Physical Design Automation for System-on-Packages and 3D-Integrated Circuits

A Thesis
Presented to
The Academic Faculty

by

Jacob Rajkumar Minz

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2006

Physical Design Automation for System-on-Packages and 3D-Integrated Circuits

Approved by:

Asst. Professor Sung Kyu Lim, Advisor
School of Electrical and
Computer Engineering
Georgia Institute of Technology, Adviser

Professor Madavan Swaminathan
School of Electrical and
Computer Engineering
Georgia Institute of Technology

Professor Abhijit Chatterjee
School of Electrical and
Computer Engineering
Georgia Institute of Technology

Professor Gabriel Rincon-Mora
School of Electrical and
Computer Engineering
Georgia Institute of Technology

Asst. Professor Gabriel H. Loh
College of Computing
Georgia Institute of Technology

Date Approved: 20th July 2006

Dedicated to my mother, Mary Ignatia Minz

and my late father, Anthony Minz

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Prof. Sung Kyu Lim for his support and guidance during my PhD at Georgia Tech. His support has been crucial for the successful completion of this research and thesis. I also thank the members of my reading committee Prof. Madhavan Swaminathan, and Prof. Abhijit Chatterjee, and other members of my dissertation committee, Prof. Gabriel Rincon-Mora, and Prof. Gabriel Loh for their time and helpful feedbacks. Prof. Partha Pratim Chakrabarti, and Prof. Dipwanita Raychaudhuri from Indian Institute of Technology, Kharagpur have been very instrumental and influential in my decision to pursue PhD. They will always have my respect and appreciation for being excellent teachers. I would also like to remember late Prof. John Uyemura who encouraged me to pursue PhD at Georgia Tech.

I would like to extend my heartfelt thanks to the members (past and present) of the GTCAD group: Mongkol Ekpanyapong, Faik Baskaya, Michael Healy, Somas Thyagaraja, Eric Wong, and Mohit Pathak for their friendship, collaboration, and encouragement. I would also like to express my thanks to Pinar Korkmaz, who has been very forthcoming with any help I needed during my PhD career. Thanks are due in no small part to all my friends in Atlanta, and elsewhere, some of whom I have known from the days of my undergrad in IIT Kharagpur. However, I would like to extend special thanks to Mrinmoy Ghosh, Atri Dutta, Bevin Perumana, and Biswajit Mitra, who have been extremely helpful, and generous with their advices. A hearty thanks is also due to Rachana Mukhopadhyay, who proofread parts of my thesis.

My family has always been supportive, but my mother has been the greatest source of inspiration for me. Surely, not sparing the rod on a certain child has resulted in this dissertation. I would not have been where I am today, if not for her constant prayers, and steadfast belief in the God. And for all the endowments that I have had, I offer my thanks to the God Almighty.

TABLE OF CONTENTS

DEDICATION . . . . **iii**

ACKNOWLEDGEMENTS . . . . **iv**

LIST OF TABLES . . . . **viii**

LIST OF FIGURES . . . . **x**

SUMMARY . . . . **xiii**

I INTRODUCTION **1**

1.1 Thesis Statement 4

1.2 Problem Statement 5

1.3 Contributions 5

1.4 Thesis Organization 7

II ORIGIN AND HISTORY OF THE PROBLEM **9**

2.1 Background 9

2.2 Global Routing 11

2.3 Optical Routing 12

2.4 Noise and Congestion Aware Floorplanning 13

2.5 Clock Distribution 14

III PERFORMANCE-DRIVEN GLOBAL ROUTING **15**

3.1 MCM vs SOP Routing 15

3.2 Problem Formulation 17

3.2.1 SOP Routing Resource 17

3.2.2 SOP Routing Problem 18

3.3 Overview of 3PGR Algorithm 20

3.3.1 SOP Pin Redistribution 22

3.3.2 Coarse Pin Distribution 24

3.3.3 Detailed Pin Distribution 26

3.3.4 SOP Layer Assignment 29

3.3.5 SOP Channel Assignment 31

3.4 Experimental Results 34

3.5 Conclusions 39

IV 3D OPTICAL ROUTING 41

4.1 SOP Optical Interconnect Technology 41

4.2 Problem Formulation 43

4.3 SOP Optical Routing Algorithm 45

4.3.1 Overview of the Algorithm 45

4.3.2 Waveguide Construction Algorithm 46

4.3.3 Optical Net Selection Algorithm 51

4.3.4 Optical Routing 51

4.4 Experimental Results 53

4.5 Conclusions 55

V CONGESTION AND NOISE-DRIVEN FLOORPLANNING 56

5.1 Problem Formulation 56

5.1.1 SOP Placement Problem 56

5.2 3D Power Supply Noise Modeling 57

5.3 3D Decoupling Capacitor Placement 59

5.4 3D Wire Congestion Computation 61

5.5 Overview of the Algorithm 63

5.6 Experimental Results 64

5.7 Conclusions 67

VI 3D CLOCK ROUTING 69

6.1 Preliminaries 69

6.1.1 Temperature Dependent Delay Model 69

6.1.2 Recent Work 71

6.1.3 Challenges 72

6.2 Problem Formulation 73

6.3 3D Clock Routing Algorithm 75

6.3.1 Overview of the Algorithm 75

6.3.2 3D Abstract Tree Generation 75

6.3.3 Clock Tree Embedding and Buffering 77

6.3.4 Thermal-aware Clock Tree Optimization 78
6.3.5 Balanced Skew Calculations 79
6.3.6 Bottom-up Options Generation 83
6.3.7 Solution Pruning 84
6.3.8 Top-down Selection 85
6.4 Experimental Results 85
6.4.1 Comparison to Existing Work 85
6.4.2 2D Clock Routing Results 86
6.4.3 3D Abstract Tree Results 86
6.4.4 3D Clock Routing Results 87
6.5 Conclusions 88

VII CONCLUSIONS 90

REFERENCES 92

PUBLICATIONS 99

VITA 101

LIST OF TABLES

1 Evolution of the Physical Design Research 9

2 Type of nets and the layer usage for a 3D SOP with K placement layers. The layer information is denoted in parenthesis. 18

3 Benchmark characteristics of GSRC and GT circuits. wcap denotes the wire capacity of each boundary in the routing tiles. CP and DP respectively denote the dimension of 2D grids used during our coarse and detailed pin distribution steps. 35

4 Area information of the four-layer SOP floorplanning for GSRC and GT benchmark designs. The min and max respectively denote the dimension of the minimum and maximum floorplan area, whereas the final column denotes the dimension of the overall footprint. 36

5 SOP pin redistribution results using our coarse and detailed pin distribution algorithms. We report the wirelength between the original and the new location (dw), total wirelength (wl), crosstalk (xt) and total number of layer pairs used (lyr) in the routing layers. 37

6 Topology generation and layer assignment results. We report the total wirelength (wl), Elmore delay of the nets with maximum sink delay (dly), and the lower bound (low) and the actual number (lyr) of layers used. 38

7 SOP channel assignment results. We report the layer usage, wirelength, and via for the baseline (wirelength minimization only) and multi-objective algorithms. 39

8 SOP local routing results for the baseline (wirelength minimization only) and multi-objective algorithms. We report the wirelength (wl), maximum (max) and average (ave) routing demand as well as the standard deviation (dev). . 40

9 Standard vs. customized waveguides 53

10 Comparison between Electrical, Congestion-driven, Standard Waveguides and Timing-driven Optical Routing with no violations. We report the total wirelength, routing layer (XY) usage, and maximum net delay (ns). . . . 53

11 Timing-driven Optical Routing with different Criticality Levels. We report the total wirelength, routing layer (XY) usage, maximum net delay (ns), and maximum violations. 54

12 area/wire-driven vs decap-driven algorithms 66

13 area/wire-driven vs congestion-driven algorithms 67

14 area/wire-driven vs area/wire/decap/congestion-driven algorithms 68

15 Power supply noise simulation results. We report SSN noise for each block placed in the top placement layer. (a) Non-optimized case without any decoupling capacitor. From Tables 12 we need 26.7 decap units to suppress SSN noise. (b) Optimized case without any decoupling capacitor. From Table 12 we need 19.9 decap units. (c) Optimized case with decoupling capacitors inserted. 68

16 Decap *vs* Congestion correlation constants 68

17 Comparison with TACO [25] for 2D non-buffered clock tree optimization. Both algorithms use the initial clock trees generated by BST-DME [1]. The delay values are in ns, skews in ps, and wirelength in μm. 87

18 2D clock routing results. The delay values are in ns, skews in ps, and wirelength in μm. 88

19 Results for the 3D Abstract tree Generation 88

20 3D clock routing results. The delay values are in ns, skews in ps, and wirelength in μm. 89

LIST OF FIGURES

1 Illustration of 3D System-On-Package, where A, D, M, and P are respectively nodes corresponding to analog chip, digital chip, memory module, and embedded passive blocks. 2

2 3D-IC with face-to-face and face-to-back bonding. 3

3 The SOP physical design detailed flow. 10

4 (a) The 3D-IC technology and (b) the corresponding 3D floorplans. 11

5 Illustration of SOP pin redistribution. The circle denotes the pins originated from the current device layer, whereas "x" denotes the x-nets that are going through the current device layer. Note that x-nets are using the routing channels in the device layer to insert feed-through vias. The initial location of the pins in soft blocks (two on the bottom) is at the center. 16

6 Illustration of the layer structure and routing resource in SOP. The block and white dots respectively denote the original and redistributed pins. The "x" denotes a feed-through pin for an x-net to pass through a placement layer using a routing channel. The solid, dotted, and arrowed lines respectively denote signal wires, vias, and feed-through vias. 17

7 Illustration of five different types of connections existing in SOP global routing. Note that MCM routing deals with only type 1 nets. 19

8 Graph-based modeling of placement layer. (a) a connection between two hard blocks, (b) b_2 is a soft block, and the new pin is located on the bottom boundary. Dotted lines denote pi assignment edges. 20

9 Illustration of crosstalk computation between two Steiner trees. The numbers denote the coupling length. (a) crosstalk is 7, (b) crosstalk-free routing. . . 21

10 Pseudocode for coarse pin distribution algorithm. 24

11 Illustration of coarse pin distribution. Pins along the external boundary are not shown for simplicity. 25

12 Illustration of the gain computation for coarse pin distribution. A net n indicated by the black node is moved from P_1 to P_2. Then, $g_d(n) = -2$, $g_w = 0$, and $g_b = -1$, where $deg(P_2) = 3$ becomes the maximum-degree partition. 25

13 Pseudocode for SOP detailed pin distribution algorithm. 27

14 Illustration of detailed pin distribution algorithm. The black and gray nodes respectively denote the old and new pin location, where the white node denotes the center of mass. The numbers denote the displacement force for each pin. 27

15 Illustration of the pin-to-location flow network. 28

16 Pseudocode for SOP layer assignment algorithm. 30

17 Illustration of (a) channel assignment, (b) local routing. 32

18 Pseudocode for SOP channel assignment algorithm. 35

19 Optical routing in System-On-Package. 42

20 Illustration of the layer structure and routing resource in SOP. Black and white dots respectively denote the original and redistributed pins. The arrowed line denotes feed-through via. 43

21 RC model of an opto-electrical interconnect. The dotted box denotes the waveguide. 45

22 Pseudocode for timing-driven waveguide construction. 47

23 Timing-driven waveguides for five timing critical nets. The nets n_1, n_3, and n_5 are multi-pin nets. Gray and black nodes respectively denote the source and sink nodes. The waveguides are shown as dotted lines. 48

24 Standard waveguides. The gray nodes indicate extry/exit points for each waveguide. 49

25 Pseudocode for congestion-driven waveguide construction. 49

26 Flow network for opto-net selection. 51

27 Rerouting of the net to go through the optical channel. The figure shows electrical routes (dashed line) before and after rerouting through the optical waveguide (bold line). 52

28 Illustration of 3D power supply modeling. (a) multi-layer power supply network, where multiple sets of device, routing, and power supply layers are stacked together, (b) 3D-grid modeling. where black and gray nodes respectively denote power supply and consumption nodes. 58

29 Illustration of SSN calculation. The dominant current source for block A is s_1, which is not located in the same layer. The dominant (= shortest) path p_0 carries $I_A/6$ amount of current, where I_A denotes the current demand of A. The block C draws current from s_2 and s_3 using p_1, p_2, and p_3 (each of these carries $I_C/3$ amount of current). The resistance of p_{34}, the overlap between p_3 and p_4, contributes to the SSN at B and C. 59

30 LP-based 3D decap allocation. 60

31 Illustration of 3D decap allocation. (a) 3D placement, (b) X-expansion, (c) XY-expansion, where the darker blocks denote the neighboring blocks of the decap (= white space) inserted. Note that blocks from other layers can utilize the white space for decap insertion. 61

32 Illustration of 3D congestion estimation. (a) 3D placement, (b) 3D pin redistribution, (c) topology generation. 62

33 Pseudocode for simulated annealing-based 3D SOP placement. 65

34 (a) 3-die IC with both face-to-face and face-to-back bonding, (b) bonding-aware 3D clock routing. . . . 70

35 The effect of hotspots on the merging points and buffer locations 72

36 3D clock routing with one and two through-vias.The squares represent the through-vias. The thin and bold lines represent routings in different layers. 76

37 Pseudocode for 3D Abstract Generation. . . . 77

38 Bottom-up generation of options. . . . 78

39 Pseudocode for 3D Clock Tree Optimization. . . . 79

40 Illustration of interconnects where the buffer and the merging point are co-located. . . . 80

41 Calculation of balanced skew. The bold lines is the skew under worst profile, while the thin line is the skew under uniform profile. . . . 82

42 Paths connecting nodes u and v to parent p. . . . 84

43 3D clock routing for $r3$ with through-via bounds of (a) one, (b) 10% of sinks, (c) 100% of sinks. The thick bold line are the via location candidates. The thick and thin lines are routing in two layers. . . . 86

SUMMARY

The focus of this research was to develop interconnect-centric physical design tools for 3D technologies. A new routing model for the SOP structure was developed which incorporated the 3D structure and formalized the resource structure that facilitated the development of the global routing tool. The challenge of this work was to intelligently convert the 3D SOP routing problem into a set of 2D problems which could be solved efficiently. On the lines of MCM, the global routing problem was divided into a number of phases namely, coarse pin distribution, net distribution, detailed pin distribution, topology generation, layer assignment, channel assignment and local routing. The novelty in this paradigm is due to the feed-through vias needed by the nets which traverse through multiple placement layers. To gain further improvements in performance, optical routing was proposed and a cost analysis study was done. The areas for the placement of waveguides were efficiently determined, which reduced delays and maximized utilization. The global router developed was integrated into a simulated-annealing based floorplanner to investigate trade-offs of various objectives. Since power-supply noise suppression is of paramount importance in SOP, a model was developed for the SOP power-supply network. Decap allocation, and insertions were also integrated into the framework. The challenges in this work were to integrate computationally intensive analysis tools with a floorplanning that works to its best efficency provided the evaluation of the cost functions is rapid. Trajectory-based approaches were used to sample representative data points for congestion analysis and interpolate the the congestion metric during the optimization schedule. Efficient algorithms were also proposed for 3D clock routing, which acheived equal skews under uniform and worst thermal profiles. Other objectives such as wirelength, through-vias, and power were also handled.

CHAPTER I

INTRODUCTION

The semiconductor industry is beginning to question the viability of the system-on-chip (SOC) approach because of its low-yield and high-cost. Recently, 3D packaging via System-On-Package (SOP) [90] has been proposed as an alternative solution to meet the rigorous requirements of today's mixed-signal system integration. The true potential of SOP technology lies in its capability to integrate both active and passive components into a single high-speed/density 3D packaging substrate. Each device layer is used to mount bare digital/analog dies (possibly using different technologies) and integrate embedded passive elements, and routing layers and vias are used to connect various elements. An illustration is shown in Figure 1. This provides a cost-effective and high-yield solution to system integration compared to SOC. In addition, 3D packaging offers a significant savings in area, delay, and power compared to the conventional 2D packaging (PCB and MCM). The layer-to-layer connection in 3D SOP is more effectively done using various types of vias compared to wire-bonding-based or stacked 3D system-in-package (SIP). Thus, innovative ideas on CAD tools for SOP technology are crucial to fully exploit the potential of this new emerging technology. However, there exist very few tools, if any, that handle the complexity of the automatic 3D SOP layout generation. Some initial works recently published on physical design for 3D SOP include [81, 77, 67, 65, 66, 68].

Recent advances in system integration has made 3D Integrated Circuits an attractive option for design implementations. Contemporary fabrication technology has made it economical to insert a large number of through vias for 3D-ICs by means of *wafer bonding technology* which is considered to be a promising method for 3D-IC integration [78]. This method achieves integration by vertically stacking the independently processed active dies interconnected to each other by means of *through-vias*, also called 3D vias. The dies can be bonded in one of the two ways: face-to-face and face-to-back (figure 2). In the face-to-face

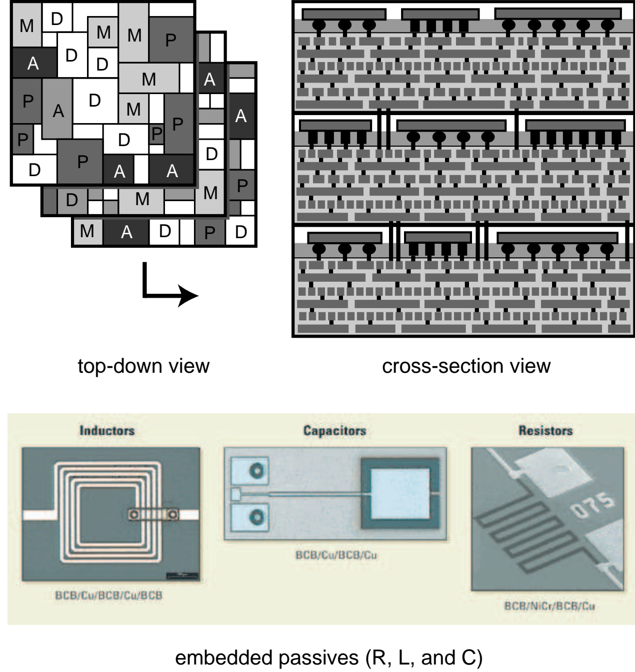

Figure 1: Illustration of 3D System-On-Package, where A, D, M, and P are respectively nodes corresponding to analog chip, digital chip, memory module, and embedded passive blocks.

bonded 3D-ICs the through-vias are manufactured through traditional metal etching technologies. This method of processing allows for a large through-via density. However only two active dies can be bonded using this scheme. In contrast, any number of active dies can be bonded using face-to-back bonding but through-vias have to be etched through the back of the die. Hence a lesser through-via density is acheivable in this case (due to lesser resolution of the etching process). The objective of the through-vias is to hold the dies together and facilitate inter-die connection. Wafer-to-wafer alignment of through-vias is an important manufacturing objective and this restricts the scaling of through-via dimensions. The typical dimensions of the through-vias vary from 1μm-by-1μm to 10μm-by-10μm depending on the technology. Hence the resistance of these vias are relatively smaller compared to the interconnect resistance and the increase in the delay is negligible. As in the case of the SOPs, the conventional CAD tools does not utilize the potential of the 3D-IC structure fully, nor address specific issues of bonding and thermal variations. Hence bonding aware placement and routing tools are needed to address the physical design concerns of 3D-ICs.

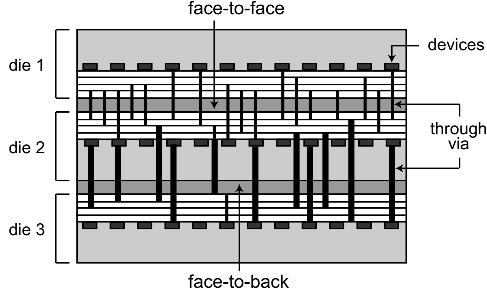

Figure 2: 3D-IC with face-to-face and face-to-back bonding.

The aim of global routing is to define the "coarse" geometry of the nets in a placed design. One of the interesting aspects of global routing in SOP is that the routing has to be done in a multi-layer placement system, which is acheived by means of feed-through vias that go through the placement layer. A number of objectives can be considered during routing, the conventional ones being performance, wirelength, vias, and the number of layers. However, signal integrity is a big issue in SOP. Crosstalk between adjacent nets causes signal deterioriation and mistimings. The problem was also interesting because it is a generalization of all traditional routing formulations. The development of a multi-phased global routing tool for SOP which optimizes performance while not ignoring crosstalk during the process, would enhance the capabilites of a traditional global router.

Recent advances in optical device integration for SOP offer drastic advantages over electrical interconnects. Optical interconnects enable faster signal propagation with virtually no crosstalk. In addition, wavelength division multiplexing allows a single waveguide to be shared among multiple interconnects, thereby significantly reducing the area used by the interconnect. An optical implementation of the 3D-technology interconnects would further improve performance and alleviate problems related to crosstalk and electromigration.

Since SOP is a multi-layer placement technology, power has to be supplied through power/ground planes to all the active circuit elements. SOP is especially susceptible to power-supply noise because of simultaneous switching within the circuit. Because of the resistive and inductive components in the power supply network, the circuits actually see a lower effective V_{dd}. If the power supply voltages are not constrained to be within tolerance limits, there may be degradation in performance and faulty operations resulting from

spurious transitions. The power/ground network design for SOP targeting reliability and robustness is a complicated and time-consuming task. This is mainly because of prohibitive runtimes for simulations and numerous iterations between placement and power-supply noise simulations. Another major concern for SOP is the thermal requirements. SOPs tend to have higher temperatures than other 2D-technologies because of their inherent structure. One of the factors indirectly responsible for higher temperatures is congestion. In addition to creating hotspots, congestion increases the cost of SOP because of its high correlation to the number of routing layers required. A tool floorplanning tool which explores the trade-offs between area, noise, and congestion would help make the design implementations more reliable.

Designing the clock distribution network is one of the most critical and performance oriented stage in the physical design of SOP. The clock distribution problem is particularly interesting for the 3D-technologies, because the routing has to be done in 3D. Various factors such as process variations, thermal profile, power dissipation need to be taken into consideration to assure reliable clock propagation. A badly designed clock distribution network has adverse implications for performance and may even result in race conditions among the synchronous circuit element leading to faulty operation. The considerations for clock network design in 3D-technologies is robustness under non-uniform thermal substrate profile and process variations. It is also desirable to minimize power consumption. Various techniques for clock distribution using different topology styles simultaneously targeting high performance and low power under 3D specific environment is possible. The developement of algorithms, which would be aware of the effects of process variations and non-uniform thermal profiles on skew and power is crucial.

1.1 Thesis Statement

The objective of this thesis is to develop interconnect-centric placement and routing tools for the 3D-technologies, especially SOPs and 3D-ICs that address 3D specific issues such as thermal and noise considerations to optimize such metrics as performance, area, signal integrity, noise tolerance and cost. To this end, a 3D global router with additional optical

routing capabalities, a congestion, and noise aware 3D floorplanner, and a thermal-aware 3D clock router have been developed. This thesis also presents experimental results to demonstrate the effectiveness of the tools.

1.2 Problem Statement

The aim of this research is to develop interconnect-centric physical design tools for high-performance and reliable SOP and 3D-ICs. The tools must address the multi-placement and multi-routing structure of 3D-technologies as well as other crucial issues such as signal integrity, and thermal and process variations. We aim to develop core place and route tools for SOP and 3D-ICs such as global routing, floorplanning, and clock routing.

1.3 Contributions

This dissertation presents the design and development of a number of interconnect-centric CAD tools for implementing the design flow for 3D technologies such as SOP and 3D-ICs. The design methodology for the conventional technologies has been well developed. A need for a flow incorporating 3D specific and modern issues has been motivated in this work.

The following items are the main contributions of this research:

- **Performance-Driven Global Routing**: We provide the formulation of the new block-level global routing problem for 3D SOP under wirelength, layer, crosstalk, and congestion minimization under various capacity constraints. We formulate and design heuristics for the following new problems: (i) SOP pin redistribution problem, (ii) SOP net distribution problem, and (iii) SOP channel assignment problem. Our linear-time, multi-phase 3PGR algorithm efficiently achieves high-quality results.

- **Optical Routing**: In this work we present the first optical router for 3D System-On-Package. Our approach is to start with a pure electrical interconnect routing solution, build a set of customized waveguides, and convert a subset of electrical wires into optical interconnects by rerouting them to use these waveguides. The WDM capability allows us to assign multiple electrical nets to each waveguide, where each waveguide has a limit on the number of different wavelengths it can handle.

In addition, various kinds of modules such as lasers, splitters, couplers, and photo detectors used for each waveguide occupy physical space in the layout. Thus, our optimization goal is to minimize the total wirelength and delays under various layout capacity constraints.

- **Congestion and Noise-Driven Floorplanning**: Our goal in this work is to perform simultaneous SSN and congestion-aware physical design for 3D SOP. Our design automation tool is aimed at reducing the amount of decap required to suppress SSN and congested areas in a 3D SOP design without compromising traditional design metrics such as area and wirelength. In our approach, we first perform automatic placement of functional modules while minimizing the amount of decap required for each module. We then place decaps near the modules that need them while minimizing the overall footprint area. We developed a compact 3D SSN model that can be used to calculate the decap demand of a 3D solution efficiently. As a result, our method is much more efficient than directly employing SSN simulation during the optimization process. To the best of our knowledge, this is the first work to perform automatic placement of modules and decaps for 3D packaging. In our 3D wire congestion analysis, we first obtain an initial 3D global routing solution and incrementally update it upon each new candidate 3D placement. Our 3D global routing process consists of 3D pin redistribution and topology generation.

- **Clock Routing**: In this research, we developed algorithms for clock routing in a 2-layer wafer-bonded 3D ICs considering the thermal gradients. An illustration is shown in Figure 34. Since the existing clock routers can easily handle the face-to-back bonding, we focus on face-to-face bonding in this work. Through-vias have a relatively smaller cost in the face-to-face bonded 3D-ICs . However, through-via planning is needed to properly manage through-vias from clock routing as well as signal routing. Morever, even distribution of through-vias during clock routing is essential for signal routability as well. There are three-fold contributions of this work. We describe a complete frame work for clock routing of 2-layered wafer bonded 3D ICs. We propose

an abstract tree generation algorithm which provides a smooth trade-off between even distribution of a given number of through-vias and the wirelength of the clock tree. To the best of our knowledge, this is the first work that discusses clock routing for 3D ICs. We show that classic clock routing algorithms are incapable of reducing skews for buffered clock trees subjected to variations in thermal profiles. We extend a recent work on thermal aware clock tree optimization to buffered clock trees and use it to refine the 3D clock tree. We prove that constructing a clock tree, which has balanced skews under two temperature profiles is equivalent to constructing a zero-skew clock tree for the average profile. This observation facilitates efficient optimization of clock tree under thermal variations.

1.4 Thesis Organization

The dissertation is organized into seven chapters.

- **CHAPTER I. INTRODUCTION**: This chapter provides an introduction on the current and emergent 3D technologies. It also discusses the contributions of this research and outlines the organization of the thesis.

- **CHAPTER II. ORIGIN AND HISTORY OF THE PROBLEM**: This chapter discusses the motivations for the thesis research. An extensive survey of related work is also provided.

