alternative Hybrid Schemes

There are two alternative schemes for a hybrid DSA-MP process for contact/via holes[MTF14]. In the first scheme, each of the N masks prints the guiding templates for DSA, then the self-assembly of the BCP will create the actual holes. We refer to this scheme as All_DSA. In the second scheme, some of the masks will directly print the contact holes and thus do not go through self-assembly, but the other masks will print the guiding templates and use self-assembly to create the holes. We refer to this scheme as $Partial\_DSA$. The main advantage of the second scheme is that the masks bypassing DSA can print shapes or sizes different from the uniform dimensions determined by DSA, for example allowing rectangular (bar) vias which have appeared starting from the 28nm node [Bru13].

In both schemes we assume that the same-mask minimum allowed pitch that applies to any two DSA groups has the same value as the same-mask minimum allowed pitch that applies between any two contacts/vias on a mask which is not applying self-assembly. This is the pitch we refer to as $litho\_pitch$.

Under these assumptions, for a via/contact layer with **all** holes having the **same square dimension required by DSA**, the $Partial\_DSA$ process scheme is more restrictive than the All_DSA scheme. This is proved in the next lemma.

Lemma 5.2.1. *Any via layer which is compliant with $Partial\_DSA$ scheme and whose vias are all squares of the same dimension determined by DSA is also manufacturable with the same number of masks in All_DSA.*

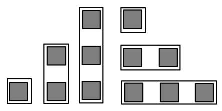


Figure 5.3: Legal DSA groups, if maximum group size (max_g) is 3

Proof. Assume that there is a via layer that is manufacturable (violation-free) in $Partial\_DSA$, but is not manufacturable with All_DSA, and all its vias are squares with the dimension compliant with DSA.

The conversion from $Partial\_DSA$ to All_DSA is performed as follows: for each contact hole that has been assigned to a mask bypassing self-assembly, a guiding template is created on the corresponding mask in All_DSA, with one contact/via. All the masks that are applying self-assembly in $Partial\_DSA$ are used in All_DSA without change.

Since there are no violations on the original masks for $Partial\_DSA$, there must be no violations for the All_DSA masks resulting from the conversion above. Thus a contradiction exists, and this design must also be manufacturable in All_DSA. □

However, designs which have bar vias in addition to the square vias may be manufacturable in $Partial\_DSA$ but not in All_DSA. This is because self-assembly of the BCP can only result in holes with a particular uniform dimension. Holes of non-uniform dimensions can be patterned in $Partial\_DSA$ by being assigned to the masks which do not apply self-assembly, assuming no coloring conflicts exist.

The work in this chapter is only concerned with contact/via hole patterning. We assume that 193i is used to print the guiding templates. Accordingly, DSA-grouped contacts are only allowed to be collinear and either vertically or horizontally aligned, because the lithography variations in guiding templates needed for more complex DSA groups can lead to high defectivity level in self-assembly [BDG15]. We assume there is a maximum number of allowed contacts per groups, which is an input value (max_g). For example, for a max_g value of 3, the legal DSA groups are the ones shown in Figure 5.3.

Thus, given a process which has Multiple Patterning (N masks) and DSA, it is required to do the DSA grouping and decompose the contact/via holes onto the N masks; in order to minimize the number of mask violations. In a violation-free solution, any two contacts/vias whose centers are separated by a distance less than the same-mask minimum pitch are either assigned to different masks or in the same DSA group on the same mask. Stitch-free decomposition has been assumed because most of the templates are expected to have small size.

5.2.2 Important Parameters in the Process

There are several important parameters in this problem:

(i) **Contact/Via hole DSA-manufacturable dimension** ($hole\_dim$): the width of contact/via that is manufacturable though the self-assembly of the BCP. Only the square contacts/vias of this dimension are assumed to be DSA-manufacturable, while contact/via holes not adhering to this dimension can only be printed in the $Partial\_DSA$ scheme.

(ii) **Minimum Grouping Pitch** (min_dsa_pitch): minimum distance that can exist between centers of two contact/via holes in a DSA group. This distance is equal to the natural pitch (L_0) of the block copolymer.

(iii) **Maximum grouping distance** (max_dsa_pitch): maximum distance that can exist between centers of two neighboring contacts/vias in one DSA group. This is derived from the properties of the block copolymer, because its self-assembly pitch can not be stretched beyond a certain threshold.

(iv) **Minimum Lithography Pitch** ($litho\_pitch$): minimum space that can occur between the centers of any two shapes on a particular mask.

(v) **Maximum DSA Group Size** (max_g): maximum number of holes that can be DSA-grouped together, and hence manufactured using the same guiding template.

(vi) **Number of masks** (N): number of masks/exposures in the process. We use b to denote the minimum number of bits required to encode N: $b = ceil[log_2(N)]$.

The distance range between centers of any two contacts/vias is divided into three regions, showing whether DSA grouping and/or assignment to different masks (MP) can be used to resolve the spacing violation, as shown in Figure 5.4 . Outside the DSA-allowed range [min_dsa_pitch,

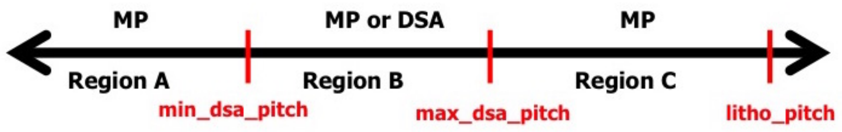


Figure 5.4: Ranges of distance between two polygons, where DSA and/or MP can resolve the spacing conflict.

max_dsa_pitch], only MP can be used to resolve the conflict between the two contacts. Note that it has been assumed that $litho\_pitch$ has larger value than max_dsa_pitch, which complies with the ranges in literature (for e.g. [CBB10, MLT15]), assuming 193i lithography is used to print the templates.

5.3 Hybrid Grouping /Spacing Graph Representation

In this section, the new graph structure which considers both DSA and MP is explained. We use a hybrid grouping/spacing graph (GG/SG). Each contact/via is represented as a graph node. There are two types of edges: spacing edges, and grouping edges. A spacing edge exists between every two contacts whose centers are within $litho\_pitch$ from each other. A grouping edge is created between every two direct-neighboring contacts/vias that can be grouped into the same guiding template. These contacts/vias are aligned on the same horizontal or vertical axis, and the distance between their centers is within the DSA grouping interval: [min_dsa_pitch, max_dsa_pitch]. An example of the hybrid graph is shown in Figure 5.5. When we are only interested in the grouping edges, we refer to the graph as Grouping Graph (**GG**), and we refer to it as Spacing Graph (**SG**) [2] when only the spacing edges are of interest. Sections 5.4 and5.5 will show how the hybrid graph is used in the proposed algorithms.

Graph Division Similar to [YYZ11], we apply the graph division technique based on connected components. The independent connected components of the hybrid graph are determined, and each independent component can be processed independently in the following algorithms. In our implementation, we use OpenMP threads in order to process the graph compo-

[2]Note that the "Spacing Graph" is similar to the "Conflict Graph" used in some mask decomposition works, for e.g. [KPX08b].

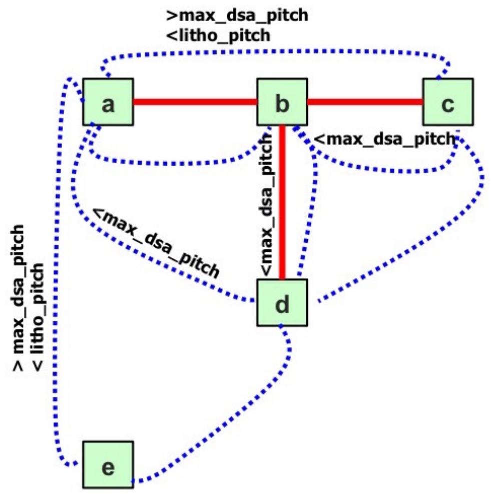


Figure 5.5: Hybrid Graph between five contacts. Grouping edges are shown as solid lines, while spacing edges are shown as dotted curves. Distance between any two contacts/vias on this graph is at least greater than min_dsa_pitch. A spacing edge exists wherever the distance between the centers of two contacts/vias is less than $litho\_pitch$ and a grouping edge exists wherever two direct-neighboring contacts can be grouped, i.e. distance between their centers is in the acceptable DSA grouping range, and are aligned on same X-axis or Y-axis.

nents in parallel.

5.4 Algorithms for the All_DSA Scheme

In this section, we present the optimal ILP formulation and heuristic algorithms for the simultaneous DSA grouping and MP decomposition problem for the All_DSA scheme.

5.4.1 ILP Formulation

In a hybrid DSA-MP process, a **conflict** between two contacts/vias means that the distance between their centers is less than the $litho\_pitch$, but are assigned to the same mask and they do not lie in the same DSA group. In this formulation, the objective is to minimize the number of conflicts. The constraints are derived from DSA as well as lithography requirements.

The constraints are generated based on the distance between centers of contacts/vias, according to the distance number line shown in Figure 5.4 which is summarized as follows: if the distance is less than min_dsa_pitch or greater than max_dsa_pitch, then the two contacts/vias have to to be assigned to different masks. If the distance is greater than min_dsa_pitch but less

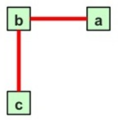


Figure 5.6: Example for a connected component on GG. Contacts a and c do not have direct grouping edge, but a grouping path exists between them. Yet, they must not be grouped.

than max_dsa_pitch then the pair of contacts/vias are either to be assigned to different masks or grouped together for DSA on the same mask. Otherwise, a conflict occurs.

To represent the problem in **linear** constraints, binary variables are used to encode the mask number, like [YYZ11]. Our ILP works for Double Patterning (DP), Triple Patterning (TP), Quadruple Patterning (QP) and other higher powers of two. However for simplicity of the notation, we only present it for QP, and we explain later the differences in the generated ILP when a different number of masks is used. For QP, two bits are required to represent the mask.

To generate the ILP constraints, it is required to construct the hybrid graph and then find the *connected components* in the GG. If two contacts/vias belong to the same connected component, then a path of grouping edges exists between them and therefore they can get grouped through that grouping path. However some of them may not be groupable because they are not manhattanly aligned, and thus their grouping has to be explicitly prohibited via special constraints. For example, in Figure 5.6, contacts a and b are allowed to be in same group, and contacts b and c can also be in same group, but these two simultaneous groupings imply the grouping of a and c which is disallowed because they are not manhattanly aligned. Therefore constraints must be added to prohibit grouping of non-groupable pairs that lie in the same connected component like the case of contacts a and c in Figure 5.6.

The variables and notation used are explained in Table 5.1. The mathematical formulation is as follows:

$minimize$

$$\sum_i \sum_j c_{ij} \tag{5.4.1}$$

$subject\ to:$

$$s_{ij}^1 + s_{ij}^2 - g_{ij} \leq c_{ij} + 1 \quad \forall (i,j) \in \textbf{SEs} \tag{5.4.2}$$

Table 5.1: Notation used in ILP Formulation

m_i^k	k^{th} bit of mask index of i^{th} contact/via
g_{ij}	Flag indicating if i^{th} and j^{th} contacts/vias are grouped
s_{ij}^k	Similarity variable indicating if k^{th} bits in masks of i^{th} and j^{th} contacts/vias are identical
c_{ij}	Flag indicating if i^{th} and j^{th} contacts/vias are in conflict
SEs	set of spacing edges in GG/SG
GEs	set of grouping edges in GG/SG
$conn(i,j)$	Flag indicating if i^{th} and j^{th} contacts belong to same connected component in GG
$fg(i,j)$	Flag indicating if grouping of i^{th} and j^{th} contacts is forbidden because they are not aligned or their inter-distance is not DSA-compliant (region A or C in Figure 5.4)
$ord(i,j,k)$	Flag indicating if i^{th} , j^{th} and k^{th} contacts are collinear and ordered, i.e. j^{th} contact lies between i^{th} and k^{th} contacts
$overlap(i,j,m,n)$	Flag indicating if group containing i^{th} and j^{th} contacts overlaps with group containing m^{th} and n^{th} contacts

$$s_{ij}^1 \geq 1 - m_i^1 - m_j^1 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3a}$$

$$s_{ij}^1 \leq 1 - m_i^1 + m_j^1 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3b}$$

$$s_{ij}^1 \leq 1 + m_i^1 - m_j^1 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3c}$$

$$s_{ij}^1 \geq -1 + m_i^1 + m_j^1 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3d}$$

$$s_{ij}^2 \geq 1 - m_i^2 - m_j^2 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3e}$$

$$s_{ij}^2 \leq 1 - m_i^2 + m_j^2 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3f}$$

$$s_{ij}^2 \leq 1 + m_i^2 - m_j^2 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3g}$$

$$s_{ij}^2 \geq -1 + m_i^2 + m_j^2 \qquad \forall\, (i,j) \in \mathbf{SEs} \tag{5.4.3h}$$

$$s^1_{ij} \geq g_{ij} \qquad \forall (i,j) \in \textbf{GEs} \tag{5.4.4a}$$

$$s^2_{ij} \geq g_{ij} \qquad \forall (i,j) \in \textbf{GEs} \tag{5.4.4b}$$

$$g_{ij} = 0 \quad \forall fg(i,j) = 1 \tag{5.4.5}$$

$$\begin{gathered} g_{ia} + g_{ja} \leq 1 + g_{ij} \\ \forall conn(i,j) = 1, conn(i,a) = 1, conn(j,a) = 1 \end{gathered} \tag{5.4.6}$$

$$\sum_{j, conn(i,j)=1, i \neq j} g_{ij} \leq max\_g - 1 \quad \forall i \tag{5.4.7}$$

$$g_{ij} \leq g_{ia}, g_{ij} \leq g_{ja} \quad \forall ord(i,a,j) = 1 \tag{5.4.8}$$

$$g_{ij} + g_{mn} \leq 1 \quad \forall \{i,j,m,n | overlap(i,j,m,n) = 1\} \tag{5.4.9}$$

The objective function in Equation (5.4.1) aims at minimizing the number of conflicts.

Each of the constraints in Equation (5.4.2) is used to set the conflict variable between two contacts/vias having a spacing graph edge, if they are assigned to the same mask and are not DSA-grouped. Constraints in Equations (5.4.3a-5.4.3h) are linear representation of the XNOR boolean relationship between two mask bits (e.g. s^1_{ij}=m^1_i XNOR m^1_j) to set the similarity variable if the corresponding mask bits are identical. For example, for a pair of contacts i and j, if they are both assigned to mask 3, then $m^1_i = m^1_j = 1$ and $m^2_i = m^2_j = 1$ and if they are not grouped, then $g_{ij} = 0$. Accordingly, the similarity variables s^1_{ij} and s^2_{ij} are set to 1 due to Equations (5.4.3a-5.4.3h). As a result, Equation (5.4.2) sets the related conflict variable c_{ij} to 1, adding 1 to the cost function in Equation (5.4.1).

The constraints in Equations (5.4.4a) and (5.4.4b) ensure that any grouping variable between two contacts/vias can only be asserted if the two contacts/vias are assigned to the same mask. Constraints in Equation (5.4.5) disallow grouping of pairs of contacts/vias that do not satisfy DSA constraints. In addition constraints in Equation (5.4.6) impose the semantics of transitive grouping, i.e. if contacts x and a are grouped and contacts y and a are also grouped, then contacts x and y are grouped as a result. The constraints in Equation (5.4.7) enforce the maximum

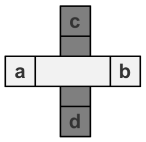


Figure 5.7: Example of overlapping groups, (a,b) and (c,d); $overlap(a,b,c,d)$=1. If the two groups are assigned to two different masks, the selection of both groups can be forbidden or allowed, according to the requirements of the process.

group size, since smaller group sizes have been found to lead to more robust assembly [BYB11, CBB10, MTF14] due to better lateral confinement.

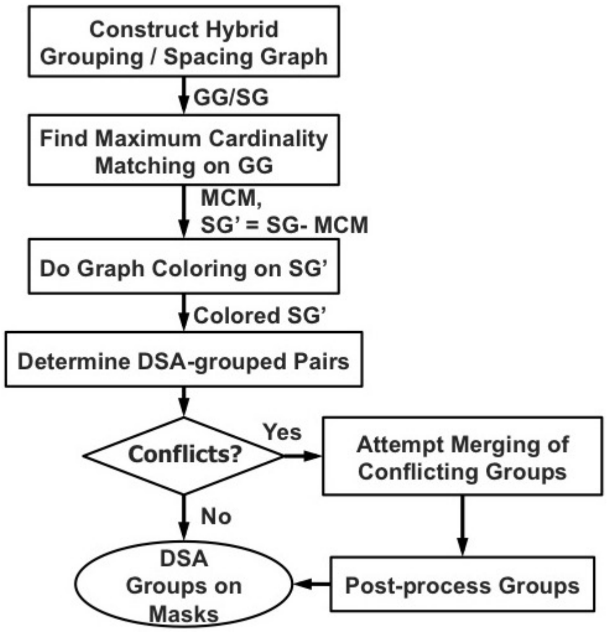
The constraints in Equation (5.4.8) make sure that DSA groups are continuous, and not interleaving, meaning that if two contact/via holes can only be grouped if the contact/via lying between them is also in the same group. Finally the constraints in Equation (5.4.9) ensure that the groups which overlap in space are mutually exclusive, even if assigned to different masks. Figure 5.7 shows an example of overlapping groups (a,b) and (c,d), thus $overlap(a,b,c,d) = 1$. If overlap of groups is disallowed by the process, then the selection of these two groups simultaneously will not happen even if both groups are assigned to different masks, due to Equation (5.4.9). Note that even if overlap between groups on different masks is allowed by the process, these two groups will never be assigned to the same mask because the distance between the two groups is less than $litho\_pitch$. The two sets of constraints of Equations (5.4.8) and (5.4.9) are optional; they depend on whether the process allows the overlap between templates on different masks. [3]

In case of Triple Patterning, one more constraint is required per polygon to prohibit using the unused mask combination [YYZ11], as shown in Equation (5.4.10).

$$m_i^1 + m_i^2 \leq 1 \quad \forall i \tag{5.4.10}$$

Note that for ease of understanding, the presented formulation hides some details which have been implemented to save memory. For example, the grouping variables are only created for pairs of contacts/vias which belong to the same connected component.

[3]In all our experiments, the overlap between templates on different masks is forbidden.


Construct Hybrid Grouping / Spacing Graph
GG/SG
Find Maximum Cardinality Matching on GG
MCM, SG' = SG- MCM
Do Graph Coloring on SG'
Colored SG'
Determine DSA-grouped Pairs
Conflicts?
Yes
Attempt Merging of Conflicting Groups
No
Post-process Groups
DSA Groups on Masks
Figure 5.8: The proposed DSA Grouping - MP Decomposition flow

5.4.2 Proposed Heuristics

Since the optimal ILP does not scale to dense full-chip designs, we present heuristics to solve the decomposition and grouping problem efficiently. The objective is to try to resolve as many conflicts as possible by grouping or assignment to different masks. To achieve this objective, we use heuristics to maximize the chance of grouping in the whole contact layer, which is in return expected to maximize the possibility of being able to fix conflicts by DSA grouping. In other words, our objective is to find the biggest number of non-contradicting groupable pairs of contacts. These will be the candidate grouping options; a subset of which will be chosen by coloring. We propose the following two heuristics:

5.4.2.1 Maximum Cardinality Matching Heuristic (MCM_H)

On the grouping graph, the problem of finding the biggest number of non-contradicting groupable pairs of contacts translates to finding the maximum number of grouping edges with no common nodes (i.e. contacts), since every grouping edge represents a grouping opportunity for the involved pair of contacts. This formulation is exactly the **Maximal Cardinality Matching** (MCM) problem, which finds the maximum number of disjoint edges and which can be solved in polynomial time using Edmond's algorithm [Edm65]. We used an $O(mn alpha(m,n))$ implementation of the algorithm, where m is the number of edges, n is the number of vertices and $alpha(m, n)$ is the inverse Ackerman function and is upper bounded by 4.

The flow of the proposed algorithm is shown in Figure 5.8. At the beginning, the hybrid grouping/spacing graph structure described in section 5.3 is constructed. Then the maximum cardinality matching of the grouping graph is found. If the technology does not allow overlap between the templates on different masks, the MCM result is processed as follows: if the MCM result includes two pairs of matched vertices such that their resulting groups will overlap, one of the two pairs is arbitrarily chosen and removed from the MCM result. Then, the spacing edges between the matched vertices (i.e. the spacing edges that are identical to the grouping edges in the MCM result) are removed from the spacing graph [4], and we have a modified spacing graph SG'. The idea is to drop the spacing edges between polygons which can be grouped. When the Multiple Patterning decomposer is then run on SG', it need not assign these groupable contacts to different masks because they can be printed as one DSA group. Then, the matched pairs of contacts are processed; if they got assigned to the same mask, then a DSA group is created for them on their mask. If they got assigned to different masks, then each is left as a DSA group of a single contact on the mask it got assigned to.

Group Merging and Post-processing

The algorithm up to this point can only produce groups of singletons (only one contact/via in a group) and doublets at the largest. We attempt to resolve the remaining coloring conflicts by merging the conflicting groups if they satisfy the DSA constraints, thus tentatively creating groups larger than two. The merging is attempted as follows: a GG (defined in Section 5.3) is constructed for the contacts/vias assigned to each mask separately. Connected components in this GG are then determined. In each connected component, if all the contacts/vias are collinear and the distance between the centers of all the neighboring vias are within the allowed DSA pitch range $[min\_dsa\_pitch, max\_dsa\_pitch]$), then one group is created for all these vias; otherwise the groups determined previously are unchanged and the merging does not happen for this connected component.

[4]Every grouping edge is also a spacing edge, because we assume max_dsa_pitch is smaller than $litho\_pitch$.

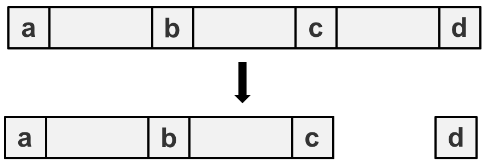


Figure 5.9: Example of the group splitting step in MCM_H, assuming max_g=3. A group of four gets split into two neighboring groups of size three and one. .

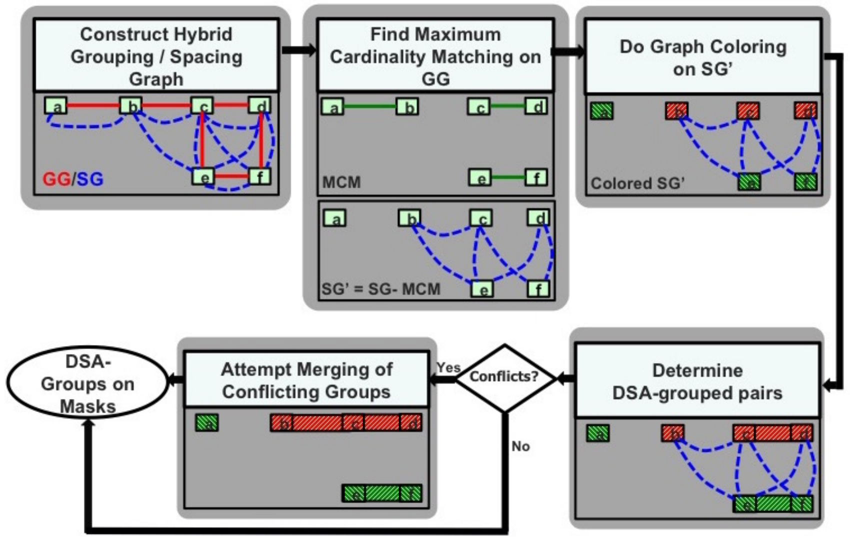

Construct Hybrid Grouping / Spacing Graph
GG/SG
Find Maximum Cardinality Matching on GG
MCM
SG' = SG- MCM
Do Graph Coloring on SG'
Colored SG'
Determine DSA-grouped pairs
Conflicts?
Yes
No
Attempt Merging of Conflicting Groups
DSA-Groups on Masks

Figure 5.10: Example of the proposed heuristic algorithm MCM_H. The post-processing step is not needed in this example and is not shown. .

