UNIVERSITY OF CALIFORNIA SAN DIEGO

Cross-Layer Pathfinding for Off-Chip Interconnects

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

in

Electrical Engineering (Computer Engineering)

by

Vaishnav Srinivas

Committee in charge:

Andrew B. Kahng, Chair
Jonathan Klamkin
Farinaz Koushanfar
Bill Lin
Naveen Muralimanohar
Dean Tullsen

2019

The dissertation of Vaishnav Srinivas is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

_____

Chair

University of California San Diego

2019

DEDICATION

To first principles.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

These acknowledgments may seem like the letter C. S. Lewis wrote to Lucy in his dedication of *The Lion, the Witch and the Wardrobe.* Having decided to pursue a Ph. D. while still working full-time, I have come to realize the amount of time it has taken for me to complete it, and this in turn has put into sharp relief how truly grateful I am to many people for their patience, guidance and collaboration. And I hope that this thesis is still relevant and useful for the approaches and methodologies used, and the cross-layer fundamentals of the design space explored, despite its somewhat lengthy journey to completion.

I'd first and foremost like to thank my advisor, Professor Andrew B. Kahng, whom I have been very fortunate to have guide me through my Ph. D. I have learnt many things from him about research and life in general, not the least of which has been the persistence to complete this thesis. I am grateful beyond these words for the opportunity he has provided along this road shared.

I'd like to thank my family and friends for their patience and encouragement - my wife Vanoli, my parents, Hema and Srinivas, my brother and sister-in-law, Vinay and Sunitha, and my many friends who have stood by me in various portions of this journey.

I would like to thank my present and former labmates in the UCSD VLSI CAD Laboratory, including Hyein Lee, Kwangsoo Han, Seokhyeong Kang, Kwangok Jeong, Wei-Ting Chan, Minsoo Kim, Mingyu Woo, Lutong Wang, Bangqi Xu, Siddhartha Nath, Jiajia Li and Tuck-Boon Chan. It has been amazing collaborating with them and I am grateful for their assistance, friendship and guidance.

I would like to thank my Thesis Committee (Professors Dean Tullsen, Bill Lin, Farinaz Koushanfar and Jonathan Klamkin, and Dr. Naveen Muralimanohar) for their time, support and insightful suggestions.

I would like to thank all my coauthors and collaborators, without whose help and support this thesis would not have been possible. All my coauthors have kindly approved inclusion of these publications in my thesis.

Chapter 2 contains a reprint of N. P. Jouppi, A. B. Kahng, N. Muralimanohar and V. Srinivas, "CACTI-IO: CACTI With Off-Chip Power-Area-Timing Models", *IEEE Transactions on Very Large Scale Integration Systems* 23(7) (2015). The dissertation author is a main contributor to, and a primary author of this paper. I would like to thank my coauthors Norm P. Jouppi, Andrew B. Kahng and Naveen

Muralimanohar.

Chapter 3 contains a reprint of A. B. Kahng and V. Srinivas, "Mobile System Considerations for SDRAM Interface Trends", *Proc. ACM International Workshop on System-Level Interconnect Prediction*, 2011. The dissertation author is a main contributor to, and a primary author of this paper. I would like to thank my coauthor Andrew B. Kahng.

Chapter 4 contains a reprint of R. Balasubramonian, A. B. Kahng, N. Muralimanohar, A. Shafiee and V. Srinivas, "CACTI 7: New Tools for Interconnect Exploration in Innovative Off-Chip Memories", *ACM Transactions on Architecture and Code Optimization* 14(2) (2017). The dissertation author is a main contributor to this paper. I would like to thank my coauthors Rajeev Balasubramonian, Andrew B. Kahng, Naveen Muralimanohar and Ali Shafiee.

Chapter 5 contains a reprint of A. Coskun, A. Gu, W. Jin, A. J. Joshi, A. B. Kahng, J. Klamkin, Y. Ma, J. Recchio, V. Srinivas and T. Zhang, "Cross-Layer Floorplan Optimization For Silicon Photonic NoCs In Many-Core Systems", *Proc. Design, Automation and Test in Europe*, 2016, pp. 1309-1314. The dissertation author is a main contributor to this paper. I would like to thank my coauthors Ayse Coskun, Anjun Gu, Warren Jin, Ajay J. Joshi, Andrew B. Kahng, Jonathan Klamkin, Yenai Ma, John Recchio and Tiansheng Zhang.