- **CHAPTER III. PERFORMANCE-DRIVEN GLOBAL ROUTING**: This chapter discusses the formulation of the new block-level 3D global routing problem under wirelength, layer, crosstalk, and congestion minimization. The chapter also presents the developement of the first global router for 3D SOP named 3PGR.

- **CHAPTER IV. 3D OPTICAL ROUTING**: This chapter presents the first optical router for 3D SOP. Efficient algorithms for the construction of timing and congestion-driven waveguides taking into account the optical resource constraints is also discussed in detail.

- **CHAPTER V. CONGESTION AND NOISE-DRIVEN FLOORPLANNING**: This chapter presents a 3D module and decap (decoupling capacitance) placement algorithm that simultaneously reduces the power supply noise and wire congestion. Efficient algorithms for 3D power supply noise and congestion analysis to guide our 3D module placement process are presented. In addition, white spaces allocation around the modules that require decaps to suppress the power supply noise is done, while minimizing the area overhead.

- **CHAPTER VI. 3D CLOCK ROUTING**: This chapter considers clock routing for 2-layer face-to-face bonded 3D-ICs while considering non-uniform thermal profiles. Efficient algorithms for bonding style-aware 3D clock synthesis and thermal-aware clock optimization have been presented.

- **CHAPTER VII. CONCLUSIONS**: This chapter concludes this dissertation with a summary of main contributions and directions for future work.

CHAPTER II

ORIGIN AND HISTORY OF THE PROBLEM

2.1 Background

VLSI chip design consists of circuit design, logic synthesis, physical design, and fabrication. Physical design can be defined as the process of automatically generating a layout from the synthesized netlist. To handle the complexity of the problem, physical design automation is subdivided into parititioning, floorplanning, global routing, detailed routing, clock routing, and compaction.A detailed discussion of each of these subproblems can be found in [82]. A brief history of the research in physical design is outlined in Table 1.

Table 1: Evolution of the Physical Design Research

Year	Design Tools
1950-1965	Manual design
1965-1975	Layout editors Automatic routers (for PCB) Efficient partitioning algorithms
1975-1985	Automatic placement tools Well defined phases of design of circuits Significant theoretical development in all phases
1985-1995	Performance driven placement and routing tools Parallel algorithms for physical design Significant development in underlying graph theory Combinatorial optimization problems for layout
1995-present	Interconnect layout optimization, Interconnect-centric design, physical-logical codesign Physical Design for 3D ICs, SoP and SoC

The aim of the physical design flow is to take a synthesized netlist as input and produce a design-rule correct placed and routed design that is ready for manufacturing. The design is also optimized for performance, power and yeild. The physical design flow for the SOP is shown in Figure 3. Similar to the case of the MCMs, the physical design for the SOPs is done in three steps: partitioning, floorplanning and routing. It can be seen that the conventional

flow is incorporated in the SOP flow. The is because the individual components for the SOP are designed using the conventional flow. The input to the physical design flow is the synthesized netlist of the design which may or may not be 3D-technology aware. However, there is a huge scope for utilizing the benefits of a 3-D technology in the back-end. Since the back-end flow cannot handle the huge-sized designed directly, the design is partitioned into managable blocks. An excellent survey of netlist partitioning techniques can be found in [5]. The partitioned netlist is then floorplanned using a number of objectives including area and wirelength. A comprehensive discussion of large-scale circuit placement techniques can be found in [35]. The placed design is finally routed, typically using a routing resource model. The routing process itself can be divided into signal routing, power routing and clock routing. The signal routing can be further be classified into global and detailed routing. [16] presents a good survey on the placement and routing of MCMs. A detailed survey on interconnect optimization including clock routing can be found in [30].

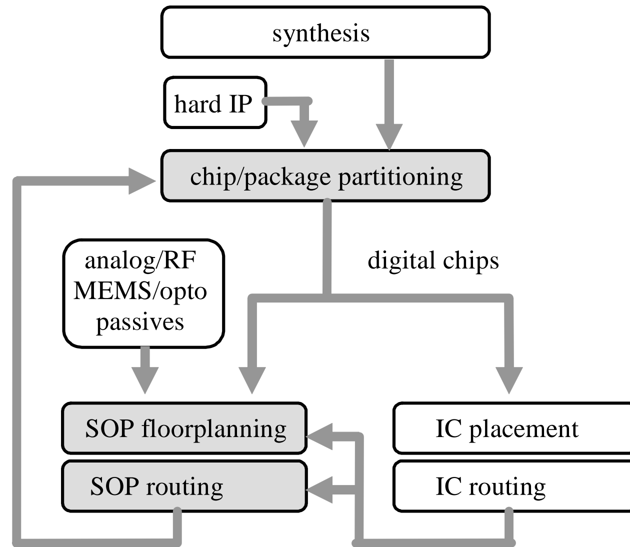

Figure 3: The SOP physical design detailed flow.

The 3D integrated ciruit is a technology that is fast gaining acceptance as a design implementation alternative in the industry and the academia. This technology is acheived through wafer bonding. There are some essential similarity between the SOPs and the 3D-ICs as far as the back-end flow is concerned. Both rely on a multi-layer placement and routing manufacturing system, although the ways of acheving it may be quite different.

Morever, the issues impeding the growth of these two technologies are similar. Hence, the tools developed for the SOPs can be used as is or with minor modifications for the 3D-ICs and vice-versa. An illustration of the 3D-IC and its floorplan is shown in Figure 4.

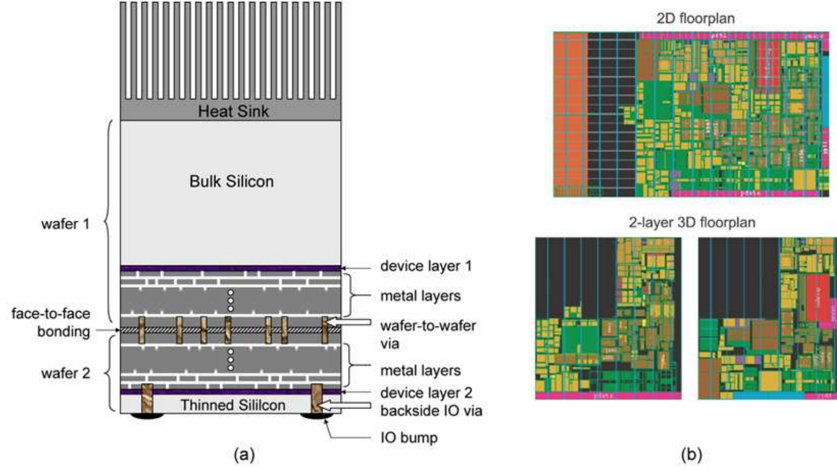

Figure 4: (a) The 3D-IC technology and (b) the corresponding 3D floorplans.

2.2 Global Routing

In this section we review the various global routers for MCM because of of its structural similarity to SOP. MCM routing is different from IC routing primarily because of the large number of layers available in MCM—a typical MCM has 60 signal layers compared to four to eight in IC. Thus, MCM routing is a multi-layer, multi-objective optimization problem. An early MCM router was presented in [50] that performs a combination of 2D and 3D routing. However, these algorithms suffer from long runtime. The SLICE [58] router constructs detailed routes through a mixture of planar routing and two-layer maze routing. While this router was superior to a 3D maze router, it requires relatively high computation times. The V4R router [57] efficiently routes layer pairs one at a time using routes with no more than four vias each. But V4R's column-by-column method may introduce more vias in the large-sized grid routing. The M2R [23] router addresses performance concerns by routing critical nets on a single layer. A lower bound on delay was proposed using wirelength and vias. In [64], a linear programming approach was presented, and randomized rounding is used to obtain integer solutions. A small number of possible topologies are considered, and

routes are restricted to a channel intersection graph. SURF [40] is a gridless MCM router based on rubber-band sketches.

The MCG [10] router considers a small number of possible routes for each net and constructs a compatibility graph. Then, this compatibility graph is reduced to yield a subset of routes that is fully compatible. Finally, a three-phase routing strategy is used to route nets with as few vias as possible. The MLR router [92] first performs layer assignment of nets, followed by Steiner optimal area routing using a hierarchical approach. The V4C [70] router completes the MCM routing with consideration of crosstalk constraint. But its crosstalk estimation method is not appropriate for high-performance MCM routing. The SEGRA [11] router performs global routing by completing pairs of layers in each pass. A simple greedy heuristic is used to select which nets are to be extended or completed during each pass. In [97], the MCM substrate is decomposed into a set of "towers"—rectangular regions containing portions of the routing surface through all layers. They first distribute routing density across a three-dimensional surface, using hierarchical decomposition, determine locations of nets on the boundaries of the towers, and finally obtain a routing for each tower using a detail router. The MINOTAUR [34] global router combines the freedom and flexibility of maze routing solutions, with the global optimization abilities of the iterative deletion method.

2.3 Optical Routing

Chang et al. [12] recently proposed a low-cost opto/digital integrated circuit on a standard printed wiring board. They successfully developed a low-temperature polymer process for fabricating and integrating optoelectronic components such as lasers, waveguides, and photo-detectors onto printed wiring boards for mixed signal SOP applications. Chen et al. demonstrated a fully embedded implementation of high-speed optical communications within one board [19]. A 12-channel linear array of thin-film polyimide waveguides, vertical-cavity surface-emitting lasers and silicon photodetectors were used for this experiment. Also, a 1-to-48 optical clock signal distribution network for Cray T-90 supercomputer was

demonstrated. The WDM concept has been experimentally demonstrated in [2]. A pseudo-random data generator circuit on the chip was used to feed 10 channels with different data that were combined at a reflective grating and transmitted through the optical fiber. GaAs diodes that were flip-chip bonded onto 0.5um silicon CMOS were used as modulators as well as detectors to spectrally split the incoming beam by wavelength. Such experiments have proved that optical routing on the package level is a promising solution. A recent work on placement for 2D SOP is presented in [80], and [79] performs clock routing using optical waveguides.

2.4 Noise and Congestion Aware Floorplanning

Floorplanning algorithms are classified into slicing floorplanning and non-slicing floorplanning. The slicing floorplanning algorithm [83, 93, 96] generates a slicing structure, which is a rectangular dissection acheived recursively by subdividing rectangles horizontally or vertically. Its optimal solution can be computed in polynomial time. However, it cannot provide a high-quality solution compared to non-slicing floorplans, especially for area objectives. Non-slicing floorplanning, on the other hand, is an NP-hard problem. The non-slicing floorplanning algorithm can be classified into two approaches: mathematic programming based [85, 86], and heuristic based, such as simulated annealing[71, 72].

Simultaneous Switching Noise (SSN) [73] is an important issue for SOP floorplanning. A popular approach for suppressing SSN is the addition of decoupling capacitances (decaps) to the floorplan. Decap placement is done for 2D PCB designs [27, 56, 20, 51] and 2D circuits [74, 99, 84, 18]. Some of the recent congestion-driven 2D circuit placement works include [75, 95, 62]. Recent placement algorithms for 3D circuit placement include [86, 42, 98, 41, 48, 17, 44]. However, the design tools for 3D circuits are not applicable since these tools target individual gates instead of chips and embedded passives. Recently, physical design algorithms for 3D SOP designs have been proposed, including 3D placement [81, 77] and 3D routing [65, 66, 68].

2.5 Clock Distribution

The performance of a system largely depends on the clock distribution to the various synchronous elements. In addition to having low delay from the insertion point to the receivers, the clock topology must satisfy the *skew* bounds, which is defined as the maximum difference of the clock arrival times at the receivers. Clock routing has inspired rich and elegant works in the last two decades. Depending on the requirement on the skew, zero-skew, bounded-skew, and associate-skew problems for clock routing can be defined. The *zero-skew tree* (ZST) problem seeks a minimum cost clock tree having zero skew [14, 15, 46]. The *Deffered-Merge Embedding* (DME) algorithm [9, 13, 46] embeds internal nodes of a topology G via (i) a bottom-up construction of a tree of merging segments, or merging tree, representing the loci of possible placements of internal nodes in the ZST; and (ii) a top-down determination of exact locations for the internal nodes of G. DME acheives minimum wirelength and tree radius for any given input topology. Related works study the topology constructions that lead to low-cost solutions when DME is applied; the most successful variant is Greedy-DME [45]. It has been known for sometime now that zero skew comes at the cost of increased wirelength and circuits can operate with a certain amount of skew. The *bounded-skew tree* (BST) problem has been addressed in [53, 55, 94, 31, 33]. More recently it was noted that bounded skew for all pairs of terminals is an over-contraint and constraints actually exist between sequentially adjacent registers i.e, pairs of registers connected by purely combinational paths of logic and interconnect. A DME-based solution for the associative-skew problem has been proposed in [21]. Buffer sizing and wire sizing have been done as a post-processing step to achieve zero skew and low power in [87]. An early work which considered the non-uniformness of the substrate temperature is outlined in [3]. Several approaches were proposed for zero skew [88, 9, 45] and bounded skew [15, 31, 33], which acheived simultaneous reduction in wirelength while meeting skew bounds. Recent works in clock routing consider variations during clock routing [25, 61, 49]. A recent work [25], which optimized clock tree for a non-uniform thermal profile, clearly demonstrated that classic deferred merge embedding (DME) based methods [88] are unequal to the task of controlling the skews of the clock tree under varying thermal profiles.

CHAPTER III

PERFORMANCE-DRIVEN GLOBAL ROUTING

The physical layout resource of an SOP is multi-layer in nature: all layers are used for both placement and routing, pins are now located in all layers, and various types of vias are available for layer-to-layer connections. Therefore, the existing design tools for PCB or MCM packaging cannot be used directly for the design of an SOP. In addition, the design tools for 3D circuits [43, 42, 48, 86, 98] are not applicable either since these tools target individual gates, whereas in 3D packaging we place and route the blocks that represent chips and embedded passives. Therefore, our primary goal is to make the best use of placement and routing layers available while automatically generating a 3D package layout under various noise constraints. In this chapter, we present 3PGR, the first global router for 3D packaging.

The chapter is organized as follows: Section 3.1 discusses the differences between the MCM and SOP routing. Section 3.2 outlines the problem formulation. An overview of the developed framework is presented in Section 3.3. The experimental results are discussed in Section 3.4. Section 3.5 concludes this chapter.

3.1 MCM vs SOP Routing

Several similarities and differences exist between MCM and SOP routing. In general, the design objectives and constraints in both MCM and SOP routing are quite similar, where performance, wirelength, via, layer, crosstalk, and congestion optimization are crucial. In addition, the routing process is divided into multiple steps: pin distribution, topology generation, layer assignment, and detailed routing. A notable difference between SOP and MCM routing is that there are *multiple* device layers in SOP, whereas in MCM there is only one device layer. Therefore, nets are now connecting pins located in *all* intermediate layers in SOP, and the blocks in each layer behave as obstacles. In MCM, however, all pins are

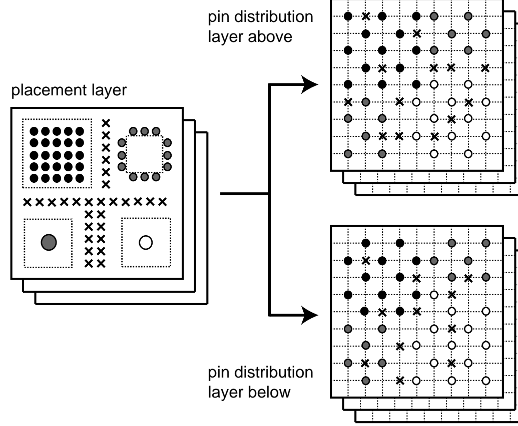

Figure 5: Illustration of SOP pin redistribution. The circle denotes the pins originated from the current device layer, whereas "x" denotes the x-nets that are going through the current device layer. Note that x-nets are using the routing channels in the device layer to insert feed-through vias. The initial location of the pins in soft blocks (two on the bottom) is at the center.

located *only* at the top layer, and no obstacles exist except for the wires themselves. This makes the SOP or 3D package routing problem more general than MCM routing.

More specifically, we note that there exist three major differences between MCM and SOP global routing: pin distribution, channel assignment, and local routing. Because of the availability of multiple device layers and their associated routing resource, some pins in each device layer can be distributed either to the layer above or below, as illustrated in Figure 5. In addition, some nets that connect blocks in non-neighboring device layers need to penetrate intermediate device layers. Since the blocks in these intermediate layers become obstacles, we use routing channels in each device layer to insert "feed-through vias." Thus, our channel assignment determines which routing channel to use for feed-through vias. We match pins in the distribution layers to channel location based on capacity constraint while minimizing wirelength and congestion. Last, the routing channels in each device layer can be used for intra-layer connection as well. Thus, after feed-through via insertion is done, our local routing step decides which remaining channels to use for the intra-layer connection. We also finish the connection between the original and distributed pins. In case the pin location is not known for some "soft blocks," we determine their location either along the boundary or underneath the block during our local routing step.

3.2 Problem Formulation

3.2.1 SOP Routing Resource

The layer structure in multi-layer SOP is illustrated in Figure 6. The *placement layers*[1] contain the blocks (such as ICs, embedded passives, opto-electric components, etc), which from a physical design point of view are just rectangular blocks with pins along the boundary. The interval between two adjacent placement layers is called the *routing interval.* A routing interval contains a stack of *routing layers* sandwiched between *pin distribution layers.* These layers are actually x-y routing layer pairs so that the rectilinear partial net topologies may be assigned to them. The pin distribution layers in each routing interval are used to evenly distribute pins from the nets that are assigned to this interval. Then, these evenly distributed pins are connected using the routing layer pairs. Each placement layer consists of a pair of x-y routing layers, so routing is permitted. A *feed-through via* is used to connect two pin distribution layers from different routing intervals. Thus, the routing channels in each placement layer are used for two purposes: (i) to accommodate feed-through vias and (ii) to perform local routing, where a limited number of intra-layer connections are made.

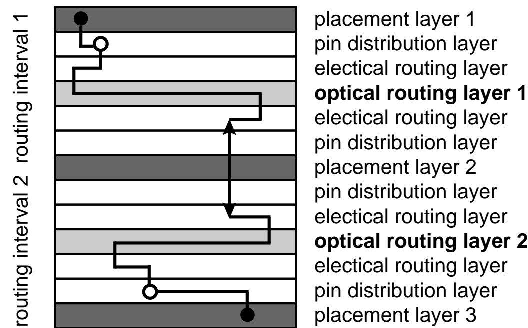

Figure 6: Illustration of the layer structure and routing resource in SOP. The block and white dots respectively denote the original and redistributed pins. The "x" denotes a feed-through pin for an x-net to pass through a placement layer using a routing channel. The solid, dotted, and arrowed lines respectively denote signal wires, vias, and feed-through vias.

In the SOP model, the nets are classified into two categories. The nets that have all their terminals in the same placement layer are called *i-nets*, while the ones having terminals in

[1] We use placement layer and device layer interchangeably.

Table 2: Type of nets and the layer usage for a 3D SOP with K placement layers. The layer information is denoted in parenthesis.

type	pins	layers used
1	$p_1(1)$ and $p_2(1)$	$L_p(1)$, $L_t(1)$, $L_r(1)$
2	$p_1(K)$ and $p_2(K)$	$L_p(K)$, $L_b(K\text{-}1)$, $L_r(K\text{-}1)$
3	$p_1(i)$ and $p_2(i)$	$L_p(i)$, $L_t(i)$, $L_r(i)$ or
		$L_p(i)$, $L_b(i\text{-}1)$, $L_r(i\text{-}1)$
4	$p_1(i)$ and $p_2(i{+}1)$	$L_p(i)$, $L_t(i)$, $L_r(i)$, $L_b(i)$, $L_p(i{+}1)$
5	$p_1(i)$ and $p_2(i{+}2)$	$L_p(i)$, $L_t(i)$, $L_r(i)$, $L_b(i)$, $L_p(i{+}1)$
		$L_t(i{+}1)$, $L_r(i{+}1)$, $L_b(i{+}1)$, L_p(i+2)

different placement layers are *x-nets*. The i-nets can be routed in a single routing interval or indeed within the placement layer itself. On the other hand, the x-nets may span more than one routing interval. Table 2 shows five different types of nets existing in SOP global routing along with their layer usage. Figure 7 shows the corresponding illustration.

The routing and pin distribution layers in each routing interval are modeled with a standard $x \times y \times z$ 3D grid, where each node represents a routing region and each x/y-direction edge represents each horizontal/vertical boundary among the regions. Each z-direction edge represents a group of vias each region can accommodate. Thus, all edges in this 3D grid are associated with capacity: x and y edges for wire capacity, and z edges for via capacity. We use the Floor Connection Graph (FCG) [28], illustrated in Figure 8, to model the placement layer, where each routing channel becomes an edge and each channel intersection point becomes a node. In addition, each soft block becomes a node, and *pin assignment edges* are added to this node to connect to all adjacent channels. This model allows us to determine which boundary to use for the pins with unknown location. Each channel edge is associated with (i) via capacity for feed-through vias, and (ii) wire capacity for local routing. In addition, pin assignment edges have pin capacity for each boundary of a soft block.

3.2.2 SOP Routing Problem

For each net n from a given netlist NL, let xt_n, wl_n, and via_n respectively denote the amount of crosstalk, wirelength, and via associated with n. The wirelength wl_n is the

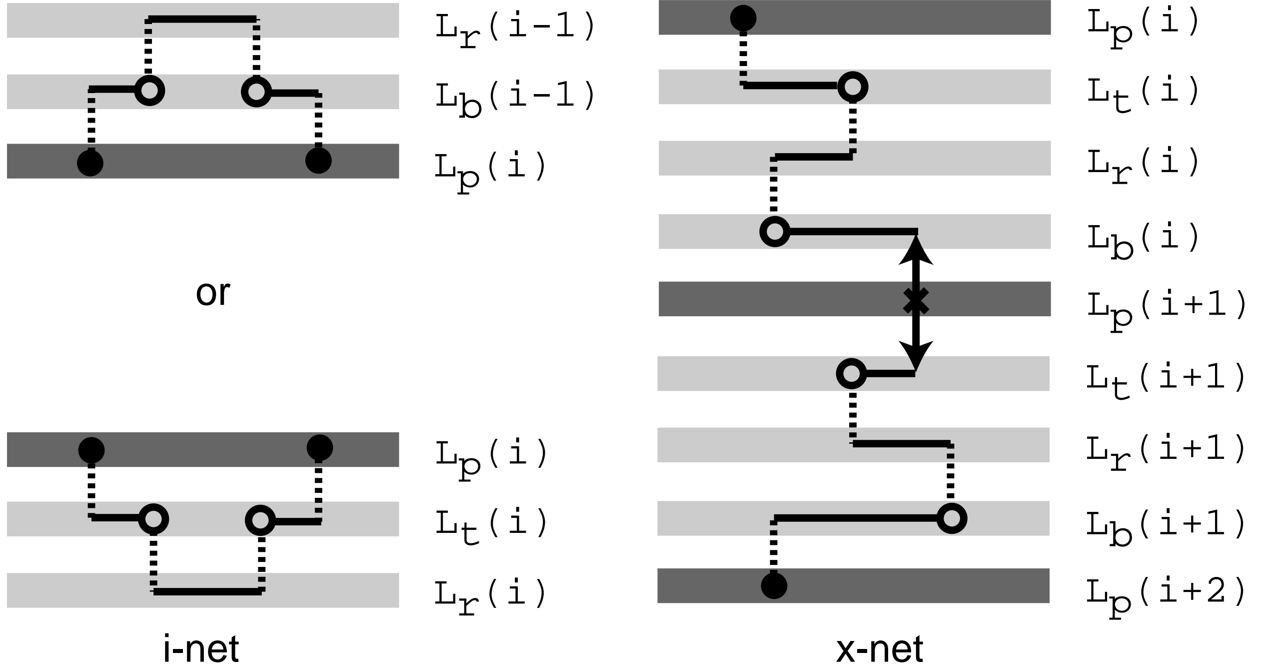

Figure 7: Illustration of five different types of connections existing in SOP global routing. Note that MCM routing deals with only type 1 nets.

sum of the Manhattan distance in the x, y, and z directions, where the z direction is the height of the associated vias.[2] Let $cl(n, m)$ denote the coupling length between n and m, as illustrated in Figure 9. We define xt_n as follows:

$$xt_n = \sum_{m \in NL,\, m \neq n} \frac{cl(n, m)}{|z(n) - z(m)|}$$

where $z(n)$ denotes the routing layer that contains net n. For each net n, let $d_n(i)$ denote the Elmore delay [47] at sink i. Then, the maximum sink delay of net n, denoted d_n, is $\max\{d_n(i)|i \in n\}$. The performance of a SOP global routing is estimated by the following:

$$D^{max} = \max\{d_n|n \in NL\}$$

For each routing interval i in a 3D package with K placement layers, let $L_t(i)$, $L_r(i)$, and $L_b(i)$ respectively denote the top pin distribution layer pairs, routing layer pairs, and bottom pin distribution layer pairs. Three kinds of connections exists in each routing interval i: top (connection between $L_p(i)$ and $L_t(i)$), middle (connection between $L_t(i)$ and $L_b(i)$), and

[2] We assume that the height of vias connecting two x-y layers in a routing and pin distribution layer pair is of one grid, whereas the height of feed-through vias that penetrate a placement layer is of 6 grid.

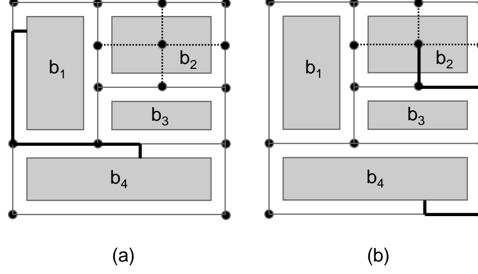

Figure 8: Graph-based modeling of placement layer. (a) a connection between two hard blocks, (b) b_2 is a soft block, and the new pin is located on the bottom boundary. Dotted lines denote pi assignment edges.

bottom (connection between $L_b(i)$ and $L_p(i+1)$). We use the routing layer pairs in $L_t(i)$, $L_r(i)$, and $L_t(i)$ respectively for the top, middle, and bottom connections. We construct the *routing grid*, denoted $G(i)$, that contains all the distributed pins from $L_t(i)$ and $L_b(i)$ in a $m \times n$ 2D grid. We use $G(i)$ to perform topology generation for various two-pin and multi-pin connections. The total number of layers used in a SOP global routing solution is given by

$$L^{tot} = \sum_{1 \le i \le K} (|L_t(i)| + |L_r(i)| + |L_b(i)|)$$

Section 3.3.4.1 discusses how to compute $|L_r(i)|$ and Section 3.3.5.1 discusses how to compute $|L_t(i)|$ and $|L_b(i)|$.

Last, the formal definition of the SOP global routing problem is as follows: Given a 3D placement and netlist, generate a routing topology for each net n, assign n to a set of routing layers, and assign all pins of n to legal locations. All conflicting nets are assigned to different routing layers while satisfying various wire/via capacity constraints. The objective is to minimize the following cost function:

$$\alpha \cdot L^{tot} + \beta \cdot D^{max} + \sum_{n \in NL} (\gamma \cdot xt_n + \delta \cdot wl_n + \epsilon \cdot via_n)$$

We minimize $|L_r|$ during our layer assignment step and $|L_t| + |L_b|$ during our channel assignment step. D^{max} is the focus during our topology generation step. The wirelength, via, and crosstalk minimization are addressed in all steps of our global router.