The merging step can result in groups exceeding the maximum allowed number of contacts/vias per group (max_g). Thus, a post-processing step is needed in which each such “large” group whose size exceeds max_g is split into two or more groups as follows: the contacts/vias in a “large” group are sorted according to their left edge coordinate (if it is a horizontal group), or bottom edge coordinate (if it is a vertical group). New groups are created for the sorted contacts resulting in groups of size max_g and possibly one group of size smaller than max_g. Note that a conflict will exist between every two neighboring groups resulting from the split. An example of group splitting is shown in Figure 5.9, where a group of four gets split into a triplet and a singleton because $max\_g = 3$.

Example

Figure 5.10 shows an example of the flow on a layout snippet. First, the hybrid grouping/spacing graph (GG/SG) is constructed where the red edges are the grouping edges and the blue

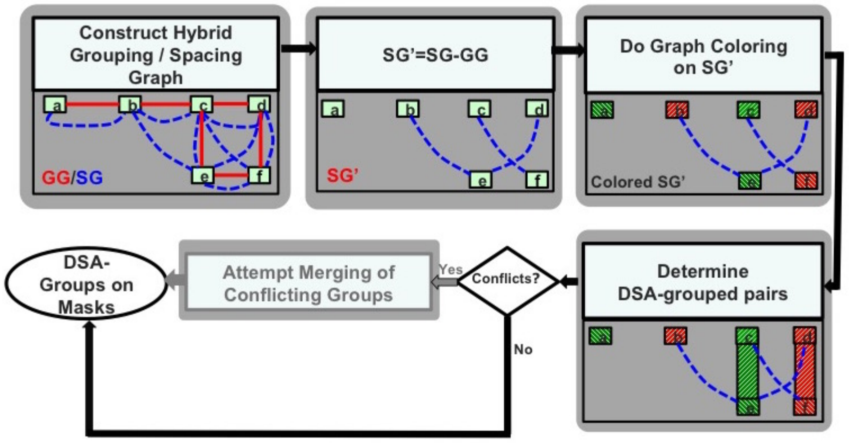

Construct Hybrid Grouping / Spacing Graph
GG/SG
a
b
c
d
e
f
SG'=SG-GG
SG'
Do Graph Coloring on SG'
Colored SG'
Determine DSA-grouped pairs
Conflicts?
Yes
No
Attempt Merging of Conflicting Groups
DSA-Groups on Masks

Figure 5.11: Example of the trivial heuristic algorithm Trivial_H. The post-processing step is not needed in this particular example and is not shown. .

edges are the spacing edges. Then, the MCM solution (not unique in this example) is computed, where it is found to consist of the six edges: a - b, b - c, c - d, e - f, c - e and d - f. After removing the spacing edges corresponding to the MCM solution, the modified spacing graph SG' is obtained. The graph is then colored using two colors (for Double Patterning), in this case assigning vias a, e and f to one of the masks and assigning vias b, c and d to the other mask. Since the matched contacts c and d were assigned to the same mask, they form a DSA group together. Also vias e and f form a DSA group. One conflict remains between vias b and c, and thus the singleton group of b is merged with the group of c and d. The final grouping and decomposition result is: a singleton group containing via a, a group of e and f on a mask; and a group of b, c and d on the other mask. None of the resulting groups contains more vias than the maximum allowed (max_g), which was assumed to be four in this example, so the post-processing step is not needed in this example and is not shown in Figure 5.10.

5.4.2.2 A Trivial Heuristic (Trivial_H)

In some cases, a much simpler heuristic algorithm can be used. This is to drop all spacing edges that coincide with grouping edges. The rest of the flow of Figure 5.8 remains the same except that MCM is not computed, and the modified spacing graph SG' is created as $SG - GG$; i.e. all the spacing edges coinciding with grouping edges are removed.

Figure 5.11 shows an example of the flow for the trivial heuristic on a layout snippet. As shown in the figure, all spacing edges that coincide with grouping edges are removed from the spacing graph, in order to create the modified spacing graph. The rest of the flow is the same as the one shown in the example of Figure 5.10.

If the technology disallows overlap between the templates on different masks (Figure 5.7), one of every two grouping edges that correspond to overlapping groups is arbitrarily chosen and deleted (the grouping edge is deleted but the corresponding spacing edge is not deleted).

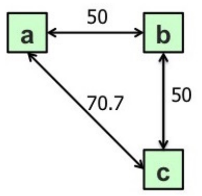
However, this heuristic requires a constraint to make sure that by dropping these spacing edges, there is no chance of ending up with illegal groups. [5] Let $litho\_dist$ be the minimum allowed distance between two vias on the same mask, which is equivalent to $litho\_pitch - hole\_dim$, and let max_dsa_dist be the maximum allowed distance between two vias in the same DSA group, which is equivalent to $max\_dsa\_pitch - hole\_dim$. The required constraint is then:

$$litho\_dist > \sqrt{2}\, max\_dsa\_dist \tag{5.4.11}$$

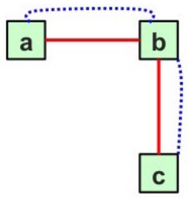
If the rule values violate this constraint (for example, max_dsa_dist=50 and $litho\_dist$=70), this trivial heuristic can not be used because it can lead to illegal groups. For example, this trivial heuristic may fail if run on the snippet shown in Figure 5.12, because all the spacing edges will be dropped and hence one possible solution is to assign the three vias to the same DSA group (and the same mask), and result in an L-shaped DSA group, which is illegal. However, if the constraint in Equation 5.4.11 had been satisfied by the values of max_dsa_dist and $litho\_dist$, there would have been a spacing edge between shapes a and c, preventing their being assigned to the same mask, and accordingly they would not be grouped.

In comparison to MCM_H, this heuristic performs the graph coloring on a simpler, less constrained graph, since all the spacing edges coinciding with grouping edges are removed, while MCM_H only removes the spacing edges that coincide with the MCM result (MCM result is subset of GG). Thus, Trivial_H can result in fewer conflicts. However, one disadvantage is that in Trivial_H, the graph coloring algorithm can unnecessarily result in big groups, potentially exceeding max_g vias, since there is no notion of maximum group size constraint, once the

[5] Remember that the allowed groups in this work are collinear manhattan groups only.



(a) Layout snippet with which the trivial heuristic may fail



(b) Corresponding hybrid graph

Figure 5.12: Example of a layout snippet with which the trivial heuristic may fail, and its corresponding hybrid graph, assuming max_dsa_pitch=64 and $litho\_pitch$=84

modified SG is used as input to the graph coloring algorithm. Each group with number of contacts/vias exceeding max_g gets split into smaller groups on the same mask, having spacing violations among them, in the post-processing step. This can lead to unnecessary conflicts in the result of the Trivial_H. However, this is not a problem in MCM_H, because the initial result (before the conflicting group-merging attempt) of MCM_H contains groups of two vias at most, thus if it results in a zero-conflict solution before group-merging , it is guaranteed to produce a final conflict-free solution.

5.5 Algorithms for the $Partial\_DSA$ Scheme

The $Partial\_DSA$ process described in Section 5.2 can allow more flexibility in the dimensions of the manufacturable contacts/vias. In such a process, some masks will be used to print the via/contact holes directly and will skip the self-assembly step. These masks can be used to print the holes whose dimensions are not compliant to DSA. We refer to these masks as *special masks* and to the vias/contacts not having the required DSA dimensions as *special holes*, i.e. contact/vias which are not squares or whose dimension is not equal to $hole\_dim$. The holes having the required DSA dimensions are referred to as *regular holes*. Note that the methods described in Section 5.4 can not be used as is, because the regular vias can also be patterned on the *special masks* without groups; otherwise, the special masks are not fully utilized and unnecessary violations can exist on the *non-special masks*. This is why, the problem can not be solved by assigning all special holes to *special masks* and using the same *ALL_DSA* methods for the regular holes on the *non-special masks* only.

In this section, we present the optimal ILP formulation and heuristic algorithms for the simultaneous DSA grouping and MP decomposition problem for the $Partial\_DSA$ scheme.

5.5.1 ILP Formulation

The same variables and notations shown in Table 5.1 are used. In addition, we use **SH** to refer to the set of *special holes*.

The cost function and constraints explained in Section 5.4.1 are used in this ILP formulation. However a few constraints are added in order to model the special case of the $Partial\_DSA$ process. Namely, special mask assignment constraints and special grouping constraints are added. The special mask assignment constraints ensure that each *special hole* can only be assigned to one of the *special masks*. The special grouping constraints prohibit the grouping of holes assigned to the *special masks* since these masks will not apply self-assembly. We assume that a process can have one or two *special masks*, which will be mask 0 or masks 0 and 1, respectively. We show the constraints here assuming four masks are used, for simplicity of the notation.

$$m_i^1 = 0 \quad \forall i \in \mathbf{SH} \tag{5.5.1a}$$

$$m_i^2 = 0 \quad \forall i \in \mathbf{SH} \tag{5.5.1b}$$

$$m_i^2 = 0 \quad \forall i \in \mathbf{SH} \tag{5.5.2}$$

$$g_{ij} \leq m_i^1 + m_i^2 \; \forall (i,j) \in GEs \tag{5.5.3}$$

$$g_{ij} \leq m_j^2 \quad \forall (i,j) \in GEs \tag{5.5.4}$$

The mathematical formulation of the special mask assignment constraints in case of one *special mask* only are shown in Equation (5.5.1). In this case, all *special holes* must be assigned to mask 0. In case of two *special masks*, each *special hole* can either be assigned to mask 0 (00_2) or mask 1 (01_2), thus the least significant bit (m_i^1) is not constrained. The constraint in this case is shown in Equation (5.5.2).

In case of one *special mask* only, the special grouping constraints are added in order to

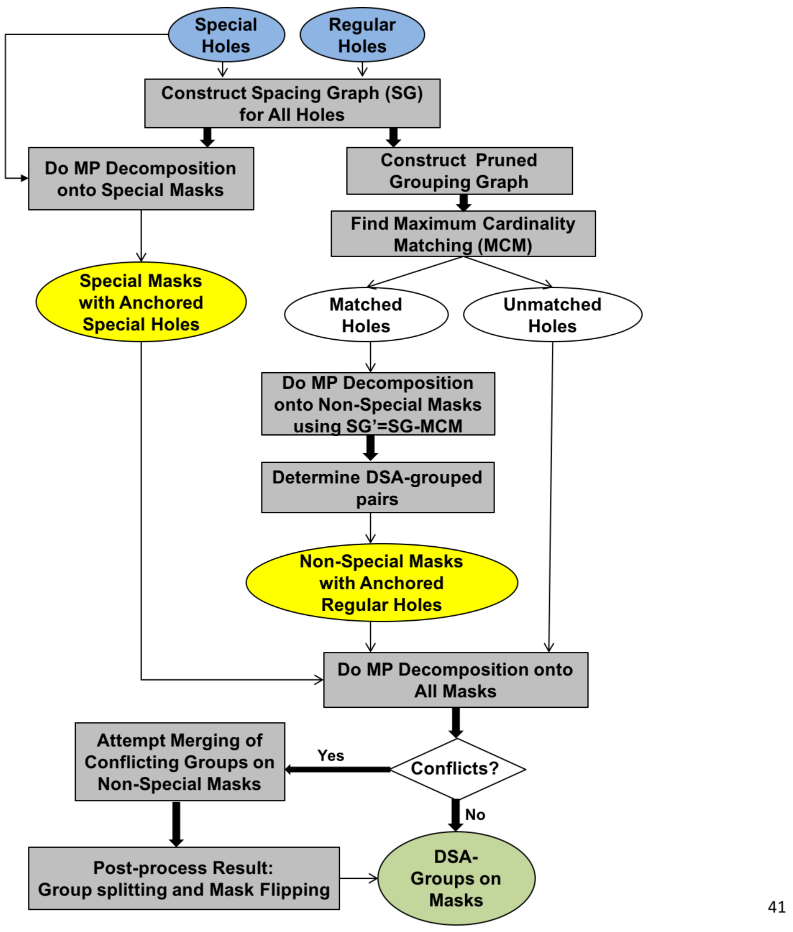

Special Holes
Regular Holes
Construct Spacing Graph (SG) for All Holes
Do MP Decomposition onto Special Masks
Construct Pruned Grouping Graph
Find Maximum Cardinality Matching (MCM)
Special Masks with Anchored Special Holes
Matched Holes
Unmatched Holes
Do MP Decomposition onto Non-Special Masks using SG'=SG-MCM
Determine DSA-grouped pairs
Non-Special Masks with Anchored Regular Holes
Do MP Decomposition onto All Masks
Attempt Merging of Conflicting Groups on Non-Special Masks
Yes
Conflicts?
No
Post-process Result: Group splitting and Mask Flipping
DSA-Groups on Masks
41

Figure 5.13: Flow of the Partial_MCM_H: MCM-based heuristic for the $Partial\_DSA$ scheme

allow grouping only for holes assigned to non-special masks (masks 2 (10_2) and 3 (11_2)), as shown in Equation (5.5.3). If a hole is assigned to mask 0 (00_2), which is a *special mask*, then all its grouping variables are forced to 0. In case of two *special masks*, the special grouping constraints are shown in Equation (5.5.4), to allow grouping to happen only if the involved holes are assigned to mask 2 (10_2) or mask 3 (11_2)), which are the non-special masks .

5.5.2 Proposed Heuristics

We propose two heuristic flows which can perform the DSA grouping and mask assignment for a via/contact holes layer, knowing that no self-assembly will take place on the *special masks*.

5.5.2.1 MCM Heuristic for Partial_DSA (Partial_MCM_H)

The first proposed flow is shown in Figure 5.13. First, we construct a spacing graph to represent the contact/via layer. The spacing graph for *Special holes* only is colored using the special masks only.

Next, we create a *pruned* grouping graph of the regular holes; a pruned grouping graph is a grouping graph which excludes some of the vias for which grouping is not essential to remove conflicts. Avoiding unnecessary grouping is a useful heuristic in $Partial\_DSA$ because grouped holes can only be assigned to one of the *non-special masks*; while non-grouped holes can be assigned to any of the masks, resulting in higher flexibility and accordingly less conflicts. We use the fact that if a graph is N-colorable, then adding a node with degree (number of edges) less than N makes the new graph N-colorable too [App04]. Thus, similar to the layout graph simplification in [YYZ11]; starting from the spacing graph of the complete layer, we perform recursive deletion of nodes representing *regular holes* with degree less than the number of masks N, but we do not delete nodes which are direct neighbors of *special holes*). This is because *special holes* have an additional constraint that they can only be assigned to *special masks*, and accordingly the coloring options for the neighbors of *special holes* can be limited, and grouping can be the only way to resolve their conflicts in some cases.

Grouping edges are added for the remaining holes only. The resulting graph is the *pruned* grouping graph. MCM is then computed on it. The edges in the MCM solution are then removed from the spacing graph of the matched holes (holes which have incident edges in the MCM). The resulting modified spacing graph (SG') of the matched holes is then provided as input to a Graph Coloring algorithm, in order to assign the matched holes to the *non-special masks* only. This is because the matched contacts are the candidates for being DSA-grouped, and special masks will not have self-assembly. After the coloring, a DSA group is created for each pair of matched holes that have been assigned to the same mask.

At this point, each *special hole* has been anchored to a *special mask* and each regular hole involved in the MCM result has been anchored to one of the *non-special masks* and potentially grouped. The remaining non-assigned holes are then decomposed onto **all** the masks, which will have some anchored holes from the earlier steps. Then we attempt to resolve any remaining conflicts on the *special masks* by merging adjacent DSA groups, in the same way that has been explained in Section 5.4.2.1.

Post-processing: Group Splitting and Mask Flipping

Finally the result is post-processed with two objectives. First, groups with more than max_g holes are split into two or more groups, in the same method explained in Section 5.4.2.1. Second, we attempt mask flipping to resolve conflicts. This is especially important because before this step, the matched holes were only allowed to be assigned to the *non-special masks* in order to be DSA-grouped if needed, but these holes can alternatively be assigned to the *special masks* without grouping. Thus, the mask of some contact/via holes can be changed in the post-processing step to reduce the number of conflicts.

Algorithm 1 Group Splitting and Mask Flipping for $Partial\_DSA$

```
Require: $break\_groups$ ( a boolean flag to allow breaking down groups to decrease conflicts)
 1: for $m = 1$ to number of masks do
 2:     for group $g$ in groups[m] do
 3:         if $size(g) > max\_g$ then
 4:             split $g$ into groups of size $max\_g$ or smaller (Section 5.4.2.1)
 5:         else
 6:             if $size(g) = 1$ then
 7:                 Attempt Mask Flipping ($g[1]$)
 8:             else if $break\_groups$ or size of $g = 1$ then
 9:                 Attempt Mask Flipping ($g[1]$)
10:                 Attempt Mask Flipping ($g[size(g)]$)
11:             end if
12:         end if
13:     end for
14: end for
```

The algorithms used in the group splitting and mask flipping algorithm are shown in Algorithms 1 and 2. Each group containing more than max_g holes is split into smaller groups as explained in Section 5.4.2.1. If a group is a singleton, then the mask flipping algorithm (Algorithm 2) is executed on it. If a group is not a singleton but has fewer holes than max_g and it is desired to break down groups for the sake of decreasing conflicts ($break\_groups = 1$ in Algorithm 2), then the group gets broken and the two holes at both ends of the group undergo mask flipping. We choose not to attempt mask flipping on holes lying between two other holes in the same group, since removing such a hole can render the group invalid because the distance between the centers of the two other holes can exceed the max_dsa_pitch, and it may also lead to overlap between groups on different masks which is prohibited in some processes.

Algorithm 2 Attempt Mask Flipping

```
Require: $contact$
 1: $min\_conf \leftarrow$ number of conflicts of $contact$ on $mask[contact]$
 2: for $m = 1$ to number of masks do
 3:     $n\_conf \leftarrow$ number of conflicts of $contact$ on mask $m$
 4:     if $n\_conf <$ number of conflicts of $contact$ on $mask[contact]$ then
 5:         $mask[contact] \leftarrow m$
 6:         $min\_conf \leftarrow n\_conf$
 7:     end if
 8: end for
 9: if $min\_conf > 0$ then
10:     for neighbor n in $neighbors[contact]$ in SG do
11:         if $n$ is not grouped then
12:             $curr\_conf \leftarrow$ number of conflicts of $n$ on $mask[contact]$
13:             if $curr\_conf = 0$ then
14:                 $n\_conf$ =number of conflicts of $contact$ on $mask[n]$
15:                 if $(n\_conf - 1) < min\_conf$ then
16:                     swap($mask[contact]$, $mask[n]$)
17:                     $min\_conf \leftarrow n\_conf$
18:                 end if
19:             end if
20:         end if
21:     end for
22: end if
```

Given, a particular contact, the mask flipping algorithm (Algorithm 2) first tries to re-assign the contact to the mask where it would have the fewest number of conflicts, without changing the mask assignment of the other contacts. Finding the number of conflicts of the contact on a mask includes the effect of grouping the contact with the neighboring groups on the new mask. If the number of conflicts of the contact is not zero, then we attempt a mask exchange between the contact and its neighbors in SG). Swapping masks does not take place unless it will result in zero conflicts for the neigboring contact, and the neighboring contact was not grouped with other contacts. Several iterations of the whole flow of Algorithm 1 can be executed, adding to runtime. In practice we found that three iterations are usually enough, first with the $break\_groups$ flag set to false in order to attempt resolving the violations with minimum change possible first. In the second iteration, we set $break\_groups$ to true and in the third, we set it to false.

5.5.2.2 A Trivial Heuristic for Partial_DSA (Partial_Trivial_H)

The trivial heuristic proposed in Section 5.4.2.2 can also be applied to the $Partial\_DSA$ scheme, by following the same flow shown in Figure 5.13 with few changes. Instead of finding the MCM on the *pruned* grouping graph and removing all the matched edges from the spacing graph, all the grouping edges in the *pruned* grouping graph are deleted from the spacing

Table 5.2: Number of vias in test cases used in Experiments for All_DSA

Test case	Number of Vias
AES	48123
CortexM0	35255
LEON3	93474
MIPS	34784

graph to obtain the modified spacing graph. All shapes having grouping edges are considered as matched holes, and the rest are considered as unmatched holes. The rest of the flow is the same. The same constraint explained in Section 5.4.2.2 must be satisfied by the rule values, in order to be able to apply this heuristic.

5.6 Experiments and Results

5.6.1 Experimental Setup

The ILP and the heuristics were implemented in C++. Open Access was used for layout manipulation; IBM CPLEX was used to solve the ILP and Boost Graph API was used for graph operations. In addition, Mentor Graphics Calibre tool was used for Multiple Patterning decomposition (Graph Coloring). All the experiments were run on a shared cluster [hof], using four cores and using up to 70GB of memory.

The test cases we use are: AES and MIPS from [ope], ARM Cortex M0 processor and LEON3 Sparc processor. These have been synthesized, placed and routed, as will be mentioned later. The experiments were performed on the Via1 layer of the layouts.

5.6.2 Results for the All_DSA Scheme

The used layouts for the experiments for the All_DSA scheme have been synthesized, placed and routed using commercial 45nm SOI libraries, then sized and scaled. After modification of the layouts, the via width is 14nm and the minimum spacing is 21nm, which is close to ITRS contact pitch for 2025. The number of vias in each test case is shown in Table 5.2.

The values that we use for our experiments are shown in Table 5.3, unless noted otherwise

Table 5.3: Parameter Values (in nm) used in Experiments

min_dsa_pitch	34
max_dsa_pitch	56
$litho\_pitch$	80
max_g	4
$hole\_dim$	14
L_0	34
N	2 (DP) and 3 (TP)

for particular experiments.

5.6.2.1 Comparison between Different Approaches

We compare the following approaches:

ILP: the ILP formulation explained in Section 5.4.1

MCM_H: the MCM heuristic explained in Section 5.4.2.1

Trivial_H: the trivial heuristic explained in Section 5.4.2.2

MP_GP: MP decomposition followed by DSA grouping on each separate mask[BTG15a]

GP_MP: DSA grouping followed by MP decomposition[BTG15a]

DP only: DP decomposition, without DSA, using Calibre Double Patterning tool

TP only: TP decomposition, without DSA, using Calibre Triple Patterning tool

MP_GP and GP_MP are the two simple sequential approaches discussed in [BTG15a]. These two approaches have been implemented by using the conventional Calibre Multiple Patterning and Directed Self Assembly tools, which are not aware of the process being hybrid DSA-MP.

In Tables 5.4 and 5.5, we show the number of spacing violations between the groups in each of the layouts for different approaches. The runtime shown for MCM_H includes the complete flow in Figure 5.8. MCM_H has a 17% increase in the total number of violations for DP and TP and has a speedup of 4x , in comparison to the ILP solution. The average group size for MCM_H is 1.026 vias and 1.014 vias for DP and TP respectively, which means that most of the vias ended up in singletons and thus are expected to result in relatively high yield [MTF14]. The average group size in the ILP solution is 1.016 vias and 1.007 vias for DP and TP respectively, even though the ILP cost function does not minimize the group size. In addition, MCM_H outperforms produces 56% fewer violations (in total) than GP_MP and MP_GP.