Chapter 6 contains a reprint of A. Coskun, F. Eris, A. Joshi, A. B. Kahng, Y. Ma, A. Narayan and V. Srinivas, "Cross-Layer Co-Optimization of Network Design and Chiplet Placement in 2.5D Systems", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, in revision. The dissertation author is a main contributor to this paper. I would like to thank my coauthors Ayse Coskun, Furkan Eris, Ajay Joshi, Andrew B. Kahng, Yenai Ma and Aditya Narayan.

Chapter 7 contains a reprint of S. Bang, K. Han, A. B. Kahng and V. Srinivas, "Clock Clustering and IO Optimization for 3D Integration", *Proc. ACM International Workshop on System-Level Interconnect Prediction*, 2015. The dissertation author is a main contributor to this paper. I would like to thank my coauthors Samyoung Bang, Kwangsoo Han and Andrew B. Kahng.

Chapter 8 contains a reprint of Y. Cao, J. Li, A. B. Kahng, A. Roy, V. Srinivas and B. Xu, "Learning-Based Prediction of Package Power Delivery Network Quality", *Proc. Asia and South Pacific Design Automation Conference*, 2019. The dissertation author is a main contributor to this paper. I would

like to thank my coauthors Yi Cao, Joseph Li, Andrew B. Kahng, Abinash Roy and Bangqi Xu.

Last but definitely not least, I would like to thank Qualcomm Inc., for its invaluable support through my Ph. D., including funding it, providing me with an environment of innovation, and collaboration and support from my colleagues and my managers.

VITA

2000	B. Tech., Electrical Engineering, Indian Institute of Technology, Madras
2002	M. S., Electrical Engineering, University of California, Los Angeles
2019	Ph. D., Electrical Engineering (Computer Engineering), University of California San Diego

PUBLICATIONS

A. Coskun, F. Eris, A. Joshi, A. B. Kahng, Y. Ma, A. Narayan and **V. Srinivas**, "Cross-Layer Co-Optimization of Network Design and Chiplet Placement in 2.5D Systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, in revision.

Y. Cao, J. Li, A. B. Kahng, A. Roy, **V. Srinivas** and B. Xu, "Learning-Based Prediction of Package Power Delivery Network Quality," *Proc. Asia and South Pacific Design Automation Conference*, 2019, pp. 160-166.

T. Ajayi, D. Blaauw, T.-B. Chan, C.-K. Cheng, V. A. Chhabria, D. K. Choo, M. Coltella, S. Dobre, R. Dreslinski, M. Fogaa, S. Hashemi, A. Hosny, A. B. Kahng, M. Kim, J. Li, Z. Liang, U. Mallappa, P. Penzes, G. Pradipta, S. Reda, A. Rovinski, K. Samadi, S. S. Sapatnekar, L. Saul, C. Sechen, **V. Srinivas**, W. Swartz, D. Sylvester, D. Urquhart, L. Wang, M. Woo and B. Xu, "OpenROAD: Toward a Self-Driving, Open-Source Digital Layout Implementation Tool Chain," *Proc. Government Microcircuit Applications and Critical Technology Conference*, 2019, pp. 1105-1110.

R. Bairamkulov, K. Xu, E. G. Friedman, M. Popovich, J. Ochoa and **V. Srinivas**, "Versatile Framework for Power Delivery Exploration," R. Bairamkulov, K. Xu, E. G. Friedman, M. Popovich, J. Ochoa and V. Srinivas, *Proc. ISCAS*, 2018, pp. 1-5.

A. Coskun, F. Eris, A. Joshi, A. B. Kahng, Y. Ma and **V. Srinivas**, "A Cross-Layer Methodology for Design and Optimization of Networks in 2.5D Systems," *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2018, pp. 101:1-101:8.

R. Balasubramonian, A. B. Kahng, N. Muralimanohar, A. Shafiee and **V. Srinivas**, "CACTI 7: New Tools for Interconnect Exploration in Innovative Off-Chip Memories," *ACM Transactions on Architecture and Code Optimization* 14(2) (2017), pp. 14:1-14:25.