3.3 Overview of 3PGR Algorithm

Our 3PGR router is divided into the following five steps:

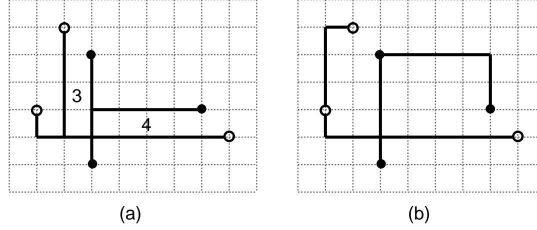

Figure 9: Illustration of crosstalk computation between two Steiner trees. The numbers denote the coupling length. (a) crosstalk is 7, (b) crosstalk-free routing.

1. Pin redistribution: We first determine which set of i-nets and x-net segments are assigned to each routing interval. The pins from these nets are then evenly distributed in the top and bottom pin distribution layers.

2. Topology generation: Steiner trees are generated for all nets in each routing interval so that the performance of the routed design is optimized.

3. Layer assignment: The routed nets are assigned to a unique routing pair in the routing layer so that the total number of layers used is minimized.

4. Channel assignment: For each x-net, its location of feed-through via in the routing channel is determined. We also assign channels and finish the connections for the i-nets that are to be routed in each placement layer.

5. Local routing: We finish connections between the pins now located in the routing channels and the pins on the block boundaries. We also determine the location of pins from soft blocks.

For step 2, we use an existing RSA/G heuristic [32] to generate the net topologies. The critical nets are routed using *Minimum Shortest Path Arborescence* (MSPA). A subset of the non-critical nets is ripped and routed for congestion control. The wirelength is minimized by using the RSA/G heuristic on rerouted nets using edge weights quantifying congestion (i.e, number of nets using the edge). In addition, we use the congestion-driven rip-up-and-reroute [63] for step 5. Therefore, the focus of this chapter is to present heuristics for steps 1, 3, and 4. Pin redistribution is performed while considering all routing intervals simultaneously. During the topology generation and layer assignment, we visit each routing

interval sequentially from top to bottom. During channel assignment and local routing, we visit placement layers sequentially from top to bottom.

3.3.1 SOP Pin Redistribution

We first present an overview of our approach and the problem formulation of SOP pin redistribution. We then discuss the details of the three steps used in our algorithm, namely, coarse pin distribution, net distribution, and detailed pin distribution.

3.3.1.1 Overview of the Approach

During 3D placement, we assume pins are located at the center of the modules (= soft modules) or at the boundary of the modules (= hard module). Thus, the pin location is highly localized and not evenly distributed. Since our plan is to use pin distribution layers and routing layers in combination to finish routing in each routing interval, one of the important steps is to evenly distribute pins in the pin distribution layer so that routing in the routing layers is done more evenly. This greatly helps reduce the number of routing layers used as well as crosstalk among nets. However, pin distribution cannot be done accurately without knowing which net is assigned to which routing interval. On the other hand, our net distribution needs to know the pin location for more accurate crosstalk measurement. Consequently, we need to iterate between pin distribution and net distribution until we converge to a good solution. We solve this issue with our three-stage effort: coarse pin distribution, net distribution, and detailed pin distribution.

1. Coarse pin distribution: We construct an $m \times n$ 2D grid and evenly distribute the pins from all nets in all routing intervals in this single grid.
2. Net distribution: We assign a routing interval for each i-net to either above or below the placement layer it belongs to. The crosstalk computation is based on the coarse pin distribution result.
3. Detailed pin distribution: We refine our pin distribution results for each routing interval based on the net distribution result. In addition, the pin location is legalized, i.e., each pin is assigned to a unique grid point in the pin distribution layers.

3.3.1.2 Problem Formulation

The following is the set of inputs to the *SOP Pin Redistribution Problem*: (i) a set of placement layers $L_p = \{L_p(1), L_p(2), \cdots, L_p(K)\}$, (ii) a set of nets (i-nets and x-nets) $NL = \{n_1, n_2, \cdots, n_k\}$ that connect the pins in L_p, (iii) a set of top pin distribution layers $L_t = \{L_t(1), L_t(2), \cdots, L_t(K)\}$, and (iv) a set of bottom pin distribution layers $L_b = \{L_b(1), L_b(2), \cdots, L_b(K)\}$. A routing interval $RI(i)$ contains $L_p(i)$, $L_t(i)$, $L_r(i)$, $L_b(i)$, and $L_p(i+1)$. An illustration is shown in Figure 6. Our goal is to determine (i) which routing interval(s) each i-net and x-net belongs to, and (ii) a one-to-one mapping from the pins in the placement layers to the pins in the redistribution layers. Each pin in L_p is assigned to a unique grid point in the $m \times n$ grid graph $G(i)$. Since each grid point represents a routing region in these pin distribution layers, each node/edge in $G(i)$ is associated with via/wire capacity. For a pin $p \in L_p$, let dw_p denote the wirelength between the original and the new location (after pin redistribution). The objective of SOP pin redistribution is to minimize the following cost function under the via/wire capacity constraints:

$$w_1 \cdot \sum_{p \in L_p} dw_p + \sum_{n \in NL} (w_2 \cdot wl_n + w_3 \cdot xt_n)$$

According to the five types of SOP nets shown in Figure 7, we note that the routing interval assignment for some i-nets and all x-nets is straightforward: all i-nets from $L_p(1)$ are assigned to $RI(1)$, and all i-nets from $L_p(K)$ are assigned to $RI(K-1)$. In addition, each x-net that spans k routing intervals is decomposed into k segments and assigned to all intermediate routing intervals. Let $N^i = \{n^i_1, n^i_2, \cdots, n^i_k\}$ denote the set of *movable i-nets* that has pins from $L_p(i)$ for $1 < i < K$. Note that these movable nets can be assigned to either $L_b(i-1)$ or $L_t(i)$, while other nets are fixed into some intervals. Thus, we formulate the *SOP Net Distribution Problem* to decide which routing interval to use for the nets in N^i.[3]

[3] We attempted to solve the SOP net distribution problem using the existing K-Way Max-cut Partitioning method [24]—we build the Net Interference Graph (NIG), where the nodes and edges respectively represent the nets and crosstalk between them. Then, the max-cut partitioning tries to separate nets with high crosstalk into different routing intervals. To our surprise, this approach had very little impact on crosstalk and produced results that were very close to a very simple heuristic: distribute the movable i-nets randomly. Our related experiments suggest that this is due to the small number of movable i-nets existing in our

```
Coarse Pin Distribution
1:  CP = m × n grid;
2:  for (each pin p ∈ NL)
3:      s = slot closest to p and under-utilized;
4:      place p in s;
5:  C = restricted multi-level clustering of pins;
6:  hgt = height of C;
7:  B(hgt) = initial m × n placement at top level;
8:  for (i = hgt downto 0)
9:      move clusters in C(i) to optimize cost;
10:     B(i) = new m × n placement at level i;
11:     project B(i) to B(i − 1);
12: return B(0);
```

Figure 10: Pseudocode for coarse pin distribution algorithm.

3.3.2 Coarse Pin Distribution

A pseudocode for our coarse pin distribution algorithm is shown in Figure 10. First, we assign all pins in the placement layers to a nearby grid point in CP, an $m \times n$ 2D grid, while trying to balance the number of pins assigned to each grid point (line 1-4). An illustration is shown in Figure 11. We impose pin capacity for each grid point so that the pins are evenly distributed in CP. Our approach is to visit the pins in a random order and find the best grid for each pin. For each pin p, a grid point that is closest to the original location of p and has not violated the pin capacity constraint is chosen (line 3). After this process is finished, CP serves the starting point of our min-cut placement-based algorithm, where each grid point corresponds to a partition. We then iteratively improve the quality of this initial solution via the move-based approach. We extend the multi-level min-cut-based global placement algorithm [39] for coarse pin distribution. In [39], recursive multi-level bipartitioning is used to divide the given netlist into an $m \times n$ grid while minimizing the number of inter-partition connections (= cutsize) as well as their estimated wirelength. In our new heuristic algorithm, our cost function is based on (i) how far the new pin location

benchmarks (less than 10%). However, this does not mean that SOP problem is insignificant. We believe that i-net distribution for crosstalk minimization will play an important role if the number of movable i-nets is huge.

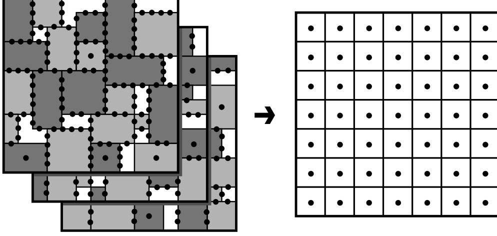

Figure 11: Illustration of coarse pin distribution. Pins along the external boundary are not shown for simplicity.

is from the initial location, (ii) total wirelength, and (iii) how evenly distributed the inter-partition connections are.

For each pin p, we define the *displacement gain*, denoted $g_d(p)$, to represent how much the distance between the original and new location is reduced if p is moved to another partition. We define the *wirelength gain*, denoted $g_w(p)$, to represent how much the length of the nets that contain p (estimated by the half-perimeter of the bounding box) is reduced if p is moved to another partition. For a partition P, let $deg(P)$ denote the number of nets that have connections to P. Then, the *cutsize balance factor* is defined as follows:

$$\max\{deg(P_i) - deg(P_j) | \forall P_i, P_j\}$$

i.e., the difference between the maximum and minimum degree among the partitions. We define the *balance gain*, denoted $g_b(p)$, to represent how much the cutsize balance factor is reduced. Our move-based multi-level mincut partitioning algorithm performs cell move based on the combined gain function:

$$g(p) = w_1 \cdot g_d(p) + w_2 \cdot g_w(p) + w_3 \cdot g_b(p)$$

Figure 12 shows an illustration of the gain computation.

In our multi-level approach, we first perform *restricted multi-level clustering* (line 5), which preserves the initial $m \times n$ placement result, where two pins that are in different partitions initially are not clustered together. At each level of the cluster hierarchy from top to bottom (line 8), we compute the combined gain $g(p)$ for each cluster and perform cluster moves. To compute the displacement and balance gain of a group of pins (= cluster), we add the individual displacement and balance gain of all pins in this cluster. When there

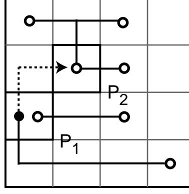

Figure 12: Illustration of the gain computation for coarse pin distribution. A net n indicated by the black node is moved from P_1 to P_2. Then, $g_d(n) = -2$, $g_w = 0$, and $g_b = -1$, where $deg(P_2) = 3$ becomes the maximum-degree partition.

is no gain at a certain level, we decompose the clusters into the next lower level and perform refinement. This process continues until we obtain a solution at the bottom level (line 12). Our initial partition computation takes $O(p \cdot m \cdot n)$ (line 2-4), where p is the number of pins. Our multi-level clustering algorithm [38] (line 5) takes $O(p \log p)$, and the multi-level partitioning (line 8-11) takes $O(p)$. Therefore, the overall time complexity of our coarse pin distribution algorithm is $O(p \cdot m \cdot n)$.

3.3.3 Detailed Pin Distribution

After coarse pin distribution and net distribution are finished, we know which set of nets is assigned to each routing interval as well as their (evenly distributed) entry/exit points in pin distribution layers. However, the coarse pin distribution is done based on the 2D grid that merged all multiple placement layers into one. The even pin distribution in this 2D grid offers good enough reference points for net distribution. But, it does not consider even pin distribution in each individual routing interval. In addition, it is also possible that the pin capacity for each routing region in each routing interval may be violated. Therefore, the goal of detailed pin distribution is to address these problems in each routing interval so that the subsequent topology generation and layer assignment truly benefit from this even pin distribution. In addition, we use a grid large enough for each routing interval to *legalize* pin location, i.e., each grid point contains only one pin. Since the crosstalk minimization is addressed during the prior steps, the major focus of the detailed pin distribution step is on (i) how far the new location is from the original location obtained from coarse pin distribution and (ii) the total wirelength.

A pseudocode for our detailed pin distribution algorithm is shown in Figure 13. Our

Detailed Pin Distribution

1: **for** (each routing interval i)
2: $\quad m_i = n_i = \lceil\sqrt{total_pin(i)}\rceil$;
3: $\quad DP[i] = m_i \times n_i$ grid;
4: $\quad$ **for** (each pin $p \in i$)
5: $\quad\quad$ assign p to $DP[i]$ using CP result;
6: **for** (each routing interval i)
7: $\quad$ **for** (each net $n \in DP[i]$)
8: $\quad\quad F_n$ = displacement force on n;
9: $\quad\quad$ obtain new location in $DP[i]$ based on F_n;
10: $\quad L_i$ = sort nets in $DP[i]$ in lexicographic order;
11: $\quad$ split L_i and form columns in $DP[i]$;
12: $\quad$ assign pins in $DP[i]$ to clusters $Cl = \{cl_j\}$;
13: $\quad$ **for** (each cluster $cl_j \in CL$)
14: $\quad\quad N_j$ = matching network for cl_j;
15: $\quad\quad$ perform min-cost maximal matching on N_j;
16: $\quad$ obtain refined location for pins in $DP[i]$

Figure 13: Pseudocode for SOP detailed pin distribution algorithm.

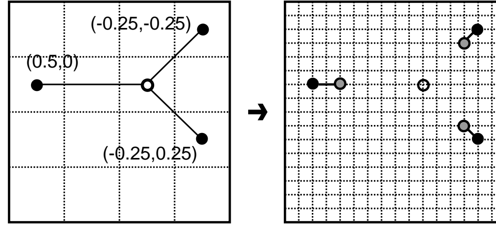

Figure 14: Illustration of detailed pin distribution algorithm. The black and gray nodes respectively denote the old and new pin location, where the white node denotes the center of mass. The numbers denote the displacement force for each pin.

force-directed heuristic algorithm encourages all pins from the same net to be placed closer to the center of mass while minimizing the distance between the old and new pin locations. The grid size for detailed pin distribution for routing interval i ($= DP[i] = m_i \times n_i$) is determined so that each pin can be assigned to a unique grid point. We compute $m_i = n_i = \lceil\sqrt{total_pin(i)}\rceil$ (line 1-3). In addition, we project the coarse pin distribution result ($= CP$) to this new set of grids (line 5). Note that there still exists overlap among the pins in $DP[i]$ at this point even though DP is usually finer than CP. To remove this overlap, we apply an additional force that slightly pulls each pin toward the center of mass. For each pin p in a net n, the *displacement force* (line 8) for x-direction is defined as follows:

$$F_x(p) = \frac{x(M_p) - x(p)}{width(n_p)}$$

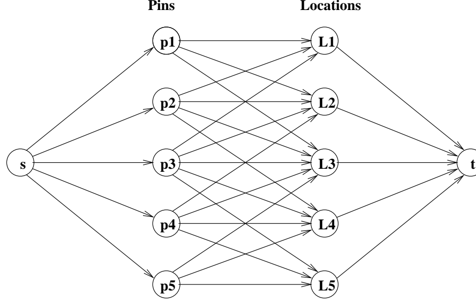

Figure 15: Illustration of the pin-to-location flow network.

where $x(M_p)$ denotes the x-coordinate of M, the center of mass of n, and $width(n_p)$ denotes the width of the bounding box of n. We compute $F_y(p)$ using the y-coordinates. Note that $-1 \leq F_x(p), F_y(p) \leq 1$. The vector $(F_x(p), F_y(p))$ is then added to (x_p, y_p) (line 9). This minor change on the original pin location helps to remove most of the overlap in $DP[i]$ while not increasing the wirelength too much. We then sort the pins based on the lexicographic order of new locations and assign each pin starting from the top-most row in the left-most column (line 10-11). Figure 14 shows an illustration. Because of its simplicity, this deterministic algorithm is quite efficient and effective in reducing the additional wirelength required for pin distribution as well as the total wirelength among all nets, as shown in Section 3.4. The complexity of this algorithm is $O(p)$, where p denotes the total number of pins.

To achieve higher solution quality, a min-cost network flow is used to further reduce wirelength and congestion. A matching network for the problem, $N{=}(P, L, E, c)$ consists of a set of pins $P{=}\{p_i\}$, a set of locations $L{=}\{l_j\}$, a set of assignments, $E{=}\{(u, v) : u \in P \; v \in L\}$, and a cost function $c : E \rightarrow R$. For each pin p_i, a set of assignments is formed by choosing neighboring locations l_j. The cost of the assignment is fixed to be $c(p_i, l_j) = distance(p_i, l_j)$. A maximal matching with minimum cost is acheived by converting the problem to a flow network. In the flow network the capacities of all the edges are set to one. The optimal solution is acheived by running Ford-Fulkerson's algorithm using minimum cost *augmenting* paths. The complexity of the algorithm is $O(p^2 \log p)$. Figure 15 illustrates the flow network formed by the pin-to-location matching network.

In practice, running the network flow algorithm on a problem size of the order of tens

of thousands of nodes can be very time consuming. This is addressed by using a clustering heuristic. The problem size for network flow is reduced to a smaller size by grouping pins into clusters and running the algorithm on clusters. To be effective, clustering should be fast. The clustering technique adopted was to assign pins in the same region to a cluster. Using the approach detailed in this section, a good-quality solution was obtained in reasonable computation time.

3.3.4 SOP Layer Assignment

3.3.4.1 Problem Formulation

For each routing interval i, the routing grid G_i contains all the (redistributed) pins from the top and bottom pin distribution layers ($L_t(i)$ and $L_b(i)$). We generate a Steiner-tree-based routing topology to connect these pins during our topology generation step. The goal is to minimize the maximum sink delay D^{max} as discussed in Section 3.2.2. The routing layer $L_r(i)$ in each routing interval i consists of several layer pairs, where each pair consists of one layer for horizontal wires and another layer for vertical wires. Thus, we can assign an entire rectilinear routing tree to a routing pair. In addition, two trees that intersect can also be assigned to the same routing pair, provided they do not violate the wire capacity of the routing regions involved. The SOP layer assignment problem is to assign each net to a routing layer pair so that the wire capacity constraint is satisfied and the total number of layer pairs used for all routing intervals is minimized.

For each routing interval i, we construct a layer constraint graph (LCG) [52], denoted LCG_i, as follows: corresponding to each net in $n \in RI(i)$ we have a node in LCG_i. Two nodes $x, y \in LCG_i$ have an edge $e = (x, y)$ between them if a net segment $s_x \in x$ and $s_y \in y$ share the same edge in G_i, i.e., s_x and s_y share the same boundary of a routing region. Then we use a node coloring algorithm to assign colors to the nodes in LCG_i such that no two nodes sharing an edge are assigned the same color. Let C^{tot} denote the total number of colors used during the node coloring, and let w denote the wire capacity of the boundary in the routing region. Then, the total number of layer pairs used in this routing interval is

```
SOP Layer Assignment
1:  for (each routing interval i)
2:      LCG_i = layer constraint graph for i;
3:      S = sort all nodes in LCG_i;
4:      color = 0;
5:      for (node j ∈ S)
6:          fin[j] = colors used by neighbors of j;
7:          tmp = lowest used color ∉ fin[j];
8:          if (tmp = ∅)
9:              color = color + 1;
10:             j ← color;
11:         else
12:             j ← tmp;
13:     assign a layer pair to each net;
```

Figure 16: Pseudocode for SOP layer assignment algorithm.

computed as follows:

$$|L_r(i)| = \lceil C^{tot}/w \rceil$$

Last, a node with color $q \cdot w + r$ ($r < w$) is assigned to layer pair q. Let h_{max} and v_{max} respectively denote the maximum number of wires used among all horizontal and vertical edges in G_i. Then, the following is a lower bound on the number of layers used in $L_r(i)$: $|L_r(i)| \geq \max\{h_{max}, v_{max}\}/w$.

We use the existing RSA/G heuristic [32] to optimize the performance of the routing topology. The *Minimum Shortest Path Steiner Arborescence* (MSPSA) generated by the heuristic guarantees the shortest path between every source-to-sink path and minimal overall weight of the tree. This routing topology is useful in high-performance SOP design because of its performance guarantee.

3.3.4.2 Layer Assignment Algorithm

Figure 16 shows the SOP layer assignment algorithm that includes our coloring heuristic. We first sort all nodes in LCG_i in a decreasing order of the number of their neighbors (line 3). Let $fin[n]$ denote the set of colors used by the neighbors of n (line 6). We visit the nodes in the sorted order (line 5). In case there exists a used color that is not included in $fin[n]$ (line 7), we assign this color to n (line 12). In case there exist multiple colors that satisfy this condition, we use the lowest color. Otherwise, we introduce a new color and

assign it to n (line 9-10). Last, we assign a layer pair to each net based on its color (line 13). Despite its simplicity, this greedy algorithm provides results that are very close to the lower bound on the total number of layers used, as demonstrated in Section 3.4. The complexity of the SOP layer assignment algorithm is $O(N \log N)$, where N is the total number of nets.

The generation of the layer conflict graph takes $O(N^2)$ time. For larger circuits the runtime and memory may be issues that cannot be ignored. These issues are resolved by using net clustering. The nets are clustered into smaller sizes and layer assignment is performed on each of the clusters. The sum of the layer usage for each cluster gives a valid layer count. High connectivity between clusters ensures good quality. The heuristic used in this work sorts the nets based on their edge length and assigns them sequentially to the clusters. The effectiveness of the clustering heuristic can be measured by comparing the layer counts with the *theoretical* minimum. Clustering was used for two of the largest benchmarks and the size of the clusters was fixed to four.

3.3.5 SOP Channel Assignment

We first present an overview of our approach, followed by the problem formulation for SOP channel assignment and local routing. We use an existing method for the local routing, so we focus on discussing our channel assignment heuristic in detail.

3.3.5.1 Overview of the Approach

Our prior topology generation and layer assignment steps focus on the connections among the distributed pins in the pin distribution layers using the routing layer pairs. After these steps are finished, two kinds of connections remain: (i) connections between a pair of neighboring pin distribution and placement layers, i.e., connection between the original and distributed pins, and (ii) connections between two non-neighboring pin distribution layers, i.e., feed-through via insertion. For both types of connections, the routing channels in the placement layers are used. Our strategy is to finish these remaining connections in two steps: channel assignment and local routing. During the channel assignment step, each pin in the pin distribution layer is mapped to a routing channel in the neighboring placement layer. In addition, each pin from an x-net that needs to penetrate a placement layer is also mapped

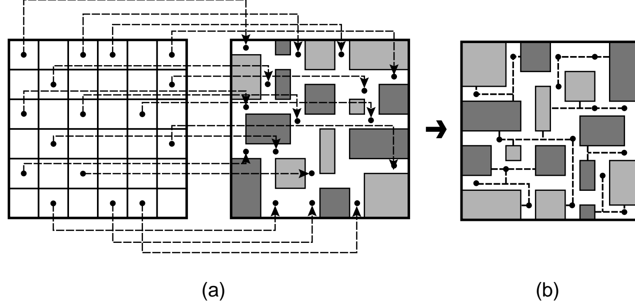

Figure 17: Illustration of (a) channel assignment, (b) local routing.

to a routing channel in the placement layer. Last, we generate a routing topology for each pin-to-channel connection and assign it to a routing layer pair in the pin distribution layer. During the local routing, we finish connections between the pins now located in the routing channels and the pins on the block boundaries. An illustration is shown in Figure 17.

The channel assignment result has a direct impact on the actual number of pin distribution layers used, so layer minimization under the routing channel capacity constraint is the primary goal. Our secondary objective is to reduce the total wirelength and the total number of bends that would necessitate the use of secondary vias. All the pin-to-channel connections are two-pin nets, so we use an L-shaped routing topology for them. The local routing step is done entirely in the routing channels of the placement layer. Since each placement layer consists of a pair of vertical/horizontal routing layers, various rectilinear routing topologies with possible intersection can be routed. In case the boundary information is not available for some soft blocks, we determine pin locations along the boundaries as well.

Problem Formulation For each placement layer $L_p(i)$, let $X(i)$ denote the set of pins that needs to be mapped to a routing channel in $L_p(i)$. $X(i)$ contains pins from $L_b(i-1)$ and $L_t(i)$. The pins in $X(i)$ are grouped into two sets: *terminal pin set* $P_t(i)$ for the pins that have terminals in $L_p(i)$ and *feed-through pin set* $P_f(i)$ for the pairs of pins that require feed-through vias to penetrate $L_p(i)$. Let $C(i)$ denote the set of routing channels in $L_p(i)$. Each channel $c \in C(i)$ is associated with the pin capacity constraint. The goal of the *SOP Channel Assignment Problem* is to map each pin $p \in X_i$ to a routing channel $c \in C(i)$

for $1 \leq i \leq K$ and finish the p-to-c connection while satisfying the pin capacity constraint A, i.e., $|C(i)| < A$. For each pair of pins $(p_1, p_2) \in P_f(i)$, we map p_1 and p_2 to the same channel $c \in C(i)$. Let $|L_t(i)|$ denote the number of layers used to finish the connection between pins from $L_p(i)$ and $L_t(i)$, and let $|L_b(i)|$ denote the number of layers used to finish the connection between pins from $L_p(i+1)$ and $L_b(i)$. The objective of SOP channel assignment is to minimize the following cost function:

$$w_1 \cdot \sum_{1 \leq i \leq K} (|L_t(i)| + |L_b(i)|) + \sum_{n \in NL} (w_2 \cdot wl_n + w_3 \cdot via_n)$$

Each placement layer is modeled with the floor connection graph (FCG) [28], as illustrated in Figure 18. The input to the *SOP Local Routing Problem* is a set of two-pin connections, where each connection is between a pin located in a routing channel and the other pin located along a block boundary. Each routing channel is associated with the wire capacity constraint, and the pin assignment edge has a pin capacity for each boundary of soft block. The goal of SOP local routing is to (i) finish the routing for the given set of two-pin connections while satisfying the wire capacity constraint, and (ii) decide the location of pins for soft blocks along their boundaries. The objective is to minimize the total wirelength, maximize pin demand, and maximize routing demand. Pin demand is the number of nets using the same block boundary, and routing demand is the number of nets using the same routing region. Both objectives have a direct relation to congestion in a 3D structure of SOP.

We use the standard two-phase method for local routing: maze routing followed by congestion-driven rip-up-and-reroute [63]. During the first phase, a shortest weighted path in FCG is found for each connection while ignoring the actual channel usage, where the weight is based on the combination of wirelength and via. During the second phase, we rip-up nets that use the most congested channels and reroute them to alleviate the congestion problem.