Table 5.4: Results on Layouts with DP, for the All_DSA scheme

	DSA + DP: ILP		DSA+DP: MCM_H		DSA+DP: Trivial_H		DSA+DP: GP_MP		DSA+DP: MP_GP		DP only
	Violations	Time (s)	Viol.	Time (s)	Violations	Time (s)	Violations	Time (s)	Violations	Time (s)	Viol.
AES	257	59	298	25	291	21	696	1	641	1	815
CortexM0	154	33	195	12	128	21	487	1	488	1	671
LEON3	268	219	303	12	280	11	680	1	642	1	779
MIPS	115	59	133	49	131	32	324	1	315	1	391
Comparison	0.96	4.35	1.12	1.15	1.00	1.00	2.63	0.05	2.51	0.05	3.20

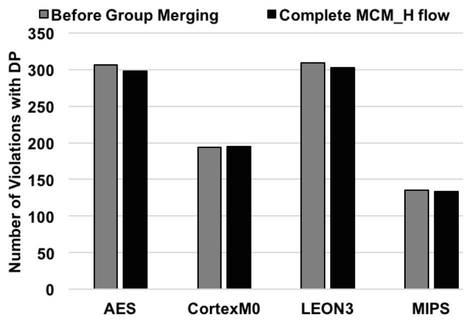
Table 5.5: Results on Layouts with TP, for the All_DSA scheme

	DSA + TP: ILP		DSA+TP: MCM_H		DSA+TP: Trivial_H		DSA+TP: GP_MP		DSA+TP: MP_GP		TP only
	Violations	Time (s)	Violations	Time (s)	Violations	Time (s)	Violations	Time (s)	Violations	Time (s)	Viol.
AES	2	87	2	24	2	25	29	1	6	1	29
CortexM0	1	27	2	8	1	19	28	1	7	1	28
LEON3	0	207	0	24	0	17	7	1	1	1	7
MIPS	0	0 62	0	30	1	31	13	1	5	1 1	3
Comparison	0.75	4.16	1.00	0.93	1.00	1.00	19.25	0.04	4.75	0.04	19.25

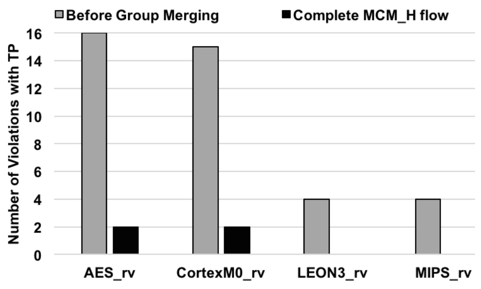
Trivial_H has a 5% increase in violations in comparison to the ILP, and has speed of 5x. The average group size for this heuristic is 1.0273 vias for DP and 1.015 vias for TP.

The trivial heuristic did outperform MCM_H in these test cases. As discussed in Section 5.4.2.2, this is because Trivial_H removes all the spacing edges that coincide with grouping edges before the graph coloring, whereas MCM_H only removes at most one spacing edge for every shape. Thus, the graph coloring algorithm needs to solve a less constrained problem (graph with fewer edges) in case of Trivial_H. Although Trivial_H is expected to form bigger groups, the average group size generated by both MCM_H and Trivial_H are found to be approximately equal. This is because with the used rule values, the spacing graph is non-planar, i.e. spacing edges existed between vias which are not direct neighbors, and since grouping edges are between direct neighbors only, these spacing edges were not removed. Accordingly, the coloring algorithm assigned these non-direct-neighboring vias to different masks, creating small groups as a result, instead of big groups on the same mask. However, as mentioned before the rules need to satisfy the constraint explained in Equation(5.4.11) in order to be able to use this trivial heuristic, and the set of rule values in this experiment satisfies this constraint. Note that in Tables 5.4, 5.5 and 5.8 only, we ran the heuristics MCM_H and Trivial_H using one thread, since the single-threaded execution for them was already fast enough, that the overhead of thread management did add to the runtime, but the ILP is run using four OpenMP threads.

These results are expected to be pessimistic since the layouts are not optimized for DSA. Since technology and design are usually co-optimized, we should expect more DSA-friendly layouts.



(a) Number of violations using MCM_H for DSA+DP



(b) Number of violations using MCM_H for DSA+TP

Figure 5.14: Number of violations using MCM_H before group merging as well as at the end of the complete MCM_H flow, for DSA+DP and DSA+TP

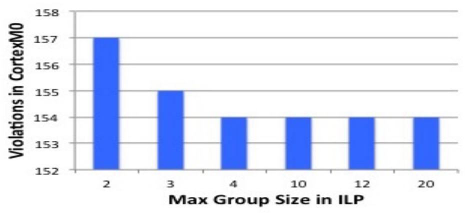


Figure 5.15: Number of Violations vs. max_g in ILP in CortexM0 testcase with DP, for the All_DSA scheme

In Figure 5.14, we show the number of violations at two points in the flow of MCM_H: before group merging and splitting, as well as the end of the whole flow shown in Figure 5.8. The group merging and splitting steps does help reduce the number of violations significantly for DSA+TP case.

5.6.2.2 Change in the Number of Violations with the Maximum Group Size

In Figure 5.15, we show how the number of violations from ILP changes as the maximum group size changes, on CORTEXM0 in DP. The size of the formed groups did not exceed four. Moreover, by restricting the maximum group size to two and three contacts per template, a 1.9% and a 1.3% increase in the number of violations are acquired respectively, which is a small penalty, given that the assembly process is more robust for small groups.

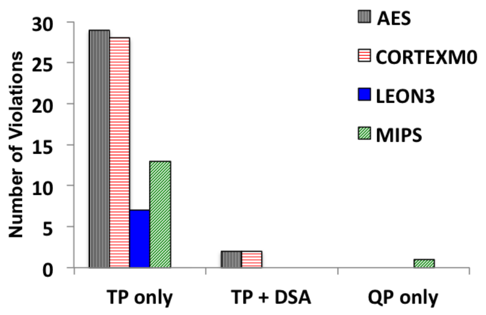


Figure 5.16: Number of Violations with TP only, TP + DSA using MCM_H, QP only, for the All_DSA scheme

5.6.2.3 Possibility of Replacing a Mask, by using DSA

Another experiment was performed in order to assess the effectiveness of the assumed hybrid DSA-MP process. In Figure 5.16, the number of violations for each of the complete layouts is shown as a result of doing TP only, TP with DSA using MCM_H and finally QP only. Without the use of DSA, it is likely that QP is needed, leading to a higher cost process. However with DSA, at most two violations existed in each test case and these violations are likely to be eliminated when DSA-friendly design rules and layouts are available.

5.6.3 Results for the $Partial\_DSA$ Scheme

In this section we show results for the second DSA-MP integration scheme which is $Partial\_DSA$. To evaluate the algorithms for $Partial\_DSA$ we synthesized, placed and routed the same designs with a 28nm library which has rectangular vias, in addition to square vias. The resulting layouts were sized and scaled. After modification of the layouts, the via width is 14nm and the minimum spacing is 20nm. Moreover, to be able to benchmark the heuristic against the optimal ILP, a snippet was used from each layout because otherwise the ILP either did run out of memory or did not finish within 12 hours using four threads.

The number of vias in each snippet and complete layout is shown in Table 5.6. The test cases are named in this way because “rv” indicates that these test cases have rectangular vias, “s” indicates that a snippet of the layout is used and “srv” indicates that the layouts have sparser usage of rectangular vias than “rv”. The used rule values are shown in Table 5.7.

Table 5.6: Number of vias in test cases used in Experiments for $Partial\_DSA$

Test case	Num. of Vias	Num. of Rect. Vias	Num. of Square Vias
AES_rv_s	1551	239	1312
CortexM0_rv_s	1758	246	1512
LEON3_rv_s	1608	178	1430
MIPS_rv_s	1691	204	1487
AES_rv	124451	16585	107866
CortexM0_rv	107481	14295	93186
LEON3_rv	374993	47794	327199
MIPS_rv	129659	17749	111910
AES_srv_s	7835	11	7824
CortexM0_srv_s	2630	105	2525
LEON3_rv_s	2678	126	2552
MIPS_srv_s	8296	384	7912
AES_srv	116974	2457	114517
CortexM0_srv	98792	3053	95739
LEON3_srv	340156	21943	318213
MIPS_srv	119238	5953	113285

Table 5.7: Parameter Values (in nm) used in Experiments for $Partial\_DSA$

min_dsa_pitch	33
max_dsa_pitch	56
$litho\_pitch$	80
max_g	4
$hole\_dim$	14
L_0	34
N	4 (QP)

5.6.3.1 Comparison between Different Approaches

We use Partial_MCM_H to refer to the flow described in Section 5.5.2.1 and Partial_Trivial_H to refer to the trivial heuristic described in Section 5.5.2.2. Both are compared to the optimal ILP in Table 5.8 with respect to the number of violations and runtime. We assume a process with Quadruple Patterning where only two of the four masks apply self-assembly (i.e. two masks are *special masks*). The number of violations is counted between the DSA-groups on the *non-special masks* and between the via holes on the *special masks*. Partial_MCM_DSA has an 18% increase in the number of violations in comparison to the ILP solution and has a speedup of 23x, and an average group size of 1.33 vias on the *non-special masks* . Partial_Trivial_H also has 14% increase in the number of violations and has a speedup of 29x speedup and an average group size of 1.69 vias on the *non-special masks*. The average group size on the *non-special masks* for ILP is 1.21 vias on the *non-special masks*. In the Partial_DSA scheme, the trivial heuristic did outperform MCM_H, and did not have larger group size, due to the same reasons explained in Section 5.6.2.1.

We also show results of the sequential approaches Partial_GP_MP and Partial_MP_GP in Table 5.8. In Partial_GP_MP, the *regular holes* are grouped using Calibre DSA tools. The *special holes* are multi-patterned onto the *special masks* only and the grouped *regular holes* are multi-patterned onto the *non-special masks* and anchored. Finally the *regular holes* which were not grouped are multi-patterned onto all the masks. Whereas in Partial_MP_GP, the *special holes* are multi-patterned on the *special masks* only, and the *regular holes* are multi-patterned onto all the masks. The *regular holes* which get assigned to each of the *special masks* undergo DSA grouping. The number of violations using QP only without DSA (done using Calibre QP tool) is also shown in Table 5.8.

Table 5.8: Results on Layout Clips with QP, for the $Partial\_DSA$ scheme. p_s is percentage of contacts assigned to special masks.

	Partial_DSA:ILP			Partial_MCM_H			Partial_Trivial_H			Partial_GP_MP		Partial_MP_GP		QP only
	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	Viol.	Time(s)	Viol.
AES_rv_s	22	78	51	26	15	47	25	7	46	29	1	31	1	1
CortexM0_rv_s	30	54	51	37	11	50	35	8	52	42	1	44	1	2
LEON3_rv_s	8	378	47	8	9	40	8	10	39	19	1	9	1	0
MIPS_rv_s	5	416	46	6	5	39	6	7	50	18	1	21	1	0
Comparison	0.88	28.94	1.04	1.04	1.25	0.94	1.00	1.00	1.00	1.46	0.13	1.42	0.13	0.04

The two heuristics Partial_MCM_H and Partial_Trivial_H as well as the sequential approached Partial_GP_MP and Partial_MP_GP have been run on four complete test cases. The results are shown in Table 5.9, along with results of QP decomposition, without DSA, using Calibre QP tool. The average group size on the *non-special masks* for $Partial\_MCM\_H$ on the complete test cases is 1.39 vias, while that of $Partial\_Trivial\_H$ is 1.38 vias.

The big number of violations is attributed to two closely-related reasons. First, these layouts are not DSA-aware. Second, there is a lot of rectangular vias in the layouts, and there are cliques of four rectangular vias in the layouts requiring four masks, as opposed to two *special masks* only in these experiments. This is why QP without DSA is more appropriate for these layouts.

As shown in Table 5.9, Partial_Trivial_H sometimes has longer runtime than Partial_MCM_H in relatively dense test cases like LEON3 where the majority of the vias lie in the same connected component of the graph. This is because the process of checking for overlap between candidate groups takes more time as the number of candidate groups increases.The candidate groups in Partial_Trivial_H are all the grouping edges in the pruned grouping graph, while in Partial_MCM_H the candidate groups are the MCM result set only. [6]

Table 5.9: Results on Complete Layouts with QP, for the $Partial\_DSA$ scheme. p_s is percentage of contacts assigned to special masks.

	Partial_MCM_H			Partial_Trivial_H			Partial_GP_MP		Partial_MP_GP		QP only
	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	Viol.	Time(s)	Viol.
AES_rv	1827	141	46	1835	136	45	2174	4	2286	3	60
CortexM0_rv	1214	268	46	1216	127	45	1543	4	1624	3	22
LEON3_rv	2842	881	38	2830	1962	38	5677	15	6264	7	93
MIPS_rv	1362	143	44	1350	141	43	1952	4	2067	3	15
Comparison	1.00	0.61	1.02	1.00	1.00	1.00	1.57	0.01	1.69	0.01	0.03

Figure 5.17 shows the number of violations using Partial_MCM_H before group merging; after grouping merging and splitting but without mask flipping; and using the complete flow Partial_MCM_H. The group merging and mask flipping improve the quality of the heuristics by decreasing violations.

[6]The process of checking overlap can be made faster by using a spatial hashing technique similar to the method in [DSB10].

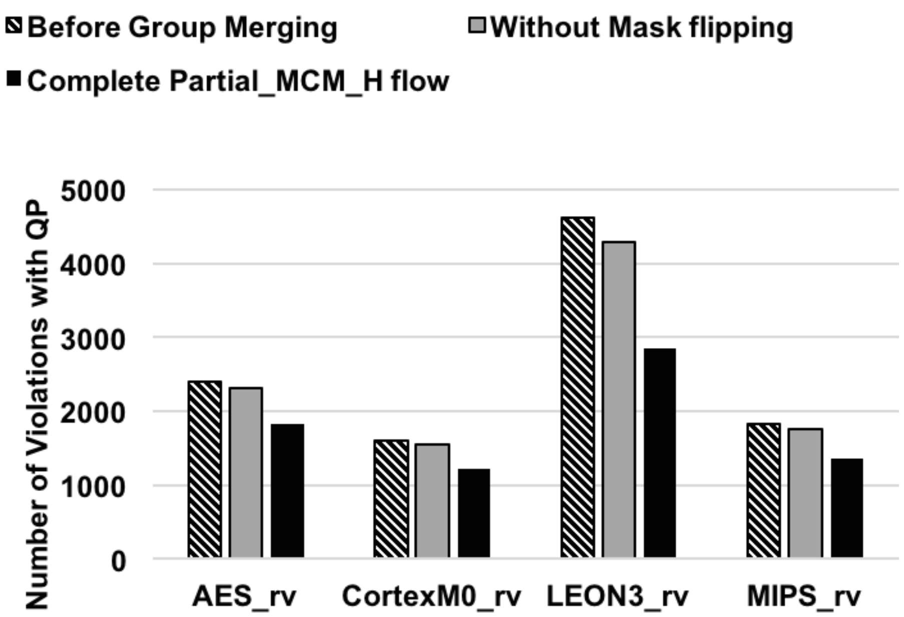


Figure 5.17: Number of violations using Partial_MCM_H before group merging, after group merging and splitting and finally using the complete flow.

5.6.3.2 Results with Sparse Usage of Rectangular Vias

The density multiplication feature of DSA can not apply to rectangular vias. Thus in order to print rectangular vias using a DSA-based technology, larger spacing design rules need to be enforced between rectangular vias. To empirically test this claim, we synthesized layouts after increasing the minimum space design rule between rectangular vias by 87% and did not instruct the router to exert high effort in using the rectangular vias. The resulting layouts have fewer rectangular vias than the ones used in Tables 5.9 and 5.8, as shown in Table 5.6.

The results are shown for the clips in Table 5.10 and for the complete layouts in Table 5.11. With the sparse usage of rectangular vias, Partial_DSA can indeed help increase the via density in comparison to QP; QP does produce a number of violations which is 8 times larger than that of Partial_Trivial_DSA on the complete layouts.

Table 5.10: Results on Layout Clips with Sparse Rectangular Vias with QP, for the $Partial\_DSA$ scheme. p_s is percentage of contacts assigned to special masks.

	ILP			Partial_MCM_H			Partial_Trivial_H			Partial_GP_MP		Partial_MP_GP		QP only
	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	Viol.	Time(s)	Viol.
AES_srv_s	0	8922	47	2	14	49	0	16	49	5	2	6	2	5
CortexM0_srv_s	0	2335	43	1	10	43	0	13	43	3	2	3	2	3
LEON3_rv_s	0	2607	41	2	14	43	0	14	42	3	1	4	2	1
MIPS_srv_s	1	4882	47	3	15	44	2	17	43	28	2	29	2	0
Comparison	0.50	312.43	1.00	4.00	0.88	0.52	1.00	1.00	1.00	19.50	0.12	21.00	0.13	4.50

Table 5.11: Results on Complete Layouts with Sparse Rectangular Vias with QP, for the $Partial\_DSA$ scheme. p_s is percentage of contacts assigned to special masks.

	Partial_MCM_H			Partial_Trivial_H			Partial_GP_MP		Partial_MP_GP		QP only
	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	p_s (%)	Viol.	Time(s)	Viol.	Time(s)	Viol.
AES_srv	13	90	45	1	90	45	164	4	195	3	33
CortexM0_srv	13	88	45	2	94	45	132	9	162	6	28
LEON3_srv	52	797	35	19	2581	34	1650	11	2320	9	119
MIPS_srv	5	105	42	4	156	41	449	4	543	3	21
Comparison	3.19	0.37	1.01	1.00	1.00	1.00	92.12	0.01	123.85	0.01	7.73

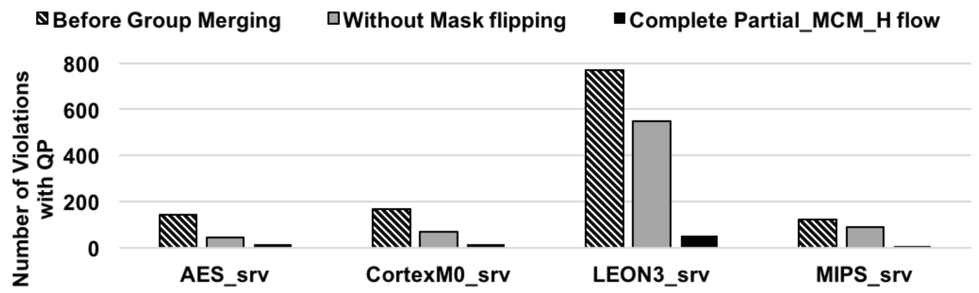


Figure 5.18: Number of violations using Partial_MCM_H before group merging; after group merging and splitting but without mask flipping; and using the complete flow, on the complete layouts with sparse rectangular vias.

Figure 5.18 shows the number of violations using Partial_MCM_H before group merging; after grouping merging and splitting but without mask flipping; and using the complete flow of Partial_MCM_H, on the layouts having sparse rectangular vias.

5.7 Conclusion

In this chapter, we presented algorithms to solve the simultaneous DSA grouping an MP Decomposition required for a hybrid DSA-MP process. Two schemes of such a hybrid process were studied. Optimal ILP formulation for the problem was presented for both schemes. Then we proposed efficient heuristic algorithms to solve the same problem, on the full-chip level, for each of the two schemes. The results of the heuristics are benchmarked against the ILP results. In our future work, we will generalize to other grouping structures that can be enabled by using EUVL to print the guiding templates.

CHAPTER 6

Hotspot-aware DSA Grouping and Mask Assignment

In Directed Self Assembly (DSA), poor printing of guiding templates can cause misassembly resulting in high defect probability. Therefore, hotspots should be avoided in the choice of the DSA groups. Accordingly, Directed Self-Assembly (DSA) technologies which use Multiple Patterning (MP) to print the guiding templates need to be aware of hotspots during the DSA grouping and MP Decomposition. In this chapter, we present a hotspot-aware heuristic for DSA grouping and MP decomposition. Results show that that the proposed heuristic eliminates 78% of the hotspots and conflicts that result from using a non-hotspot-aware grouping and decomposition algorithm. In comparison to the optimal solution using Integer Linear Programming, the proposed heuristic results in 24% more violations.

6.0.1 What is considered a hotspot?

A hotspot can be one of the following:

(i) Lithographic hotspot. This is a low-yield pattern due to photolithography, which is likely to cause a printing failure [DTP11, BG17].

(ii) Complex design rule. In advanced nodes, foundries introduced a lot of complex 2D and conditional rules. These rules can require pattern-based representation [BG17, DYR07]

(iii) Forbidden pattern due to using a restrictive patterning technology like Self-Aligned Multiple Patterning [BG17].

(iv) Some guiding templates result in high probability of defects in self-assembly. Thus the prohibition of low yield DSA groups or guiding templates can be achieved by representing these guiding templates as hotspots [BG17, SCS15].

6.0.2 Prior Work in DSA-MP

Several works [BTG15a, BTG15b, KY16, XLW16] have addressed the problem of simultaneous DSA Grouping and Mask assignment for hybrid DSA-MP technologies. Ou et al. [OYP16] solved the same problem while adding redundant vias, while Lin et al. [LC16b] added cut redistribution. DSA-aware routing has been addressed in [OYX17] for DSA+Double Patterning (DP) technology. In [SCS15], the authors perform DSA-friendly post-placement optimization, to avoid DSA groups of high defect probability, for contacts layer. However, they do not address low yield patterns made up of a neighborhood of DSA groups. The problem of Hotspot-aware DSA Grouping and Mask Assignment was considered in [BG17] for the purpose of Technology Path-finding, and an Integer Linear Program (ILP) was used for optimal evaluation, which is not scalable for full-chip layouts.

6.0.3 Contribution

In this work, we propose a scalable heuristic to achieve hotspot-aware DSA grouping and MP decomposition for hybrid DSA + MP technologies where MP is used to pattern the DSA guiding templates. Given, a contacts/via layer, it is required to cluster the vias into DSA groups, where each group will result in a guiding template, and assign the groups to the different masks, while avoiding **hotspots** as shown in Figure 6.1. We propose a heuristic which can be applied with different DSA Grouping and MP decomposition algorithms. Therefore, one key advantage of this heuristic is that it doesn't disrupt the flow of DSA Grouping and MP decomposition, but rather can be integrated nicely into the several threads of research coming up with fast methods for DSA grouping and MP decomposition. Up to the authors' knowledge, this is the first work which presents a scalable method for considering hotspots in the DSA Grouping and MP decomposition problem.

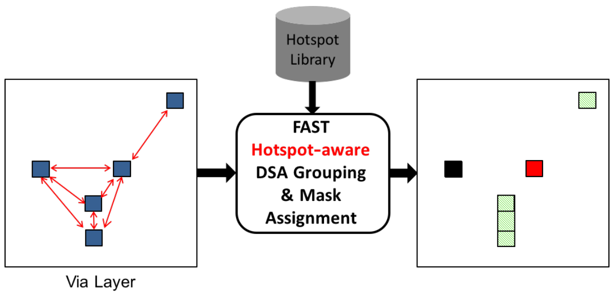


Figure 6.1: Objective of this work. In this example, DSA+Triple Patterning is assumed.

6.1 The Hotspots-avoidance Heuristic

6.1.1 Pattern and Graph Representation

We use the pattern representation proposed in [BG17], shown here for convenience in Figure 6.2. The segment representation encodes the DSA groups in the pattern while the node representation expresses the singletons (DSA groups containing one via each) in the pattern. For a hotspot pattern and a layout window, we use "**constituent vias**" to refer to all the vias, in layout window, which appear in the DSA groups or the singletons in the pattern. Vias which lie in the layout window but are not constituent vias are simply called "**non-constituent vias**". In the following, we use "**potential hotspot**" to refer to a layout window where a hotspot from the hotspots library can possibly exist, after grouping and coloring of the vias in the window. This means that the prerequisites of a potential hotspot is that for every via in the hotspot pattern, there is a via in the layout window in the corresponding location. For example in Figure 6.3, the shown layout window is a potential hotspot; because if vias *a* and *b* are grouped and assigned to the same mask as *d* and if via *c* is assigned to a different mask, the hotspot shown in the same figure will occur in this layout window. More about that will be described in Section 6.1.2.

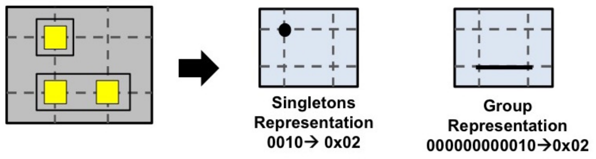


Figure 6.2: A hotspot and its corresponding representation [BG17]

We assume the layout is initially represented using a hybrid hyper-graph, where a conflict hyper-edge/edge is created between every two vias separated by a distance less than the minimum allowed distance on a single mask, and a grouping hyper-edge is created between every set of vias which can form a legal DSA group.

6.1.2 How to eliminate potential hotspots?

A potential hotspot will be realized only if:

(i) all the constituent vias are assigned to the same mask, and

(ii) none of the non-constituent vias is assigned to that mask.