J. L. Abellan, A. K. Coskun, A. Gu, W. Jin, A. Joshi, A. B. Kahng, J. Klamkin, C. Morales, J. Recchio, **V. Srinivas** and T. Zhang, "Adaptive Tuning of Photonic Devices in a Photonic NoC Through Dynamic Workload Allocation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36(5) (2017), pp. 801-814.

A. Coskun, A. Gu, W. Jin, A. J. Joshi, A. B. Kahng, J. Klamkin, Y. Ma, J. Recchio, **V. Srinivas** and T. Zhang, "Cross-Layer Floorplan Optimization For Silicon Photonic NoCs In Many-Core Systems," *Proc. Design, Automation and Test in Europe*, 2016, pp. 1309-1314.

N. P. Jouppi, A. B. Kahng, N. Muralimanohar and **V. Srinivas**, "CACTI-IO: CACTI With Off-Chip Power-Area-Timing Models," *IEEE Transactions on Very Large Scale Integration Systems* 23(7) (2015), pp. 1254-1267.

S. Bang, K. Han, A. B. Kahng and **V. Srinivas**, "Clock Clustering and IO Optimization for 3D Integration," *Proc. ACM International Workshop on System-Level Interconnect Prediction*, 2015, pp. 1-8.

N. Jouppi, A. B. Kahng, N. Muralimanohar and **V. Srinivas**, "CACTI-IO: CACTI with Off-chip Power-Area-Timing Models," *Proc. IEEE/ACM International Conference on Computer-Aided Design*, 2012, pp. 294-301.

A. B. Kahng and **V. Srinivas**, "Mobile System Considerations for SDRAM Interface Trends," *Proc. ACM International Workshop on System-Level Interconnect Prediction*, 2011, pp. 1-8.

V. J. Vijayakrishna, **V. Srinivas**, N. DasGupta and A. DasGupta, "Unified Analytical Model of HEMTs for Analogue and Digital Applications," *IEE Proceedings on Circuits, Devices and Systems* 152(5) (2005), pp. 425-432.

V. Srinivas, K. J. Bois, D. Knee, D. Quint, M. F. Chang and I. Verbauwhede, "Gigabit Simultaneous Bi-directional Signaling Using DS-CDMA," *Proc. EPEP*, 2002, pp. 15-18.

A. DasGupta, **V. Srinivas**, V. P. Reddy and N. DasGupta, "A New Unified Analytical Model for I-V characteristics of HEMTs," *10th International Conference on the Physics of Semiconductor Devices*, 1999, pp. 467-474.

ABSTRACT OF THE DISSERTATION

Cross-Layer Pathfinding for Off-Chip Interconnects

by

Vaishnav Srinivas

Doctor of Philosophy in Electrical Engineering (Computer Engineering)

University of California San Diego, 2019

Andrew B. Kahng, Chair

Off-chip interconnects for integrated circuits (ICs) today induce a diverse design space, spanning many different applications that require transmission of data at various bandwidths, latencies and link lengths. Off-chip interconnect design solutions are also variously sensitive to system performance, power and cost metrics, while also having a strong impact on these metrics. The costs associated with off-chip interconnects include die area, package (PKG) and printed circuit board (PCB) area, technology and bill of materials (BOM). Choices made regarding off-chip interconnects are fundamental to product definition, architecture, design implementation and technology enablement. Given their *cross-layer* impact, it is imperative that a cross-layer approach be employed to architect and analyze off-chip interconnects up front, so that a top-down design flow can comprehend the cross-layer impacts and correctly assess the system

performance, power and cost tradeoffs for off-chip interconnects. Chip architects are not exposed to all the tradeoffs at the physical and circuit implementation or technology layers, and often lack the tools to accurately assess off-chip interconnects. Furthermore, the collaterals needed for a detailed analysis are often lacking when the chip is architected; these include circuit design and layout, PKG and PCB layout, and physical floorplan and implementation.