SOP Channel Assignment A pseudocode for the SOP channel assignment algorithm is shown in Figure 18. We visit each placement layer and assign the feed-through pins and terminal pins to the channels. In addition, an L-shaped routing topology for each

pin-to-channel connection is constructed in a 2D grid $G(i)$ (line 3). The signal delay of feed-through vias is larger than that of other types of vias. Since each channel is under a capacity constraint, it is important to assign the feed-through pins to the nearest channels first. Therefore, our strategy is to perform channel assignment for the feed-through pins first to minimize the delay of x-nets that require feed-through vias. In addition, we give priority to the pins that are included in long nets. Thus, we sort the pins based on the wirelength (line 4). Our heuristic algorithm assigns pins to channels based on the cost of mapping—we seek a channel with the best mapping cost for a given pin (line 6-8). We compute the cost of mapping for a given pin p and a channel c as follows:

$$cost(p, c) = \frac{room(c)}{dist(p, c) \cdot bend(p, c) \cdot cong(p, c)}$$

where $room(c)$ denotes the number of pins c can accommodate until it violates the capacity constraint, $dist(p, c)$ is the Manhattan distance between p and c, $bend(p, c)$ is the number of bends in the connection between p and c, and $cong(p, c)$ denotes the total number of existing connections along the proposed L-shaped route. We choose the channel with the maximum $cost(p, c)$. Note that a channel is represented with a line instead of a point. Thus, $distance(p, c)$ and $bend(p, c)$ are based on the shortest connection between p and any point on c. Upon a pin-to-channel mapping, we update the usage of channel in $C(i)$ and edges in $G(i)$ (line 10). After the channel assignment and topology generation for all pin-to-channel connections are finished, we perform layer assignment using our coloring heuristic presented in Section 3.3.4.2 and compute the total number of layers used. The complexity of the SOP channel assignment algorithm is $O(|P| \cdot |C|)$, where P and C respectively denote the total number of pins and the channels in the given SOP design.

3.4 Experimental Results

We implemented our algorithms in C++/STL and ran our experiments on Linux Beowulf clusters. We tested our algorithms with two sets of benchmarks. The first set is from the standard GSRC floorplan circuits, where the hard blocks are placed into four-layer SOPs using our SOP floorplanner [81]. The second set, named the GT benchmark, was synthesized

```
SOP Channel Assignment
1:  for (each placement layer i)
2:      C(i) = channels in i;
3:      G(i) = 2D routing grid;
4:      S = sorted feed-through pins;
5:      for (each pin p ∈ S)
6:          best = ∅;
7:          for (each channel c ∈ C(i));
8:              compute cost(p, c) and update best;
9:          T_p = p-to-best routing topology in G(i);
10:         update channel and edge usage in G(i);
11:     repeat line 4-10 for terminal pin set;
12:     perform layer assignment on G(i);
```

Figure 18: Pseudocode for SOP channel assignment algorithm.

Table 3: Benchmark characteristics of GSRC and GT circuits. wcap denotes the wire capacity of each boundary in the routing tiles. CP and DP respectively denote the dimension of 2D grids used during our coarse and detailed pin distribution steps.

ckts	blk	i-net	x-net	pin	wcap	CP	DP
n30	30	97	252	723	10	3 x 3	24 x 22
n50	50	76	409	1050	10	4 x 4	28 x 26
n100	100	189	696	1873	10	5 x 5	34 x 35
n200	200	297	1288	3599	10	6 x 6	46 x 48
n300	300	339	1554	4358	10	8 x 8	51 x 50
gt50	50	2102	7317	20835	20	4 x 4	110 x 101
gt100	100	4361	11823	36033	20	6 x 6	138 x 140
gt300	300	4534	15538	45901	20	8 x 8	163 x 165
gt1000	1000	6908	25561	79830	20	15 x 15	218 x 219
gt1500	1500	6554	28171	87785	20	15 x 15	225 x 227

from the IBM circuits [4], where we use our multi-level partitioner [38] to divide the gate-level netlist into multiple blocks first and then use our SOP floorplanner [81] to floorplan them again to four-layer SOPs. Table 3 shows the characteristics of the GSRC and GT benchmark designs. The GSRC benchmarks are small to medium sized in terms of both the number of blocks and nets. The GT benchmarks contain a medium to large number of blocks with *dense* netlists. We note that in both cases the number of i-nets is only a small fraction of the total nets. The area information of the benchmarks is shown in Table 4. The final area refers to the overall footprint area of the four-layer floorplan, which is determined by the maximum width and height among the individual floorplan layers.

Table 4: Area information of the four-layer SOP floorplanning for GSRC and GT benchmark designs. The min and max respectively denote the dimension of the minimum and maximum floorplan area, whereas the final column denotes the dimension of the overall footprint.

ckts	min	max	final
n30	239 x 218	251 x 233	251 x 237
n50	226 x 212	247 x 228	247 x 230
n100	204 x 211	221 x 231	221 x 231
n200	199 x 210	218 x 225	218 x 227
n300	262 x 252	289 x 274	289 x 274
gt50	88 x 108	76 x 139	97 x 139
gt100	111 x 117	131 x 143	131 x 143
gt300	125 x 124	127 x 161	127 x 161
gt1000	128 x 128	122 x 172	128 x 172
gt1500	132 x 143	136 x 183	137 x 183

In Table 5 we compare pin redistribution results. Under the DPD columns, we perform detailed pin distribution (DPD) presented in Section 3.3.3 only, where we skip coarse pin distribution (CPD) presented in Section 3.3.2 and assign all i-nets to the routing interval below for net distribution. Under the CPD+DPD, we perform CPD using the algorithm, assign all i-nets to the routing interval determined by net distribution, and perform DPD. DPD serves as our baseline, where CPD+DPD demonstrates the impact of our coarse pin distribution. In all cases, we perform detailed pin distribution to legalize the pin location, i.e., remove overlaps among the pins. We use the following metrics to evaluate our solutions: wirelength between the original and the new location (dw), total wirelength (wl), crosstalk (xt), and total number of layer pairs used (lyr) in the routing layers. The displacement (dw) and wirelength (wl) results are scaled by 10^6, and the time reported is the average runtime among the GSRC/GT circuits. From the comparison between DPD and CPD+DPD, we note that the displacement result (dw) increases by an average of 1%. However, CPD lowers the total wirelength (wl) consistently by 12% on average and the number of layers (lyr) increases by 1% on average. The metric dw is the measure of routing from the redistributed pins to the originating pins.

In Table 6 we show our topology generation (RSA/G) and layer assignment (LAYER) results. We used the technology parameters for the 0.13 μ process for Elmore delay computation. Specifically, the driver resistance of 29.4 $k\Omega$, input capacitance of 0.050 fF,

Table 5: SOP pin redistribution results using our coarse and detailed pin distribution algorithms. We report the wirelength between the original and the new location (dw), total wirelength (wl), crosstalk (xt) and total number of layer pairs used (lyr) in the routing layers.

	DPD				CPD+DPD			
ckt	dw	wl	xt	lyr	dw	wl	xt	lyr
n30	0.15	0.07	0	3	0.16	0.07	0	3
n50	0.23	0.11	5	4	0.24	0.10	5	4
n100	0.41	0.18	10	6	0.41	0.16	10	6
n200	0.76	0.34	51	9	0.80	0.29	52	9
n300	1.13	0.50	219	11	1.18	0.44	220	11
gt50	1.96	0.87	92	21	1.98	0.71	90	24
gt100	4.10	1.78	441	34	4.08	1.44	446	35
gt300	5.71	2.48	2200	43	5.08	2.03	2186	46
gt1000	10.53	4.53	13365	58	10.75	3.72	13496	57
gt1500	12.55	5.58	14296	63	12.78	4.43	14366	61
TIME	529				547			

unit-length resistance of 0.82 $\Omega/\mu m$, and unit-length capacitance of 0.24 $fF/\mu m$ are used. We report the total wirelength (wl), Elmore delay of the nets with maximum sink delay (dly), and the lower bound and the actual number of layers used for the top-most routing interval. In general, GSRC benchmarks have a greater delay than GT benchmarks because of the larger average wirelength. Our layer assignment algorithm presented in Section 3.3.4.2 is able to achieve results very close to the lower bounds discussed in Section 3.3.4.1. For the GT circuits, the layer assignment results are within 10% of the lower bound. For the GSRC circuits, we were able to achieve results equal to the lower bound.

Our channel assignment results are shown in Table 7. The baseline case is where we optimize the wirelength only. We then compare it to our multi-objective channel assignment algorithm that simultaneously optimizes the wirelength, via, and layer usage. Our comparison indicates that the number of layers is consistently and significantly reduced, especially for the larger GT benchmarks, where an average improvement of 38% is observed. In the case of the second largest benchmark gt1000, we achieved a 59% improvement. These savings on the layer usage come at the cost of an increase in wirelength and vias. The average increase in wirelength is 20% and 23% for GSRC and GT benchmarks, respectively.

Table 6: Topology generation and layer assignment results. We report the total wirelength (wl), Elmore delay of the nets with maximum sink delay (dly), and the lower bound (low) and the actual number (lyr) of layers used.

	RSA/G		LAYER	
ckt	wl	dly	low	lyr
n30	389.0	2.759	2	2
n50	404.6	2.872	3	3
n100	361.6	2.564	4	4
n200	378.7	2.687	6	6
n300	506.7	3.599	7	7
gt50	199	1.410	16	16
gt100	261	1.849	28	28
gt300	257	1.821	36	36
gt1000	331	2.346	26	30
gt1500	374	2.649	30	34
TIME	90		150	

The average increase in via usage is 85% of the benchmarks. The number of layers, wirelength, and via are conflicting objectives. We noted that the channel assignment result is very sensitive to the weighting constants among the objectives used in our cost function. This indicates that the solution space of the channel assignment problem offers many useful trade-off points. The primary objective for optimized channel assignment was the number of layers and wirelength. Via was the secondary objective.

Table 8 reports our local routing results. We report the wirelength and maximum and average routing demand, as well as the standard deviation. Our baseline is the local routing optimized for wirelength only. We then compare it with our multi-objective local routing algorithm that simultaneously optimizes wirelength and routing demands. In both cases, the same pin demand constraint is imposed. We note that the improvement of our multi-objective algorithm over the baseline is significant, especially for GSRC circuits—the routing demands were reduced by 35% on average, while the wirelength increased by only 14%. In addition, we reduced the routing demands for the GT benchmarks by 37% on average, with a wirelength increase of 10%. In our largest benchmarks (gt1500), our routing demand reduction is the largest (54%), which comes with the maximum increase in wirelength (23%). This again indicates that the local routing result is very sensitive to the weighting constants among the objectives used in our cost function. The lower standard

Table 7: SOP channel assignment results. We report the layer usage, wirelength, and via for the baseline (wirelength minimization only) and multi-objective algorithms.

	wl-only			lyr+wl+via		
ckt	lyr	wl	via	lyr	wl	via
n30	8	0.030	107	8	0.033	144
n50	10	0.036	157	8	0.040	241
n100	9	0.038	303	8	0.047	470
n200	11	0.059	595	10	0.078	1231
n300	13	0.067	728	12	0.084	1292
gt50	11	0.383	3647	9	0.543	7772
gt100	16	0.893	7263	11	1.117	16133
gt300	25	1.291	12370	15	1.490	29285
gt1000	59	2.705	30869	24	3.220	68555
gt1500	63	3.430	38869	35	3.971	81093
TIME	194			210		

deviation of our multi-objective algorithm indicates that the routing demand is more evenly distributed (lower congestion) compared to the wirelength-only case.

3.5 Conclusions

In this work, a new paradigm for global routing for SOP was introduced that looks into various aspects such as crosstalk, wirelength, and layer minimization. A modular approach toward global routing is advocated because it facilitates the handling of various objective functions efficiently. The impact of various algorithms such as coarse pin distribution, net distribution and detailed pin distribution is shown on the overall global routing flow of SOP. Experimental results show that our algorithms efficiently handle the various objectives. As emphasized in this chapter, a fresh approach toward the SOP physical design to handle issues unique to this emergent technology is needed. Our work is an attempt in this direction. An interesting direction in this research is the optical routing for SOP and its impact on the various objectives.

Table 8: SOP local routing results for the baseline (wirelength minimization only) and multi-objective algorithms. We report the wirelength (wl), maximum (max) and average (ave) routing demand as well as the standard deviation (dev).

	wl-only				wl+rd			
ckt	wl	max	avg	dev	wl	max	avg	dev
n30	0.080	50	7.75	9.22	0.081	39	7.53	7.57
n50	0.129	68	10.18	11.33	0.134	61	9.96	9.53
n100	0.251	206	15.22	20.12	0.284	117	15.23	15.08
n200	0.514	316	22.35	29.16	0.592	146	22.77	19.65
n300	0.781	318	22.35	29.41	0.917	146	23.18	19.45
gt50	1.414	3428	296.70	455.76	1.678	2100	311.03	358.58
gt100	2.780	4479	341.85	604.55	3.314	3308	364.70	459.78
gt300	5.121	6702	331.11	609.49	6.269	3721	362.94	442.70
gt1000	8.143	6460	264.71	500.61	9.682	3757	305.09	383.31
gt1500	9.593	9221	256.47	504.77	11.801	4259	292.94	377.14
TIME	547				573			

CHAPTER IV

3D OPTICAL ROUTING

The System-on-Chip (SOC) paradigm is a new system integration approach, where not only more and more transistors but also various mixed-signal active and passive components are integrated into a single chip. However, the systems community has begun to realize that SOC presents fundamental, engineering, and investment limits [89]. This has led to the 3D System-On-Package (SOP), where the package and not the chip becomes the medium for system integration. Optoelectronics, which today finds use primarily in the back planes and high-speed board interconnects, is fast moving towards SOP to obtain high I/O and high-speed interconnections. Optical interconnects which are replacing copper interconnects address both the resistance and crosstalk issues of electronic ICs [12]. Wide area, high speed optical clock, and data transport simplify the digital architecture because fewer parallel transmission lines are needed for the same bandwidth. Also, optical links have low crosstalk and are not susceptible to electromagnetic interference (EMI) noise, thus they reduce the need for decoupling capacitors. In addition, Wavelength Division Multiplexing (WDM) allows a single waveguide to be shared among multiple optical interconnects, thereby significantly reducing the area used by an interconnect.

This chapter has been organized as follows: Section 4.1 presents an overview of the optical routing technology for SOP. Section 4.2 presents the problem formulation. Section 4.3 presents our 3D SOP optical routing algorithm. Experimental results are shown in Section 4.4, and we conclude in Section 4.5.

4.1 SOP Optical Interconnect Technology

Basic technologies required for a fully integrated digital-optical micro-system are shown in Figure 19. The text boxes in the figure show the enabling integration technologies that have been developed to achieve full digital-optical functionality. Each high-frequency output port

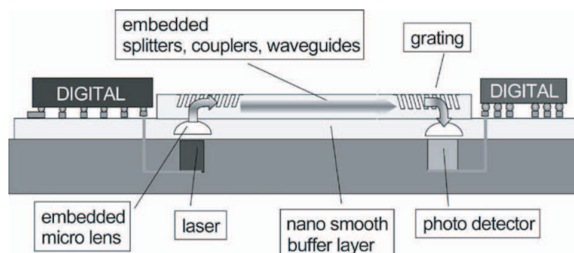

Figure 19: Optical routing in System-On-Package.

of the processor modulates a specific laser in an array. The digitized optical signal is coupled through a micro-lens array, to the optical signal distribution network, which comprises of waveguides, splitters, couplers, gratings, etc., and is transported to its destination. The optical signal is detected by a specific photodiode in an array of optical receivers, and converted to an electrical signal that is input to a specific port of the receiving processor. The signaling is bi-directional and nonblocking.

Optical signals are coupled in and out of the optical transport network by a number of means such as gratings, lenses, waveguide end-mirrors, directional couplers, and evanescent coupling. The entire opto/digital micro-system is built directly on the buffer layer which is fabricated on low cost FR-4 and APPE boards.

There are different types of signal losses in optical routing. First, internal signal losses associated are associated with certain optical devices. For example, the splitting loss of an optical splitter is about 0.4 dB per splitter, as demonstrated in [8]. Optical waveguides have internal as well as external losses. Internal losses are primarily due to material absorption and propagation. Absorption effects in general are more prominent at the higher wavelengths, while propagation losses in waveguides are estimated to be 0.36 dB/cm at a wavelength of 1.3um [26]. The length of a waveguide is critical from a physical design point of view as a greater length corresponds to a greater propagation loss. Extrinsic losses typically result from scattering off profile roughness and defects. Second, signal losses resulting from optical waveguide bending. For example, when a signal passes through a bent optical waveguide, the tangential velocity of the signal in a cladding layer exceeds the velocity of light. Hence, this portion cannot stay in phase and splits away from the guided mode signal, and results in signal loss. As a result, a greater bending radius corresponds to a lower signal

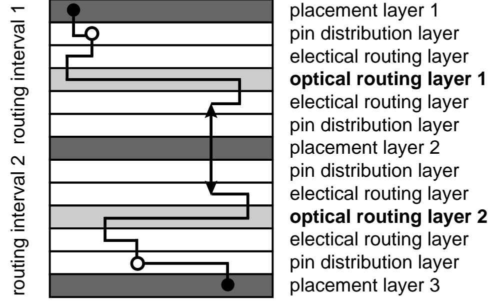

Figure 20: Illustration of the layer structure and routing resource in SOP. Black and white dots respectively denote the original and redistributed pins. The arrowed line denotes feed-through via.

loss. Thus, we impose a limit on the maximum waveguide length and the number of bends while constructing waveguides in order to minimize signal loss.

4.2 Problem Formulation

The layer structure in multi-layer SOP is illustrated in Figure 20. The *placement layers* contain the blocks (such as ICs, embedded passives, opto-electric components, etc), which from the point of view of physical design are just rectangular blocks with pins along the boundary. The interval between two adjacent placement layers is called the *routing interval*. A routing interval contains a set of *routing layers* sandwiched between *pin distribution layers*. At the core of the routing layer set is a single *optical routing layer* that contains both waveguides and optical modules such as laser, lens, detector, splitter, and grating. The pin distribution layers in each routing interval are used to evenly distribute pins from the nets that are assigned to this interval. Then, these evenly distributed pins are connected using the routing layer pairs. A *feed-through via* is used to connect two pin distribution layers from different routing intervals. The nets which have all their terminals in the same placement layer are called *i-nets*, while the ones having terminals in different placement layers are *x-nets*.

The layer structure in multi-layer SOP is illustrated in Figure 6. Given a 3D SOP placement, a set of nets N, and the number of placement layers L_p, the *3D SOP Optical Routing Problem* can be defined formally as follows: generate a routing topology for each net

$n \in N$, assign n to *optical* or *electrical* routing layer or both, and assign all pins of n to legal locations. All conflicting nets are assigned to a different routing layer. The number of optical layers is fixed to *one* per routing interval. Let L_r be the number of electric routing layers, D_{max} be the maximum net delay, and wl_n be the wirelength of net n. The computation of electrical interconnect is based on Elmore delay model: $D_{elec}(s, v) = \sum_{k \in P} R_k \cdot C_k$, where P is the path from source node s to sink v, R_k is the local resistance seen at node k, C_k is the downstream capacitance seen at node k. The computation of an optical interconnect is based on the linear delay model: $D_{opt}(s, v) = b \times l$, where l is the path length between nodes s and v, and b is the constant representing delay per unit length in the optical interconnect. Thus, the delay of any source-to-sink interconnect is the sum of D_{elec} and D_{opt} parts. The objective of SOP optical routing problem is to minimize the total wirelength and routing layers used while keeping the D_{max} within the specified value.

The optical waveguide structure consists of an electrical to optical convertor, the optical medium, and optical-to-electrical convertor. The convertors have intrinsic delays are modeled after buffers with input capacitance and output resistance. The optical signal delay depends linearly on the length of the waveguide. The delay of the line containing a waveguide is then given by its Elmore delay (see figure 21), which is:

$$
\begin{aligned}
D(x, y) = R_d(cx + C_w) + rx(\frac{cx}{2} + C_w) + \\
R_w(c(L - y) + C_L) + r(L - y)(\frac{c}{2}(L - y) + C_L) + \\
t_d + \frac{1}{v_w}(y - x)
\end{aligned}
$$

where R_d is the driver resistance, C_L is the load capacitance, r, and c are the unit wire resistance and capacitance respectively, R_w is the opto-device output resistance, C_w is the opto-device input capacitance, t_d is the opto-devices intrinsic delay, v_w is the waveguide signal velocity, L is the length of the wire, and x, and y are the start and end points of the waveguide. Section 4.4 presents the values of these parameters used in our experiment.

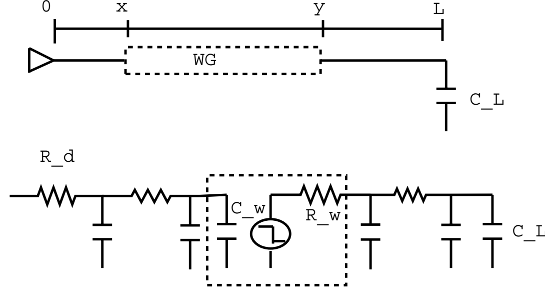

Figure 21: RC model of an opto-electrical interconnect. The dotted box denotes the waveguide.

4.3 SOP Optical Routing Algorithm

We develop the following algorithms for various steps required during the SOP optical routing process: (i) construction of optical waveguides based on performance and congestion consideration, (ii) optimum mapping of net-to-waveguide using maximum-flow minimum-cost network flow model, (iii) ripping and rerouting of existing nets to relieve congestion and make maximum utilization of waveguides.

4.3.1 Overview of the Algorithm

Compared with the conventional "single device multiple routing layer" model as used in IC, PCB, and MCM routing, SOP routing is more general in that it requires "multiple device multiple routing layer" model. Because of the complexity involved in this 3D routing process, we divide the entire process into the following steps:

1. Pin redistribution: We first determine which set of net segments are assigned to each routing interval. The pins from these nets are then evenly distributed in the pin distribution layers.

2. Topology generation: Steiner trees are generated for all nets in each routing interval so that the performance of the routed design is optimized.

3. Waveguide construction: a planar set of optical waveguides is constructed based on the topology generation in each routing interval. Each waveguide has a maximum of single bend and its maximum length is also constrained.

4. Optical net selection: a subset of nets that makes the best use of the waveguides is chosen. Each net is assigned to a unique waveguide based on its proximity and capacity.

5. Optical routing: each optical net is rerouted so that it uses the entire waveguide to which it is assigned to while minimizing the length of its electrical portion.

6. Layer assignment: the routed nets are assigned to a unique routing pair in the routing layer so that the total number of layers used is minimized.

7. Local routing: the location of feed-through via for each net in the routing channel is determined. In addition, we finish the connection between the original and distributed pins.

We use [69] for pin redistribution, and layer assignment, and [32] for topology generation. We use the congestion-driven rip-up-and-reroute [63] for local routing. Therefore, the focus of this chapter is to present heuristics for the waveguide construction, optical net selection, and optical routing.

4.3.2 Waveguide Construction Algorithm

Our goal during optical waveguide construction is to choose the most optimum nets for optical routing based on their locations and properties such as wirelength, delays, and number of bends. In our model, each routing layer is represented as a grid graph $G(V, E)$, where each node specifies a region in the layer and each edge represents the adjacency between neighboring regions. The nodes and edges are characterized by capacities. Specifically, the nodes are annotated with optical device capacities and the edge capacities correspond to number of waveguides. A simple calculation based on the dimensions given in [12] for a 1 *cm* by 1 *cm* package, represented as a 10×10 routing grid, results in the node capacity of 4 and edge capacity can vary between 4 to 8 [22]. In this work we assume the edge capacity to be 4. In addition, each waveguide can carry up to a maximum of 10 multiplexed signals. This means a maximum of 40 nets can be assigned to each routing edge in G.

4.3.2.1 Timing-Driven Waveguides

The construction of the timing-driven waveguides considers the timing critical nets. The aim of the waveguide construction primarily is to reduce the worst delays of the design and secondarily to improve electrical wirelength and routing layers cost. The timing criticality of a net is defined as the ratio between the net delay and the worst delay among all nets in the routing layer. The delay values are available since nets have already been routed using electrical routing resources. A net is said to be a critical net if its criticality exceeds a predefined *criticality level*.

The pseudocode of the timing-driven waveguide construction algorithm is shown in figure 22. Our timing-driven approach is based on the following important observation:

Lemma 1 *Given a multi-pin net n routed with pure electrical wires, converting a part of n into optical will not degrade the performance if (i) the overall topology is maintained, and (i) the delay reduction caused by optical waveguides exceeds the delay penalty caused by optical modules.*

Thus, our timing-driven approach is to preserve the routing topology of the timing critical nets and convert parts of the nets into optical only when the overall delay reduction is positive.

```
Timing-driven Waveguide Construction
1:  W = ∅;
2:  N = selected critical nets;
3:  sort N by criticality;
4:  while (N is not empty)
5:      B = set of segmented root-to-sink paths;
6:      for each  b ∈ B do
7:          L = candidate waveguide locations;
8:          l_i = minimum-cost waveguide location;
9:          P = constrained length waveguide at l_i;
10:         add P to the set of waveguides W;
11:     update waveguide usages;
12: return W;
```

Figure 22: Pseudocode for timing-driven waveguide construction.

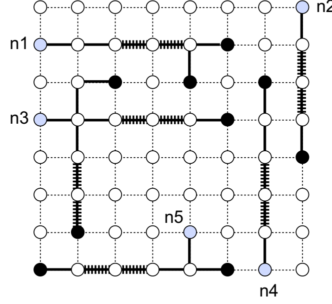

Figure 23: Timing-driven waveguides for five timing critical nets. The nets n_1, n_3, and n_5 are multi-pin nets. Gray and black nodes respectively denote the source and sink nodes. The waveguides are shown as dotted lines.

The algorithm starts with a set of pre-selected timing critical nets, which are sorted by their criticalities. The paths for each of the critical sinks (sinks with delays above a threshold) of the net are computed. These paths are segmented at their branch points, whereas the set B is a set of no-branch segments.[1] The candidate locations for waveguides are determined for each of the segmented paths. The optimal delay location for the waveguide is achieved by differentiating $D(x, y)$ with respect to x and solving it for zero.

$$\frac{\partial D(x,y)}{\partial x} = (cR_d - cR_w\frac{\partial y}{\partial x}) + rc(x - (L - y)\frac{\partial y}{\partial x}) + \\ (rC_w - rC_L\frac{\partial y}{\partial x}) + \frac{1}{v_w}(\frac{\partial y}{\partial x} - 1)$$

The optimal delay location is close to the center of the line if $R_d \approx R_w$ and $C_L \approx C_w$ and the size of the waveguide is constrained (i.e $y \approx x + constant$). The cost of waveguide insertion at each of the locations (l_i) is calculated based on the number of bends and on its distance from the midpoint (mid_i) of the path. The node and edge usages are also taken into consideration for device and waveguide allocation. The cost of the location is given by:

$$cost(l_i) = \alpha \cdot |l_i - mid_i| + \beta \cdot bends + \gamma \cdot \frac{usage}{capacity}$$

where α, β, and γ are the weights for the cost factors. In our experiments, α and γ were chosen to be 1, whereas β was chosen to be 2. The waveguide is constructed at the location with the minimum cost (see Figure 23). It is possible that waveguides cross each other at some nodes. However, such waveguides can be split at those junction points, and

[1] We assume that our timing-driven optical waveguides allow up to a single bend but prohibit branching.