For example, the hotspot in Figure 6.3 will not be realized in the shown layout window unless vias *a* and *b* are grouped and assigned to the same mask as *d*, and via *c* is assigned to a different mask. Thus, to avoid this hotspot in the shown layout window, a conflict edge can be added between vias *b* and *d* to force them onto different masks, or between vias *a* and *d*. Alternatively, the hotspot can be avoided if vias *c* and *d* are grouped (which of course means that they are assigned to the same mask).

Thus, eliminating a potential hotspot can be done by one of the following two ways:

(i) Adding a **conflict edge** between two of the constituent vias of the potential hotspot. The two chosen constituent vias must not be group-able.

(ii) Forcing a DSA group between at least one constituent via and one non-constituent via. We use "**affinity hyper-edge**" to refer to a forced DSA group.

However, adding a conflict edge or forcing a DSA group adds additional constraints to the grouping and coloring problem, which can increase the number of unresolved conflicts. Thus, it is desired to eliminate the hotspots by adding the minimum number of conflict edges and forced groups.

6.1.3 Mapping the problem to a Set Cover problem

A hotspot can be eliminated by adding one edge from a set of conflict edges, or forcing a group represented as an affinity hyper-edge between vias. In some cases, one conflict or affinity hyper-edge can be used to eliminate more than one potential hotspot. The objective is to choose the

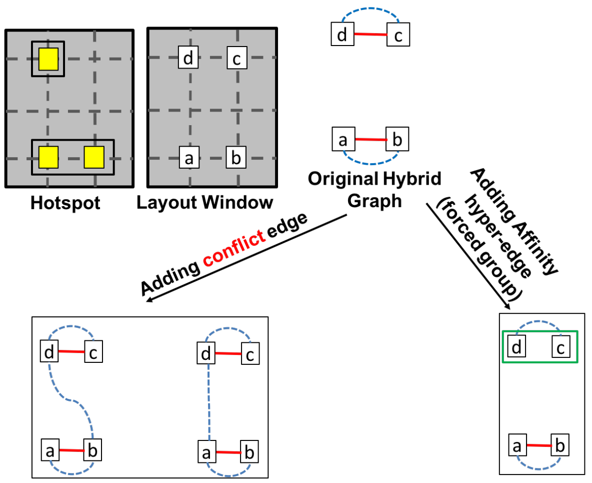


Figure 6.3: Elimination of the potential hotspots

minimum number of edges which cover the potential hotspots. This is the classic Set Covering problem where each set contains all the potential hotspots that can be eliminated by the addition of a conflict or an affinity hyper-edge. [Kar72].

6.1.4 Greedy Heuristic to solve the Set Covering problem

Set Covering is an NP-complete problem. We use the Greedy heuristic to solve the Set Covering problem [Joh74]. The algorithm is to iteratively choose the subset covering the largest number of elements from the universe set. Applying the greedy heuristic to the Hotspot avoidance problem results in the flow shown in Figure 6.4, where in each iteration the conflict or affinity hyper-edge which eliminates the largest number of potential hotspots is chosen. The graph is updated according to the chosen hotspot avoidance method.

6.1.5 Implementation

We develop an $O(m)$ implementation by using the bucket list data structure similar to the one used in [FM88], where m is the number of potential hotspots. For every potential hotspot, we identify the pairs of constituent vias that are not groupable and do not have a conflict edge between them. These are potential to-add conflict edges. Furthermore, the potential DSA groups including a constituent via and a non-constituent via are identified. These are potential forced groups.

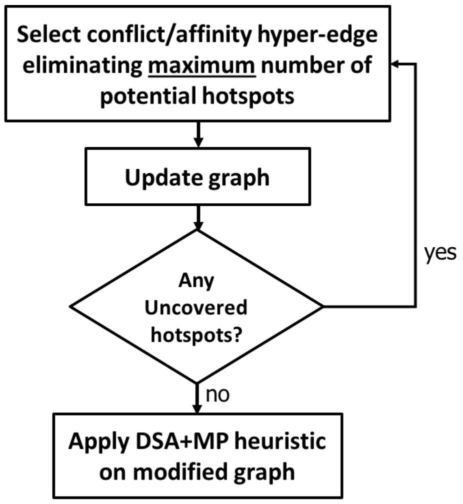

Select conflict/affinity hyper-edge eliminating maximum number of potential hotspots
Update graph
Any Uncovered hotspots?
yes
no
Apply DSA+MP heuristic on modified graph

Figure 6.4: Hotspot avoidance Flow using Greedy Set Cover heuristic

Each edge or forced group is attached to the list of edges of the same frequency, i.e. the number of potential hotspots that will be avoided by this edge. Frequencies are saved in a linked list in decreasing order. Then in each iteration in Figure 6.4, one edge from the highest frequency is picked in $O(1)$.

6.1.6 Which DSA Grouping and MP Decomposition algorithm can be used?

As shown in Figure 6.3, the proposed hotspot heuristic adds conflict edges as well as forced groups. Handling an added conflict edge is straightforward, since it is to be exactly considered as an original conflict edge (that exists because the distance between two vias is smaller than the minimum allowed distance on a mask).

Some algorithms can handle forced groups easily. In [KYY16], the forced groups are to be modeled by assigning a negative cost to the grouping edges representing them. Finally, the algorithm in [OCY17] can handle the forced groups very simply by setting the ILP variables corresponding to the selected groups.

6.2 Experiments and Results

6.2.1 Generating Hotspots

In order to generate hotspots, we randomly generated 4x2 patterns (four rows and two columns). The randomly generated patterns were then simulated using Calibre Litho-Friendly Design tool [Bus] to generate the Process Variation (PV) bands shown in Figure 6.5. We use the Process Manufacturability Index (PMI) metric proposed in [TB05] to gauge the sensitivity of each pattern to process variation. Patterns with PMI higher than 20% were considered as hotspots and there were 36 hotspots.

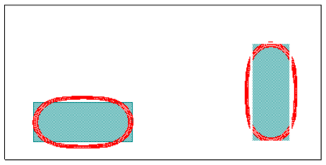


Figure 6.5: PV Band in red

6.2.2 Results

Optimal ILP formulation for Hotspot-aware Grouping and Coloring

In [BG17], an ILP formulation was proposed for hotspot-aware simultaneous DSA Grouping and MP decomposition. We apply the same formulation in order to find the optimal solution and benchmark the developed heuristics against the optimal solution.

Experimental Setup

We synthesized, placed and routed the following designs: AES [1] and MIPS from OpenCores, and ARM Cortex M0 using a projected 7nm library that was scaled down to 5nm. We apply the greedy set cover heuristic followed by DSA grouping and Mask Assignment. For grouping and coloring, we used the ILP in [BG17] -without providing a hotspot library- in order to

[1] In order to compare to the optimal solution using ILP, we had to use a clip of AES instead of the full layout because otherwise the ILP needed more than 24 hours

purely measure the quality of the hotspot-related heuristic independently from the quality of the grouping and coloring heuristics. To use the ILP of [BG17] for grouping and coloring only, we do not provide any hotspot library. We only used collinear DSA groups, assuming 193i is used to pattern the guiding templates [2], and used the parameter values shown in Table 6.1.

Table 6.1: Parameter values used for ILP in [BG17] for DSA grouping and Mask Assignment

Parameter	**Value**
L_0	30 nm
$max_dsa_$pitch	51 nm
max_g	2 or 3
$min\_pitch\_same\_mask$	75 nm
$min\_pitch\_diff\_mask$	10 nm
via_width	15nm

We compare our results to the ILP formulation for hotspot-aware DSA grouping and coloring in [BG17]. We also compare the results of the hotspot heuristic to the results of using an optimal DSA grouping and coloring algorithm which is unaware of hotspots, and for that we use the ILP of [BG17] **without** using any hotspots library as input to the ILP. The results are shown in Tables 6.2 and 6.3 for maximum group size=2 and maximum group size=3 respectively. The shown number of violations is the sum of number of unresolved conflicts as well as number of existing hotspots. The Greedy set cover heuristic produces a total of 24% increase in violations in comparison to the optimal solution. In comparison to an optimal grouping and coloring algorithm that is unaware of hotspots, our hotspot heuristic eliminated 78% of the total number of conflicts and hotspots in the output masks. The average runtime of the heuristic (excluding the DSA grouping and MP assignment time) is 45 seconds for the used test cases.

6.3 Conclusion

We proposed an algorithm for Hotspot-aware DSA Grouping and MP decomposition. The proposed heuristic was benchmarked against the optimal solution, resulting in 24% more violations. In comparison to an optimal DSA grouping and mask assignment algorithm that is

[2]The same hotspot heuristic can be used if the set of allowed DSA groups is different, and hence can be used if Extreme Ultraviolet Lithography (EUV) is used instead of 193i.

Table 6.2: Results of Greedy Set cover heuristic using maximum group size=2

	N Vias	DSA_Pathfind [BG17]	Non-hotspot-aware	Our heuristic	
		N Viols	N Viols	N Viols	Time(s)
aes_clip	35051	71	350	79	36
mips	35971	12	129	21	73
m0	44530	65	280	74	34
Ratio	–	1	5.13	1.18	–

Table 6.3: Results of Greedy Set cover heuristic using maximum group size=3

	N Vias	DSA_Pathfind [BG17]	Non-hotspot-aware	Our heuristic	
		N Viols	N Viols	N Viols	Time(s)
aes_clip	35051	50	292	63	36
mips	35971	12	130	20	35
m0	44530	39	237	52	57
Ratio	–	1	6.52	1.34	–

unaware of hotspots, the proposed heuristic eliminates 78% of the conflicts and hotspots.

Part III

Scaling Boosters

CHAPTER 7

Assessing Benefits of a Buried Interconnect Layer in Digital Designs

In sub-15nm technology nodes, local metal layers have witnessed extremely high congestion leading to pin-access-limited designs, and hence affecting the chip area and related performance. In this work we assess the benefits of adding a buried interconnect layer below the device layers for the purpose of reducing cell area, improving pin access and reducing chip area. Results show that buried interconnect, as an integration primitive, is very promising as a booster to density scaling. [1]

7.1 Introduction

With feature dimensions reaching the nanometer scale, local metal layers have become extremely valuable routing resources since they are heavily used for standard cell routing and pin access. In FinFET technologies [Aut12, Hoo12, LGN14], the introduction of the Local Interconnect (LI), which is used to connect fins or gates to make a multi-fin or multi-poly device [LW15], helped reduce congestion on these layers. However, it is unlikely that adding more LI layers will give significant benefits since the contact (V_0) holes that are connected to the device layers are also highly congested. Moreover, pin access has also become one of the biggest challenges to scaling density, since technology scaling and the design of cells with compact area has made it extremely difficult for the routers to access the pins [TLA10]. Metal design rules, have become limited by the lithography resolution. Thus, in a lot of cases, chip area is routing-limited, which reduces the potential benefit of technology scaling.

[1]The material in this chapter is based on the published work [ZBW17].

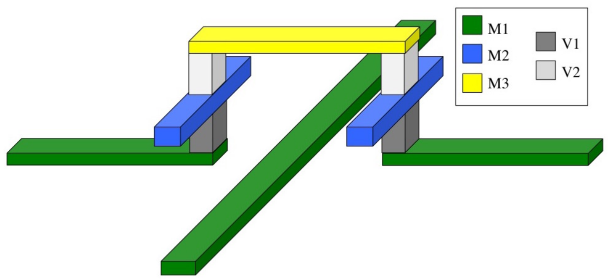


Figure 7.1: Via Blockage with the use of three metal layers. Notice the wasted area on M_2 due to the need for landing pads satisfying minimum area rule.

The use of more metal layers above the device layers for intra-cell routing adds to the severity of the congestion problem. This is because for every route on a higher metal layer, vias and landing pads have to be placed on the lower layer, leading to what is known as via blockage problem. Moreover, the landing pads must satisfy the minimum area rule, and thus routing resources on the lower layers are wasted. This problem is only getting worse as metal minimum area rules have not scaled as much as pitch rules (the latter being aided by multiple patterning). For example, Figure 7.1 shows three metal layers (M_1-M_3), along with the required vias. Because of the congestion on M_1 and M_2, the route was resumed on M_3. However, landing pads are still needed on M_2, consuming M_2 space. Earlier research [Sai95] has studied the problem of via blockage, and it has been postulated that via blockage limits the benefit of increasing the number of metal layers. Moreover, it has been shown in [CDZ00] that the via blockage factor can be as high as 50% on the first metal layer.

In order to solve these problems, we propose using a buried metal layer in the standard cells. This buried metal layer (M_{-1}) and its contact layer (V_{-1}), lie underneath the device layers. Figure 7.2(a) shows a cross-section of the traditional interconnect stack on top of the device layers, and Figure 7.2(b) shows the interconnect stack after adding the proposed buried layer under the device layers.

Having an interconnect layer under the transistors in the standard cells is expected to be much more beneficial than adding a new LI layer on top of the device layers or using the upper metal layers (M_2 and possibly higher) for intra-cell routing because of several reasons. First, as mentioned above, the V_0 layer, which is the main bridge from the device layers to the metal layers, has become congested as well. Second, using the higher metal layers will add to the via blockage problem, due to the need for vias and landing pads as explained earlier. Third, sharing

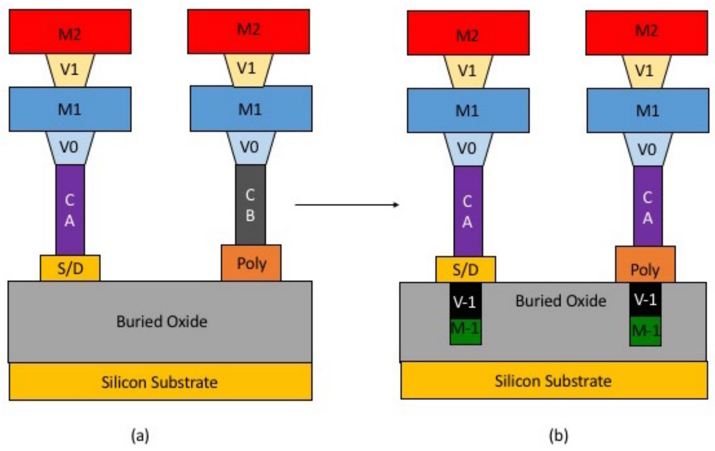


Figure 7.2: (a) Original stack of interconnect layers on top of device layers in SOI process (b) New stack with M_{-1} and V_{-1}

a layer (e.g. M_2) for intra-cell and inter-cell routing makes pin access even more difficult for the router by blocking some of the tracks that would have otherwise been available for pin access. Fourth, using M_1 intensely for intra-cell routing has urged designers to create short pins, which again complicates pin access [TLA10].

The idea of using a buried metal layer under silicon is not totally novel, but using it in logic by introducing it to the standard cell library to make transistor connections has not been studied before, to the best of our knowledge. A buried layer has been used in DRAM for buried word lines [FNA99]. For many years, extensive work has been done in direct wafer bonding [FNA99] to include a buried metal layer in the dielectrically isolated substrate [MGC88]. It has been proven that the buried metal layer can provide potential benefits to SOI substrates in MOS IC's [Las86, NDL94]. The electrical performance of this buried layer has also been experimentally studied to verify that the silicon substrates including the buried metal layer can exhibit good device behavior [GMR97]. Finally, a buried source/drain contact is proposed in [ZYY16] for FinFET devices. All these works have different contexts and purposes of using a buried layer and their proposed fabrication methods can not be applied when the buried layer is used to make local connections in standard cells.

In this chapter, we study the performance and density implications of a buried interconnect layer for random digital logic. The contributions of this work can be summarized as follows:

- This is the first work to propose and evaluate the buried interconnect layer concept for random digital logic. An algorithm is implemented to modify a standard cell library, in order to introduce the buried layer and reduce the area of the standard cells as a result.

- Pin-access benefits of the buried layer are evaluated. In addition, several benchmarks are synthesized, placed and routed using the standard cell library with the buried layer, in order to assess the chip area savings.

- TCAD and Spice simulations are used to evaluate the performance impact of the buried layer. Effect of the buried layer on the chip-level performance is also evaluated.

The rest of the chapter is organized as follows: Section 7.2 describes the introduction of the buried metal layer into a 7nm standard cell library, followed by the cell-area analysis. Section 7.3 presents the pin access improvement and the chip-area analysis. Section 7.4 describes the process flow that we propose to manufacture the buried interconnect. Performance evaluation is shown in Section 7.5. Finally, conclusions and future work are shown in Section 7.6.

7.2 Buried layer in Standard Cells

How Buried Interconnect contacts the Transistors. All the work in this chapter assumed SOI-based FinFET process[2]. The buried interconnect layer and the buried contact layer lie underneath the device layers. As shown in Figure 7.2 the buried V_{-1} connects the buried M_{-1} layer to the gate and source/drain by going through the SOI buried oxide.

The buried interconnect connects to source/drain region through a V_{-1} hole which goes through the buried oxide and contacts the source/drain region. A cartoon diagram of source/drain contacts through V_{-1} is shown in Figure 7.3. With FinFET devices, the source/drain region is usually formed by merging fins by epitaxial growth of SiGe [KBY09]. Therefore, the buried vias do not need to contact the thin fins directly (which would have been very hard to control due to the overlay error).

To contact the gates, the contact V_{-1} is placed between two adjacent fins as shown in Figure

[2]The cost comparison of SOI vs. Bulk is outside the scope of this study. This work presents the idea on SOI. However, our future work will address the buried layer concept in a bulk Si process.

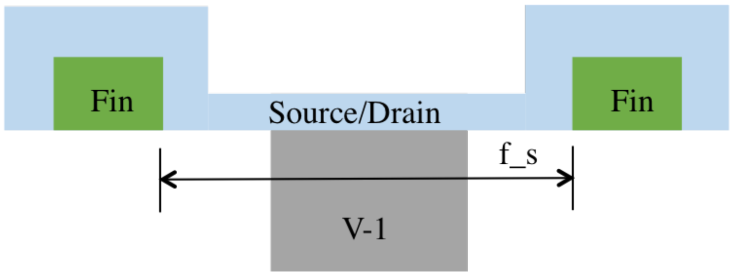


Figure 7.3: Contacting Source/drain contacts (V_{-1})

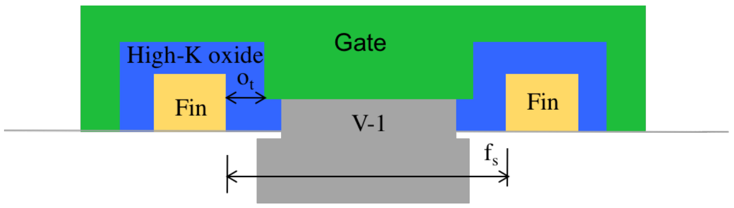


Figure 7.4: Contacting Gate contacts (V_{-1})

7.4 to reach the gate. This requires the contact width (c_w) to satisfy this constraint: $c_w < f_s - 2o_t$ where f_s is the fin to fin spacing and o_t is the thickness of the high-K oxide that is underneath the gate.

Experimental Setup. We introduced the buried layer to a projected 7nm FinFET standard cell library, from a leading IP provider. The cells have been modified in order to use a buried layer, as shown in the following sections. All M_{-1} segments are horizontal, keeping the layer unidirectional.

Layout Changes. The buried layer (M_{-1}) is used to completely replace the horizontal LI layer (*CB*), to make gate-to-gate connections. For example, Figure 7.5 shows the interconnect layers in a snippet of a hypothetical standard cell, where a gate-to-gate connection labeled x got transferred from *CB* to the buried layer M_{-1}. Only one LI layer (*CA*) is used in the final standard cell library; *CA* is still preserved in order to connect the fins, create the power straps and connect the transistors to M_1. Therefore, *CA* is used for both gate and source/drain contacts like the contact layer in pre-LI technologies [Jan10]. V_0 layer is used to connect M_1 to the *CA* layer, as shown in Figure 7.2.

In this 7nm standard library, all of the I/O pins are on M_1 layer, and the pins are accessed by dropping a via from M_2 to M_1. Thus, moving all the intra-cell M_2 routing segments to M_{-1} is a top priority, because it is expected to improve pin access, and accordingly can result in chip-area reduction.

An algorithm is implemented to automate the relocation of M_1 and M_2 routes to M_{-1}. Nets containing more than two endpoints are broken down into multiple connections; each connection is between two endpoints. The resulting connections are handled as follows:

Gate-to-Gate Connection or Source/Drain-to-Gate Connection: An available track on M_{-1} is picked for the route resulting in a horizontal M_{-1} segment. Figure 7.5 shows a gate-to-gate route labeled z, which used to span M_1 and M_2, but got relocated to M_{-1} (the input pin is kept on M_1, though).

Source/Drain-to-Source/Drain Connection: If both endpoints belong to P-FET transistors or both belong to N-FET transistors, then an available track on M_{-1} is used for this connection. However, if one endpoint is a P-FET and the other is an N-FET, a vertical segment is needed which cannot be done on on M_{-1}, and thus part of the connection is done on M_1. For example in Figure 7.5, net y connects the drain regions of one P-FET and two N-FET transistors and it used to exist on layer M_1, thus the P-FET to P-FET connection got relocated to M_{-1} but the P-FET to N-FET connection remained on layer M_1.

If relocating the route to M_{-1} results in design rule violations or no available track is found on M_{-1}, the route is kept on its original layer. In this library, all the routes on M_2 were successfully relocated to M_{-1}, thus M_2 is no longer used for intra-cell routing. In addition some of the M_1 routes were successfully moved to M_{-1}, reducing the M_1 congestion, as shown in Figure 7.5. However, some routes remain on M_1 because they are I/O pins, vertical connections (M_{-1} is horizontal layer), or because of the capacity of M_{-1}.

Cell- level area reduction. Due to the relocation of M_1 and M_2 to M_{-1} as described in the previous sections, the consumed routing resources on M_1, M_2 and V_1 have been greatly reduced. Therefore, standard cells which used to be routing-limited, meaning that their areas were determined by the need for more routing resources rather than by the transistors, are no longer routing-limited and their areas are reduced. This has been performed by removing the dummy polysilicon shapes (shapes on the polysilicon layer, that are not gates) that used to exist to provide more space for routing. The cell-level area reduction algorithm attempts to delete the area occupied by such dummy poly shapes, and compacts the cells by shifting the contents

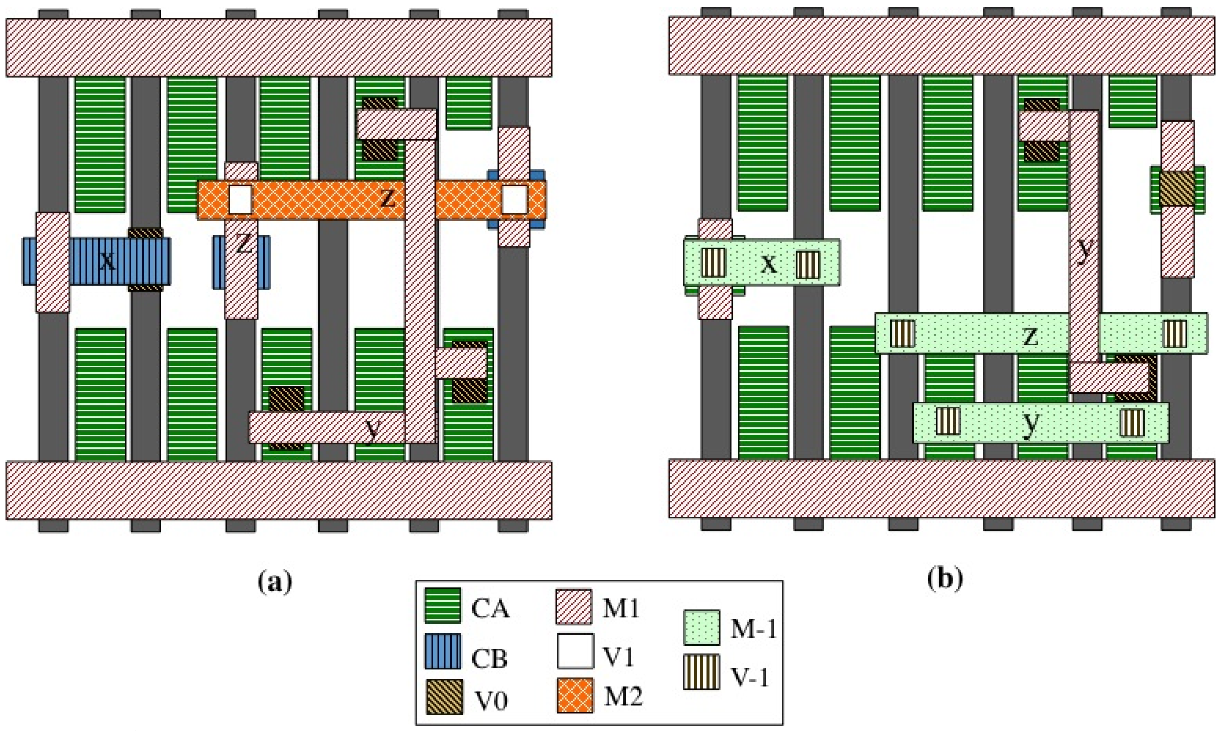


Figure 7.5: (a) Interconnect layers of a snippet of a hypothetical Standard Cell without buried layer (b) Same snippet with buried layer

of the cell. The routes that used to cross that area are then re-wired by being moved to other tracks, if they have design rule violations with the other routing segments. This is done as follows: M_1 routes with vertical steiner-tree trunks are changed by relocating the trunk to an available vertical track on M_1. Horizontal connections passing through the eliminated area are made shorter easily without introducing problems.