To address the need for a framework that enables architects to assess the system-level impact of off-chip interconnects, this thesis presents power-area-timing (PAT) models for off-chip interconnects, optimization and planning tools with the appropriate abstraction using these PAT models, and die/PKG/PCB co-design methods that help expose the off-chip interconnect cross-layer metrics to the die/PKG/PCB design flows. Together, these models, tools and methods enable cross-layer optimization that allows for a top-down definition and exploration of the design space and helps converge on the correct off-chip interconnect implementation and technology choice. The tools presented cover off-chip memory interfaces for mobile and server products, silicon photonic interfaces, 2.5D silicon interposers and 3D through-silicon vias (TSVs). The goal of the cross-layer framework is to assess the key metrics of the interconnect (such as timing, latency, active/idle/sleep power, and area/cost) at an appropriate level of abstraction by being able to do this across layers of the design flow. In additional to signal interconnect, this thesis also explores the need for such cross-layer pathfinding for power distribution networks (PDN), where the system-on-chip (SoC) floorplan and pinmap must be optimized before the collateral layouts for PDN analysis are ready.

Altogether, the developed cross-layer pathfinding methodology for off-chip interconnects enables more rapid and thorough exploration of a vast design space of off-chip parallel and serial links, inter-die and inter-chiplet links and silicon photonics. Such exploration will pave the way for off-chip interconnect technology enablement that is optimized for system needs. The basis of the framework can be extended to cover other interconnect technology as well, since it fundamentally relates to system-level metrics that are common to all off-chip interconnects.

Chapter 1

Need for Cross-Layer Off-Chip Interconnect Optimization and Exploration

Ever-increasing compute requirements have led to a consequent increase in off-chip interconnect bandwidth. Power and cost considerations have led to various advancements in interconnect technology to address this need for higher bandwidths. These include advanced signaling schemes on traditional interconnect on a printed circuit board (PCB), 2.5D and 3D interconnect technology, and silicon photonic interconnects. The tradeoffs for performance, power and cost of off-chip interconnects are often complex and subtle, involving various layers of abstraction in the design flow to help assess them. At the same time, the tradeoffs are so fundamental for system metrics and architecture that they must be considered up front in the design cycle of a chip. This requires new enablement of *system-level pathfinding* for off-chip interconnects, to allow reasonably accurate assessment of system-level metrics early in the design cycle. System-level pathfinding requires a framework wherein the key cross-layer aspects of the off-chip interconnect are modeled and exposed at a suitable level of abstraction. Such a framework would allow architects to model and optimize the off-chip interconnect based on the system needs of performance, power and cost. The key metrics for the off-chip interconnect include (i) timing, latency and bandwidth to

evaluate system performance; (ii) active, idle and sleep currents and supply voltage to evaluate system power; and (iii) die area, die technology, bump and ball cost, package (PKG) and PCB area, stackup and bill of materials (BOM) cost to evaluate system cost. In addition, thermal and power distribution network (PDN) considerations are additional metrics that are important. Evaluating these metrics requires a cross-layer model and assessment, as they are dependent on the architecture, design, implementation and technology of the off-chip interconnects.

This thesis presents the models, tools and methods to address the need for cross-layer off-chip interconnect optimization and exploration. The models include Power-Area-Timing (PAT) and PDN impedance models. The tools and methods presented use such models for the key metrics for the off-chip interconnect enumerated above at a suitable level of abstraction such that the model captures the key tradeoffs within the physical design, circuit design and technology layers, while still being able to assess the metrics without the detailed collaterals of design and layout of the die, PKG and PCB. The modeling methods presented here employ various techniques to achieve the suitable level of abstraction, including directed graphs, analytical models, lookup tables, metamodeling and machine learning techniques. The tools include optimization tools and planning tools, including stand-alone excel frameworks, scripts to assist existing flows, including place-and-route (P&R) and floorplanning, global optimizers, and application program interfaces (APIs) that allow for easy integration into higher-level simulators. These tools integrate into die, PKG and PCB design flows for floorplanning, package bump and ball planning and assignment, interface and PDN definition, and interconnect technology exploration. As described below, the chapters of this thesis cover different types of interconnects, the cross-layer challenges that each present, as well as the tools and methods built to address these challenges and enable off-chip interconnect optimization and exploration.