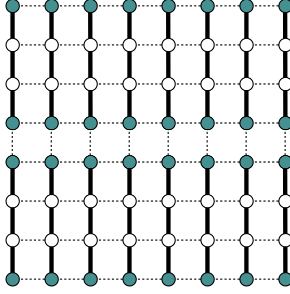

Figure 24: Standard waveguides. The gray nodes indicate extry/exit points for each waveguide.

Congestion-driven Waveguide Construction

1: $W = \emptyset$;
2: calculate routing edge usage;
3: L = set of all heavily used edges;
4: **while** (L is not empty)
5: $P = \emptyset$;
6: **while** ($|P| < max_size$)
7: e = edge $\in L$ located closest to P;
8: **if** (adding e to P does not cause a bend)
9: add e to P;
10: fill gap between e and P with edges in L;
11: add P to W if W is still planar;
12: **return** W;

Figure 25: Pseudocode for congestion-driven waveguide construction.

opto-devices plus a local electrical connection can be used to bridge the broken optical connections. A small penalty is incurred at those points for opto-conversions and local electrical wiring. Therefore, the waveguides constructed are guaranteed to be planar.

4.3.2.2 Congestion-Driven Waveguides

The objectives in the design of congestion-driven waveguides is to reduce the costs of electrical wiring and routing layers while not compromising design performance. Heavily used edges are obtained by noting the number of nets that pass through each edge and edges that surpass a certain threshold are chosen. In our *Customized Waveguide Construction*, heavily used edges are noted in each routing interval and planar waveguides are constructed in those areas to reduce the number of electrical nets using these edges and thus reduce congestion.

Figure 25 shows our customized waveguide construction algorithm. Our greedy approach starts with a set of heavily used routing edges L from a given global routing solution. Our "cluster growth" based method attempts to grow the current waveguide P by adding an edge e from L based on the distance between e and P. The waveguide has to be a straight line with no bends any time[2], and its maximum length is also limited. In case there exists a discontinuity between e and P, we fill the gap using any intermediate edges to keep P a straight line. Upon completion of constructing a single waveguide, we check for the planarity, i.e., we check to see if adding P to the current set of waveguide W does not cause any crossing. Last, the set W contains all legal and optimized waveguides. A minimum length of 2 edges and a maximum length of 4 edges is allowed for any waveguide in our experiment. This is because single-edge waveguides have high cost, whereas waveguides with length greater than 4 edges suffer from signal loss, and low utilization, and so are impractical. The described methods are illustrated in Figure 24.

4.3.2.3 Standard Waveguides

The method described for the congestion-driven waveguides requires a different layout for each optical layer that is generated for a specific routing interval. From a fabrication point of view, this would be more time consuming as there exists no standard design for any optical layer. In our *Standard Waveguide Construction*, we address this issue by choosing a standard subset of edges in each routing interval irrespective of their individual usages. This approach, however, can result in low utilization of waveguides making it an inefficient solution. Our standard waveguide construction is straightforward in that a fixed subset of edges is used for the waveguides for every optical layer in the SOP. As shown in Figure 24, each waveguide has a length of 4 edges and there are 16 waveguides in each routing layer. The waveguides could also have been constructed using horizontal edges, but that would make a negligible difference in the overall results.

[2]Note that we do not allow any bends in our congestion-driven waveguides unlike timing-driven waveguides. This is because the strict requirement reduces the signal loss significantly but has very little impact on the utilization under the congestion objective.

4.3.3 Optical Net Selection Algorithm

The next step is to select a set of nets and assign them to the waveguides we constructed. We use a minimum-cost maximum-flow network flow model to select the most optimal nets to be assigned to each waveguide. The goal of this model is to minimize the overall cost while respecting the individual edge capacities. This model is illustrated in Figure 26, where a source node is connected to net nodes (each net is assigned a node) and all waveguide nodes (each waveguide is assigned a node) are connected to a sink node. A net node is connected to a particular waveguide node if all the routing edges in that waveguide are contained in the net. Each edge in this flow graph is assigned a capacity and cost value. We assign a capacity of 40 and cost of 0 to every waveguide-to-sink edge because a maximum of 40 nets can use the waveguide and no waveguide costs different from another. The minimum length of nets considered for optical routing is 4 as it needs to utilize *all* edges of its waveguide. Each net-to-waveguide edge has a cost equal to number of edges in the net that are not in any waveguide because such edges will have to be routed electrically and it is better to minimize these connections in an optically routed net. Finally, the flow analysis provides flow values for each edge that tells us how useful it is to retain that edge in our final model. For example, in case of a waveguide with more than 40 nets mapped to it, we utilize these flow values to only retain nets with 40 maximum flow values.

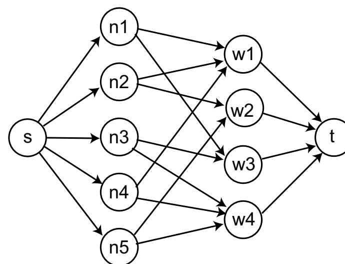

Figure 26: Flow network for opto-net selection.

4.3.4 Optical Routing

The customized waveguides either cover timing critical net segments or heavily congested areas of the routing layer. Hence after routing the net through the optical waveguides, it is desirable for the electrical part of the net to *not* go through these optical regions (see Figure

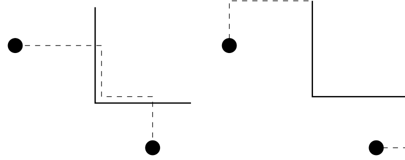

Figure 27: Rerouting of the net to go through the optical channel. The figure shows electrical routes (dashed line) before and after rerouting through the optical waveguide (bold line).

27). In order to achieve this, all nets that utilize optical waveguides are segmented into three parts and rerouted: (i) the electrical part of the net topology containing the entry of the waveguide, (ii) the optical waveguide, and (iii) the electrical part of the net topology containing the exit point of the waveguide. Then our goal is to reroute (i) and (iii) while not using (ii). The nodes in the routing graph corresponding to the waveguides are deleted, ensuring that the net subtrees do not use them, thereby reducing overall electrical routing congestion. However, if the worst delay for the net deteriorates beyond the performance budget after optical rerouting, the net is converted back to electrical.

Rectilinear Steiner arborescence creates a routing tree with minimum source to sink distances for all sinks, and the weight of the tree is also minimum. It was shown that constructing minimal rectilinear Steiner arborescence is NP-complete [76]. We modify the RSA/G heuristic [32] to consider congestion-driven weights during the arborescence construction for simultaneous performance and congestion consideration. Our algorithm starts by constructing the *weighted shortest path subgraph* for a given net n to be rerouted by finding out all shortest paths from the source to sink in n. The weight is based on the current usage of the routing resource graph. The nodes in the subgraph are ranked according to the weighted distance from the source node (higher distance translates to higher ranks). The heuristic constructs the topology by considering the nodes in descending order of their ranks and merging them iteratively to form Steiner nodes. During the tree construction we do not allow merging at the routing nodes that are occupied by the existing waveguides.

Let s_n be the source of net n, T_n be the set of sink nodes of n, and i_n and o_n be the entry and exit point of the waveguide that n contains. Then, I_n includes i_n as well as the

Table 9: Standard vs. customized waveguides

	Standard Waveguides				Customized Waveguides			
ckt	nets	WG	WL	Util	nets	WG	WL	Util
n30	55	60	212	8.83	224	41	740	54.63
n50	87	60	348	14.5	277	35	906	79.14
n100	123	60	492	20.5	259	27	889	95.92
n200	235	60	940	39.16	190	19	640	100.00
n300	218	60	872	36.33	358	36	1242	99.44

Table 10: Comparison between Electrical, Congestion-driven, Standard Waveguides and Timing-driven Optical Routing with no violations. We report the total wirelength, routing layer (XY) usage, and maximum net delay (ns).

	Electrical			Congestion Optical			Standard Optical			Timing Optical		
ckt	WL	lyrs	D_{max}	WL	lyrs	D_{max}	WL	lyrs	D_{max}	WL	lyrs	D_{max}
n30	0.619	3.5	2.609	0.561	3.5	2.609	0.601	3.5	2.609	0.592	3.5	2.083
n50	0.899	5	2.722	0.799	4.5	2.722	0.843	5	2.721	0.857	4.5	2.079
n100	1.458	7.5	2.666	1.349	7.5	2.666	1.439	7.5	2.503	1.399	7.5	2.036
n200	2.710	16	2.553	2.524	13	2.553	2.582	16	2.553	2.650	16	2.143
n300	4.098	13	3.237	3.781	12.5	3.059	3.956	13	3.037	4.041	12.5	2.837
TIME	20			108			33			52		

set of all sink nodes of n located close to i_n. O_n is the set of all sink nodes of n located close to o_n. Thus, $I_n \subset T_n$ and $O_n \subset T_n$. Let X be the set of all nodes included in the waveguide of n except i_n and o_n. Finally, we construct the Steiner arborescence from s to I_n and from o_n to O_n while excluding X. This formulation can be easily extended to consider the nets with multiple waveguides.

4.4 Experimental Results

We implemented our algorithms in C++/STL and ran our experiments on Linux Beowulf clusters. For our experiments we used the standard GSRC floorplanning benchmarks. The blocks are placed into four-layer SOPs using our SOP floorplanner [81]. [3]

Table 9 shows the preliminary results obtained after the construction of waveguides using two techniques (congestion-driven customized and standard). Two sets of waveguides were passed through the same opto-net selection algorithm. Hence, these results are solely based

[3] Our attempt to compare to the routing results reported in [80] was not successful for several reasons. First, [80] reports only 2D SOP results. Second, [80] only reports delay improvement in percentage and does not provide details on delay models used.

Table 11: Timing-driven Optical Routing with different Criticality Levels. We report the total wirelength, routing layer (XY) usage, maximum net delay (ns), and maximum violations.

	cLevel=0.90				cLevel=0.85				cLevel=0.80			
ckt	WL	lyrs	D_{max}	viol	WL	lyrs	D_{max}	viol	WL	lyrs	D_{max}	viol
n30	0.606	3.5	2.261	0	0.604	3.5	2.134	0	0.592	3.5	2.083	0
n50	0.884	5	2.283	0	0.872	4.5	2.234	0	0.857	4.5	2.079	0
n100	1.442	7.5	2.356	0	1.426	7.5	2.212	0	1.399	7.5	2.036	0
n200	2.677	16	2.283	0	2.650	16	2.143	0	2.598	16	1.987	1
n300	4.041	12.5	2.837	0	3.979	12.5	2.638	4	3.934	12	2.453	6
TIME	27				48				57			

on the difference in waveguide construction. Wirelength saving is calculated as the amount of electrical wiring (in terms of grid edges) that has been converted into optical routing. Waveguide utilization is the ratio of number of nets actually using the waveguides to the maximum number of nets that could use these waveguides. It is evident that customized waveguides provide better waveguide utilization and wirelength savings. Also, they use less number of optical waveguides but convert more electrical nets into optically routed nets.

We used the technology parameters for 0.13 μ process [29] for the Elmore delay computation. Specifically, the driver resistance (R_d) of 29.4 $k\Omega$, input capacitance (C_L) of 0.050 fF, unit-length resistance (r) of 0.82 $\Omega/\mu m$ and unit-length capacitance (c) of 0.24 $fF/\mu m$ were used. For calculating the optical delay, v_w was chosen to be $1.936 \times 10^8 m/s$, and t_d was assigned 200 ps [12]. The values of R_w and C_w were chosen to be the same as R_d and C_L. The wirelengths reported have been scaled down by 10^5 μm. All routing results are evaluated against the baseline pure-electrical routing. The runtimes reported are in seconds.

Table 10 compares pure electrical routing with congestion-driven optical routing, optical routing using standard waveguides, and timing-driven optical routing. In all cases, only one optical layer is introduced per routing interval. An average wirelength saving of 8.5% was achieved using customized waveguides. The maximum wirelength improvement was 11% for $n50$. The amount of wirelength saved is less with the standard waveguides (3.7%). The number of layers reduce by one for most cases in congestion-driven optical routing

but mostly remain the same for standard waveguides. The performances of the designs are preserved for all benchmarks, with only nominal improvement in some instances. The wirelength saving in timing-driven optical routing is 3.6% and the performance enhancement is 19% on the average.

Table 11 compares timing-driven optical routing with different *criticality levels*. Reducing criticality levels increases the number of critical nets considered for waveguide constructions. While the chances that delay improves increase, the probability of optical resources capacity violation increases. The wirelength savings using criticality levels of 0.90, 0.85, and 0.80, are 1.5%, 2.6%, and 4.3% respectively. Performance improvements are 13%, 17.6%, and 22.8% respectively. However, as the criticality level reduces, the violations for larger circuits grow. The number of violations is an indicator of the difficulty in fixing the violations as a post process. A maximum of 23.7% delay improvement without resource violations is achieved with a criticality level of 0.80 for $n50$ and $n100$.

4.5 Conclusions

Optical waveguides are useful for extremely high propagation speeds, low crosstalk and transmission of multiple signals simultaneously. However, device costs and signal losses associated with optical devices call for careful consideration during optical routing within SOP.

In this chapter, we presented the first optical router for 3D System-On-Package. We developed algorithms to handle the construction of optical waveguides based on performance and congestion objectives, optimum net-to-waveguide mapping using maximum-flow minimum-cost network flow model, and ripping and rerouting of existing nets to relieve congestion and make maximum utilization of waveguides. Our experimental results suggest that smart placement of waveguides coupled with other routing techniques can reduce electrical wirelength by 11% and improve performance by 23%, when a single optical layer is introduced for every placement layer. Future work should tackle routing and layer assignment for multiple optical routing layers per each routing interval.

CHAPTER V

CONGESTION AND NOISE-DRIVEN FLOORPLANNING

The continuing trend of reducing power supply voltage has resulted in reduced noise margin, which affects reliability and may even cause functional failures as a result of spurious transitions. Active devices in 3D packaging draw a large volume of instantaneous current during switching, which causes simultaneous switching noise (SSN). Existing approaches consider the SSN issue as an afterthought, which may require an excessive amount of decoupling capacitance (decap) to suppress SSN. In addition, many iterations are required between full-length SSN simulation and manual layout repair until we converge to a satisfactory result. Moreover, ignoring the impact of placement on wire congestion may create congested spots and thus increase the manufacturing cost of the 3D SOP because of the additional routing layers and vias needed.

The organization of this chapter is as follows: Section 5.1 presents the problem formulation. Sections 5.2 and 5.4 explains the noise and the congestion models. The decoupling capacitor placement has been detailed in Section 5.3. The algorithms used for solving the problem has been outlined in Section 5.5. The experimental results are reported in discussed in 5.6. The conclusions are provided in Section 5.7.

5.1 Problem Formulation

5.1.1 SOP Placement Problem

The following are given as the input to our 3D SOP placement problem: (i) a set of blocks $B = \{B_1, B_2, \cdots, B_m\}$ that represents the various active and passive components in the given SOP design, (ii) width, height, and maximum switching currents for each block, (iii) a netlist $NL = \{n_1, n_2, \cdots, n_k\}$ that specifies how the blocks are connected via electrical wires, (iv) K, the number of placement layers ($= |L_p|$) in the 3D packaging structure, (v) the number of power/ground signal layers along with the location of the power/ground pins,

and (vi) tolerance on simultaneous switching noise.

For each net n from a given netlist NL, let wl_n denote the wirelength of n. The wirelength wl_n is the sum of Manhattan distance in the x, y, and z directions, where the z direction is the height of the associated vias. Let $w(i)$, $h(i)$, and $A(i)$ respectively denote the width, height, and area of the placement layer $L_p(i)$. Let $A^t = \max\{w(i)\} \cdot \max\{h(j)\}$, i.e., the product of the maximum width and the maximum height among all placement layers. Let $A^s = \sum_i A(i)$. Let w_{ave} and h_{ave} denote the average width and height among all placement layers. Let $A^d = \sum_i \{|w_{ave} - w(i)| + |h_{ave} - h(i)|\}$, i.e., the sum of the deviation of the dimension among all placement layers. Last, the area objective of 3D SOP placement, denoted A^{tot}, is the weighted sum of A^t, A^s, and A^d.

Let D^{tot} denote the total amount of decoupling capacitance required to suppress the simultaneous switching noise (SSN) under the given tolerance value. Let C^{tot} denote the amount of wire congestion in the given 3D SOP design. Sections 5.2 and 5.4 respectively discuss how D^{tot} and C^{tot} are computed. Lastly, the goal of the *SOP Placement Problem* is to find the location of each block in the placement layers such that the following cost function is minimized while SSN constraint is satisfied:

$$w_1 \cdot A^{tot} + w_2 \cdot \sum_{n \in NL} wl_n + w_3 \cdot D^{tot} + w_4 \cdot C^{tot}$$

In addition, decaps are required to be placed adjacent to the blocks that require them.

5.2 3D Power Supply Noise Modeling

We model the P/G network for 3D SOP as a 3D grid graph, as shown in Figure 28. Each P/G layer in the multi-layer structure is represented as a mesh. The mesh is connected by edges that represent the via in the P/G network. The edges in the mesh have inductive and resistive impedances. The mesh contains power-supply points and connection points. The connection points consume currents. The current is drawn from all the sources by the consumers, and the amount of current drawn along a path is inversely proportional to the impedance of the path in the power supply mesh. If for a particular block I_1, I_2, $\cdots$, I_N are the currents drawn from N power sources in the grid, then $I_1 + I_2 + \cdots + I_N = I$, where

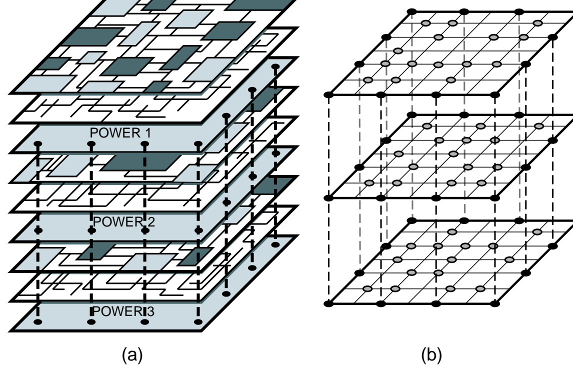

Figure 28: Illustration of 3D power supply modeling. (a) multi-layer power supply network, where multiple sets of device, routing, and power supply layers are stacked together, (b) 3D-grid modeling. where black and gray nodes respectively denote power supply and consumption nodes.

I is the switching current demand of the block. Then, $I_1 Z_1 = I_2 Z_2 = \cdots = I_N Z_N$, where Z_i is the impedances of path i. If $Y_j = 1/Z_j$, then

$$I_j = \frac{Y_j}{\sum_{i=1}^{N} Y_i} I, \ 1 \leq j \leq N$$

The current distribution for all blocks can be calculated using the above equations.

The *dominant current source* for a block is defined as the voltage source supplying significantly more power to the block than any other neighboring sources. The *dominant path* for a block is the path from the dominant supply to the block causing the most drop in voltage. It has been shown experimentally in [99] that the shortest path between the dominant current source (nearest Vdd pins) and the block offers highly accurate SSN estimation within reasonable runtime. In our 3D SSN analysis engine, we compute dominant paths (shortest paths to the nearest Vdd pins) for all blocks. This information is then dynamically updated whenever a new placement solution is evaluated in terms of SSN. Let P_k be a dominant current path for block k. Then, $T^k = \{P_j : P_j \cap P_k \neq \emptyset\}$ denotes the set of dominating paths overlapping with P_k (T^k includes P_k itself). Let P_{jk} be the overlapping segments between path P_j and P_k. Let $R_{P_{jk}}$ and $L_{P_{jk}}$ denote the resistance and inductance of P_{jk}. After the current paths and their values have been determined for all blocks, the SSN for B_k is given by

$$V_{noise}^k = \sum_{P_j \in T^k} (i_j \cdot R_{P_{jk}} + L_{P_{jk}} \frac{di_j}{dt})$$

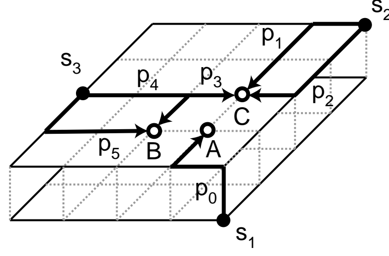

Figure 29: Illustration of SSN calculation. The dominant current source for block A is s_1, which is not located in the same layer. The dominant (= shortest) path p_0 carries $I_A/6$ amount of current, where I_A denotes the current demand of A. The block C draws current from s_2 and s_3 using p_1, p_2, and p_3 (each of these carries $I_C/3$ amount of current). The resistance of p_{34}, the overlap between p_3 and p_4, contributes to the SSN at B and C.

where i_j is the current in the path P_j, which is the sum of all currents through this path to various consumers. An illustration is shown in Figure 29. The weight of i_j and its rate of change are the resistive and inductive components of the path. Let Q^k denote the maximum charge drawn from the power supply by block B_k. If $\theta = \max(1, V^k_{noise}/V^{lim}_{noise})$, where V^{lim}_{noise} is the noise tolerance, the decap allocated to block B_k is given by

$$D^k = \frac{(1 - 1/\theta)Q^k}{V^{lim}_{noise}},\ 1 \leq k \leq M$$

where M denotes the total number of blocks to be placed. Finally, the decap cost is given by $D^{tot} = \sum_{k=1}^{M} D^k$.

5.3 3D Decoupling Capacitor Placement

In our LP-based 3D decap allocation shown in Figure 30, $x_k^{(j)}$ denotes the amount of decap allocated from white space k to block j. The objective is to maximize the utilization of white spaces for decap allocation. The constraint (2) limits the total allocation of a white space to its total area, where A_k denotes the area of white space k, and N_k denotes the neighboring blocks of k. The constraint (3) ensures that the total amount of decap allocated to each block does not exceed its demand, where $S^(j)$ denotes the demand.

Unlike the 2D decap assignment as done in [99], the neighboring blocks in the 3D case are the adjacent blocks either from the same placement layer or from neighboring layers. The assumption here is that the white space from different layers can be used to allocate decap. To facilitate this, we introduce predetermined parameters β_{kj} to control decap allocation

LP-based 3D Decap Allocation

$$\text{Maximize} \quad S = \sum_{k=1}^{H} \sum_{j \in N_k} \beta_{kj} x_k^{(j)} \quad (1)$$

Subject to

$$\sum_{j \in N_k} x_k^{(j)} \leq A_k, \; k = 1, 2, \cdots, H \quad (2)$$

$$\sum_{k=1}^{k=H} x_k^{(j)} \leq S^{(j)}, \; j = 1, 2, \cdots, M \quad (3)$$

$$x_k^{(j)} \geq 0, \; \forall k, \forall j \quad (4)$$

Figure 30: LP-based 3D decap allocation.

to block j from white space module k. For a given pair of white space k and block j, β_{kj} evaluates the usefulness of k to be used as a decap for j. Even if k is located in a different layer than j, it is possible that $\beta_{kj} > \beta_{ij}$ if another white space i in the same layer as j is too far away from k.

The second aspect of the formulation deals with the generation of A_k, the optimal area that will accomodate all decap without too much penalty on the final footprint area. The decap budget $S^{(j)}$ for each block is determined through *SSN* analysis of a particular 3D placement. The decap allocation involves several iterations between white space insertion and LP solving to reach a good solution both in terms of completeness of decap allocated and the additional increase in area. The white space is generated by expanding the floorplan in the X and Y direction, as illustrated in Figure 31. In a multi-layer placement, however, we have to take additional care to balance the expansion in each layer, so as to minimize the expansion of the total footprint area. The expansion is proportional to the decap demand of each module. In our Sequence Pair-based 3D placement, we modify the horizontal and vertical constraint graphs to expand the placement into X and Y directions, respectively.

Note that we may have to iterate between white space insertion and allocation if the current expansion does not satisfy all the decap needs. To prevent iterating between LP and white space insertion, we start by adding white spaces generously in the floorplan and

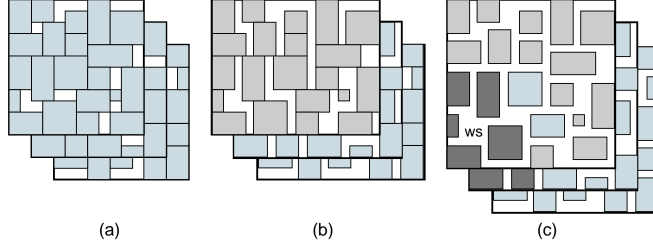

Figure 31: Illustration of 3D decap allocation. (a) 3D placement, (b) X-expansion, (c) XY-expansion, where the darker blocks denote the neighboring blocks of the decap (= white space) inserted. Note that blocks from other layers can utilize the white space for decap insertion.

then use LP to perform compaction. In our scheme, the XY-expansion is controlled by β parameters discusssed earlier, where the existing white spaces have higher ranks than the newly inserted ones. The actual amount of expansion is determined by a parameter $\alpha \geq 1$, where we increase the amount of expansion by a factor of α for each block that needs decap. This ensures that the initial expansion satisfies the LP constraint. Since LP determines the individual $x_k^{(j)}$, we use these assignment values to decide which white space insertion was not necessary and thus removed.

As exact decap allocation is a time-consuming process, it is impractical to solve an LP problem for every candidate solution. However, our white space insertion takes $O(n^2)$, which is computationally equivalent to computing block location using longest path computation. Therefore, we can afford white space insertion (and the calculation of area increases because of decap insertion) at every move. We perform the actual LP-based decap allocation at the end of the annealing process as a compaction stage.

5.4 3D Wire Congestion Computation

Estimating congestion at a reasonable level of accuracy during the placement process is at least as hard as global routing itself. The congestion profile itself is very sensitive to the placement result, so an inaccurate congestion measure can easily mislead placement. The process of global routing for 3D packaging is very different from that of conventional technologies (PCB, MCM, standard cells) because of the multiple placement layers existing in 3D packaging. Let $G_i = (V_i, E_i)$ be the grid graph representing the routing resource

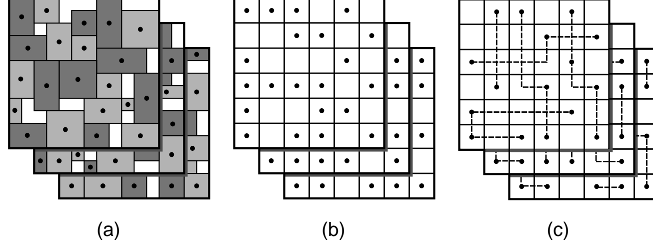

Figure 32: Illustration of 3D congestion estimation. (a) 3D placement, (b) 3D pin redistribution, (c) topology generation.

for routing interval i. The routing density of an edge e in G_i, denoted d_e^i, is the total number of nets/subnets that use e. Then, $d^i = \max_{e \in E_i}\{d_e^i\}$ is the local congestion in routing interval i. The congestion of the 3D placement with K placement layers is given by $C^{tot} = \sum_i d^i$. Finally, the lower bound in the routing layers required for all routing intervals in a 3D package is $\sum_{i \in RI} d^i/cap$, where RI and cap respectively denote the set of all routing intervals and capacity of edges in G_i. A more uniform usage of the grid graph results in low congestion. We perform the following steps presented in [65, 68] to accurately measure C^{tot}:

1. Net distribution: we determine which set of nets is assigned to each routing interval.
2. Pin redistribution: The pins in the placement layers are evenly distributed into the top and bottom pin distribution layers.
3. Topology generation: Steiner trees are generated for all nets in each routing interval.