Since the cell-level area reduction only targets cells which are routing limited, only the complex cells in the library can benefit from it. Accordingly, other cells whose areas are defined by the transistors do not get area reduction by using the buried interconnect. Thus in our experiments, which were performed on a 59-cell projected 7nm library, the three most complex cells witnessed an area reduction between ˜6%-13%, as shown in Table 7.1. For the entire 7nm standard library, the average area reduction is around 0.48% over all cells since this is a very small library (59 cells) and most of the cells are simple ones that are not routing-limited.

The cell area reduction could not have been done on the original library, without the buried layer, even if M_3 is used for intra-cell routing. Routing the reduced-area flip flop cell with three metal layers above the device (M_1-M_3) and one local interconnect layer is impossible while

Table 7.1: Results of the Standard Cell-level Area Reduction

Cell	Original Area (u^2)	New Area (u^2)	Area Difference
LAT_X1	0.2916	0.2527	13.34%
MX2_X1	0.2138	0.1944	9.09%
SDFF_X1	0.62208	0.5832	6.15%

routing it with one metal layer above device (M_1), one local interconnect layer (CA) and one buried layer is feasible. This supports our claim that the buried layer can solve problems and achieve benefits that above-device routing layers can not.

7.3 Pin Access and Chip-level Benefits of the Buried Layer

In this section, we discuss two benefits gained from the addition of the new metal layer M_{-1}: pin-access improvement and chip-level area reduction

Pin Access improvement. To quantify pin access, we use the metric proposed in [XYG16], where a hit point is defined as the overlap of M_2 routing track and the I/O pin; and a valid hit point combination (VHC) is the set of hit points containing one hit point for each of the input and output pins, with no design rule violations. The total number of VHC is the used metric. The number of VHC in the original cells is constrained by intra-cell routes on M_2. Since all the M_2 routes were successfully replaced by M_{-1}, the different ways to access a specific pin from both directions increase significantly. The hit point calculation algorithm has been applied to recursively count the total number of VHC for the cell libraries before and after using the buried layer. Since all pin access is assumed to be done through M_2, only the cells that used to have M_2 routes can observe an improvement in pin access, when the buried layer is introduced. The results of the pin access improvement are summarized in Table 7.2. More M_2 used in the original standard cell results in greater pin access improvement that can be obtained due to M_{-1}. The average pin access improvement is 126.32% for the listed four cells only. [3]

Chip-Area Saving. In order to check if these cell-level improvements can lead to final chip-level benefits, chip-level experiments have been performed. Using the new cell library made

[3]The other cells in the library did not have any routes on M_2.Reducing congestion on M_1 does not result in a change in the used pin access metric. Thus among the whole library, the average pin access improvement is 8.6% only.

Table 7.2: Pin Access Improvement Results using Number of Valid Hit Point Combinations (VHC) as a metric

Cell	No. of VHC Without Buried Layer	No. of VHC With Buried Layer	Improvement (%)	No. of M_2 routes
SDFF_X1	35	84	140.0%	4
XNOR2_X1	2	8	300.0%	3
LAT_X1	30	46	53.3%	2
MX2_X1	183	205	12.02%	1

Table 7.3: Benchmarks used in Place and Route Experiments

Testcase	No. of Gates	No. of Routing Layers
Cortex M0	9800	4 (M_2 - M_5)
FPU	27140	3 ($M_2 - M_4$)
MIPS	7967	2 ($M_2 - M_3$)

of 59 cells with M_{-1} layer, three benchmarks were synthesized, placed and routed. A Layout Exchange Format (LEF) file reflecting the modified cells geometry, due to the buried layer, is used. The used test cases are FPU and MIPS from Open Cores as well as a Cortex M0 processor. These have been placed and routed using Cadence Encounter. The number of gates in each benchmark and the number of metal layers used in routing is shown in Table 7.3.

Table 3.

The highest utilization factor, at which the design is routable with no design rule violations, is used in each experiment. Table 7.4 shows the chip area reduction due to the decrease in congestion on M_1 and M_2 without decreasing the cell area, and the final area reduction due to the reduction in cell area as well as congestion relief on M_1 and M_2. The final chip-level area saving is around 9 %-13%, due to the M_{-1} buried layer.

Table 7.4: Results of the final chip area reduction

	Cortex M0	FPU	MIPS
Replacement of *M1* and *M2* without cell-level area reduction	9.9%	11.9%	7.5 %
Replacement of *M1* and *M2* with cell-level area reduction	11.8%	12.9%	8.9%

7.4 Manufacturing Process Flow and MP Decomposition

Process Flow In order to pattern the buried interconnect and via layers, we propose the following process. Cross-sections of the proposed process steps are shown in Figure 7.6.

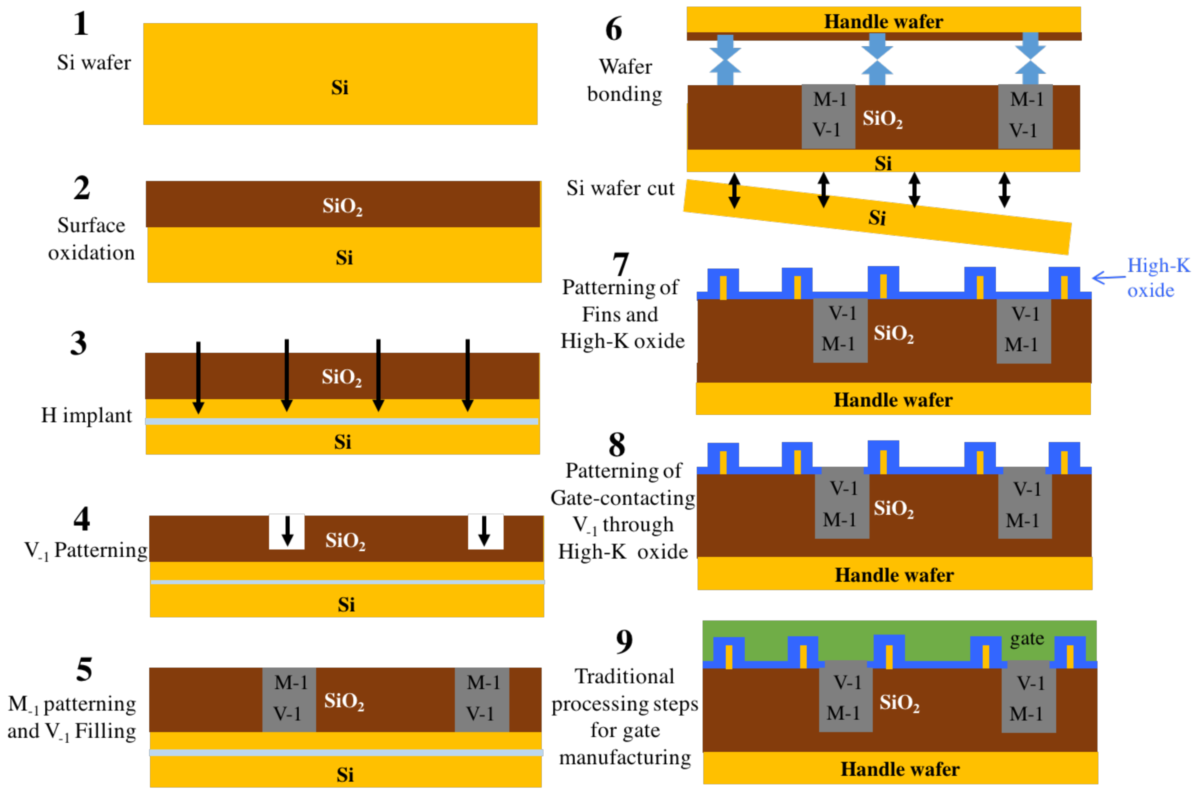


Figure 7.6: Manufacturing Process Flow for Buried Interconnect. In step 8, only the gate-contacting V_{-1} holes are patterned. The shown cross-section is in the gate region, not in the source/drain region

Steps (1-3) are similar to conventional SOI; they form oxide and separation layers for later SOI wafer cutting. In steps (4-5), buried interconnect lines and the buried vias are patterned in Tungsten above the Si wafer. Then the SOI wafer is cut and bonded to the handle wafer (step 6). The buried interconnect layer does not get damaged by the high temperature used in wafer bonding, because of the high melting point of Tungsten (3,422°C). The patterning of fins (potentially using Sidewall Image Transfer (SIT)) and gate oxide is carried out in step 7. In addition, the epitaxial growth of SiGe to merge fins in the source/drain regions takes place in step 7. Thus, the buried vias contacting the source/drain regions are already in physical contact with the respective regions. Next, the buried vias which contact the gate need to be patterned through the high-K oxide (step 8). In this work, we have assumed the dimension of the gate-contacting V_{-1} patterned in step 8 is 16nm. Depending on the technology and its required dimensions, alternative schemes for patterning contacts/vias (e.g., directed self assembly (DSA) [GTA14], E-beam direct write or EUV) can be used to pattern these vias. Finally, the remaining conventional steps for gate manufacturing take place in step 9.

MP Decomposition Advanced nodes have used MP technology [GG13], especially for lower metal/via layers. We evaluate the number of masks required to pattern the interconnect layers, with and without buried layer. We assumed Extreme Ultraviolet Lithography (EUV) is used in this technology. Currently the challenges in EUV patterning are developing high NA projection systems [SIB16] as well as fine resolution resists [KLH15]. Integrating multiple patterning with EUV is a candidate to replace the challenging high-NA EUV [KWH14]. As EUV comes closer to production, the allowed pitch will be smaller and thus fewer masks will be needed per layer. For the 7nm node, a single exposure of EUV is expected to achieve a metal pitch of 48nm with a preferred orientation [LCG15]. Since our M_1 layer is bidirectional, we use a more conservative EUV pitch of 51nm and accordingly multiple patterning steps are needed since the metal pitch in the used library is even smaller than 48nm. Mentor Graphics Calibre MP Decomposer for Double Patterning (DP), Triple Patterning (TP) and Quadruple Patterning (QP) has been run on the following layers*: CA, CB*, V_0*,* M_{-1}, V_{-1}*,* M_1-M_5 *and* V_1-V_4 on the standard cells in the library as well as the three chip-level testcases. The minimum number of masks needed to print these layers has been computed. Results are shown in Table 7.5. Even though one mask is necessary to pattern V_{-1} layer in step 4 in Figure 6, another mask is required

Table 7.5: Results of the MP decomposition for EUV. 'Orig.' is before introducing buried interconnect, and 'New' is after using buried interconnect.

Layer	Whole library		Cortex M0		FPU		MIPS	
	Orig.	New	Orig.	New	Orig.	New	Orig.	New
M_{-1}	N/A	SP	N/A	SP	N/A	SP	N/A	SP
V_{-1}	N/A	2*SP	N/A	2*SP	N/A	2*SP	N/A	2*SP
CA	DP	TP	DP	TP	DP	TP	DP	TP
CB	DP	N/A	DP	N/A	DP	N/A	DP	N/A
V_0	DP	DP	DP	DP	DP	DP	DP	DP
M_1	QP	TP	QP	TP	QP	TP	QP	TP
V_1	SP	N/A	DP	DP	DP	DP	DP	DP
M_2	DP	N/A	QP	QP	QP	QP	QP	QP
V_2	N/A	N/A	DP	DP	DP	DP	DP	DP
M_3	N/A	N/A	SP	SP	SP	SP	SP	SP
V_3	N/A	N/A	SP	SP	SP	SP	N/A	N/A
M_4	N/A	N/A	SP	SP	SP	SP	N/A	N/A
V_4	N/A	N/A	SP	SP	N/A	N/A	N/A	N/A
M_5	N/A	N/A	SP	SP	N/A	N/A	N/A	N/A
Total masks	N/A	N/A	23	24	22	23	20	21

for the gate-contacting V_{-1} in step 8. Thus, two masks are needed to pattern V_{-1}.

The number of required masks for patterning M_1 with EUV has decreased from four masks (QP) to three (TP) since the buried layer has relieved the congestion on M_1. In addition, M_2 is no longer used for intra-cell routing, and *CB* has been eliminated altogether. One mask only (Single Patterning (SP)) is required for each of $M_{-1} and V_{-1}$. However, as shown in Figure 7.6 step 8, another mask is required to pattern gate-contacting V_{-1}. *CA* needs one more mask since *CA* has become gate as well as source/drain contact layer. From the chip-level MP decomposition results, using the buried layer interconnect adds one mask only to the masks required for the interconnect stack, even though two layers (M_{-1} and V_{-1}) have been added. Note that the used router is not MP-aware, so the reduction of the number of masks reduced on M_1 is only due to the decrease in congestion due to the introduction of the buried layer.

7.5 Performance Evaluation

In order to assess the possible performance loss introduced by M_{-1} and V_{-1}, TCAD simulations have been performed for FinFETs with the buried M_{-1} layer and a via V_{-1} layer as shown in

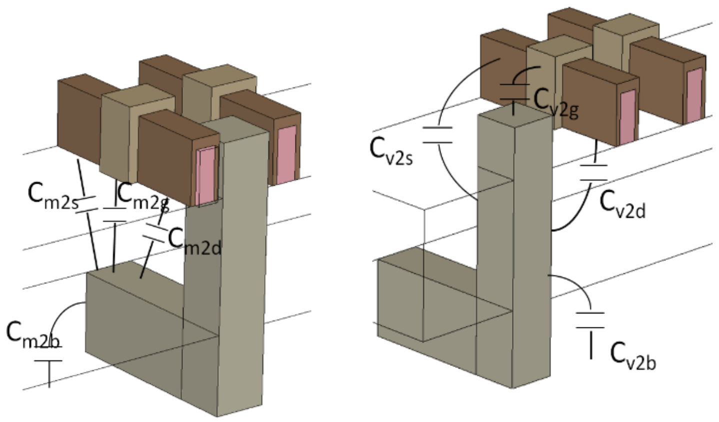


Figure 7.7: TCAD simulations of FinFETs with M_{-1} and V_{-1} (left: V_{-1} between fins, right: V_{-1} beside fins)

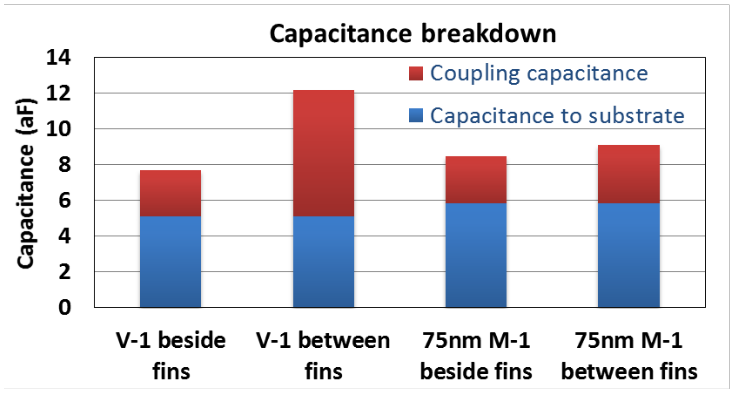


Figure 7.8: Capacitance breakdown for M_{-1} and V_{-1}. Coupling capacitance contains C_{v2s}, C_{v2d}, and C_{v2g}. Capacitance to substrate is C_{v2s}.

Figure 7.6. The different types of extracted capacitance of the buried via and metal lines are shown in Figure 7.7. The capacitance breakdown is shown in Figure 7.8.

The coupling capacitance of V_{-1} (C_{v2s}, C_{v2d}, and C_{v2g}) is larger when it exists between fins than beside fins. The coupling capacitances between two V_{-1} vias and between two M_{-1} segments are not analyzed because V”-$_{-1}$ and M_{-1} layers are relatively sparsely utilized indicating that they have less coupling capacitance than M_1, M_2 and V_0.

The coupling effect may also introduce threshold voltage shift and leakage current increase of the fins that are electrically disconnected from the near vias. The introduced leakage decreases with the increase of V_{-1}-to-fin distance. A severe leakage increase only occurs when a via is placed very close to an active FinFET, e.g., a V_{-1} is placed between fins with 4nm distance can increase leakage by 20%. However, this never happens in our library, and the smallest V_{-1}-to-fin distance is greater than 40nm, which may have less coupling effect than other metal

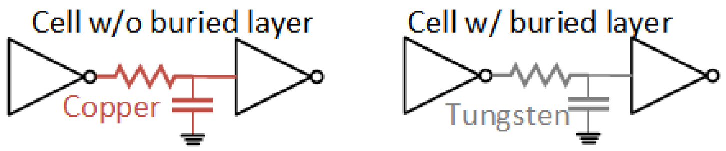


Figure 7.9: Simplified logic gate stage for standard cell delay change estimation due to using buried layer

and via layers, e.g., some V_0 close to fins. Thus, the leakage penalty is negligible and we ignore it in the timing evaluation.

The buried layer is assumed to be made of Tungsten to be able to withstand the high temperature involved in wafer bonding. Thus, the buried layer has higher resistivity (5.6x10^{-6} ohm cm) than the copper metal layers (1.7x10^{-6} ohm cm). We estimate the effect of that on the propagation delay of the standard cells and chip-level performance. TCAD simulations on entire library will take infinite time and is out of the scope of this chapter. We use a simplified logic gate stage shown in Figure 7.9 as in [WPC16] , which contains a driving gate (e.g., inverter), a wire load (resistance and capacitance), and a gate load (e.g., inverter), to simulate the propagation delay change of standard cells after using the buried layer. The accuracy of the simplified gate model for chip-level speed estimation has been verified in [WPC16] against synthesis, placement, and routing. In our library, when a cell is redesigned with the additional buried layer, total copper wire length (M_1 and M_2) is reduced and the use of Tungsten (M_{-1} and CA) increases due to the routes that get relocated to M_{-1} from M_1 and M_2. The reduced copper wire and two copper vias (V_0) are used as the wire load for simulating delay of standard cells without buried layer, while the increased tungsten wire and two tungsten vias (V_{-1}) serve as the wire load for cells with buried layer. The unit capacitance of copper wire is assumed to be same as tungsten, while in reality M_1 and M_2 layers are more utilized than M_{-1} and should have larger coupling capacitance than M_{-1}, indicating that the performance evaluation is pessimistic for the buried layer. The driving and load gate sizes vary from 3 to 12 fins according to standard cell size. The SPICE simulation results show that 22 out of 59 cells have decreased propagation delay after using the buried layer, while 37 cells see delay increase. The cell delay change ranges from -3.6% to 2.1%. In addition, on the average the buried layer led to 3.5% overall wire length reduction per cell, which explains the delay reduction of some standard cells.

To perform chip-level performance evaluation, a timing library is generated by applying the change in propagation delay to the original timing library. Timing analysis is performed after Place and Route using the modified library. The delay of the most critical path changes by 0.13%, 0%, and -0.01% in Cortex M0, MIPS and FPU respectively. Thus the chip-level performance change is too small and negligible.

7.6 Conclusion

A buried interconnect layer has been introduced to a standard cell library in order to alleviate the congestion on the traditional interconnect layers. It has been shown that the buried layer can improve pin access by 126%, and save chip area by 9-13%, which makes it a valuable technology scaling primitive.

CHAPTER 8

Supervia: Relieving Routing Congestion using Double-height Vias

With the increase in transistor packing density and the use of uni-directional metal routing, resources on local metal layers are increasingly limited. In addition, the minimum metal area (minArea) design rule has been steadily increasing over the past few technology nodes. For a net which crosses multiple metal layers (e.g., M2 to M4), polygons on intermediate layers (e.g., M3), i.e. via landing pads, must satisfy the minArea rule; this creates unnecessary routing blockage, which can lead to area overhead.

In this chapter, we investigate the benefits of using a new "supervia" structure, which is a double-height via spanning two metal layers without a landing pad on an intermediate metal layer. We study the benefit of supervia using (i) routing clip-based evaluation using an optimal ILP-based router (OptRouterSV) and (ii) chip-level evaluation using a commercial routing tool in conjunction with MILP-based supervia aware legalization. [1]

8.1 Introduction

Local metal congestion has increased in the recent nodes. In addition, the minimum metal area rule has not scaled down (due to challenges in deposition and lithography processes) – e.g., going from the 65nm node to the 20nm node, this rule has worsened from 3x to 6x (i.e., multiples of minimum metal width). [2] Moreover, there is increased via blockage as more nets

[1]The work in this chapter is a collaboration with Saptadeep Pal, Hyein Lee, Kwangsoo Han and Professor Andrew Kahng.

[2]In the following, we adopt the convention that an "Nx" minimum-area (minArea) rule requires metal area of Nx^2, where x is the minimum metal width.

are routed on intermediate or global metal layers due to performance reasons, as well as due to enforcement of unidirectional routing. Finally, pin access has become more challenging, particularly with emerging device architectures which scale the front-end well and accordingly increase pin density.

Local metal congestion has been addressed throughout the design flow, even up to such early stages as logic synthesis [CHR11]. Via minimization[XK89, DB89] has received substantial attention as well, both to minimize via-blockage and improve yield/reliability. A comprehensive via blockage model [CDZ00] has shown that via blockage on local interconnect layers can waste up to 50% of the wiring area. Several recent works have tried to incorporate via minimization explicitly in global routing [XZC09, LW08, STD17]. Pin access improvement also has been discussed in physical design [XYG16, Nie11] as well as standard cell design [XCY15].

Though all the above approaches are helpful in improving routability, they fail to address the increasing trend of the minArea rule. Therefore, in this work we propose a new *supervia* technology primitive aimed at addressing this challenge. A supervia is a double-height via which (unlike conventional stacked vias) does not require a landing pad in the intermediate layer. We develop a supervia-aware legalization flow which allows us to assess supervia benefits at chip-scale. We then analyze density benefits coming from supervias for a variety of designs and minArea rule values.

The remainder of this chapter is organized as follows. Section 8.2 explains the minArea rule and introduces the physical structure and manufacturing process of supervia. In Section 8.3, we describe our two approaches that are used to evaluate supervia benefit: (i) an optimal MILP-based router *OptRouterSV* for clip-level evaluation, and (ii) an MILP-based legalizer used in conjunction with commercial place-and-route (P&R) tools for chip-level evaluation. Experimental setup and results are discussed in Section 8.4. Section 8.5 concludes our work.

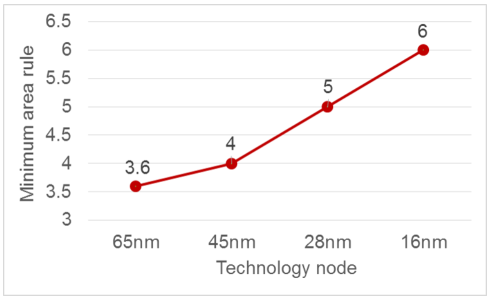


Figure 8.1: Minimum metal area rule vs. technology nodes.