Shown in Figure 1.1 is the organization and scope of the chapters of this thesis. The columns show the various interconnect types studied, including memory interconnect, 2.5D, 3D, photonic interconnect, as well as PDN which forms a key piece of the connections off-chip. The rows show the system-level metrics and considerations covered, including PPA+Cost (power, performance, area and cost), thermal, HW/SW (hardware/software) co-design for off-chip interconnects, and die/PKG/PCB co-design. These are the key system-level metrics and considerations impacted by off-chip interconnect, including the need

for co-design of the system-on-chip (SoC) with the PKG/PCB and runtime considerations for off-chip interconnects. Runtime considerations for off-chip interconnect include off-chip bandwidth and latency management at runtime, workload allocation, and thermal and PDN mitigation techniques. Die/PKG/PCB co-design is strongly connected with off-chip interconnect as the interactions between the three respective design flows largely involve off-chip interconnect definition and design.

Various aspects of the cross-layer pathfinding framework for off-chip interconnect tie in with die/PKG/PCB co-design. Floorplanning and pin planning, including bumps and balls, are key aspects that define the top-down configuration, including the bandwidth, capacity/loading, latency, bus width, signaling and clocking. The interconnect technology is also a key die/PKG/PCB co-design aspect, as the technology roadmap for the PKG and PCB drive the interface definition. In addition to the signal interconnect, PDN exploration is crucial, as a large part of the PKG and PCB are dedicated for PDN. Figure 1.2 shows the design flows for SoC, PKG and PCB. The portion highlighted in yellow is where the methods and tools presented in this thesis fit and enable die/PKG/PCB co-design. The SoC flow covers an RTL (register-transfer level) to GDS (graphic data system) type flow, with the architecture definition having been made before the RTL definition. The PKG and PCB flows start from the technology and size definition and industrial design. The tools and methods in this thesis aid in connecting the dots between the three flows, especially from architecture to pinmap, floorplanning and PKG/PCB definition. They consider the key system metrics that affect die/PKG/PCB co-design, including the thermal, signal and power integrity impact, in addition to performance, power, area and cost.

Chapter 2 describes a tool that enables memory interface exploration. Specifically, the chapter presents CACTI-IO, an extension to CACTI [171] that includes power, area, and timing models for the IO and PHY of the off-chip memory interface for various server and mobile configurations. CACTI-IO enables design space exploration of the off-chip IO along with the DRAM and cache parameters. We describe the models added and four case studies that use CACTI-IO to study the tradeoffs between memory capacity, bandwidth, and power. The case studies show that CACTI-IO helps (i) provide IO power numbers that can be fed into a system simulator for accurate power calculations; (ii) optimize off-chip configurations including the bus width, number of ranks, memory data width, and off-chip bus frequency, especially for novel buffer-based topologies; and (iii) enable architects to quickly explore new interconnect technologies,

Mapping of Thesis Chapters to System Metrics and Interconnect Types

System Metric	Interconnect Types				
	Memory	2.5D	3D	Photonic	PDN
Die/PKG/PCB Co-Design	4	6			8
PPA+Cost	2, 3, 4	6	7	5	8
Thermal		6		5	
HW/SW Co-Design	2, 4	6		5	

Figure 1.1: Organization of this thesis.