An illustration is shown in Figure 32.

Our congestion estimation involves several time-consuming steps, unlike decap estimation, which makes it impractical to perform it for every new candidate 3D placement solution. In order to tackle this problem, we make use of the trajectory method proposed in [60]. We briefly describe the method here. Trajectory methods for floorplan optimization consist of *sampling* solutions at appropriate points in the trajectory traced by the optimization routine and *collecting* the metrics of the sampled solution. The original work used trajectory optimization in the context of minimizing clocks per instruction(CPI). We extend the ideas presented in that work to handle other cost functions as well, which are too time

consuming to be calculated at each step. The data points are collected apriori using the trajectory of area/wirelength optimized floorplanning. During congestion-driven floorplanning this values are used to estimate the congestion metric of all the candidate solutions. Let there be n *collected* points. The i_{th} point is A_i, WL_i, and CN_i respectively, for area, wirelength, and congestion. The congestion CN_c of the *current* solution is estimated using the following equations. A_c, WL_c are the area and wirelength of the current solution.

$$d_i = \|A_c - A_i\| + \|WL_c - WL_i\|$$

$$\bar{d} = min\ d_i,\ i = 1, \cdots, n.$$

$$\hat{w}_i = e^{-\beta d_i / \bar{d}}.$$

$$w_i = \hat{w}_i / \sum_{i=1}^{n} \hat{w}_i,\ i = 1, \cdots, n.$$

$$CN_c^i = CN_i + \left(\frac{\partial CN}{\partial A}\right)_i \cdot (A_c - A_i) + \left(\frac{\partial CN}{\partial WL}\right)_i \cdot (WL_c - WL_i).$$

$$CN_c = \sum_{i=1}^{n} w_i \cdot CN_c^i.$$

The partial derivatives are calculated by taking the *previous* two values in the trajectory and calculating the gradient. Since the estimation would only be accurate if the calculations were done in the vicinity of the point, we use clipping of estimated congestion values (CN_c^i) for pathological swings in the optimization. The value of estimated congestion varies only within a certain range ($CN_i - L < CN_c^i < CN_i + L$). L can be taken to be any appropriate value. In our experiments β was fixed to 25. In the *adaptive* method, the trajrctory is created during the optimization process. It has been shown that the adaptive method is more efficient than the non-adaptive one where the trajectory points are calculated beforehand.

5.5 Overview of the Algorithm

Simulated annealing [59] is a very popular approach for module placement because of its high-quality solutions and flexibility in handling various constraints. We extend the existing 2D sequence pair scheme [72] to represent our 3D module placement solutions. A pseudocode of our simulated annealing-based SOP placement algorithm is shown in Figure

33. The simulated annealing procedure starts with an initial multi-layer placement and its cost in terms of area, wirelength, decap, and congestion (line 3). We then make random perturbations (move) to the initial solution to generate a new 3D placement solution (line 6) and measure its cost (line 12). We perform 3D SSN analysis to compute the amoung of decap needed (line 7). As for the congestion measurement, we either perform a 3D global routing (line 8-9) or estimate via an interpolatation (line 10-11) discussed in Section 5.4. White space insertion (discussed in Section 5.3) is done and the increase in footprint area is calculated (line 12). If the new cost is lower than the old one, the solution is accepted (line 14-15); otherwise the new solution is accepted based on some probability that is dependent on the temperature of the annealing schedule (line 16-17). We examine a pre-determined number of candidate solutions at each temperature (line 5). The temperature is decreased exponentially (line 18), and the annealing process terminates when the freezing temperature is reached (line 4). The actual decap allocation is done after optimization using the LP solver (line 19) discussed in Section 5.3.

5.6 Experimental Results

We implemented the proposed algorithms and analysis tools using C++/STL. Our program was evaluated using the GSRC benchmarks. In our simulated annealing schedule, 20 temperature levels are used, and 100 moves are made in each temperature level. We chose area and wirelength-driven algorithms as our baseline. We use the following metrics for comparison: area, wirelength, additional area for decap, decap demand, and congestion. The number of layers is fixed to *four* for all benchmarks. We note that the final results obtained are highly sensitive to the weights of the metrics used in the cost function. In our experiment, we use the same parameters across all the benchmarks. However, we noticed that the solution quality can be improved by fine tuning the parameters per benchmark. The technology parameters used are 0.01 $\Omega/\mu m$ for wire resistance per unit length, $1pH/\mu m$ for wire inductance per unit length, and $10fF/\mu m$ for wire capacitance per unit length. We report the average runtime for each benchmark measured in seconds. The *area-driven* floorplanning was our baseline. We summarize our observations from Tables 12, 13, and 14

```
SOP Placement
1:  T = initial temperature;
2:  old_cost = INFINITY;
3:  init_3D_placement();
4:  while (T > final_temp)
5:      for (itr from 0 to tot_move)
6:          generate_new_placement();
7:          decap = analyze_SSN();
8:          if (allowed_schedule())
9:              cong = 3D_global_routing();
10:         else
11:             cong = estimate_congestion();
12:         ws = estimate_decap_alloc();
13:         new_cost = compute_cost();
14:         if (old_cost > new_cost)
15:             accept_move();
16:         else
17:             prob_accept_move();
18:     update T;
19: ws = actual_decap_alloc();
20: return best_placement;
```

Figure 33: Pseudocode for simulated annealing-based 3D SOP placement.

as follows:

- Our *decap-aware* placement achieves a consistent decap savings (average of 20%) for all circuits at the cost of a small decrease in area (7%) and wirelength (15%). The area reported is the final area reported after inserting the decap.

- We used adaptive trajectory optimization for the *congestion-driven* floorplanning. The trajectory was created during the floorplanning process. We achieved an overall 2% improvement with 57% increase in area compared to the baseline. We achieved a maximum improvement of 12%.

- Our multi-objective algorithm that considers all four objectives achieves a consistent improvement in both congestion and decap over the baseline, with only a slight increase in area (13%) and wirelength (10%). An average improvement of 10% on decap and a reduction of congestion by 3% , were acheived over the baseline. A look at the

Table 12: area/wire-driven vs decap-driven algorithms

ckts		area/wire-driven				decap-driven			
name	size	area	wire	decap	cong	area	wire	decap	cong
n50	50	22107	2.66E+04	18.08	21	24674	2.99E+04	9.15	22
n50b	50	25497	2.51E+04	13.18	26	26263	2.76E+04	4.1	22
n50c	50	21233	2.35E+04	15.97	21	22940	2.92E+04	2.23	23
n100	100	31551	6.66E+04	78.23	41	34307	7.31E+04	69.27	42
n100b	100	28014	4.83E+04	73.04	35	30301	5.61E+04	59.18	32
n100c	100	32886	6.21E+04	80.68	41	34217	6.76E+04	67.56	45
n200	200	56017	1.71E+05	226.33	86	69395	2.05E+05	223.22	74
n200b	200	55624	1.82E+05	233.20	83	65354	2.13E+05	224.34	79
n200c	200	54320	1.69E+05	237.45	61	61356	1.96E+05	228.6	76
n300	300	84620	2.86E+05	393.87	101	84467	2.86E+05	393.87	101
gt100	100	19156	1.32E+06	60.85	1275	20743	1.68E+06	42.55	1090
gt300	300	23898	1.96E+06	342.52	1357	24858	2.23E+06	334.99	1410
gt400	400	27017	2.81E+06	493.14	1989	26851	3.25E+06	482.05	1944
gt500	500	31665	3.03E+06	645.35	1910	32169	3.54E+06	632.41	2041
gt600	600	47541	6.08E+06	806.50	2957	50903	7.11E+06	794.8	2758
RATIO		1.00	1.00	1.00	1.00	1.07	1.15	0.80	0.99
TIME		360.00				392.54			

table confirms that the decap, congestion, or both are improved for all circuits.

We report our correlation study of the decap and congestion metric in Table 16. We notice that the correlation varies from small to medium. In fact, most of the circuits have moderate correlation. The trajectory optimization method works well with the circuits that have more correlation.

Table 15 shows power supply noise simulation results for three 3D placement schemes: no-decap-aware, decap-aware, and decap-aware+decap-placement. The P/G plane structure size is 246 $mm\times$ 254 mm, and the top P/G plane pair was modeled using a cavity resonator model [73] and simulated in HSPICE. The placement layer that used this P/G plan includes fouteen active devices. The DC $5V$ sources are located at four edges in the plane pair and fourteen current sources exist in the plane pair. As can be seen from Table 15, the SSN for the decap-aware algorithm is lower compared to the non-decap-aware algorithm. The SSN of the noisiest block blk4 is $1.58V$, which is reduced to $1.42V$ by our decap-aware scheme. With the insertion of decap, the noise is suppressed to $0.22V$. In addition, the total amount of decap required for the non-decap-aware algorithm is 26.7,

Table 13: area/wire-driven vs congestion-driven algorithms

ckts	area/wire-driven				congestion-driven			
name	area	wire	decap	cong	area	wire	decap	cong
n50	22107	2.66E+04	18.08	21	40930	3.24E+04	31.95	21
n50b	25497	2.51E+04	13.18	26	34200	2.92E+04	24.12	24
n50c	21233	2.35E+04	15.97	21	26798	2.79E+04	18.05	22
n100	31551	6.66E+04	78.23	41	41482	7.60E+04	85.49	39
n100b	28014	4.83E+04	73.04	35	59916	6.56E+04	108.23	35
n100c	32886	6.21E+04	80.68	41	47210	6.98E+04	105.59	40
n200	56017	1.71E+05	226.33	86	81494	1.90E+05	260.6	76
n200b	55624	1.82E+05	233.20	83	100009	2.09E+05	253.7	78
n200c	54320	1.69E+05	237.45	61	123868	2.11E+05	272.57	72
n300	84620	2.86E+05	393.87	101	154232	3.50E+05	418.04	93
gt100	19156	1.32E+06	60.85	1275	34690	1.72E+06	88.13	1205
gt300	23898	1.96E+06	342.52	1357	23630	1.96E+06	341.86	1496
gt400	27017	2.81E+06	493.14	1989	31858	2.85E+06	512.56	1878
gt500	31665	3.03E+06	645.35	1910	52813	3.98E+06	685.8	1887
gt600	47541	6.08E+06	806.50	2957	59078	6.26E+06	828.3	2697
RATIO	1.00	1.00	1.00	1.00	1.57	1.17	1.24	0.98
TIME	360.00				401.15			

which is reduced to 19.9 with our decap-aware scheme. The largest amount of decap is used for blk5 ($0.50nF$) because of an increase in its SSN after optimization. The numbers show that the SSN was efficiently suppressed and the amount of decap reduced by using our algorithms.

5.7 Conclusions

In this chapter, a 3D module and decap placement algorithm that simultaneously reduces the power supply noise and wire congestion was proposed. We provided efficient algorithms for 3D power-supply noise and congestion analysis to guide our 3D module placement process. In addition, we allocated white spaces around the modules that require decaps to suppress the power supply noise while minimizing the area overhead. We achieved improvements in both decap amount and congestion with only a small increase in area, wirelength, and runtime. Future work should consider thermal issues during 3D placement. In addition, 3D clock routing and interconnect optimization such as wire sizing and buffer insertion should also be addressed.

Table 14: area/wire-driven vs area/wire/decap/congestion-driven algorithms

ckts	area/wire				area/wire/decap/congestion			
name	area	wire	decap	cong	area	wire	decap	cong
n50	22107	2.66E+04	18.08	21	26217	3.02E+04	18.26	20
n50b	25497	2.51E+04	13.18	26	28383	2.67E+04	6.88	24
n50c	21233	2.35E+04	15.97	21	22169	2.84E+04	3.15	26
n100	31551	6.66E+04	78.23	41	37203	7.72E+04	73.94	37
n100b	28014	4.83E+04	73.04	35	33656	5.44E+04	71.01	34
n100c	32886	6.21E+04	80.68	41	35368	7.05E+04	72.08	42
n200	56017	1.71E+05	226.33	86	72780	1.99E+05	224.29	79
n200b	55624	1.82E+05	233.20	83	86011	2.28E+05	237.31	67
n200c	54320	1.69E+05	237.45	61	54315	1.69E+05	237.45	66
n300	84620	2.86E+05	393.87	101	84607	2.86E+05	393.88	97
gt100	19156	1.32E+06	60.85	1275	17509	1.32E+06	54.79	1211
gt300	23898	1.96E+06	342.52	1357	23642	1.98E+06	338.24	1367
gt400	27017	2.81E+06	493.14	1989	29873	2.94E+06	505.18	1852
gt500	31665	3.03E+06	645.35	1910	36653	3.22E+06	664.61	1871
gt600	47541	6.08E+06	806.50	2957	58405	6.73E+06	821.09	2842
RATIO	1.00	1.00	1.00	1.00	1.13	1.10	0.90	0.97
TIME	360.00				432.41			

Table 15: Power supply noise simulation results. We report SSN noise for each block placed in the top placement layer. (a) Non-optimized case without any decoupling capacitor. From Tables 12 we need 26.7 decap units to suppress SSN noise. (b) Optimized case without any decoupling capacitor. From Table 12 we need 19.9 decap units. (c) Optimized case with decoupling capacitors inserted.

blk	wire only	decap aware	decap used	blk	only	decap aware	decap used
blk1	1.21	1.01	0.16	blk8	1.33	1.21	0.36
blk2	1.31	1.15	0.17	blk9	1.44	1.16	0.32
blk3	1.33	1.18	0.20	blk10	1.34	1.38	0.32
blk4	**1.58**	1.42	0.22	blk11	1.43	**1.44**	0.20
blk5	1.26	1.41	**0.50**	blk12	1.41	1.35	0.33
blk6	1.54	1.27	0.31	blk13	1.37	1.27	0.27
blk7	1.47	1.33	0.43	blk14	1.05	1.15	0.22

Table 16: Decap *vs* Congestion correlation constants

ckt	correlation	ckt	correlation
n50	0.2188	n50b	0.0791
n50c	0.0013	n100	0.0662
n100b	0.0656	n100c	0.0048
n200	0.1433	n200b	0.1551
n200c	0.0909	n300	0.1786
gt100	0.2353	gt300	0.2422
gt400	0.1648	gt500	0.2028
gt600	0.1934	-	-

CHAPTER VI

3D CLOCK ROUTING

The 3D-ICs [78] have a tremendous potential for reducing the wirelength and the area of a design, thereby improving performance and cost. However, its widespread adoption has been hampered primarily by thermal issues. Heating of the substrate has proven to be a formidable challenge to the design community. The substrates experience higher thermal gradients due to multiple active layers plus the use of insulating dielectric layers of low thermal conductivity. Higher thermal gradients have adverse implications for performance and reliability, which undermines the advantages of 3D-ICs.

Physical design aspects of 3D-ICs have generated immense interest from the researchers in the recent past. Several studies on 3D floorplanning [36] and routing [37] have been proposed in the current literature. However, to the best of our knowledge, clock routing work for 3D-ICs has not been studied. Two major considerations for the clock routing in the 3D-ICs are the management of through vias and the consideration of the thermal issues. These challenges warrant new approaches for clock routing, which has a defining impact on the overall performance of the 3D-ICs.

The chapter is organized as follows. Section 6.1 describes preliminaries about the delay model and challenges of this work. The problem is formulated in Section 6.2. The 3-D clock routing algorithms including optimization considering buffers are explained in Section 6.3. Experimental results are presented in Section 6.4. We conclude in Section 6.5.

6.1 Preliminaries

6.1.1 Temperature Dependent Delay Model

It is widely known that interconnect resistance is linearly dependent on temperature. This research uses the recent temperature dependent interconnect delay model proposed in [3,

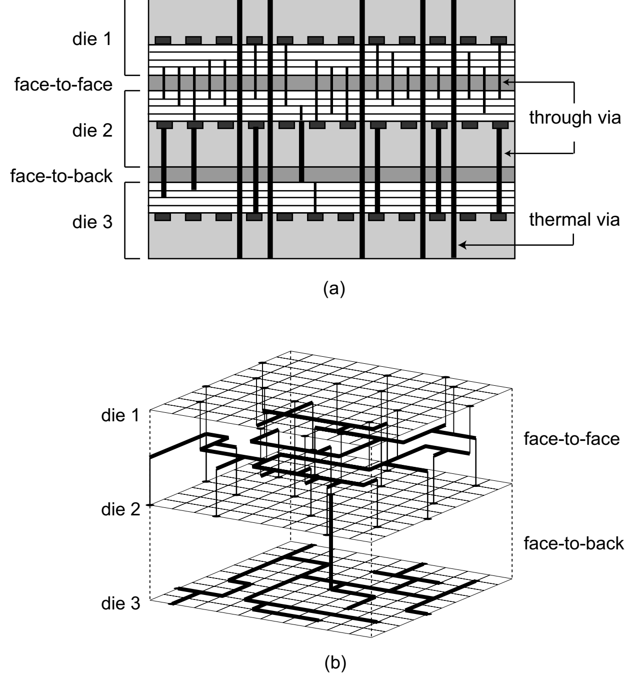

Figure 34: (a) 3-die IC with both face-to-face and face-to-back bonding, (b) bonding-aware 3D clock routing.

6, 7]. The line resistance per unit length can be calculated as:

$$r(x,y) = r_0(1 + \beta \cdot T(x,y)) \tag{5}$$

where r_0 is the resistance at 0^oC, β is the temperature coefficient of resistance and $T(x,y)$ denotes the temperature at location (x,y). The elmore delay [47] of a wire of length L driving a load C_L is given by:

$$D = \int_0^L r(x)(\int_x^L (c(y) + C_L)dy)dx \tag{6}$$

The temperature effect on the driving resistance of the buffer is captured by the following relation:

$$R_d(x,y) = R_{d0}(1 + \beta_c \cdot T(x,y)) \tag{7}$$

The buffer capacitance C_b and the unit length wire capacitance c have been assumed to be independent of temperature. It is worthwhile to note here that random or systematic variations in wire width affect both wire resistance and capacitance unlike temperature in this model.

6.1.2 Recent Work

A comprehensive survey of the clock routing problem has been presented in [30]. Recently, the focus of clock routing research has been on optimization for environmental and deterministic variations. Optimizations for a non-buffered clock tree in the presence of process variations has been tackled in [61, 49]. A pioneering work on optimization of the clock trees for thermal variations is named TACO [25]. Thermal variations induce substantial skews in the zero-skew design clock trees. TACO reduces thermally induced skews in the non-buffered clock trees by building *merging diamonds* that correspond to every internal clock tree node in a bottom-up fashion. Equal delay merging diamonds are constructed considering various paths (four L-shaped paths) centered on the merging nodes. Balanced delay merging diamonds are derived from equal delay merging diamonds by finding suitable points through heuristics. These points have nearly equal skews under a uniform and the worst case thermal profile. The work also considers a method for "shrinking" the diamonds for wirelength refinement. The optimization algorithm is run iteratively until thermal closure is achieved. The complexity of the algorithm is linear for each iteration. Our work differs from TACO in the following aspects:

- We have proposed a method for the exact calculation of balanced delay points, which is not harder than the computation of equal delay merging points under a non-uniform profile. We prove that the calculation of a balanced delay point for a given uniform and worst profile is equivalent to finding a zero-skew point for the average profile.

- This property can also be applied to buffered merging. We also derive expression for exact balanced skew computations of buffered interconnect.

- The skew properties lead to efficient calculation of candidate merging points with linear complexity. We use our algorithm to optimize 3D clock trees for thermal variations. However, these algorithms can be used for optimizing traditional 2D clock trees as well with minimal changes.

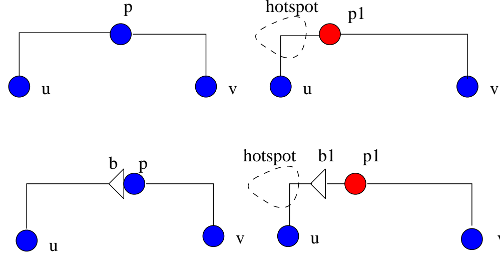

Figure 35: The effect of hotspots on the merging points and buffer locations

6.1.3 Challenges

The challenges for 3D clock routing in 3D-ICs include (i) skew minimization under thermal variations, (ii) even distribution of through-via connecting clock networks from various dies, and (iii) minimization of the total wirelength of the clock network. In addition, the number of through-vias used is also a design constraint because a significant number of through-vias is also used for signal routing. Hence, an effective through-via planning is essential for the quality of the 3D clock routing. Thermal issue in 3D-ICs adversely affects clock tree skew, so the clock tree design must consider the thermal profile.

Figure 35 illustrates the movement of the zero-skew merging points in the presence of hotspots. In the case of non-buffered clock trees, the merging point moves in the direction of the hotspot to balance out the greater delay on that side. The presence of buffers in the clock tree causes merging points to move towards hotspots as well. However, repositioning the buffers can reduce the displacement of the merging points. It should be noted that the zero-skew clock tree under uniform profiles at different temperatures ($T_1 \neq T_2$) remain unaffected. The situation is different in the presence of buffers, where the merging points may change across different uniform temperatures. Hence, tackling buffer clock trees with thermal considerations is a difficult task. As the temperature of the uniform profile increases, the merging points move linearly towards the buffer due to the increased buffer resistance. If the merging point is to be preserved, the buffer has to move towards the sink. These facts together with *path dependence* make the exact calculation of the merging segments intractable.

In this work, the merging segments are not computed explicitly. Instead, the starting

points for the merging segment construction are zero-skew merged nodes under uniform temperature. A small number of points from the merging segment is computed efficiently from the initial merged node. The advantage of this approach is that we achieve significant reduction in skew while the wirelength of the initial zero-skew tree increases minimally.

The objective of the thermal-aware clock tree synthesis or optimization is to ensure equal skews across all profiles over time. This is also true for other sources of variations such as voltage, where the power-supply profile is also time-varying and the fluctuations in the skew across profiles is inevitable. This is not true however for process variations across die, since the correlation profile is manufacturing dependent. The skew in this case can be optimized for one set of randomly varying parameters. The clock routing by classic algorithms will continue to work provided the induced skews due to variations are negligible. However, as demonstrated by numerous studies, the skews are getting large enough to cause a noticeable and adverse impact on system performance.

It was pointed out in a recent work that the delay at the internal node of the clock tree is *path-dependent* under non-uniform temperature profiles. It is worthwhile to note that under uniform temperature, zero-skew merging point between two nodes can be analytically calculated, and the locus of such points has been shown to be the intersection of manhattan arcs. However, under non-uniform temperature the determination of such locus is non-trivial and merging points cannot be computed by a closed form expression, unless the temperature profile itself was described by an analytical expression. In practice, this is seldom the case. The evaluation of the merging point is done on the two dimensional grid representing the thermal profile. Moreover, due to the discrete nature of the profile, the locus is non continuous. The locus in this case is a set of points which can be curve-fitted to approximate the actual locus. Each point in this locus characterizes a specific topology that connects to the merging point, which may have different lengths at different points.

6.2 Problem Formulation

Given a 3D-IC with two-die bonded face-to-face, the following are the inputs to the Thermal-aware 3D Buffered Clock Tree Construction problem: two sets of sinks $S_1 = \{s_{11}, s_{12} \cdots\}$

from die 1 and $S_2 = \{s_{21}, s_{22} \cdots\}$ from die 2, their corresponding locations and input capacitances, layer-specific line resistances (r_1, r_2) and capacitances (c_1, c_2), a through-via bound B, an ambient (= uniform) thermal profile P_u and a worst thermal profile P_w, and buffer load $C_{bufload}$. The goal is to construct a buffered interconnect tree $T(r)$ that connects the sinks in $S_1 \cup S_2$ so that the number of through-vias does not exceed B. In addition, the distribution of through-vias across the die is required to be even, and each buffer drives a load up to $C_{bufload}$. The objective is to minimize the skew between uniform and non-uniform thermal distribution and the total wirelength.

The problem of synthesizing a clock tree with a bounded skew under thermal consideration while considering the additional third dimension in 3D-ICs is non-trivial. The time varying nature of the thermal profiles makes the task of generating the clock tree extremely time-consuming, if not impossible. Instead, we adopt a formulation similar to [25], wherein the given zero-skew clock tree is *optimized* for a worst thermal profile P_w. This work attempts to construct a bonding-aware buffered clock tree for wafer bonded 3D-ICs with equal skews under uniform and worst thermal profiles.

The problem can be subdivided into three parts:

1. **Bonding-Aware Abstract Merging Tree Construction:** *Given a set of clock pin locations from each layer of the 3D-IC, construct a merging tree $AbsT(r)$, which specificies the order in which the pins are merged, while not exceeding the through-via budget B.*

2. **Zero-skew Simultaneous Buffering and Topological Embedding of the Given Abstract Merging Tree under Uniform Temperature:** *Given a 3-D abstract merging tree $AbsT(r)$, determine the exact geometric locations of all the nodes and the buffer positions, such that the wirelength of the embedded and buffered clock tree $T(r)$ is minimized, the capacitance seen by the buffer (buffer load) does not exceed $C_{bufload}$ and the total skew is zero.*

3. **Optimization of the Given Buffered Clock Tree for Balanced Skew under Thermal Variation:** *Given a 3-D zero skew buffered clock tree $T(r)$, an uniform*

ambient thermal profile P_u and a worst thermal profile P_w, optimize $T(r)$ such that the worst skews under profiles P_u and P_w are balanced and minimized with minimal changes in the total wirelength.

The motivation for the above formulation is to enable incremental clock routing for 3D-ICs which is essentially linear time.

6.3 3D Clock Routing Algorithm

6.3.1 Overview of the Algorithm

Traditionally, the clock routing problem has been solved in two ways [30]. The first approach separates the abstract tree generation and geometric embedding into different steps. The other approach does not use an abstract tree but constructs a merging tree and embedding solutions simultaneously [45, 46]. The simultaneous approach has been known to achieve better wirelength. However, due to the complexity of the problem we are solving, we adopt a three-step approach for our 3D clock routing problem: (1) 3D abstract tree generation, (2) topological embedding and buffering, and (3) thermal-aware optimization. The goal of the 3D abstract tree generation is to facilitate an even distribution of through-vias while minimizing the wirelength under through via bound. During the topological embedding and buffering step, the internal nodes of the abstract tree are placed and buffers are inserted under zero-skew constraint. Our thermal-aware optimization step refines the clock node locations for skew minimization under thermal variations and determines the actual routes among the nodes.

6.3.2 3D Abstract Tree Generation

The objective of this step is to generate a merging tree and additionally to get a smooth trade-off between the number of through-vias inserted and wirelength. The through-vias should also be evenly distributed. The insertion of through-vias can help reduce wirelength (see Figure 36). If only one through-via is used, the clock routing can be done independently for both dies and the roots can be connected by a through-via. However, if the through via

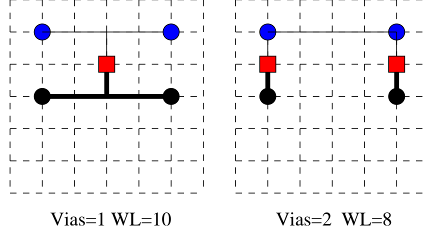

Figure 36: 3D clock routing with one and two through-vias.The squares represent the through-vias. The thin and bold lines represent routings in different layers.

bound is more than one, generating a clock-tree with minimum wirelength is not straightforward. The MMM algorithm [54] has been extended to generate abstract merging tree for the 3D clock sinks in a top-down manner. The algorithm has been outlined in Figure 37. The traditional MMM algorithms work on a set of sinks located on one layer. Essentially, it is recursive geometric partitioning. The objectives of 3D-MMM is to obtain a balance between geometric partitioning and layer-wise partitioning. The layer-wise partitioning of sinks is constrained by the number of through-vias available and hence must be done judiciously. The even distribution of the through-vias should be monitored throughout the process.