8.2 Min-Area Rule and SuperVia

8.2.1 minArea Rule

Current technologies enforce a minArea requirement on every metal polygon. This rule has experienced an increasing trend over the past few technology nodes, as shown in Figure 8.1. Two main contributing factors to this trend are lithography and deposition. Local metal layers have started to extensively use multiple-patterning (MP), which decreases the metal pitch but does not help the minArea requirement which is still dictated by a single exposure. Further, copper metallization requires metal trenches to be lined with a barrier material (e.g., Ta, TaN, TiN, TiW) to prevent highly reactive copper atoms from leaching into silicon [ZLW05, Wan94]. However, forming a uniform thin barrier layer is a challenging deposition task, more so when the metal line trench and via openings are small. So, this puts a constraint on the minimum size of a trench opening in a dual damascene process.

As noted above, the minArea rule has added to the congestion challenge on local metal layers. Vias that traverse two layers will require a minArea landing pad on the middle layer – for example, an M2-M4 via will need a minimum-area landing pad on M3 – which can cause excessive via induced blockage. This wastes routing resources on intermediate metal layers. Our experiments on a projected 7nm library and two small design blocks indicate that such intermediate layer blockage can be 15-20% of total via blockage on the M2 and M3 layers, and that more than 50% of signal net edges traverse more than two metal layers while routing.

Table 8.1 shows how the minArea rule affects achievable maximum *utilization* in a P&R block, for three small designs AES, MIPS and ARM CORTEX M0. The experimental setup used to obtain these utilization values is described in Section 8.4.2 below. The utilization of a block is the fraction of block area occupied by the standard cells; we use utilization (equivalently, layout density) as an indicator of chip area in this chapter. As seen from the table, increasing the minArea rule has a negative impact on utilization. For sub-10nm technology, this minArea rule is expected to be $\geq$ 6x. For the testcases we have used, the resulting drop in achievable utilization could be $\geq 7\%$. This is obviously of great concern, as such a loss of utilization would cancel recent node-to-node layout density gains.

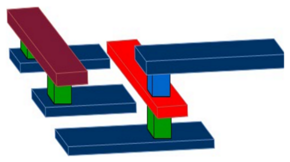
Table 8.1: Maximum achievable utilization with different minArea rules (1x, 3x, 5x, 7x) and different numbers of metal layers.

Testcase	#Layers	MinArea rule			
		1x	3x	5x	7x
MIPS_4	4	97%	97%	95%	94%
MIPS_5	5	98%	98%	97%	95%
M0_5	5	91%	89%	88%	84%
M0_6	6	95%	95%	92%	91%
AES_5	5	91%	88%	86%	84%
AES_6	6	97%	95%	94%	93%

8.2.2 Supervia

Figure 8.2(a) shows a normal stacked via with landing pad on the intermediate metal layer. Our proposed *supervia* structure is presented in Figure 8.2(b). A supervia is a single double-height via fabricated at once. As seen from the figures, the landing pad on the intermediate layers blocks the track surrounding it and thus increases congestion.

The supervia structure can be realized with deep-etch technologies, which have been developed and used for high-aspect ratio cut layers in various emerging technologies such as 3D-NAND flash [GM06, XPM15, SPM11].


(a) Stacked via with landing pad

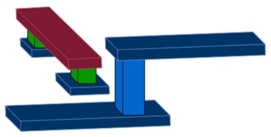

(b) Supervia

Figure 8.2: Non-supervia vs. supervia.

8.3 Supervia Aware Layout Legalization

In this section, we describe our supervia and minArea-aware legalizers. We propose two legalizers: (i) an optimal MILP-based router for clip-level evaluation and (ii) an MILP-based chip-level legalizer. The notations used in this section are summarized in Table 8.2.

Table 8.2: Notations.

Notation	Meaning
N	set of multi-pin nets
n_k	k^{th} multi-pin net
s_k	source of n_k
T_k	set of sinks of n_k
$t_{k,i}$	i^{th} sink of n_k
$G(V, A)$	routing graph
U	set of vertices (of the routing graph)
u_i	a vertex with the location (x_i, y_i, z_i)
A	set of directed arcs
$a_{i,j}$	a directed arc from u_i to u_j
$e^k_{i,j}$	0-1 indicator whether $a_{i,j}$ is used in the routing of n_k
$\beta^k_{i,j}$	cost for $a_{i,j}$ in the routing of n_k
$f^k_{i,j}$	flow variable for $a_{i,j}$ in the routing of n_k
$p^k_{i,c}$ ($q^k_{i,c}$)	c^{th} left (right) EOL extension option for via vertex u_i, net n_k
Γ	set of metal layer numbers
Π	set of via layer numbers
W	set of all horizontal wire segments
m^γ_w	minimum width on metal layer $\gamma \in \Gamma$
m^γ_s	minimum space on metal layer $\gamma \in \Gamma$
m^γ_l	minimum length on metal layer $\gamma \in \Gamma$
h^π_w	via width on via layer $\pi \in \Pi$
z_i	layer number of wire segment w_i
l_i (r_i)	left (right) variable of wire segment $w_i \in W$
l^{orig}_i (r^{orig}_i)	left (right) original location of wire segment $w_i \in W$
v_j	location variable for a via connected to $w_i \in W$
$b_{j,j'}$	0-1 flag indicating whether via j and j' are vertically aligned. 0 means vias j and j' are vertically aligned
$land(j, j')$	the wire segment which is only a landing pad for both vias j and j'
Q	Set of wire segments which are not landing pads
Δ^l_i (Δ^r_i)	left (right) perturbation to wire segment $w_i \in W$
S_i	Elastic/Slack variable representing a minArea violation (Non-negative)

8.3.1 Clip-level Legalizer (OptRouterSV)

In *OptRouter* [HKL15], minArea rule constraints and supervias are not considered. We extend the ILP formulation in [HKL15] to comprehend minArea rules and supervia. Note that a minArea rule can be converted to a minLength rule (m_l^γ) for each metal layer, by assuming 1D routing.

For a given three-dimensional routing resource, horizontal metal track x_i, vertical metal track y_i and metal layer z_i, we formulate our MILP optimization as follows.

$$\text{Minimize:} \quad \sum_{n_k \in N} \sum_{a_{i,j} \in A} \beta_{i,j}^k \cdot e_{i,j}^k$$

Subject to:

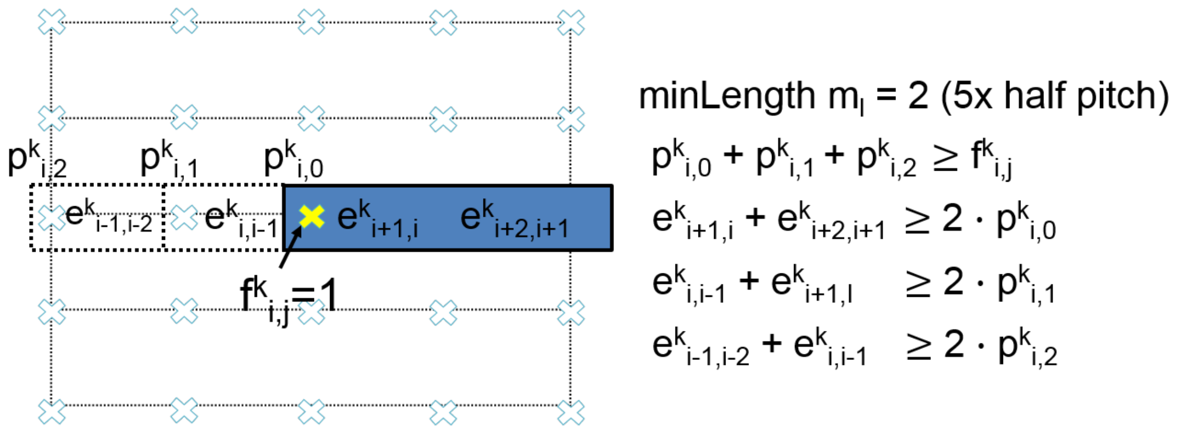
$$\sum_{n_k \in N} (e_{i,j}^k + e_{j,i}^k) \leq 1 \quad a_{i,j}, a_{j,i} \in A \tag{8.3.1}$$

$$e_{i,j}^k \geq \frac{f_{i,j}^k}{|T_k|} \quad a_{i,j} \in A, n_k \in N \tag{8.3.2}$$

$$\sum_{u_j : a_{i,j} \in A} f_{i,j}^k - \sum_{u_j : a_{j,i} \in A} f_{j,i}^k = \begin{cases} |T_k| & \text{if } u_i = s_k, n_k \in N \\ -1 & \text{else if } u_i \in T_k, n_k \in N \\ 0 & \text{otherwise} \end{cases} \tag{8.3.3}$$

The objective is to minimize the weighted sum of total wirelength and the number of vias. Note that we can change the objective to a constant value to check feasibility with the constraints. Constraints (8.3.1), (8.3.2) and (8.3.3) enable multi-commodity flow for multi-pin-net routing. Constraint (8.3.1) ensures that each arc is used by only one net. Constraint (8.3.2) pertains to the binary variable $e_{i,j}^k$, which indicates whether there is a flow through $e_{i,j}$. Constraint (8.3.3) ensures source-sink connectivities (flow conservation).

End-of-line (EOL) extension and minArea rule constraints. As we can identify the locations of EOL from via locations in 1D routing if we do not consider EOL extension, we might be able to control minLength rule by using via spacing rules. However, this is not correct and realistic, and tight via spacing rules coming from a larger minLength rule will restrict routing solutions severely in sub-10nm technology nodes. Thus, we introduce EOL extension variables to allow



minLength m_l = 2 (5x half pitch)
p^k_{i,0} + p^k_{i,1} + p^k_{i,2} ≥ f^k_{i,j}
e^k_{i+1,i} + e^k_{i+2,i+1} ≥ 2 · p^k_{i,0}
e^k_{i,i-1} + e^k_{i+1,I} ≥ 2 · p^k_{i,1}
e^k_{i-1,i-2} + e^k_{i,i-1} ≥ 2 · p^k_{i,2}
p^k_{i,2}
p^k_{i,1}
p^k_{i,0}
e^k_{i-1,i-2}
e^k_{i,i-1}
e^k_{i+1,i}
e^k_{i+2,i+1}
f^k_{i,j}=1

Figure 8.3: minLength rule constraints with EOL extension variables.

wire segments to be extended to fix minLength violations.

The EOL variables represent the locations of EOL extension options for a wire segment. Each wire segment is extended according to the EOL location defined by the corresponding EOL variable selected by ILP. Note that EOL variables are created at via locations, where EOLs exist, since we assume 1D routing. Figure 8.3 shows an example of EOL variables, where a wire segment with a flow $f^k_{i,j}$ on a via, where $x_j = x_i$, $y_j = y_i$ and $z_j = z_i \pm 1$ (i.e., a via is placed at location (x_i,y_i) between layers z_j and z_i), and the flow continues to the right direction (i.e., the wire segment has a left EOL). In the example of Figure 8.3, the minLength $m^\gamma_l = 2$. For the left EOL, we introduce three EOL extension options $p^k_{i,0}$, $p^k_{i,1}$, $p^k_{i,2}$ which indicate EOL extensions toward left. We force only one of the options to be chosen among the three EOL options. Note that we do not need more than three options since $m^\gamma_l = 2$. For each EOL option, we force edges between the EOL location and the via location of the wire segment to be one to make the wire segment extended. To do so, the sum of corresponding e variables must meet the minLength m^γ_l. If $p^k_{i,0}$ is selected, $e^k_{i+1,i}$ and $e^k_{i+2,i+1}$ must be one.

A generalized formulation is given in Constraint (8.3.4).

$$\sum_{c \in C} p^k_{i,c} \geq f^k_{i,j} \qquad a_{i,j} \in A \tag{8.3.4}$$

$$\sum_{a_{i,j} \in A'} e^k_{i,j} \geq m^\gamma_l \cdot p^k_{i,c} \qquad c \in C \tag{8.3.5}$$

If a flow variable $f^k_{i,j}$ on a via is used (i.e., whenever there is a via at location (x_i, y_i), Constraint (8.3.4) forces that one of the left EOL extension variables must be selected. Given a minLength m^γ, if an EOL extension variable $p^k_{i,c}$ is selected, the sum of corresponding $e^k_{i,j} \in A'$ must be larger than or equal to the minLength (Constraint (8.3.5)). We can similarly treat right EOL extension variables $q^k_{i,c}$ and minLength rule constraints.

Constraints (8.3.4) and (8.3.5) handle only left EOLs of wire segments. To handle right EOLs of wire segments together, the constraints are rewritten as follows.

$$\sum_{c \in C} (p^k_{i,c} + q^k_{i,c}) \geq f^k_{i,j} \qquad a_{i,j} \in A \tag{8.3.6}$$

$$\sum_{a_{i,j} \in A'} e^k_{i,j} \geq m^\gamma_l \cdot p^k_{i,c} \qquad c \in C \tag{8.3.7}$$

$$\sum_{a_{i,j} \in A'} e^k_{i,j} \geq m^\gamma_l \cdot q^k_{i,c} \qquad c \in C \tag{8.3.8}$$

Supervia constraints. To enable supervia, minLength rule constraint should not be applied for the intermediate layer when two vertically aligned consecutive vias are used. In our formulation, we have additional constraints as follows.

$$\sum_{c \in C} (p^k_{i,c} + q^k_{i,c}) \geq f^k_{i,j} - f^k_{i,j'} \qquad a_{i,j} \in A \tag{8.3.9}$$

$f^k_{i,j}$ and $f^k_{i,j'}$ are the flows of two vertically aligned vias, where $x_i = x_j = x_{j'}$, $y_i = y_j = y_{j'}$, $z_j = z_i + 1$ and $z_{j'} = z_i - 1$.

8.3.2 Chip-level Legalizer

We propose a supervia-aware legalization method based on Mixed Integer Linear Programming (MILP). The input of the legalizer is a routed layout with minArea-rule violations and the objective is to minimize the minArea violations by applying supervias. There are two main differences between our legalizer and the classic migration/legalization algorithm [HCT97]. First, our legalizer does not change the front-end-of-line (FEOL) of standard cells but only the BEOL, and in addition it keeps pin locations of standard cells and intra-cell routes unchanged. Second, we consider supervias in the MILP formulation to take the advantage of the supervias during the legalization. The migration is performed in the X-direction followed by the Y-direction. Two iterations of migration are performed (Experimentally no further iterations were needed). We explain our layout representation and MILP formulation in the following paragraphs.

MILP formulation.

In this section, we show the MILP formulation for migration in the X-direction. The MILP for the Y-direction is similar.

An example of our layout representation is shown in Figure 8.4, where the variables representing three metal segments and a via are shown. All the routes are assumed to be unidirectional. Each layout rectangle on the metal layers M2 and above is represented by its four edges. The variables l_i and r_i correspond to the left and right edges of rectangle i respectively, as shown in Figure 8.4.

Since we assume all vias of each layer have the same dimensions, only the bottom left corner (thus left edge and bottom edge) are enough to represent the via.

$$\text{Minimize:} \quad minimize \sum_i (\Delta_i^l + \Delta_i^r) + \lambda * \sum_i S_i \tag{8.3.10}$$

Subject to:

$$\Delta_i^l \geq l_i - l_i^{orig} \qquad \forall w_i \in W \tag{8.3.11}$$

$$\Delta_i^l \geq l_i^{orig} - l_i \qquad \forall w_i \in W \tag{8.3.12}$$

$$\Delta_i^r \geq r_i - r_i^{orig} \qquad \forall w_i \in W \tag{8.3.13}$$

$$\Delta_i^r \geq r_i^{orig} - r_i \qquad \forall w_i \in W \tag{8.3.14}$$

$$l_i - r_{i'} \geq m_s^\gamma \qquad \forall (w_i, w_{i'}), \gamma \in \Gamma \tag{8.3.15}$$

$$r_i - l_i \geq m_w^\gamma \qquad \forall \gamma \in \Gamma \tag{8.3.16}$$

$$b_{j,j'} = 0 \implies r_i - l_i + S_i \geq m_l^\gamma \;\; \forall w_i | w_i = land(v_j, v_j') \tag{8.3.17}$$

$$r_i - l_i + S_i \geq m_l^\gamma \qquad \forall w_i \in Q \tag{8.3.18}$$

$$l_i \leq v_j \qquad \forall w_i \in W, v_j \in w_i \tag{8.3.19}$$

$$r_i \geq v_j + h_w^{z_i} \qquad \forall w_i \in W, v_j \in w_i \tag{8.3.20}$$

We use the cost function of the relaxed 1D minimum layout perturbation problem proposed in [HCT97], which aims at minimizing the change to the layout while fixing the design rule violations. However, we only allow relaxation of the minArea constraints, using the elastic variables S_i.

The non-linear **cost function** in the X-direction is:

$$minimize \sum_i |l_i - l_i^{orig}| + |r_i - r_i^{orig}| + \lambda * \sum_i S_i$$

This cost function is linearized by adding a variable (Δ_i^r) for each edge (r_i), representing the absolute value of the perturbation done to the edge. Constraints (8.3.11), (8.3.12), (8.3.13) and (8.3.14) are added accordingly. Summation of the absolute variables are then minimized as shown in Equation (8.3.10).

The constraints are explained below.

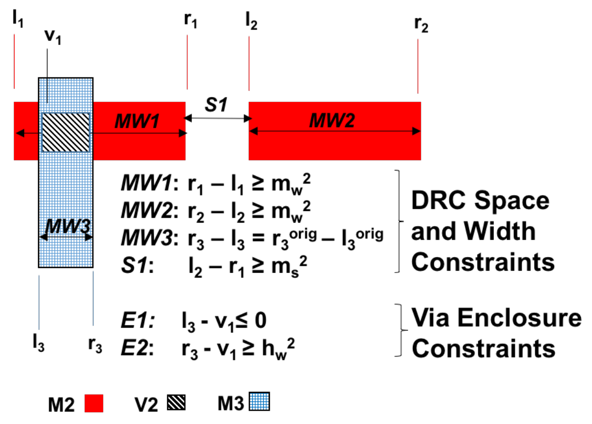


Figure 8.4: Example of our layout representation and the MILP variables and constraints for the X-direction legalization. The generated DRC and via enclosure constraints are shown. m_w^2 is the minimum width value DRs for M2. m_s^2 is the minimum space allowed on M2. h_w^2 is the dimension of each via hole on via2 layer.

Design rule constraints. The enforced design rules are minimum width and minimum space on metal layers.

The minimum space constraints are generated between every two opposite edges from different polygons, as shown in S1 constraint in Figure 8.4. Constraint (8.3.15) shows the minimum space constraint. Along the direction of the routing, the length of the segment must not decrease below the minimum width rule. The width of the segment is constrained to be equal to the original width. For example, in Figure 8.4, MW1, MW2 and MW3 represent the minimum width constraints, which are described in Constraint (8.3.16). Non-minimum width polygons are preserved like [HCT97], but non-min-space distances are not preserved.

MinArea and supervia constraints. If a metal segment overlaps with exactly one via above it and one via underneath (hence the segment only exists to be a landing pad for both vias) and the two vias are perfectly aligned, then a supervia can be created. Otherwise, the metal segment has to obey minArea rule. These two cases are shown in Figure 8.2. This is created as a conditional Constraint (8.3.17), which is linearized using Indicator Constraint Transformation [ABG10]. We only allow supervias to replace a stack of via1 and via2 shapes. The general minArea rule is enforced through Constraint (8.3.18). In both Constraints (8.3.17) and (8.3.18), an elastic variable S_i [Chi07] is added to allow the violation of the minArea rule, with a penalty in the cost function.

Via enclosure constraints. The only multi-layer constraints are the via enclosure constraints which are Constraints (8.3.19) and (8.3.20). These force the via to remain enclosed within its top and bottom metal layers; i.e. all edges of the via lying within the two metal polygons. Accordingly the connectivity is preserved. An example is shown in Figure 8.4, where the via enclosure constraints are E1 and E2.

8.4 Experimental Setup and Results

In this section, we present our experimental setup and results. We experiment on the utilization used to route the design and the number of violations that exist after each stage of the flow. These experiments show the chip-scale benefit of using supervia against conventional non-supervia case.

8.4.1 Experimental Setup

In our experiments, we use two designs (AES and MIPS) from OpenCores and an ARM Cortex M0 design (M0). We synthesize the RTL netlists of the testcases and then perform P&R with an abstracted 7nm library from an industrial IP provider using *Cadence Innovus 16.10*. The testcase information is summarized in Table 8.3. The naming convention follows '{design name}_{#metal layer}'. #Inst column shows the total number of instances, #Nets column shows the total number of nets; fourth to eighth columns show the number of vias, e.g., V12, V23, V34, V45 and V56. Each testcase is implemented with $minArea = 1x$, with the maximum achievable utilization. The utilization numbers are reported in Table 8.1.

Table 8.3: Testcases Details.

Testcase	Inst	Nets	V1	V2	V3	V4	V5
MIPS_4	11330	7942	29320	41637	13672	0	0
MIPS_5	11345	7942	29395	38502	14421	3737	0
M0_4	79059	9709	34579	50429	13672	0	0
M0_5	15209	9826	34573	46427	14421	5784	0
M0_6	14555	9769	34398	44921	14090	5748	2864
AES_5	20366	13867	48480	65017	20512	9302	0
AES_6	18410	13858	48362	61437	20203	9430	4533

OptRouterSV and *Legalizer* are written in C++ with *OpenAccess* to support LEF/DEF and extract the polygon shape information from the routed layout. We use *CPLEX* [IBM] as our MILP solver. Geometric operations in the Legalizer are performed using the Boost library.

8.4.2 Clip-level Evaluation Flow

It is challenging to quantify the benefits of using supervia due to (i) lack of support for double-height vias in commercial detailed routers, and (ii) the large turnaround time of the entire P&R flow. Thus, we use a routing clip-based evaluation framework (*OptRouterSV*), to study the benefit of supervia. Our clip-level evaluation framework provides *routing completion rates*, i.e., how much percentage of routing clips are routable with a particular configuration, for routing clips for the given minArea rule and via option.

For the clip-level evaluation, we use 10 $\times$ 10 tracks routing clips extracted from routed designs as input instances. The routing clips are selected based on violating points that cannot be solved by the chip-level legalization method. Figure 8.5 shows how we convert extracted routing clips to routing problems. We first map all routing and pin segments into a 3D routing map. We then remove internal routing segments, except internal pin segments for incoming and outgoing nets and routing segments at boundaries for feed-through nets. We then run *OptRouterSV* for each routing clip to see whether the clip is routable for a given minArea rule. In our experiments, we run 100 routing clips and check routing completion rate. We note that although the clip-level evaluation might not be a proxy for chip-level evaluation, it can provide statistics for multiple evaluation data points which can be used for a projection to chip-level evaluation. More specifically, the routing completion rates can be used to calculate a projected number of minArea violations.

We study routing completion rate for different minArea rules, along with supervia (SV) option and non-supervia (non-SV) options across different designs. The results are shown in Table 8.4. We run *OptRouterSV* for 100 routing clips for each testcase. 100% means that 100 clips among the total 100 clips have legal routing with respect to a particular minArea rule and via option.