Figure 37 shows our 3D abstract three generation algorithm. The recursive algorithm takes as input a set of 3D clock sinks and a through-via bound. If the size of the sink ($|A|$) is one then abstract tree root is returned (lines 3-4). If the through-via bound is one, the sink set is partitioned into two subsets L and R such that the sinks of each layer goes to a different set (lines 7-8). In case the above conditions are not satisfied, the set is flattened to 2-D (z-dimension is ignored) and the set is partitioned geometrically by a horizontal or vertical line, depending on the direction of the cut (line 10). The line is drawn at the median of the x or y coordinates of the sinks. The through-via bound is divided between the two sets (line 11). The bound for each subset is calculated by (i) estimating the number of through-vias required by each set and (ii) dividing the given bound B according to the ratio of estimated vias. We assumed the minimum of the sink size in each layer in each set as an estimation of vias. The routine is called recursively for each of the subsets L and R with different cut direction (lines 12-13). The roots of the subtrees are connected by the

```
3D Abstract Tree Generation
Input: clock sinks from 2 dies and through via bound
Output: Root of 3D merging tree with bounded vias

1:  AbsTreeGen3D (SinkSet A, cutDir D, bound B)
2:      L and R = sink sets;
3:      if |A| = 1
4:          return root(A);
5:      else
6:      if B = 1 then
7:          Partition A into L, R layer-wise;
8:          B1 = B2 = 0;
9:      else
10:         Geometrically divide A into L, R using D;
11:         Find B1, B2 such that B1 + B2 = B;
12:     root(L) = AbsTreeGen3D (L, !D, B1);
13:     root(R) = AbsTreeGen3D (R, !D, B2);
14:     leftChild(root(A)) = root(L);
15:     rightChild(root(A)) = root(R);
16:     return root(A);
```

Figure 37: Pseudocode for 3D Abstract Generation.

root of the higher level tree (lines 14-16). The complexity of the algorithm is $O(nlogn)$, where n is the number of nodes.

6.3.3 Clock Tree Embedding and Buffering

The classic DME [9] algorithm is used to generate topology embedding for the given 3D abstract tree. This work inserts the buffer into the given tree by using a cost function to decide buffer insertion. Merging segments are obtained based on the merging distance computed as a result of buffer insertion as in [91]. The main difference between our approach and the approach described in [91] is that our approach works on a merging tree while the latter directly generates merging segments by considering buffers in the merging cost, thereby bypassing abstract tree generation. While the latter approach guarantees better quality solutions, it suffers from prohibitive complexity ($= O(n^3)$). The complexity of our approach is $0(n)$, which makes it feasible for incremental clock routing or inclusion in a solution search framework. The legal location of the buffers are near the nodes, i.e., above

or below the merging nodes.

6.3.4 Thermal-aware Clock Tree Optimization

The goal of our thermal-aware optimization step is to refine the clock node locations for skew minimization under thermal variations and determine the actual routes among the nodes. We introduce the term *options* to describe the set of points stored at each internal node, along with the topologies and its child options (see Figure 38). In a broader context, options could mean a set of local solutions at each node. The algorithm generates options at each internal node by a bottom-up traversal. The number of options is bounded at each node by solution pruning, where the weighted sum of skews and wirelength is used. The options with the best weights are retained while the others are discarded since the other options are less likely to improve either skew or wirelength of the final tree. At the root, the option with the best skew value is selected and the solution is propagated down to obtain the optimized clock tree. Figure 39 presents this optimization algorithm.

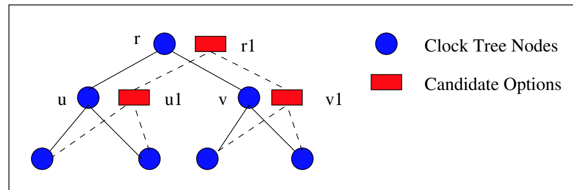

Figure 38: Bottom-up generation of options.

The algorithm works in a bottom-up manner on the clock tree. Lines 2-5 handle the case when the left and right children of the node are leaves. If the subtree at this level is buffered, the buffer is repositioned to a thermally optimal location. The new buffer location becomes a dummy node. A set of options is then created for the node by considering different paths to the parent with the same length. The option is actually a balanced skew point that balances out the skew between uniform and worst thermal profiles. The options are then stored for the node. Lines 6-9 tackle a node when at least one of its children nodes is not a leaf. The options from each child node is used to create a new option by considering all combinations of children nodes and topology. To restrict the number of options to be stored at each node, the inferior solutions are pruned (line 10). When options have been created

```
3D Clock Tree Optimization
Input: clock tree, uniform and worst thermal profiles
Output: optimized clock tree with balanced skew

1:  for (each node u in the bottom-up traversal)
2:      if (left_child(u) and right_child(u) are leaves)
3:          if (u is buffered)
4:              do buffer repositioning for u;
5:          create a set of options for u;
6:      else
7:          if (u is buffered)
8:              do buffer repositioning for u;
9:          create options for u from children options;
10:         prune out inferior options;
11: if (u is the root)
12:     choose option with best solution;
13: propagate solution top-down;
```

Figure 39: Pseudocode for 3D Clock Tree Optimization.

at the node, the node is checked to see if it is the root of the clock tree. If so, the solution is chosen to be the option with the best balanced skew and wirelength (lines 11-12). After the options have been created for all nodes, the best solution at the root is propagated top-down to determine the new optimized tree. The complexity of the algorithm is $O(n)$.

6.3.5 Balanced Skew Calculations

In this section, we discuss various properties of the clock tree skew under various thermal profiles, which enables us to compute balanced skews efficiently under any given uniform and worst profiles.

Definition 1 *Skew Function $S_P(t_1, t_2, x)$: A skew function is a function which gives the value of skew with respect to a temperature profile P at a point x in a path between two nodes with delays t_1 and t_2, respectively.*

Definition 2 *Skew Gradient: The rate of change of the skew function along the path connecting two clock nodes, i.e., $\frac{dS_P}{dx}$.*

Definition 3 *Absolute Skew: The absolute value of the skew function, i.e., $skew(P, x) =$*

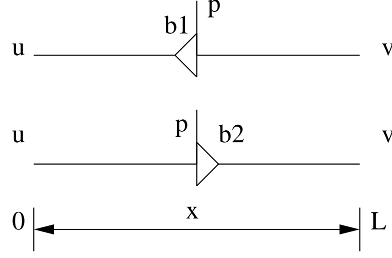

Figure 40: Illustration of interconnects where the buffer and the merging point are co-located.

$|S_P(t_1, t_2, x)|$. *We also use the notation* $skew_u(x)$ *to denote absolute skew under profile* P_u.

Definition 4 *Balanced Skew Point: A point* p *with* $skew(P_u, p) = skew(P_w, p)$*, where* P_u *and* P_w *are the uniform and worst thermal profiles.*

Lemma 1 *The skew gradient is a linear function of the temperature profile for unbuffered interconnects.*

Proof: We have

$$\begin{aligned} t_1(x) &= t_1 + \int_0^x r(z) \cdot \left(C_1 + \int_0^z c(y)dy \right) dz \\ t_2(x) &= t_2 + \int_x^L r(z) \cdot \left(C_2 + \int_z^L c(y)dy \right) dz \end{aligned} \tag{8}$$

$$\begin{aligned} s(x) &= t_1(x) - t_2(x) \\ &= t_1 - (t_2 + \int_0^L r(z) \cdot (C_2 + c(L-z))dz) + \\ &\quad \int_0^x r(z) \cdot (C_1 + C_2 + c \cdot L)dz \end{aligned} \tag{9}$$

$$\frac{ds}{dx} = (C_1 + C_2 + c \cdot L)r(x) \tag{10}$$

■

Lemma 2 *If the buffer and the merging point are co-located (see Figure 40), the skew*

gradients are given by:

$$\begin{aligned}\frac{ds_{b1}}{dx} &= (C_u + C_v + c \cdot L) \cdot r(x) + c \cdot r_d(x) + \\ &\quad (C_u + c \cdot x) \cdot \frac{dr_d}{dx} \\ \frac{ds_{b2}}{dx} &= (C_u + C_v + c \cdot L) \cdot r(x) + c \cdot r_d(x) - \\ &\quad (C_v + c \cdot (L - x)) \cdot \frac{dr_d}{dx}\end{aligned}$$

Proof: The proof of Lemma 2 is similar to the proof of Lemma 1. ■

Corollary 2.1 *The following expressions immediately follow from the preceding lemmas:*

$$S_P(t_1 + s_1, t_2, x) = S_P(t_1, t_2, x) + s_1$$

$$S_P(t_1, t_2 + s_2, x) = S_P(t_1, t_2, x) - s_2$$

$$S_P(t_1, t_2, x + \Delta x) \approx S_P(t_1, t_2, x) + \tfrac{dS_P}{dx} \cdot \Delta x$$

The properties mentioned above can be used for finding equal delay merging pointis in any given thermal profile. The balanced skews can be found recursively for each nodes as shown in the following theorem provided that children nodes have exactly balanced skews.

Theorem 3 *Let a and b be clock tree nodes such that $skew(P_u, a) = skew(P_w, a) = s_1$ and $skew(P_u, b) = skew(P_w, b) = s_2$. A balanced skew merging point p can be calculated by solving $S_u(t_{1u}, t_{2u}, x) = -S_w(t_{1w}, t_{2w}, x)$, where $t_{1u}, t_{2u}, t_{1w}, t_{2w}$ are the median delay value for uniform and worst profiles at a and b.*

Proof: Figure 41 provides an illustration of the proof.

$$\begin{aligned} S_u(t_{1u,min} + \frac{s_1}{2},\ t_{2u,min} + \frac{s_2}{2},\ x) = -S_w(t_{1w,min} + \frac{s_1}{2}, \\ t_{2w,min} + \frac{s_2}{2},\ x) \\ \Rightarrow S_u(t_{1u,min} + s_1,\ t_{2u,min},\ x) = -S_w(t_{1w,min}, \\ t_{2w,min} + s_2,\ x) \\ \Rightarrow S_u(t_{1u,max},\ t_{2u,min},\ x) = -S_w(t_{1w,min},\ t_{2w,max},\ x) \\ \Rightarrow |t_{1u,max}(x) - t_{2u,min}(x)| = |t_{1w,min}(x) - t_{2w,max}(x)| \end{aligned}$$

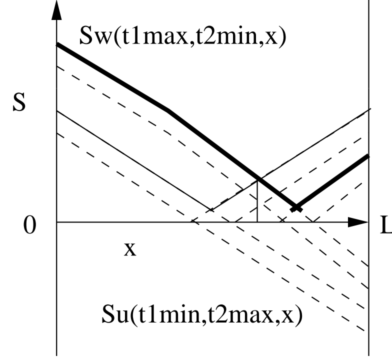

Figure 41: Calculation of balanced skew. The bold lines is the skew under worst profile, while the thin line is the skew under uniform profile.

Similarly,

$$|t_{1u,min}(x) - t_{2u,max}(x)| = |t_{1w,max}(x) - t_{2w,min}(x)|$$

$$\begin{aligned}
& S_u(t_{1u,min} + \frac{s_1}{2}, t_{2u,min} + \frac{s_2}{2}, x) = \\
& -S_w(t_{1w,min} + \frac{s_1}{2}, t_{2w,min} + \frac{s_2}{2}, x) \\
\Rightarrow \quad & S_u(t_{1u,min} + s_1, t_{2u,min}, x) = \\
& -S_w(t_{1w,min}, t_{2w,min} + s_2, x) \\
\Rightarrow \quad & S_u(t_{1u,max}, t_{2u,min}, x) = \\
& -S_w(t_{1w,min}, t_{2w,max}, x) \\
\Rightarrow \quad & |t_{1u,max}(x) - t_{2u,min}(x)| = \\
& |t_{1w,min}(x) - t_{2w,max}(x)|
\end{aligned}$$

Similarly,

$$\Rightarrow \quad S_u(t_{1u,min}, t_{2u,min} + s_2, x) =$$

$$-S_w(t_{1w,min} + s1, t_{2w,min}, x)$$

$$\Rightarrow \quad S_u(t_{1u,min}, t_{2u,max}, x) =$$

$$-S_w(t_{1w,max}, t_{2w,min}, x)$$

$$\Rightarrow \quad |t_{1u,min}(x) - t_{2u,max}(x)| =$$

$$|t_{1w,max}(x) - t_{2w,min}(x)|$$

■

Theorem 4 *The construction of a clock tree (buffered or non-buffered) with equal skews for given profiles P_1 and P_2 is equivalent to constructing a zero-skew tree for a profile $P_{avg} = (P_1 + P_2)/2$ (i.e $P_{avg}(x_1, y_1) = (P_1(x_1, y_1) + P_2(x_1, y_1))/2$).*

Proof: From Theorem 3, we have:

$$S_1(t_{11}, t_{21}, x) + S_2(t_{12}, t_{22}, x) = 0 = 2 \cdot S_{avg}(t_1, t_2, x)$$

From this, we can derive the following : $t_1 = (t_{11} + t_{12})/2$, $t_2 = (t_{21} + t_{22})/2$ and $P_{avg} = (P_1 + P_2)/2$. The balanced skew tree can be constructed in a bottom-up manner. The balanced skew can be computed using t_1 and t_2.

■

These theorems facilitate efficient calculation of balanced clock skews.

6.3.6 Bottom-up Options Generation

The clock tree can be recursively described by its children subtrees and the interconnection topology between the children nodes and the root. An optimal optimization is possible if the tree has the optimal substructure, i.e if given optimal solutions at the subtree, optimal solutions at the root can also be constructed. In this case however, the clock tree for thermal optimization does not posess the optimality substructure, as the number of paths to be examined would have to be exponential. However good enough solution can be

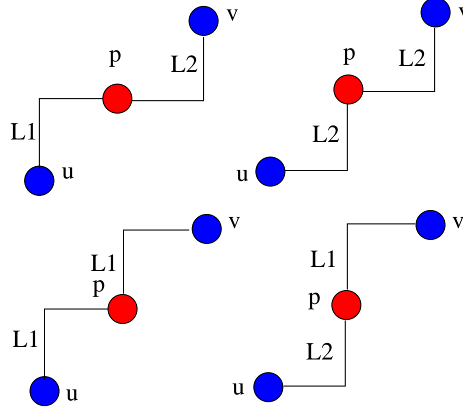

Figure 42: Paths connecting nodes u and v to parent p.

constructed given good enough solution for the subtrees. The options stores the attributes of the subtrees rooted at each node, i.e among others wirelength and skews.

Options are generated at each node by examining child options. Two important cases have to be handled during option generation: nodes connecting to parents without buffer and with buffer. We look at each of the cases in detail here.

- Case 1: When there are no buffers in the children-to-parent interconnection, the options are generated by identifying four paths as in [25]. The balanced skew points for each of the paths is determined by using the properties developed in Section 6.3.5.

- Case 2: When the children-to-parent interconnect is buffered, the options are generated by finding the balanced skew points for all the L-topologies. The movement of the buffers is computed by using Theorem 4. While generating the options, the relative positions of the buffer and the nodes are maintained.

6.3.7 Solution Pruning

The merging of the internal nodes for each combination of options may result in an explosion in the number of options that needs to be stored at each node. The number of options at each internal node is bounded by using solution pruning and removal of inferior solutions. In solution pruning, the weighted sum of skews and wirelength is considered. The options with the best weights are retained while the others are discarded, since the other options are less likely to improve either skew or wirelength of the final tree.

6.3.8 Top-down Selection

The options in the root of the clock tree represent the possible locations of the root and the associated location of its subtrees. Selecting the option with the least skew, determines the best solution. The optimized tree is obtained by recursively selecting the left and the right options in a top-down traversal. The selection determines the parent-to-child topological path as well as the new location of the internal nodes and buffers.

6.4 Experimental Results

The algorithms were implemented using C++/STL, and the experiments were run on linux xbeowulf cluster. The IBM benchmarks $r1$-$r5$ from [1] are used for the experiments. We generate 2-die clock sink placement by randomly partitioning the clock sinks into two layers.[1] The technology parameters used were $r_0 = 0.003\Omega/\mu m$, $c_0 = 2.0$ x $10^{-17}F/\mu m$, $\beta = 0.0068(1/^oC)$. The buffer parameters were $R_{d0} = 122\Omega$ and $C_b = 400fF$ and the intrinsic buffer delay was $t_b = 75ps$. The critical buffer load was chosen to be $4pF$. In our experiments, the RC parasitics were assumed to be the same in both dies, but our formulation can handle the heterogeneous case too. The through-via resistance is set to 50 $m\Omega$, which is small compared to the interconnect resistance.

6.4.1 Comparison to Existing Work

Table 17 compares the results of TACO [25] and our non-buffered 2D clock trees. The initial zero-skew clock trees for both cases were derived from [1]. The thermal profiles in TACO were randomly generated based on the min/max/average temperatures in a gaussian distribution. Our temperature numbers were generated in the same way. TACO reported 50-70% reduction of skews with 0.35% increase in wirelength. Although a direct comparison of all metrics between TACO and our approach may not be possible due to the random temperature values, we achieved a slightly better skew improvement percentages. Note

[1] Note that the footprint area will significantly reduce in the 2-die implementation, but we use the 2D footprint so that the cross-comparison is possible between our 2D ad 3D results. In this case, the wirelength of 3D clock tree approaches that of 2D clock tree (= 3D with the z-dimension ignored) if an unrestricted through-via bound is used.

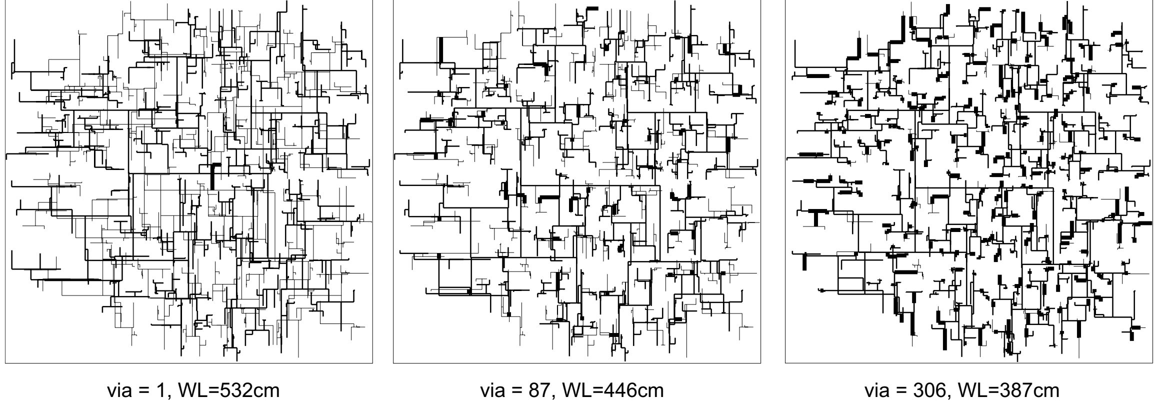

Figure 43: 3D clock routing for $r3$ with through-via bounds of (a) one, (b) 10% of sinks, (c) 100% of sinks. The thick bold line are the via location candidates. The thick and thin lines are routing in two layers.

that the skew values under the uniform and non-uniform temperatures have been perfectly balanced in our approach. TACO does not guarantee this.

6.4.2 2D Clock Routing Results

The results for 2D clock tree routing are presented in Table 18. By optimizing the unbuffered clock tree under thermal considerations, we reduced the skew by 65% on the average. The maximum reduction of skew for one benchmark ($r3$) was about 79%. The wirelength increase was about 0.21%. For the buffered clock tree, a skew reduction of 43% was achieved. A wirelength decrease was observed for some of the benchmarks. The overall increase in wirelength increase was 0.02%. The total capacitances of the clock trees have been reported as a measure of dynamic power consumption. Buffers improve the slew at the sinks, which is typically measured by the insertion delays, i.e., lesser insertion delays means better slew [91]. The wirelength of the unbuffered clock trees increased by 13% and the dynamic power increased by 8% when buffers were inserted. However, the insertion delay decreased by 34%.

6.4.3 3D Abstract Tree Results

Table 19 reports the results for the bonding-aware 3D abstract tree synthesis. A smooth trade-off between wirelength and the usage of through-vias was achieved by our bonding-aware 3D abstract tree generation. The baseline was the clock routing with only one

Table 17: Comparison with TACO [25] for 2D non-buffered clock tree optimization. Both algorithms use the initial clock trees generated by BST-DME [1]. The delay values are in ns, skews in ps, and wirelength in μm.

	TACO [25]						Ours (non-buffered)					
	T-uniform		T-worst			skew	T-uniform		T-worst			skew
ckt	dly	skew	dly	skew	wire	imprv %	dly	skew	dly	skew	wire	imprv %
r1	1.8	23.1	1.9	11.4	1323174	57.1	1.8	14.8	2.0	14.8	1322793	61.2
r2	4.5	86.2	4.9	75.7	2615431	62.3	4.4	99.4	5.2	99.4	2609009	49.7
r3	7.1	107.9	8.1	107.9	3399060	50.1	7.0	78.9	8.3	78.9	3395771	79.4
r4	14.7	220.7	17.6	220.7	6845621	64.9	14.7	367.5	17.7	367.5	6844030	68.2
r5	35.9	629.1	44.7	675.1	10302650	71.0	35.7	379.3	44.8	379.3	10265969	67.9
RATIO	1.00	1.00	1.00	1.00	1.00	1.00	0.99	0.96	1.03	1.11	0.998	1.09

through-via. The wirelength was calculated after running DME on the abstract tree. A wirelength reduction of 15% was achieved when the through-via bound was set to 10% of the clock sinks. When the through-via bound was set to the value of 100%, the wirelength reduction was about 27%. The number of vias increases 3.6 times, as the via bound is increased by 10X. A metric $Vdist$ is defined to be the deviation of the via distribution from the uniform via distribution. We found that the distribution of vias in all the examples were uniform. Figure 43 shows a snapshot of three 3D clock trees under various through-via bounds.

6.4.4 3D Clock Routing Results

The results of 3D clock tree optimization are presented in Table 20. The initial clock trees are generated by setting the via bound to 10% of the clock sinks. For the unbuffered 3-D clock tree, a skew reduction of about 67% on the average is achieved. The maximum skew reduction for one of the benchmarks ($r5$) is 85%. The increase in the wirelength was about 0.20%. The skew in buffered 3D clock tree was decreased by nearly 46%. The wirelength increase was about 0.03%. Wirelength decrease was also observed for one of the benchmarks ($r3$). After the buffer insertion and optimization, the wirelengths of the 3D unbuffered clock trees increased by 20% and the dynamic power increased by 13%. The insertion delay decreased by 42% with buffer insertion.

Table 18: 2D clock routing results. The delay values are in ns, skews in ps, and wirelength in μm.

			initial tree						optimized tree					
			T-uniform		T-worst				T-uniform		T-worst			
ckt	#sink	#buf	dly	skew	dly	skew	wire	cap	dly	skew	dly	skew	wire	cap
2D non-buffered														
r1	267	0	1.8	0	2.0	38.2	1320666	40.7	1.8	14.8	2.0	14.8	1322793	40.8
r2	598	0	4.4	0	5.3	198.0	2602908	84.6	4.4	99.4	5.2	99.4	2609009	84.8
r3	862	0	7.0	0	8.5	384.3	3388951	115.3	7.0	78.9	8.3	78.9	3395771	115.4
r4	1903	0	14.4	0	17.7	1159.2	6828510	241.5	14.7	367.5	17.7	367.5	6844030	241.8
r5	3101	0	35.3	0	44.7	1183.7	10242660	375.3	35.7	379.3	44.8	379.3	10265969	375.8
RATIO			1.00	-	1.00	1.00	1.00	1.00	1.01	-	0.99	0.35	1.002	1.002
2D buffered														
r1	267	13	2.32	0	2.69	153.54	1478434	44.00	2.28	88.92	2.72	88.92	1476646	43.91
r2	598	28	3.69	0	4.38	355.80	2910782	90.95	3.72	121.60	4.21	121.60	2905106	90.73
r3	862	34	4.23	0	5.14	250.99	3843117	124.56	4.34	106.60	5.11	106.60	3849891	124.56
r4	1903	83	4.80	0	5.95	305.37	7873062	262.74	4.96	246.69	6.02	246.69	7875873	262.46
r5	3101	127	6.16	0	7.95	435.35	11701835	405.03	6.48	303.40	7.92	303.40	11724289	404.97
RATIO			1.00	-	1.00	1.00	1.00	1.00	1.02	-	0.99	0.57	1.00	0.99
UNBUF:BUF			0.65	1.00	0.66	1.42	1.13	1.08	0.65	2.01	0.66	2.01	1.13	1.08

Table 19: Results for the 3D Abstract tree Generation

Input	ViaBound=one				ViaBound=10%				ViaBound=100%			
	wire	delay (ns)	vias	dV	wire	delay (ns)	vias	dV	wire	delay (ns)	vias	dV
r1 (267)	1934487	1.32	1	0	1681185	2.36	27	0	1496630	2.15	97	0
r2 (598)	4147404	3.77	1	0	3531971	6.19	60	0	2996259	5.54	217	0
r3 (862)	5319061	6.17	1	0	4462767	9.35	87	0	3871595	8.60	306	0
r4 (1903)	10788290	15.62	1	0	9139617	28.16	191	0	7788189	25.12	716	0
r5 (3101)	15848429	28.88	1	0	13232093	49.13	310	0	11380541	45.22	1135	0
RATIO	0.65	1.00	-	-	0.85	1.69	1.00	-	0.73	1.53	3.63	-

6.5 Conclusions

This chapter described clock routing for 2-layered wafer bonded 3-D ICs. The proposed clock tree algorithms consider specific 3-D structure features like bonding-awareness. The issue of thermal variations which poses a major problem for the 3D ICs is also considered during post clock tree synthesis. The optimization of the 3-D buffered clock tree for thermal variations can easily be specialized for a 2-D clock tree. Experimental results show that our algorithms reduce clock skews by 55-85% for non-buffered clock trees and 35-60% for buffered clock trees with minimal wirelength overhead. Future researchs should handle process, voltage and temperature (PVT) variations simultaneously for 3-D clock tree optimization.

Table 20: 3D clock routing results. The delay values are in ns, skews in ps, and wirelength in μm.

			initial tree						optimized tree					
			T-uniform		T-worst				T-uniform		T-worst			
ckt	#sink	#buf	dly	skew	dly	skew	wire	cap	dly	skew	dly	skew	wire	cap
3D non-buffered														
r1	267	0	2.36	0	2.67	40.40	1681185	48.00	2.38	17.85	2.68	17.85	1683603	48.05
r2	598	0	6.19	0	7.17	128.86	3531971	103.26	6.27	56.74	7.19	56.74	3538297	103.39
r3	862	0	9.35	0	11.22	300.75	4462767	136.82	9.45	91.00	11.24	91.00	4473010	137.02
r4	1903	0	28.16	0	34.06	555.27	9139617	287.73	28.41	186.08	34.24	186.08	9160805	288.16
r5	3101	0	49.13	0	63.13	2979.75	13232093	435.13	49.72	445.42	63.06	445.42	13258917	435.66
RATIO			1.00	-	1.00	1.00	1.00	1.00	1.01	-	1.002	0.33	1.002	1.001
3D buffered														
r1	267	15	2.76	0	3.21	145.32	1961838	53.67	2.79	61.17	3.19	61.17	1962953	53.64
r2	598	32	4.55	0	5.40	355.15	4138673	115.52	4.55	224.02	5.49	224.02	4141285	115.45
r3	862	44	5.18	0	6.23	381.66	5311238	153.96	5.38	232.38	6.18	232.38	5308883	153.74
r4	1903	101	6.41	0	8.04	468.22	11410433	333.56	6.71	253.96	8.00	253.96	11414633	333.24
r5	3101	127	8.61	0	11.09	853.55	16190834	494.81	8.74	430.73	11.48	430.73	16198152	494.45
RATIO			1.00	-	1.00	1.00	1.00	1.00	1.02	-	1.01	0.54	1.00	0.99
UNBUF:BUF			0.57	1.00	0.58	1.75	1.2	1.13	0.58	2.45	0.58	2.45	1.2	1.13

CHAPTER VII

CONCLUSIONS

The main objective of this thesis was to develop interconnect-centric tools for the physical design automation of high performance and reliable System-on-Packages (SOP) and 3D-ICs. In this research, global routing, floorplanning and clock routing phases of physical design were of prime interest. SOP is a multi-layer placement and routing technology which envisions system level integration of digital, analog, RF, opto and MEMS components. This heterogenous package level assembly is made possible by making use of the state-of-the-art knowledge in chip design, microelectronic packaging and fabrication. The wafer bonding technologies provide an alternative means to achieve system integration by bonding two or more dies together in a face-to-face or face-to-back configuration. The three dimensional structures offers an opportunity to stay in the interconnect curve as projected by Moore's Law. However the design complexities of SOPs and 3D-ICs cannot be addressed adequately by conventional layout tools due to the inherent three dimensional structure. The aim of this research was to develop a global routing tool for 3D technologies which considers issues specific to SOPs and 3D-ICs like thermal problems and crosstalk. As an futher improvement on performance, optical routing was proposed as a strong alternative. A cost-performance trade-off study was also done for the optical routing which substantiated the claims.

In order to acheive a reliable design of a SOP, this research proposed tight-coupling of the global routing with SOP floorplanning. A major consideration is the effective handling of power supply noise caused by simultaneous switching. This research proposed to address power supply noise by handling it simultaneously during floorplanning where decoupling capacitors (decaps) are also allocated to bring noise down to acceptable margins. In order to acheive maximum performance, the clock distribution has to be done efficiently.

This research maximized 3D design performance by effective clock routing strategies

considering substrate thermal profile. It was shown that a smooth trade-off between wirelength and the number of 3D vias can be achieved. The thermal variation was addressed by intelligently designing buffered clock trees which balanced the skews between uniform and the worst themal profile with only a nominal increase in wirelength.

Several interesting directions for future research were identified during the course of this work. Thermal issue continues to be the dominating factor preventing the 3D technologies from acheiving its desired potential. This fact only strengthens the need for the thermal awareness at all levels of physical design i.e floorplanning, placement, and routing, and even higher levels of the design cycle. Noise issues related to power-supply, crosstalk and substrate also need to be addressed with urgency as the technology scales down and the effects of these second-order factors become dominant. The ever decreasing feature sizes and the relative complex procedures for manufacturing 3D technologies also introduce the need for manufacturability-awareness. Hence, a future design methodology for 3D technology has to be necessarily yeild-conscious. Process variations is bound to permeate all levels of the design cycle in the near future. The 3D technology CAD must be variation-aware and multi-objective to tackle the challenges of the future. It should be kept in mind that 3D physical design would solve a more general problem than what the current technology offers. The 3D technology CAD may draw on the existing work for the the established technologies. Nevertheless, any novel approach proposed for a 3D technology may also be retroactively used to enhance the effectiveness of the conventional CAD tools. New researches in this direction would not only solve problems in a new domain but would also be proactive in contributing to the current state-of-the-art conventional CAD tools.

REFERENCES

[1] http://vlsicad.ucsd.edu/GSRC/bookshelf/Slots/BST/ (06/2005).

[2] Agarwal, D., Keeler, G. A., Nelson, B. E., and Miller, D. A. B., "Wavelength division multiplexed optical interconnects using femtosecond optical pulses," in *IEEE Lasers and Electro-Optics Society Twelfth Annual Meeting*, 1999.

[3] Ajami, A., Banerjee, K., and Pedram, M., "Effects of non-uniform substrate temperature on the clock signal integrity in high performance designs," in *Proc. of IEEE Custom Integrated Circuits Conference*, pp. pp. 233–236, May 2001.

[4] Alpert, C. J., "The ISPD98 circuit benchmark suite," in *Proc. Int. Symp. on Physical Design*, pp. 80–85, 1998.

[5] Alpert, C. J. and Kahng, A. B., "Recent directions in netlist partitioning: a survey," *Integration, the VLSI Journal*, pp. 1–81, 1995.

[6] Banerjee, K., Ajami, A. H., and Pedram, M., "Analysis and optimization of thermal issues in high-performance VLSI," in *Proc. Int. Symp. on Physical Design*, pp. 230–237, April 2001.

[7] Banerjee, K., Souri, S. J., Kapur, P., and Saraswat, K., "3D ICs: A novel chip design for improving deep submiscron interconnect performance and systems-on-chip integration," in *Proc. IEEE, Special Issue on Interconnects,*, pp. 602–633, May 2001.

[8] Bihari, B., Gan, J., Wu, L., Liu, Y., Tang, S., and Chen, R. T., "Optical clock distribution in super-computers using polyimide-based waveguides," in *SPIE*, 1999.

[9] Boese, K. D. and Kahng, A. B., "Zero-skew clock routing trees with minimum wirelength," in *Proc. IEEE Intl. Conf. on ASIC*, pp. pp. 1.1.1–1.1.5, 1992.

[10] Carothers, J. and Li, D., "A multilayer MCM auto-router based on the correct-by-design approach," in *Proc. 8th Annual IEEE Intl. ASIC Conf. and Exhibit*, 1995.

[11] Cha, Y., Rim, C., and Nakajima, K., "A simple and effective greedy multilayer router for MCMs," in *Proc. Int. Symp. on Physical Design*, 1997.

[12] Chang, G., Guidotti, D., Liu, F., Chang, Y., Huang, Z., Sundaram, V., Balaraman, D., Hegde, S., and Tummala, R., "Chip-to-chip optoelectronics sop on organic boards or packages," *IEEE Trans. on Advanced Packaging*, pp. 386–397, 2004.

[13] Chao, T.-H., Hsu, Y.-C., and Ho, J.-M., "Zero skew clock net routing," in *Proc.ACM/IEEE Design Automation Conf.*, pp. pp. 518–523, 1992.

[14] Chao, T.-H., Hsu, Y. C., Ho, J. M., Boese, K. D., and Kahng, A. B., "Zero skew clock routing with minimum wirelength," in *IEEE Trans. Circuits and Systems*, pp. 39(11):799–814, November 1992.

[15] Charikar, M., Kleinberg, J., Kumar, R., Rajagopalan, S., Sahai, A., and A.Tomkins, "Minimizing wirelength in zero and bounded skew clock trees," in *Proc.ACM/SIAM Symp. on Discrete Algorithms*, pp. pp. 177–184, 1999.

[16] Chattopadhyay, S., Bouldin, D. W., and Dehkordi, P. H., "An overview of placement and routing algorithms for multi-chip modules,"

[17] Chen, G. and Sapatnekar, S., "Partition-driven standard cell thermal placement," in *Proc. Int. Symp. on Physical Design*, 2003.

[18] Chen, H., Huang, L., Liu, I., Lai, M., and Wong, D., "Floorplanning with power supply noise avoidance," in *Proc. Asia and South Pacific Design Automation Conf.*, 2003.

[19] Chen, R. T. and et al, "Fully embedded board-level guided-wave optoelectronic interconnects," in *Proceedings of the IEEE*, 2000.

[20] Chen, Y., Chen, Z., and Fang, J., "Optimum placement of decoupling capacitors on packages and printed circuit boards under the guidance of electromagnetic field simulation," in *IEEE Electronic Components and Technology Conference*, 1996.

[21] Chen, Y., Kahng, A. B., and Zelikovsky, A., "On the associative-skew clock routing problem," in *Proceedings of ICCAD*, 1999.

[22] Chiang, T., Banerjee, K., and Saraswat, K., "Effect of Via Separation and Low-k Dielectric Materials on the Thermal Characteristics of Cu Interconnects," in *IEEE International Electron Devices Meeting*, 2000.

[23] Cho, J. D., Liao, K., Raje, S., and Sarrafzadeh, M., "M2R: Multilayer routing algorithm for high-performance MCM," *IEEE Trans. on Circuits and Systems Part I*, 1994.

[24] Cho, J. D., Raje, S., Sarrafzadeh, M., Sriram, M., and Kang, S. M., "Crosstalk-minimum layer assignment," in *IEEE Custom Integrated Circuits Conference*, 1993.

[25] Cho, M., Ahmed, S., and Pan, D. Z., "TACO: Temperature aware clock optimization," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, November 2005.

[26] Cho, S., S.Seo, Brooke, M., and Jokerst, N., "Integrated detectors for embedded optical interconnections on electrical boards, modules, and integrated circuits," in *IEEE J. Speical Topics in Quantum Electronics*, 2002.

[27] Choi, J., Chun, S., Na, N., Swaminathan, M., and Smith, L., "A methodology for the placement and optimization of decoupling capacitors for gigahertz systems," in *VLSI Design Symposium*, 2000.

[28] Cong, J., "Pin assignment with global routing for general cell design," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1401–1412, 1991.

[29] Cong, J., He, L., Khoo, K. Y., Koh, C. K., and Pan, Z., "Interconnect design for deep submicron ICs," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, pp. 478–485, 1997.

[30] Cong, J., He, L., Koh, C. K., and Madden, P. H., "Performance optimization of VLSI interconnect layout," *Integration, the VLSI Journal*, pp. 1–94, 1996.

[31] Cong, J., Kahng, A. B., Koh, C. K., and Tsao, C.-W. A., "Bounded-skew clock and steiner routing," in *ACM Trans. on Design Automation of Electronic Systems 3(3)*, pp. pp. 341–388, 1998.

[32] Cong, J., Kahng, A. B., and Leung, K.-S., "Efficient algorithms for the minimum shortest path steiner arborescence problem with applications to VLSI physical design," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 24–39, 1999.

[33] Cong, J. and Koh, C.-K., "Minimum-cost bounded-skew clock routing," in *Proc.IEEE Intl. Symp. Circuits and Systems*, pp. volume 1, pp. 215–218, April 1995.

[34] Cong, J. and Madden, P., "Performance driven multi-layer general area routing for PCB/MCM designs," in *Proc. ACM Design Automation Conf.*, 1998.

[35] Cong, J., Shinnerl, J. R., and Xie, M., "Large-scale circuit placement," *ACM Trans. on Design Automation of Electronics Systems*, pp. 389–430, 2005.

[36] Cong, J., Wei, J., and Zhang, Y., "A thermal-driven floorplanning algorithm for 3D ICs," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2004.

[37] Cong, J. and Zhang, Y., "Thermal-driven multilevel routing for 3-D ICs," in *Proc. Asia and South Pacific Design Automation Conf.*, 2005.

[38] Cong, J. and Lim, S. K., "Edge separability based circuit clustering with application to multi-level circuit partitioning," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 3, pp. 346–357, 2004.

[39] Cong, J. and Lim, S. K., "Retiming-based timing analysis with an application to mincut-based global placement," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 23, no. 12, pp. 1684–1692, 2004.

[40] Dai, W. W., "Topological routing in surf: Generating a rubberband sketch," in *Proc. ACM Design Automation Conf.*, pp. 39–48, 1991.

[41] Das, S., Chandrakasan, A., and Reif, R., "Timing, energy, and thermal performance of three-dimensional integrated circuits," in *Proc. Great Lakes Symposum on VLSI*, 2004.

[42] Das, S., Chandrakasan, A., and Reif, R., "Design tools for 3-D integrated circuits," in *Proc. Asia and South Pacific Design Automation Conf.*, 2003.

[43] Deng, Y. and Maly, W., "Interconnect characteristics of 2.5-D system integration scheme," in *Proc. Int. Symp. on Physical Design*, 2001.

[44] Deng, Y. and Maly, W., "Physical design of the 2.5D stacked system," in *Proc. IEEE Int. Conf. on Computer Design*, 2003.

[45] Edahiro, M., "A clustering-based optimization algorithm in zero-skew routings," in *Proc. ACM Design Automation Conf.*, pp. pp. 612–616, june 1993.

[46] Edahiro, M., "Delay minimization for zero-skew routing," in *Proc. IEEE Intl. Conf. Computer-Aided Design*, pp. pp, 563–566, November 1993.

[47] Elmore, W., "The transient response of damped linear networks with particular regard to wideband amplifiers," *Journal of Applied Physics*, pp. 55–63, 1948.

[48] Goplen, B. and Sapatnekar, S., "Efficient thermal placement of standard cells in 3D ICs using a force directed approach," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 2003.

[49] Guthaus, M. R., Sylvester, D., and Brown, R. B., "Process-induced skew reduction in nominal zero-skew clock trees," in *Proc. Asia and South Pacific Design Automation Conf.*, pp. 84–89, Jan 2006.

[50] Hanafusa, A., Yamashita, Y., and Yasuda, M., "Three-dimensional routing for multilayer ceramic printed circuit boards," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, 1990.

[51] Hattori, I., Kamo, A., Watanabe, T., and Asai, H., "Optimal placement of decoupling capacitors on PCB using poynting vectors obtained by FDTD method," in *Proc. IEEE Int. Symp. on Circuits and Systems*, 2002.

[52] Ho, M., Sarrafzadeh, M., Vijayan, G., and Wong, C., "Layer assignment for multichip modules," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1272–1277, 1990.

[53] Huang, J. H., Kahng, A. B., and Tsao, C.-W. A., "On the bounded-skew clock and steiner routing problems," in *Proc. ACM/IEEE Design Automation Conf.*, pp. pp. 508–513, 1995.

[54] Jackson, M., Srinivasan, A., and Kuh, E., "Clock routing. for high-performance ICs," in *Proc. ACM Design Automation Conf.*, 1990.

[55] Kahng, A. B. and Tsao, C.-W. A., "More practical bounded-skew clock routing," in *Proc. ACM/IEEE Design Automation Conference*, pp. pp.594–599, 1997.

[56] Kamo, A., Watanabe, T., and Asai, H., "An optimization method for placement of decoupling capacitors on printed circuit board," in *Proc. IEEE Electrical Performance of Electronic Packaging*, 2000.

[57] Khoo, K. and Cong, J., "An efficient multilayer MCM router based on four-via routing," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1277–1290, 1995.

[58] Khoo, K.-Y. and Cong, J., "A fast multilayer general area router for MCM designs," *IEEE Trans. Circuits and Syst. II*, pp. vol.39, pp. 841–51, Nov. 1992.

[59] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P., "Optimization by simulated annealing," *Science*, pp. 671–680, 1983.

[60] Long, C., Simonson, L., Liao, W., and He, L., "Floorplanning optimization with trajectory piecewise-linear model for pipelined interconnects," in *Proc. ACM Design Automation Conf.*, 2004.

[61] Lu, B., Hu, J., Ellis, G., and Su, H., "Process variation aware clock tree routing," in *Proc. ACM International Symposium on Physical Design*, pp. pp. 174–181, 2003.

[62] M. Wang, X. Yang, M. S., "Congestion minimization during placement," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 2000.

[63] McMurchie, L. and Ebeling, C., "Pathfinder: A negotiation based performance-driven router for FPGAs," in *ACM Intl Symposium on Field-Programmable Gate Arrays*, pp. 111–117, 1995.

[64] Mehrotra, S., Franzon, P., and Steer, M., "Performance driven global routing and wiring rule generation for high speed PCBs and MCMs," in *Proc. ACM Design Automation Conf.*, 1995.

[65] Minz, J. and Lim, S. K., "A global router for system-on-package targeting layer and crosstalk minimization," in *Proc. IEEE Electrical Performance of Electronic Packaging*, 2004.

[66] Minz, J. and Lim, S. K., "Layer assignment for system on packages," in *Proc. Asia and South Pacific Design Automation Conf.*, 2004.

[67] Minz, J., Lim, S. K., Choi, J., and Swaminathan, M., "Module placement for power supply noise and wire congestion avoidance in 3D packaging," in *Proc. IEEE Electrical Performance of Electronic Packaging*, 2004.

[68] Minz, J., Pathak, M., and Lim, S. K., "Net and pin distribution for 3D package global routing," in *Proc. Design, Automation and Test in Europe*, 2004.

[69] Minz, J., Wong, E., Pathak, M., and Lim, S. K., "Placement and Routing for 3D System-On-Package Designs," *IEEE Trans. on Components and Packaging Technologies*, 2005.

[70] Miyoshi, T., Wakabayashi, S., Koide, T., and Yoshida, N., "An MCM routing algorithm considering crosstalk," in *Proc. IEEE Int. Symp. on Circuits and Systems*, 1995.

[71] Murata, H., Fujiyoshi, K., and Kaneko, M., "VLSI/PCB placement with obstacles based on sequence pair," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 60–68, 1998.

[72] Murata, H., Fujiyoshi, K., Nakatake, S., and Kajitani, Y., "Rectangle packing based module placement," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, pp. 472–479, 1995.

[73] Na, N., Choi, J., Swaminathan, M., Libous, J. P., and O'Connor, D. P., "Modeling and simulation of core switching noise for asics," *IEEE Trans. Advanced Packaging*, pp. 4–11, 2002.

[74] Pant, M., Pant, P., and Wills, D., "On-chip decoupling capacitor optimization using architectural level prediction," *IEEE Trans. on VLSI Systems*, 2002.

[75] Ranjan, A., Bazargan, K., and Sarrafzadeh, M., "Fast hierarchical floorplanning with congestion and timing control," in *Proc. IEEE Int. Conf. on Computer Design*, 2000.

[76] Rao, S. K., Sadayappan, P., Hwang, F. K., and Shor, P. W., "The rectilinear steiner arboresence problem," *Algorithmica*, pp. 277–288, 1992.

[77] Ravichandran, R., Minz, J., Pathak, M., Easwar, S., and Lim, S. K., "Physical layout automation for system-on-packages," in *IEEE Electronic Components and Technology Conference*, 2004.

[78] Reif, R. and et al, "Fabrication technologies for three-dimensional integrated circuits," in *Proc. Int. Symp. on Quality Electronic Design*, 2002.

[79] Seo, C., Chatterjee, A., Cho, S., and Jokerst, N., "Design and optimization of board-level optical clock distribution network for high-performance optoelectronic system-on-a-packages," in *Proc. Great Lakes Symposum on VLSI*, 2004.

[80] Seo, C., Chatterjee, A., and Jokerst, N., "Physical design of optoelectronic system-on-a-package: a CAD tool and algorithms," in *Proc. Int. Symp. on Quality Electronic Design*, 2005.

[81] Shiu, P., Ravichandran, R., Easwar, S., and Lim, S. K., "Multi-layer floorplanning for reliable system-on-package," in *Proc. IEEE Int. Symp. on Circuits and Systems*, 2004.

[82] Sriram, M. and Kang, S., *Physical design of Multichip Modules.* Kluwer Academic Publishers, 1993.

[83] Stockmeyer, L., "Optimal orientation of cells in slicing floorplan designs," *Information and Control*, pp. 91–101, 1983.

[84] Su, H., Sapatnekar, S., and Nassif, S. R., "An algorithm for optimal decoupling capacitor sizing and placement for standard cell layouts," in *Proc. Int. Symp. on Physical Design*, pp. 68–73, 2002.

[85] Sutanthavibul, S., Shragowitz, E., and Rosen, J., "An analytical approach to floorplan design and optimization," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 761–769, 1991.

[86] Tanprasert, T., "An analytical 3-D placement that reserves routing space," in *Proc. IEEE Int. Symp. on Circuits and Systems*, 2000.

[87] Tsai, J.-L., Chen, T.-H., and Chen, C. C.-P., "Epsilon-optimal zero-skew clock tree wire-sizing in pseudo-polynomial time," in *ACM International Symposium on Physical Design*, 2003.

[88] Tsay, R. S., "An exact zero-skew clock routing algorithm," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 1993.

[89] Tummala, R., "SOP: what is it and why? a new microsystem-integration technology paradigm-moore's law for system integration of miniaturized convergent systems of the next decade," *IEEE Trans. on Advanced Packaging*, pp. 241–249, 2004.

[90] Tummala, R. and Madisetti, V., "System on chip or system on package?," *IEEE Design & Test of Computers*, pp. 48–56, 1999.

[91] Vittal, A. and Marek-Sadowska, M., "Power optimal buffered clock tree design," in *Proc. ACM Design Automation Conf.*, pp. 497–502, 1995.

[92] Wang, D. and Kuh, E., "A new timing-driven multilayer MCM/IC routing algorithm," in *Proc. IEEE Multi-Chip Module Conf.*, 1997.

[93] Wong, D. and Liu, C., "A new algorithm for floorplan design," in *Proc. ACM Design Automation Conf.*, pp. 101–107, 1986.

[94] Xi, J. G. and Dai, W. W.-M., "Useful-skew clock routing with gate sizing for low power design," in *Proc. ACM/IEEE Design Automation Conf.*, pp. pp. 383–388, 1996.

[95] Yang, X., Kastner, R., and Sarrafzadeh, M., "Congestion estimation during top-down placement," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 2002.

[96] Young, F. and Wong, D., "Slicing floorplans with pre-placed modules," in *Proc. IEEE Int. Conf. on Computer-Aided Design*, pp. 252–258, 1998.

[97] Yu, Q., Badida, S., and Sherwani, N., "Algorithmic aspects of three dimensional MCM routing," in *Proc. ACM Design Automation Conf.*, 1994.

[98] Zhang, R., Roy, K., Koh, C.-K., and Janes, D. B., "Exploring SOI device structures and interconnect architectures for 3-dimensional integration," in *Proc. ACM Design Automation Conf.*, 2001.

[99] Zhao, S., Koh, C.-K., and Roy, K., "Decoupling capacitance allocation and its application to power supply noise aware floorplanning," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, pp. 81–92, 2002.

PUBLICATIONS

[1] Jacob Minz and Sung Kyu Lim, "Layer Assignment for System on Packages", in *ACM/IEEE Asia and South Pacific Design Automation Conference*, p31-37, 2004.

[2] Jacob Minz, Mohit Pathak, and Sung Kyu Lim, "Net and Pin Distribution for 3D Package Global Routing", in *Design, Automation and Test in Europe*, p1410-1411, 2004.

[3] Ramprasad Ravichandran, Jacob Minz, Mohit Pathak, Siddharth Easwar, and Sung Kyu Lim, "Physical Layout Automation for System-On-Packages", *IEEE Electronic Components and Technology Conference*, p41-48, 2004.

[4] Mongkol Ekpanyapong, Jacob Minz, Thaisiri Watewai, Hsien-Hsin Lee, and Sung Kyu Lim, "Profile-Guided Microarchitectural Floorplanning for Deep Submicron Processor Design", *ACM Design Automation Conference*, p634-639, 2004.

[5] Jacob Minz, Sung Kyu Lim, Jin Woo Choi, and Madhavan Swaminathan, "Module Placement for Power Supply Noise and Wire Congestion Avoidance in 3D Packaging", 13th Topical Meeting on *Electrical Performance of Electronic Packaging*, p123-126, 2004.

[6] Jacob Minz and Sung Kyu Lim, "A Global Router for System-on-Package Targeting Layer and Crosstalk Minimization", 13th Topical Meeting on *Electrical Performance of Electronic Packaging*, p99-102, 2004.

[7] Jacob Minz, Sung Kyu Lim, and Cheng-Kok Koh, "3D Module Placement for Congestion and Power Noise Reduction," *ACM Great Lake Symposium on VLSI*, p458-461, 2005.

[8] Jacob Minz, Eric Wong, and Sung Kyu Lim, "Thermal and Crosstalk-Aware Physical Design For 3D System-On-Package," *IEEE Electronic Components and Technology Conference*, 2005.

[9] Jacob Minz, Eric Wong, and Sung Kyu Lim, "Thermal and Power Integrity-aware Floorplanning for 3D Circuits," to appear in *IEEE International SOC Conference*, 2005.

[10] Eric Wong, Jacob Minz, and Sung Kyu Lim, "Power Noise-aware 3D Floorplanning for System-On-Package," *IEEE Electrical Performance of Electronic Packaging*, pp. 259-262, 2005.

[11] Jacob Minz, Somaskanda Thyagaraja, and Sung Kyu Lim, "Optical Routing for 3D System-On-Package," to appear in *Design, Automation and Test in Europe*, 2006.

[12] Eric Wong, Jacob Minz, and Sung Kyu Lim, "Effective Thermal Via and Decoupling Capacitor Insertion Targeting 3D System-On-Package," to appear in *IEEE Electronic Components and Technology Conference*, 2006.

[13] Eric Wong, Jacob Minz, and Sung Kyu Lim, "Decoupling Capacitor Planning and Sizing for Noise and Leakage Reduction," to appear in *IEEE International Conference on Computer Aided Design*, 2006.

[14] Mongkol Ekpanyapong, Jacob Minz, Thaisiri Watewai, Hsien-Hsin Lee, and Sung Kyu Lim, "Profile-Guided Microarchitectural Floorplanning for Deep Submicron Processor Design," to appear in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2006.

[15] Jacob Minz, Eric Wong, Mohit Pathak, and Sung Kyu Lim, "Placement and Routing for 3D System-On-Package Designs," to appear in *IEEE Transactions on Components and Packaging Technologies*, 2006.

[16] Jacob Minz and Sung Kyu Lim, "Block-level 3D Global Routing With an Application to 3D Packaging," to appear in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*

[17] Eric Wong, Jacob Minz, and Sung Kyu Lim, "Multi-objective Module Placement For 3D System-On-Package," to appear in *IEEE Transactions on Very Large Scale Integration Systems.*

VITA

Jacob Rajkumar Minz received B. Tech (Hons.) degree in Computer Science and Engineering from Indian Institute of Technology, Kharagpur in 2001. He was employed in the Advanced VLSI Design Lab, IIT Kharagpur for a year where his main responsibility was building expertise for the design of digital chips. He joined the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta in August 2002. His research towards PhD was done in the GTCAD lab under the guidance of Dr. Sung Kyu Lim. His areas of interest are physical design automation and algorithms for VLSI CAD.