For $minArea = 3x$, all designs show close to 100% routing completion rate for both SV and

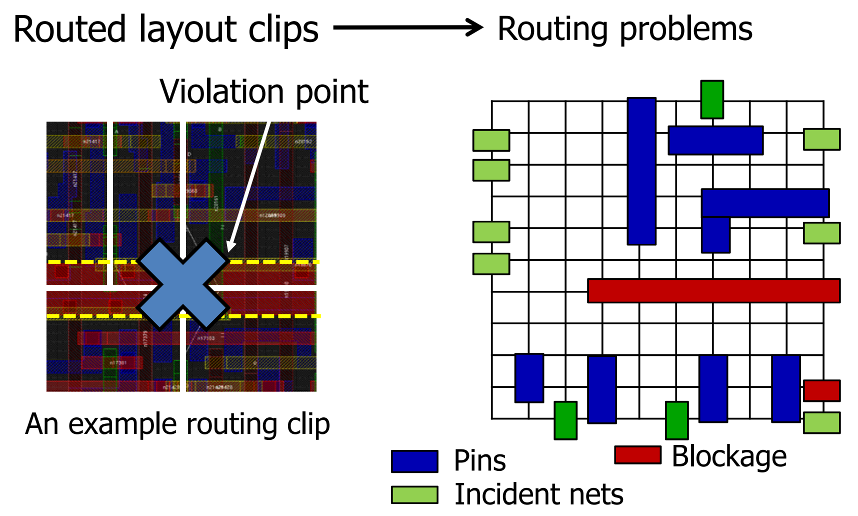


Figure 8.5: An example of a routing clip. We convert the extracted routing clips to routing problems by removing internal routing segments except for internal pins and routing segments at the boundaries.

non-SV cases. For $minArea = 5x$, we observe 77%∼99% completion rates. We note that M0_4 is a special case; the utilization is exceptionally low (i.e., 65%) as shown in Table 8.1. Due to the low utilization, standard cells are placed sparsely and thus routability and pin accessiblity are improved, which also helps to legalize routing segments for the minArea rules. We observe that as the number of used metal layers increases, the completion rates decreases, except for M0_4,5,6, $minArea = 7x$, SV case.

From the results, one can see that there is no need for supervias in the $minArea = 3x$ case though the need for routers to handle minimum area rule efficiently is exhibited. On the other hand there is a dramatic improvement in routability when $5x$ or $7x$ rules are used. Clip completion rates improve almost by a factor of two in the $minArea = 7x$ case showing the promise of supervias.

Table 8.4: Routing completion rate results on clips.

Testcase	3x		5x		7x	
	non-SV	SV	non-SV	SV	non-SV	SV
MIPS_4	100%	100%	87%	97%	57%	88%
MIPS_5	100%	100%	87%	96%	50%	77%
AES_5	100%	100%	85%	95%	29%	50%
AES_6	100%	100%	77%	90%	34%	71%
M0_4	100%	100%	99%	100%	71%	87%
M0_5	99%	99%	87%	99%	39%	70%
M0_6	99%	99%	46%	79%	40%	74%

8.4.3 Chip-level area assessment

Although the clip-level evaluation flow can project routability for a given minArea rule and via option, it is not sufficient to derive chip-level benefits in terms of area. Thus our *Legalizer* is used for chip-level evaluation flow.

The supervia-aware legalizer takes routed layouts and minArea rules, via options as inputs and indicates whether the input layout is legal with respect to the input rule and option. We start with a DRC clean $minArea = 1x$ design and legalize the design by solving the MILP shown in Section 8.3.2 such that minArea rule is enforced on all metal polygons except the intermediate metal segment of stacked via routes in case supervia is used. In case no supervia is used, then the minArea rule is enforced on all metal polygons.

If the legalization is not successful, the legalizer outputs the list of violations which it could not fix.

We discuss the experiments performed using *Cadence Innovus* (with optimal via generation option enabled) and supervia-aware legalizer.

In figure 8.6 and figure 8.7, total number of DRC violations vs utilization is shown for the following cases:

- Innovus run with 5x minArea rule
- Legalizer run with supervia option
- Legalizer run with without supervia option

As seen from these figures, the benefits of supervia is marginal ($\sim$2% for CORTEX_M0). This is because the number of DRC violations with supervia using the legalizer (i.e. the green curve) is lower than that of supervia-oblivious routing using Innovus (red curve) for only a few higher utilization points, after which the red curve gets better than the green curve. The key factors resulting in this trend are the following:

- minArea-aware via generation and routing used in Innovus

- Legalization is not the optimal tool to show the benefits of supervia due to its numerous limitations.

Thus, we can conclude from these results that minArea-aware P&R can recover most of the area hit coming from aggressive minArea rules in advanced technology nodes. However, from the small design testcases we used, we can observe that about a ∼2% density benefit can be achieved using supervia.

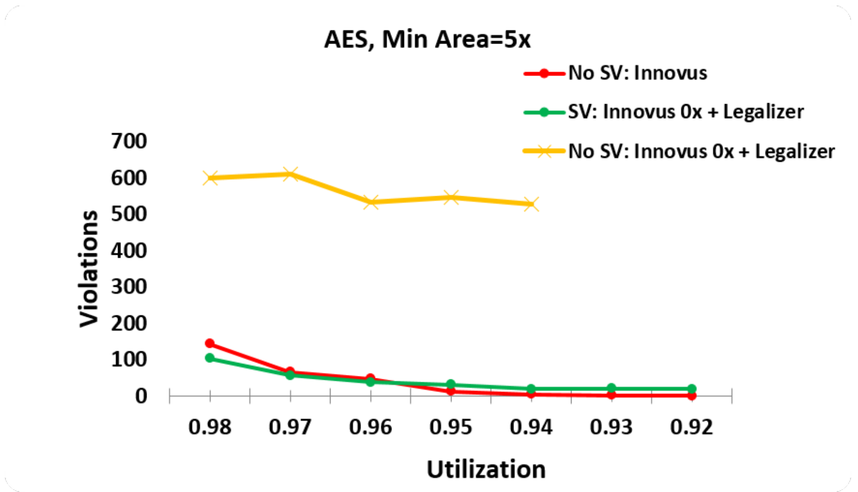


Figure 8.6: Chip Level Evaluation for AES where the "minArea = 1x" design from Innovus is legalized for 5x minArea rule

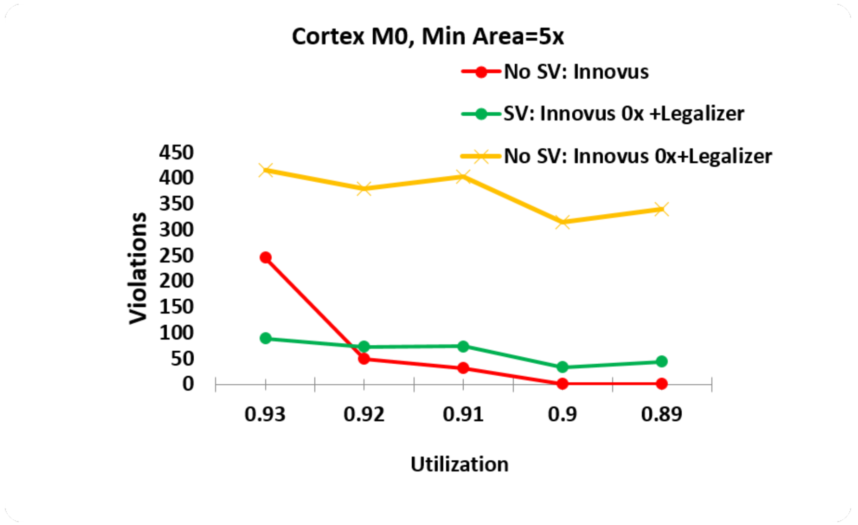


Figure 8.7: Chip Level Evaluation for CORTEX_M0 where the "minArea = 1x" design from Innovus is legalized for 5x minArea rule

8.5 Conclusions

In this work, we evaluate the benefit of supervia using (i) routing clips using an optimal ILP-based router (*OptRouterSV*) and (ii) chip-level evaluation using a supervia-aware legalizer on commercial router results.

Our results show that minArea-aware routing can recover most of the area required for minArea rule, and that the additional benefit of supervia is limited to around 2%.

Thus, in conclusion supervia isn't promising enough for digital logic routing. However, supervia can be an interesting option to consider for density scaling in the following few applications/ use-cases among possibly many:

- STT-RAM memory cell design where the memory cells are placed in the BEOL stack and supervias can be used to provide access connection directly from the access transistors to the memory cells, instead of wasting space for landing pads on local interconnect and lower metal layers. The use of supervia in STT-RAM has been demonstrated in [AWR17], even though the general idea of supervia was independently proposed in both works: ours and [AWR17].
- Since supervias are double-height vias realized by digging through multiple layers of SiO_2 at once, the resistive barrier layer on top of the bottom via can now be eliminated. This might find interest in on-chip power-distribution network design, since via resistance is becoming a major issue in the advanced nodes where the via dimensions are shrinking resulting in increased via resistance.

In the current technologies, it is unlikely that supervia can achieve density scaling in SRAM cells [IWP16], since the area is determined by the transistors (as opposed to routing).

CHAPTER 9

Conclusion

Sub-wavelength photolithography has made the co-optimization of design and technology a necessity. The work in this dissertation provided frameworks to facilitate design and technology co-optimization (DTCO). These DTCO frameworks have been made public [Lab].

9.1 Research Contribution

Part I proposed frameworks for evaluation of design rules. Chip-DRE provides a means for chip-level evaluation of design rules to address the shortcomings inherent in the traditional standard cell-based evaluation. Using Chip-DRE, foundries can estimate the number of good chips per wafer achieved by a set of design rules under evaluation. Pattern-DRE provides a framework for pattern-based evaluation of design rules. Foundries can use Pattern-DRE to find out the patterns that are important from a design perspective and accordingly optimize the process to this set of patterns.

Part II focused on Directed Self Assembly, and provided a framework for Path-finding for this technology. The main purpose of the framework is to find specifications for DSA-based technologies and design in order to develop a technology that is friendly to the design as well as the fabrication. We also proposed heuristics for simultaneous DSA grouping and Multiple Patterning decomposition for hybrid DSA+MP technologies.

Part III proposed two technology scaling boosters that are beyond patterning. The first one is the use of a buried interconnect layer for inter-cell connections. The buried layer showed up to a 13% reduction in the chip area. Then we proposed the use of supervia which is double height via which can avoid the need for a landing pad on the intermediate routing layer when a metal connection spans three or more routing layers. Supervia is especially useful when the minimum metal area rule results in wasting so many routing resources. However, for the average logic layouts the benefit of supervia is so small (2%).

9.2 Future Direction

The future direction for this research is to close the loop between the design optimization and the technology optimization. Using the DTCO frameworks that were proposed in this thesis, the output is a set of technology and design recommendations as a result of the optimization. Therefore a set of tools is needed to use the framework recommendations to smartly update/regenerate the used benchmarks, and feed them into the DTCO frameworks along with the optimized technology. This may need to be done iteratively until the iterative flow converges to benchmarks which are result of the co-optimized design and technology.

REFERENCES

[ABG10] Ashish Agarwal, Sooraj Bhat, Alexander Gray, and Ignacio E Grossmann. "Automating mathematical program transformations." In *International Symposium on Practical Aspects of Declarative Languages*, pp. 134–148. Springer, 2010.

[App04] Andrew W Appel. *Modern compiler implementation in C*. Cambridge university press, 2004.

[Aut12] Chris Auth. "22-nm fully-depleted tri-gate CMOS transistors." In *Custom Integrated Circuits Conference (CICC), 2012 IEEE*, pp. 1–6. IEEE, 2012.

[AW12] Islam S Abed and Amr G Wassal. "Double-patterning friendly grid-based detailed routing with online conflict resolution." *Design, Automation and Test in Europe (DATE)*, pp. 1475–1478, 2012.

[AWR17] Raf Appeltans, Pieter Weckx, Praveen Raghavan, Ryoung-Han Kim, Gouri Sankar Kar, Arnaud Furnémont, Liesbet Van der Perre, and Wim Dehaene. "The effect of patterning options on embedded memory cells in logic technologies at iN10 and iN7." In *SPIE Advanced Lithography*, pp. 101480G–101480G. International Society for Optics and Photonics, 2017.

[BDG15] Joost Bekaert, Jan Doise, Roel Gronheid, Julien Ryckaert, Geert Vandenberghe, Germain Fenger, YoungJun Her, and Yi Cao. "N7 logic via patterning using templated DSA: implementation aspects." In *Proc. SPIE*, volume 9658, p. 965804, 2015.

[BG17] Yasmine Badr and Puneet Gupta. "Technology path-finding framework for directed-self assembly for via layers." *Journal of Micro/Nanolithography, MEMS, and MOEMS*, **16**(1):013505–013505, 2017.

[BMG14] Yasmine Badr, Ko-wei Ma, and Puneet Gupta. "Layout pattern-driven design rule evaluation." *JM3*, **13**(4):043018, 2014.

[BRB07] Charles T Black, Ricardo Ruiz, Gregory Breyta, Joy Y Cheng, Matthew E Colburn, Kathryn W Guarini, H-C Kim, and Ying Zhang. "Polymer self assembly in semiconductor microelectronics." *IBM Journal of Research and Development*, **51**(5):605–633, 2007.

[Bru13] Jean-Marie Brunet. "Semiconductor Engineering- Let's All Meet At The Via Bar.", 2013.

[BSI] "BSIM4.6.2 User Manual.".

[BTG15a] Yasmine Badr, Andres Torres, and Puneet Gupta. "Incorporating DSA in multipatterning semiconductor manufacturing technologies." In *SPIE Advanced Lithography*, 2015.

[BTG15b] Yasmine Badr, Andres Torres, and Puneet Gupta. “Mask assignment and synthesis of DSA-MP hybrid lithography for sub-7nm contacts/vias.” In *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*, pp. 1–6. IEEE, 2015.

[BTG17] Yasmine Badr, Andres Torres, and Puneet Gupta. “Mask Assignment and DSA Grouping for DSA-MP Hybrid Lithography for Sub-7 nm Contact/Via Holes.” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **36**(6):913–926, 2017.

[Bus] Mentor A Siemens Business. “Calibre Litho-friendly Design tool.”.

[BYB11] Xin-Yu Bao, He Yi, C. Bencher, Li-Wen Chang, Huixiong Dai, Yongmei Chen, P.-T.J. Chen, and H.-S.P. Wong. “SRAM, NAND, DRAM contact hole patterning using block copolymer directed self-assembly guided by small topographical templates.” In *Electron Devices Meeting (IEDM), IEEE International*, 2011.

[cal31] *Mentor Graphics Calibre nmDP*, March 2013.1.

[CBB10] Li-Wen Chang, Xinyu Bao, C. Bencher, and H.-S.P. Wong. “Experimental demonstration of aperiodic patterns of directed self-assembly by block copolymer lithography for random logic circuit layout.” In *Electron Devices Meeting (IEDM), IEEE International*, 2010.

[CBP09] Simon Chang, James Blatchford, Steve Prins, Scott Jessen, Thuc Dam, Guangming Xiao, Linyong Pang, and Bob Gleason. “Exploration of complex metal 2D design rules using inverse lithography.” In *Proc. SPIE*, volume 7275, p. 72750D, 2009.

[CDZ00] Qiang Chen, Jeffrey A Davis, Payman Zarkesh-Ha, and James D Meindl. “A compact physical via blockage model.” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, **8**(6):689–692, 2000.

[CGK04] Luigi Capodieci, Puneet Gupta, Andrew B Kahng, Dennis Sylvester, and Jie Yang. “Toward a methodology for manufacturability-driven design rule exploration.” In *Proceedings of the 41st annual Design Automation Conference*, pp. 311–316. ACM, 2004.

[Chi07] John W Chinneck. *Feasibility and Infeasibility in Optimization:: Algorithms and Computational Methods*, volume 118. Springer Science & Business Media, 2007.

[CHR11] Mike Clarke, Diego Hammerschlag, Matt Rardon, and Ankush Sood. “Eliminating routing congestion issues with logic synthesis.” *Whitepaper, Cadence Design Systems*, 2011.

[Cou] Compact Model Council. “Guidelines for Extracting Well Proximity Effect Instance Parameters.”.

[CW08] Chris Chu and Yiu-Chung 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):70–83, 2008.

[CYB08] Minsik Cho, Kun Yuan, Yongchan Ban, and David Z Pan. "Eliad: Efficient lithography aware detailed router with compact post-opc printability prediction." *Design Automation Conference (DAC)*, pp. 504–509, 2008.

[DB89] Jitender S Deogun and Bhargab B Bhattacharya. "Via minimization in VLSI routing with movable terminals." *IEEE transactions on computer-aided design of integrated circuits and systems*, **8**(8):917–920, 1989.

[DC13] Vito Dai, Luigi Capodieci, et al. "Pattern matching for identifying and resolving non-decomposition-friendly designs for Double Patterning Technology (DPT)." *SPIE Advanced Lithography*, pp. 868409–868409, 2013.

[DCY09] Vito Dai, Luigi Capodieci, Jie Yang, and Norma Rodriguez. "Developing DRC Plus rules through 2D pattern extraction and clustering techniques." In *Proc. SPIE*, volume 7275, p. 727517, 2009.

[DGW13] Yuelin Du, Daifeng Guo, Martin D. F. Wong, He Yi, H.-S. Philip Wong, Hongbo Zhang, and Qiang Ma. "Block Copolymer Directed Self-assembly (DSA) Aware Contact Layer Optimization for 10 Nm 1D Standard Cell Library." In *Proc. ICCAD*, ICCAD '13. IEEE Press, 2013.

[DGY11] Duo Ding, Jhih-Rong Gao, Kun Yuan, and David Z Pan. "AENEID: A generic lithography-friendly detailed router based on post-RET data learning and hotspot detection." *Design Automation Conference (DAC)*, pp. 795–800, 2011.

[DMY11] Yunfei Deng, Yuangsheng Ma, Hidekazu Yoshida, Jongwook Kye, Harry J Levinson, Jason Sweis, Tamer H Coskun, and Vishnu Kamat. "DPT restricted design rules for advanced logic applications." *SPIE Advanced Lithography*, pp. 79730H–79730H, 2011.

[DSB10] Hoda A Darwish, Hoda N Shagar, Yasmine A Badr, Yasmine H Arafa, and Amr G Wassal. "A hashing mechanism for rule-based decomposition in Double Patterning Photolithography." In *2010 International Conference on Microelectronics*, pp. 363–366. IEEE, 2010.

[DTP11] Duo Ding, Andres Torres, and David Z Pan. "High performance lithography hotspot detection with successively refined pattern identifications and machine learning." *TCAD*, 2011.

[DWG09] Duo Ding, Xiang Wu, J. Ghosh, and D.Z. Pan. "Machine learning based lithographic hotspot detection with critical-feature extraction and classification." *IEEE International Conference on IC Design and Technology (ICIDT)*, 2009.

[DXW14] Yuelin Du, Zigang Xiao, Martin DF Wong, He Yi, and H-S Philip Wong. "DSA-aware detailed routing for via layer optimization." In *SPIE Advanced Lithography*, pp. 90492J–90492J, 2014.

[DYG12] Duo Ding, Bei Yu, Joydeep Ghosh, and David Z Pan. "EPIC: Efficient prediction of ic manufacturing hotspots with a unified meta-classification formulation." *Asia and South Pacific Design Automation Conference (ASPDAC)*, 2012.

[DYR07] Vito Dai, Jie Yang, Norma Rodriguez, and Luigi Capodieci. "DRC Plus: augmenting standard DRC with pattern matching on 2D geometries." In *SPIE Advanced Lithography*, 2007.

[Edm65] Jack Edmonds. "Paths, trees, and flowers." *Canadian Journal of mathematics*, **17**(3):449–467, 1965.

[Elm48] William C Elmore. "The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers." *Journal of Applied Physics*, **19**(1):55–63, 1948.

[FCC12] Shao-Yun Fang, Szu-Yu Chen, and Yao-Wen Chang. "Native-conflict and stitch-aware wire perturbation for double patterning technology." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **31**(5):703–716, 2012.

[FHL15] Shao-Yun Fang, Yun-Xiang Hong, and Yi-Zhen Lu. "Simultaneous Guiding Template Optimization and Redundant Via Insertion for Directed Self-Assembly." In *ICCAD*, pp. 410–417. IEEE Press, 2015.

[FM88] Charles M Fiduccia and Robert M Mattheyses. "A linear-time heuristic for improving network partitions." In *Papers on Twenty-five years of electronic design automation*, pp. 241–247. ACM, 1988.

[FNA99] Leonard Forbes, Wendell P Noble, and Kie Y Ahn. "Method of making memory cell with vertical transistor and buried word and body lines.", June 1 1999. US Patent 5,909,618.

[Fre] "FreePDK." http://www.eda.ncsu.edu/wiki/FreePDK.

[GBD16] Roel Gronheid, Carolien Boeckx, Jan Doise, Joost Bekaert, Ioannis Karageorgos, Julien Ruckaert, Boon Teik Chan, Chenxi Lin, and Yi Zou. "EUV patterned templates with grapho-epitaxy DSA at the N5/N7 logic nodes." In *SPIE Advanced Lithography*, pp. 97761W–97761W. SPIE, 2016.

[GBG14] Rani S Ghaida, Yasmine Badr, Mukul Gupta, Ning Jin, and Puneet Gupta. "Comprehensive die-level assessment of design rules and layouts." In *ASP-DAC*, pp. 61–66, 2014.

[GG12] Rani S Ghaida and Puneet Gupta. "DRE: A framework for early co-evaluation of design rules, technology choices, and layout methodologies." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **31**(9):1379–1392, 2012.

[GG13] Rani S Ghaida and Puneet Gupta. "Role of design in multiple patterning: technology development, design enablement and process control." In *Proceedings of the Conference on Design, Automation and Test in Europe*, pp. 314–319. EDA Consortium, 2013.

[GM06] Bahram Azizollah Ganji and Burhanuddin Yeop Majlis. “Deep trenches in silicon structure using DRIE method with aluminum as an etching mask.” In *Semiconductor Electronics, 2006. ICSE’06. IEEE International Conference on*, pp. 41–47. IEEE, 2006.

[GM10] Xin Gao and Luca Macchiarulo. “Enhancing double-patterning detailed routing with lazy coloring and within-path conflict avoidance.” *Design, Automation and Test in Europe (DATE)*, pp. 1279–1284, 2010.

[GMM09] Justin Ghan, Ning Ma, Sandipan Mishra, Costas Spanos, Kameshwar Poolla, Norma Rodriguez, and Luigi Capodieci. “Clustering and pattern matching for an automatic hotspot classification and detection system.” *SPIE Advanced Lithography*, 2009.

[GMR97] Wang Ling Goh, H Montgomery, SH Raza, BM Armstrong, and HS Gamble. “Electrical characterization of dielectrically isolated silicon substrates containing buried metallic layers.” *IEEE Electron Device Letters*, **18**(5):232–234, 1997.

[GP12] Jhih-Rong Gao and David Z Pan. “Flexible self-aligned double patterning aware detailed routing with prescribed layout planning.” *International Symposium on Physical Design (ISPD)*, pp. 25–32, 2012.

[GSK12] Rani S Ghaida, Tanaya Sahu, Parag Kulkarni, and Puneet Gupta. “A methodology for the early exploration of design rules for multiple-patterning technologies.” *International Conference on Computer-Aided Design*, **12**:50–56, 2012.

[GTA14] A Gharbi, R Tiron, M Argoud, X Chevalier, J Belledent, J Pradelles, P Pimenta Barros, C Navarro, C Nicolet, G Hadziioannou, et al. “Contact holes patterning by directed self-assembly of block copolymers: What would be the Bossung plot?” In *SPIE Advanced Lithography*, pp. 90491N–90491N, 2014.

[HCT97] Fook-Luen Heng, Zhan Chen, and Gustavo E Tellez. “A VLSI artwork legalization technique based on a new criterion of minimum layout perturbation.” In *Proceedings of the 1997 international symposium on Physical design*, pp. 116–121. ACM, 1997.

[HKL15] Kwangsoo Han, Andrew B Kahng, and Hyein Lee. “Evaluation of BEOL design rule impacts using an optimal ILP-based detailed router.” In *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*, pp. 1–6. IEEE, 2015.

[hof] “UCLA Hoffman2 Cluster.”.

[Hoo12] Terence B Hook. “Fully depleted devices for designers: FDSOI and FinFETs.” In *Custom Integrated Circuits Conference (CICC), 2012 IEEE*, pp. 1–7. IEEE, 2012.

[HSP12] “Synopsys HSPICE.”, 2012.

[HT73] John Hopcroft and Robert Tarjan. “Algorithm 447: Efficient Algorithms for Graph Manipulation.” *Commun. ACM*, **16**(6):372–378, June 1973.

[IBM] IBM. “CPLEX.”.

[IWP16] Motoi Ichihashi, Youngtag Woo, and Sanjay Parihar. “SRAM cell performance analysis beyond 10-nm FinFET technology.” In *VLSI Technology, Systems and Application (VLSI-TSA), 2016 International Symposium on*, pp. 1–2. IEEE, 2016.

[Jan10] Dirk Jansen. *The electronic design automation handbook*. Springer Science & Business Media, 2010.

[Jar13] Nathan Jarnagin. *High X Block Copolymers For Sub 20 Nm Pitch Patterning: Synthesis, Solvent Annealing, Directed Self Assembly, And Selective Block Removal*. PhD thesis, Geogia Institute of Technology, December 2013.

[JCS08] Vivek Joshi, Brian Cline, Dennis Sylvester, David Blaauw, and Kanak Agarwal. “Stress aware layout optimization.” In *Proceedings of the 2008 international symposium on Physical design*, pp. 168–174. ACM, 2008.

[JHJ12] Daehyun Jang, Naya Ha, Junsu Jeon, Jae-Hyun Kang, Seung Weon Paek, Hungbok Choi, Kee Sup Kim, Ya-Chieh Lai, Philippe Hurat, and Wilbur Luo. “In-design process hotspot repair using pattern matching.” *SPIE Advanced Lithography*, pp. 83270S–83270S, 2012.

[JKS09] Kwangok Jeong, A.B. Kahng, and K. Samadi. “Impact of Guardband Reduction On Design Outcomes: A Quantitative Approach.” **22**(4):552 –565, November 2009.

[Joh74] David S. Johnson. “Approximation algorithms for combinatorial problems.” *Journal of Computer and System Sciences*, **9**(3):256–278, dec 1974.

[JPR06] Tejas Jhaveri, Larry Pileggi, Vyacheslav Rovner, and Andrzej J Strojwas. “Maximization of layout printability/manufacturability by extreme layout regularity.” In *SPIE 31st International Symposium on Advanced Lithography*, pp. 615609–615609. International Society for Optics and Photonics, 2006.

[JRL10] Tejas Jhaveri, Vyacheslav Rovner, Lars Liebmann, Larry Pileggi, Andrzej J Strojwas, and Jason D Hibbeler. “Co-optimization of circuits, layout and lithography for predictive technology scaling beyond gratings.” *TCAD*, **29**(4):509–527, 2010.

[Kar72] Richard M Karp. “Reducibility among combinatorial problems.” In *Complexity of computer computations*, pp. 85–103. Springer, 1972.

[KBY09] H Kawasaki, VS Basker, T Yamashita, C-H Lin, Y Zhu, J Faltermeier, S Schmitz, J Cummings, S Kanakasabapathy, H Adhikari, et al. “Challenges and solutions of FinFET integration in an SRAM cell and a logic circuit for 22 nm node and beyond.” In *Electron Devices Meeting (IEDM), 2009 IEEE International*, pp. 1–4. IEEE, 2009.

[KCH11] Huiman Kang, Gordon SW Craig, Eungnak Han, Padma Gopalan, and Paul F Nealey. “Degree of perfection and pattern uniformity in the directed assembly of cylinder-forming block copolymer on chemically patterned surfaces.” *Macromolecules*, **45**(1):159–164, 2011.

[KHS13] Myungwoong Kim, Eungnak Han, Daniel P Sweat, and Padma Gopalan. "Interplay of surface chemical composition and film thickness on graphoepitaxial assembly of asymmetric block copolymers." *Soft Matter*, **9**(26):6135–6141, 2013.

[KIN13] Chikaaki Kodama, Hirotaka Ichikawa, Koichi Nakayama, Toshiya Kotani, Shigeki Nojima, Shoji Mimotogi, Shinji Miyamoto, and Atsushi Takahashi. "Self-Aligned Double and Quadruple Patterning-aware grid routing with hotspots control." *Asia and South Pacific Design Automation Conference (ASPDAC)*, pp. 267–272, 2013.

[KLH15] Marie Krysak, Michael Leeson, Eungnak Han, James Blackwell, and Shane Harlson. "Extending resolution limits of EUV resist materials." In *SPIE Advanced Lithography*, pp. 942205–942205. International Society for Optics and Photonics, 2015.

[KM02] Andrew B Kahng and Stefanus Mantik. "Measurement of inherent noise in EDA tools." In *Quality Electronic Design, 2002. Proceedings. International Symposium on*, pp. 206–211. IEEE, 2002.

[KPX08a] Andrew B Kahng, Chul-Hong Park, and Xu Xu. "Fast dual-graph-based hotspot filtering." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **27**(9):1635–1642, 2008.

[KPX08b] Andrew B Kahng, Chul-Hong Park, Xu Xu, and Hailong Yao. "Layout decomposition for double patterning lithography." In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*, pp. 465–472. IEEE Press, 2008.

[KRG16] Ioannis Karageorgos, Julien Ryckaert, Roel Gronheid, Maryann C Tung, H-S Philip Wong, Evangelos Karageorgos, Kris Croes, Joost Bekaert, Geert Vandenberghe, Michele Stucchi, et al. "Design method and algorithms for directed self-assembly aware via layout decomposition in sub-7 nm circuits." *Journal of Micro/Nanolithography, MEMS, and MOEMS*, **15**(4):043506–043506, 2016.

[KRT16] Ioannis Karageorgos, Julien Ryckaert, Maryann C. Tung, H. S. P. Wong, Roel Gronheid, Joost Bekaert, Evangelos Karageorgos, Kris Croes, Geert Vandenberghe, Michele Stucchi, and Wim Dehaene. "Design strategy for integrating DSA via patterning in sub-7 nm interconnects." In Luigi Capodieci and Jason P. Cain, editors, *SPIE Advanced Lithography*, p. 97810N. SPIE, mar 2016.

[KWH14] Patrick A. Kearney, Obert Wood, Eric Hendrickx, Greg McIntyre, Soichi Inoue, Frank Goodwin, Stefan Wurm, Jan van Schoot, and Winfried Kaiser. "Driving the industry towards a consensus on high numerical aperture (high-NA) extreme ultraviolet (EUV)." In *SPIE Proceedings Extreme Ultraviolet (EUV) Lithography V*, volume 9048, 2014.

[KY16] Jian Kuang and Evangeline F.Y. Young. "Simultaneous template optimization and mask assignment for DSA with multiple patterning." In *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 75–82. IEEE, jan 2016.

[KYY16] Jian Kuang, Junjie Ye, and Evangeline FY Young. "Simultaneous template optimization and mask assignment for DSA with multiple patterning." In *Design Automation Conference (ASP-DAC), 2016 21st Asia and South Pacific*, pp. 75–82. IEEE, 2016.

[Lab] NanoCAD Lab. "NanoCAD Lab Downloads page, EE, UCLA.".

[Las86] JB Lasky. "Wafer bonding for silicon-on-insulator technologies." *Applied Physics Letters*, **48**(1):78–80, 1986.

[LC16a] Zhi-Wen Lin and Yao-Wen Chang. "Cut redistribution with directed self-assembly templates for advanced 1-D gridded layouts." In *2016 21st ASP-DAC*, pp. 89–94. IEEE, jan 2016.

[LC16b] Zhi-Wen Lin and Yao-Wen Chang. "Double-Patterning Aware DSA Template Guided Cut Redistribution for Advanced 1-D Gridded Designs." In *Proceedings of ISPD '16*, pp. 47–54, New York, New York, USA, apr 2016. ACM Press.

[LCG15] Lars Liebmann, Albert Chu, and Paul Gutwin. "The daunting complexity of scaling to 7nm without EUV: Pushing DTCO to the extreme." In *SPIE Advanced Lithography*, pp. 942702–942702. International Society for Optics and Photonics, 2015.

[LGN14] CH Lin, B Greene, S Narasimha, J Cai, A Bryant, C Radens, V Narayanan, B Linder, H Ho, A Aiyar, et al. "High performance 14nm SOI FinFET CMOS technology with 0.0174 μm 2 embedded DRAM and 15 levels of Cu metallization." In *Electron Devices Meeting (IEDM), 2014 IEEE International*, pp. 3–8. IEEE, 2014.

[LMP12] Gerard Luk-Pat, Alex Miloslavsky, Ben Painter, Li Lin, PD Bisschop, and Kevin Lucas. "Design compliance for spacer is dielectric (SID) patterning." In *Proc. SPIE*, volume 8326, p. 83260D, 2012.

[LPG11] Lars Liebmann, David Pietromonaco, and Matthew Graf. "Decomposition-aware standard cell design flows to enable double-patterning technology." In *SPIE Advanced Lithography*, pp. 79740K–79740K. International Society for Optics and Photonics, 2011.

[LW08] Tsung-Hsien Lee and Ting-Chi Wang. "Congestion-constrained layer assignment for via minimization in global routing." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **27**(9):1643–1656, 2008.

[LW15] Ning Lu and Richard A Wachnik. "Modeling of resistance in FinFET local interconnect." *IEEE Transactions on Circuits and Systems I: Regular Papers*, **62**(8):1899–1907, 2015.

[McG] Dylan McGrath. "Globalfoundries looks to leapfrog fab rivals." http://www.eetimes.com/.

[MGC88] WP Maszara, Goetz Goetz, A Caviglia, and JB McKitterick. "Bonding of silicon wafers for silicon-on-insulator." *Journal of Applied Physics*, **64**(10):4943–4950, 1988.

[MGM08] Ning Ma, Justin Ghan, Sandipan Mishra, Costas Spanos, Kameshwar Poolla, Norma Rodriguez, and Luigi Capodieci. "Automatic hotspot classification using pattern-based clustering." *SPIE Advanced Lithography*, 2008.

[MLT15] Yuansheng Ma, Junjiang Lei, Andres Torres, Le Hong, James Word, Germain Fenger, Alexander Tritchkov, George Lippincott, Rachit Gupta, Neal Lafferty, et al. "Directed self-assembly graphoepitaxy template generation with immersion lithography." *JM3*, **14**(3):031216–031216, 2015.

[MMJ11] Prateek Mishra, Anish Muttreja, and Niraj K Jha. "Finfet circuit design." In *Nanoelectronic Circuit Design*, pp. 23–54. Springer, 2011.

[MTF14] Yuansheng Ma, Andres Torres, Germain Fenger, Yuri Granik, Julien Ryckaert, Geert Vanderberghe, Joost Bekaert, and James Word. "Challenges and opportunities in applying grapho-epitaxy DSA lithography to metal cut and contact/via applications." In *European Mask and Lithography Conference*, pp. 92310T–92310T. SPIE, 2014.

[MWW16] Yuansheng Ma, Yan Wang, James Word, Junjiang Lei, Joydeep Mitra, Andres Torres, Le Hong, Germain Fenger, Daman Khaira, Moshe Preil, Lei Yuan, Jongwook Kye, and Harry J. Levinson. "Directed Self Assembly (DSA) compliant flow with immersion lithography: from material to design and patterning." In *SPIE Advanced Lithography*, p. 97770N. SPIE, mar 2016.

[MYP05] Joydeep Mitra, Peng Yu, and David Z Pan. "RADAR: RET-aware detailed routing using fast lithography simulations." *Design Automation Conference (DAC)*, pp. 369–372, 2005.

[MZW12] Qiang Ma, Hongbo Zhang, and Martin DF Wong. "Triple patterning aware routing and its comparison with double patterning aware routing in 14nm technology." *Design Automation Conference (DAC)*, pp. 591–596, 2012.

[Nana] "Nangate Open Cell Library v1.3. 2009." http://www.si2.org/openeda.si2.org/projects/nangatelib.

[Nanb] *Nangate open cell library v1.3. 2009 http://www.si2.org/openeda.si2.org/projects/nangatelib*.

[NCX] "Synopsys Liberty NCX.".

[NDL94] ED Nowak, L Ding, YT Loh, and C Hu. "Speed, power, and yield comparison of thin bonded SOI versus bulk CMOS technologies." In *SOI Conference, 1994 Proceedings., 1994 IEEE International*, pp. 41–42. IEEE, 1994.

[Nie11] Tim Nieberg. "Gridless pin access in detailed routing." In *Proceedings of the 48th Design Automation Conference*, pp. 170–175. ACM, 2011.

[Nor11] Greg Northrop. "Design technology co-optimization in technology definition for 22nm and beyond." In *VLSI Technology (VLSIT), 2011 Symposium on*, pp. 112–113. IEEE, 2011.

[OCY17] Jiaojiao Ou, Brian Cline, Greg Yeric, and David Z Pan. "Efficient DSA-DP Hybrid Lithography Conflict Detection and Guiding Template Assignment." In *SPIE Advanced Lithography*, pp. 101480C–101480C. International Society for Optics and Photonics, 2017.

[OLD16] Gaddiel Ouaknin, Nabil Laachi, Kris Delaney, Glenn Fredrickson, and Frederic Gibou. "Shape optimization for DSA." In Christopher Bencher and Joy Y. Cheng, editors, *SPIE Advanced Lithography*, p. 97770Y. SPIE, mar 2016.

[ope] "http://www.opencores.org.".

[OYP16] Jiaojiao Ou, Bei Yu, and David Z. Pan. "Concurrent Guiding Template Assignment and Redundant via Insertion for DSA-MP Hybrid Lithography." In *Proceedings of the 2016 on International Symposium on Physical Design - ISPD '16*, pp. 39–46, New York, New York, USA, apr 2016. ACM Press.

[OYX17] Jiaojiao Ou, Bei Yu, Xiaoqing Xu, Joydeep Mitra, Yibo Lin, and David Z Pan. "DSAR: DSA aware Routing with Simultaneous DSA Guiding Pattern and Double Patterning Assignment." In *ISPD*, pp. 91–98, 2017.

[PLY15] David Z Pan, Lars Liebmann, Bei Yu, Xiaoqing Xu, and Yibo Lin. "Pushing multiple patterning in sub-10nm: are we ready?" In *Proceedings of the 52nd Annual Design Automation Conference*, p. 197. ACM, 2015.

[RCN04] Jan M. Rabaey, Anantha Chandrakasan, and Borivoje Nikolic. *Digital integrated circuits- A design perspective*. Prentice Hall, 2nd edition, 2004.

[RCP] "Cadence RTL Compiler Advanced Physical Option.".

[RKD08] Ricardo Ruiz, Huiman Kang, François A Detcheverry, Elizabeth Dobisz, Dan S Kercher, Thomas R Albrecht, Juan J de Pablo, and Paul F Nealey. "Density multiplication and improved lithography by directed block copolymer assembly." *Science*, **321**(5891):936–939, 2008.

[Sai95] Geroge A Sai-Halasz. "Performance trends in high-end processors." *Proceedings of the IEEE*, **83**(1):20–36, 1995.

[SCS15] Seongbo Shim, Woohyun Chung, and Youngsoo Shin. "Defect Probability of Directed Self-Assembly Lithography: Fast Identification and Post-Placement Optimization." In *ICCAD*. IEEE Press, 2015.

[SDS04] Deepak Sundrani, SB Darling, and SJ Sibener. "Hierarchical assembly and compliance of aligned nanoscale polymer cylinders in confinement." *Langmuir*, **20**(12):5091–5099, 2004.

[SIB16] Jan van Schoot, Koen van Ingen Schenau, Gerardo Bottiglieri, Kars Troost, John D Zimmerman, Sascha Migura, Bernhard Kneer, Jens Timo Neumann, and Winfried Kaiser. "EUV high-NA scanner and mask optimization for sub-8nm resolution." In *SPIE Advanced Lithography*, pp. 97761I–97761I. International Society for Optics and Photonics, 2016.

[Sou81] Jiri Soukup. "Circuit layout." *Proceedings of the IEEE*, **69**(10):1281–1304, 1981.

[SPM11] Z Sanaee, M Poudineh, M Mehran, and S Mohajerzadeh. "High-aspect-ratio deep Si etching of micro/nano scale features with SF6/H2/O2 plasma, in a low plasma density reactive ion etching system." pp. 325–328, 2011.

[STD17] Daohang Shi, Edward Tashjian, and Azadeh Davoodi. "Dynamic planning of local congestion from varying-size vias for global routing layer assignment." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017.

[SYS13] Yuriko Seino, Hiroki Yonemitsu, Hironobu Sato, Masahiro Kanno, Hirokazu Kato, Katsutoshi Kobayashi, Ayako Kawanishi, Tsukasa Azuma, Makoto Muramatsu, Seiji Nagahara, et al. "Contact hole shrink process using graphoepitaxial directed self-assembly lithography." *JM3*, **12**(3), 2013.

[TB05] Juan A Torres and C Neil Berglund. "Integrated circuit DFM framework for deep sub-wavelength processes." In *Microlithography 2005*, pp. 39–50. International Society for Optics and Photonics, 2005.

[TLA10] Taraneh Taghavi, Zhuo Li, Charles Alpert, Gi-Joon Nam, Andrew Huber, and Shyam Ramji. "New placement prediction and mitigation techniques for local routing congestion." In *Computer-Aided Design (ICCAD), 2010 IEEE/ACM International Conference on*, pp. 621–624. IEEE, 2010.

[TZX14] Haitong Tian, Hongbo Zhang, Zigang Xiao, and Martin DF Wong. "Hybrid lithography for triple patterning decomposition and e-beam lithography." In *SPIE Advanced Lithography*, pp. 90520P–90520P. International Society for Optics and Photonics, 2014.

[Wan94] Shi-Qing Wang. "Barriers against copper diffusion into silicon and drift through silicon dioxide." *MRS bulletin*, **19**(8):30–40, 1994.

[WLZ14] Wei-Long Wang, Azat Latypov, Yi Zou, and Tamer Coskun. "A full-chip DSA correction framework." In *SPIE Advanced Lithography*, pp. 90491J–90491J. SPIE, 2014.

[WPC16] Shaodi Wang, Andrew Pan, Chi On Chui, and Puneet Gupta. "PROCEED: A Pareto optimization-based circuit-level evaluator for emerging devices." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, **24**(1):192–205, 2016.

[WPM11] Jen-Yi Wuu, Fedor G Pikus, and Malgorzata Marek-Sadowska. "Efficient approach to early detection of lithographic hotspots using machine learning systems and pattern matching." *SPIE Advanced Lithography*, 2011.

[WRG13] Ling-Shu Wan, Paulina A. Rincon Delgadillo, Roel Gronheid, and Paul F. Nealey. "Directed self-assembly of ternary blends of block copolymer and homopolymers on chemical patterns." *Journal of Vacuum Science Technology B: Microelectronics and Nanometer Structures*, **31**(6):06F301–06F301–6, November 2013.

[XCY15] Xiaoqing Xu, Brian Cline, Greg Yeric, Bei Yu, and David Z Pan. "Self-aligned double patterning aware pin access and standard cell layout co-optimization." *TCAD*, **34**(5):699–712, 2015.

[XDW15] Zigang Xiao, Yuelin Du, Martin DF Wong, He Yi, HS Wong, and Hongbo Zhang. "Contact pitch and location prediction for Directed Self-Assembly template verification." In *ASPDAC*, pp. 644–651. IEEE, 2015.

[XDZ13] Zigang Xiao, Yuelin Du, Hongbo Zhang, and Martin DF Wong. "A polynomial time exact algorithm for overlay-resistant self-aligned double patterning (sadp) layout decomposition." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **32**(8):1228–1239, 2013.

[XK89] X-M Xiong and Ernest S Kuh. "A unified approach to the via minimization problem." *IEEE Transactions on Circuits and Systems*, **36**(2):190–204, 1989.

[XLW16] Zigang Xiao, Chun-Xun Lin, Martin D.F. Wong, and Hongbo Zhang. "Contact layer decomposition to enable DSA with multi-patterning technique for standard cell based layout." In *2016 ASPDAC*, pp. 95–102. IEEE, jan 2016.

[XPM15] Qing Xu, Alex Paterson, Jon McChesney, Russell Dover, Yoko Yamaguchi, and Aaron Eppler. "Enhanced etch process for TSV & deep silicon etch." In *Advanced Semiconductor Manufacturing Conference (ASMC), 2015 26th Annual SEMI*, pp. 426–428. IEEE, 2015.

[XPN09] Bing Xu, Rafael Piñol, Merveille Nono-Djamen, Sandrine Pensec, Patrick Keller, Pierre-Antoine Albouy, Daniel Lévy, and Min-Hui Li. "Self-assembly of liquid crystal block copolymer PEG-b-smectic polymer in pure state and in dilute aqueous solution." *Faraday discussions*, **143**, 2009.

[XYG16] Xiaoqing Xu, Bei Yu, Jhih-Rong Gao, Che-Lun Hsu, and David Z Pan. "PARR: Pin-Access Planning and Regular Routing for Self-Aligned Double Patterning." *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, **21**(3):42, 2016.

[XZC09] Yue Xu, Yanheng Zhang, and Chris Chu. "FastRoute 4.0: global router with efficient via minimization." In *Proceedings of the 2009 Asia and South Pacific Design Automation Conference*, pp. 576–581. IEEE Press, 2009.

[YBT13] He Yi, Xin-Yu Bao, Richard Tiberio, and H.-S. Philip Wong. "Design strategy of small topographical guiding templates for sub-15nm integrated circuits contact hole patterns using block copolymer directed self assembly." volume 8680, 2013.

[YBT15] He Yi, Xin-Yu Bao, Richard Tiberio, and H-S Philip Wong. "A General Design Strategy for Block Copolymer Directed Self-Assembly Patterning of Integrated Circuits Contact Holes using an Alphabet Approach." *Nano letters*, **15**(2):805–812, 2015.

[YCS13] Greg Yeric, Brian Cline, Saurabh Sinha, David Pietromonaco, Vishal Chandra, and Robert Aitken. "The past present and future of design-technology co-optimization." In *Custom Integrated Circuits Conference (CICC), 2013 IEEE*, pp. 1–8. IEEE, 2013.

[YLJ13] Yen-Ting Yu, Geng-He Lin, Iris Hui-Ru Jiang, and Charles Chiang. "Machine-learning-based hotspot detection using topological classification and critical feature extraction." *Design Automation Conference (DAC)*, 2013.

[YLP09] Kun Yuan, Katrina Lu, and David Z Pan. "Double patterning lithography friendly detailed routing with redundant via consideration." *Design Automation Conference (DAC)*, pp. 63–66, 2009.

[YLZ15] Yunfeng Yang, Wai-Shing Luk, Hai Zhou, Changhao Yan, Xuan Zeng, and Dian Zhou. "Layout decomposition co-optimization for hybrid e-beam and multiple patterning lithography." In *The 20th Asia and South Pacific Design Automation Conference*, pp. 652–657. IEEE, 2015.

[YYZ11] Bei Yu, Kun Yuan, Boyang Zhang, Duo Ding, and D.Z. Pan. "Layout decomposition for triple patterning lithography." In *Proc. ICCAD*, 2011.

[ZBW17] Liheng Zhu, Yasmine Badr, Shaodi Wang, Subramanian Iyer, and Puneet Gupta. "Assessing Benefits of a Buried Interconnect Layer in Digital Designs." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **36**(2):346–350, 2017.

[ZCY08] Yunqiang Zhang, Jonathan Cobb, Amy Yang, Ji Li, Kevin Lucas, and Satyendra Sethi. "32nm design rule and process exploration flow." In *Proc. SPIE*, volume 7122, p. 71223Z, 2008.

[ZDW11] Hongbo Zhang, Yuelin Du, Martin DF Wong, and Rasit Topaloglu. "Self-aligned double patterning decomposition for overlay minimization and hot spot detection." In *Design Automation Conference (DAC), 2011 48th ACM/EDAC/IEEE*, pp. 71–76. IEEE, 2011.

[ZLW05] Shawn Xiaodong Zhang, Shi-Wei Ricky Lee, LT Weng, and Sylvia So. "Characterization of copper-to-silicon diffusion for the application of 3D packaging with through silicon vias." In *Electronic Packaging Technology, 2005 6th International Conference on*, pp. 51–56. IEEE, 2005.

[ZYC14] Wei Zhao, Hailong Yao, Yici Cai, Subarna Sinha, and Charles Chiang. "Fast and scalable parallel layout decomposition in double patterning lithography." *Integration, the VLSI Journal*, **47**(2):175–183, 2014.

[ZYY16] Hui Zang, Chun-Chen Yeh, Tenko Yamashita, and Veeraraghavan Basker. "Buried local interconnect in finfet structure and method of fabricating same.", April 26 2016. US Patent 9,324,842.