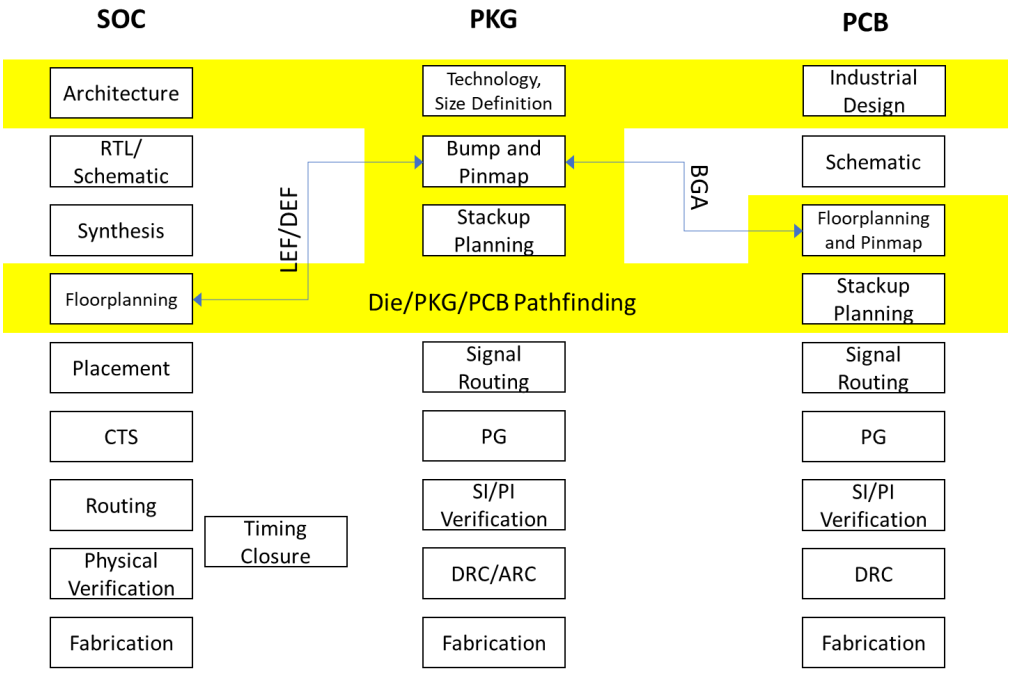


Figure 1.2: Die/PKG/PCB co-design and pathfinding.

including 3D interconnect. We find that buffers on board and 3D technologies offer an attractive design space involving power, bandwidth, and capacity when appropriate interconnect parameters are deployed.

Chapter 3 includes an early work focused on mobile memory interfaces that connect the SoC to the SDRAM (Synchronous Dynamic Random Access Memory). A variety of interconnect technologies and standards (dual inline memory modules (DIMMs), multi-chip package (MCP), package on package (POP), stacked-die and 3D-stack) enable a controller integrated circuit (IC) to communicate with an external SDRAM, or with multiple SDRAMs over a shared interconnect. Low-power requirements have driven mobile controllers to mobile-SDRAM low-power double data rate (LPDDR) memory solutions. However, LPDDR configurations do not scale to match the throughput and capacity requirements of mobile processors, or of emerging tablet products that bring new and divergent tradeoffs among memory subsystem metrics. As a result, identifying the memory configuration best suited to a given mobile application becomes quite challenging. This chapter considers some of the basic methods and approaches to enable a cross-layer framework for mobile memory interfaces. We highlight considerations that apply when choosing a particular memory configuration for a mobile processor based on capacity, throughput, latency, power, cost and thermal concerns. Further, we distinguish various choices according to interconnect implementation and performance, including power and timing in the IO and interconnect. To do this, we develop a three-part framework: (i) driving questions in the form of a *decision tree*; (ii) a *calculator* that projects power and timing for mobile IO implementations; and (iii) propagated top-down requirements and bottom-up capabilities that distinguish interconnect implementations.

Historically, server designers have opted for simple memory systems by picking one of a few commoditized DDR memory products. We are already witnessing a major upheaval in the off-chip memory hierarchy, with the introduction of many new memory products – buffer-on-board (BoB), latency-reduced DIMM (LRDIMM), hybrid memory cube (HMC), high bandwidth memory (HBM), and non-volatile memories (NVMs) to name a few. Chapter 4 extends CACTI-IO to custom memory solutions for server designs. We use the resulting capability to explore and define two new memory solutions: (i) we introduce a new relay-on-board chip that partitions a DDR channel into multiple cascaded channels; and (ii) we design a custom DIMM to reduce energy and improve performance, where the data channel is split into three narrow parallel channels and the on-DIMM interconnects are operated at a lower frequency. In

addition, this allows us to design a two-tier error protection strategy that reduces data transfers on the interconnect. The cascaded channel and narrow channel architectures serve as case studies for the new tool, and show the potential for benefit from re-organizing basic memory interconnects.

Chapter 5 considers cross-layer floorplan optimization for silicon photonic NoCs (PNoCs). Many-core chip architectures are now feasible, but the power consumption of electrical networks-on-chip does not scale well. PNoCs are more scalable and power efficient, but floorplan optimization is challenging. Prior work optimizes PNoC floorplans through simultaneous P&R, but does not address *cross-layer* effects that span optical and electrical boundaries, chip thermal profiles, or effects of job scheduling policies. This chapter proposes a more comprehensive, cross-layer optimization of the silicon PNoC and core cluster floorplan. Our simultaneous placement (locations of router groups and core clusters) and routing (waveguide layout) considers scheduling policy, thermal tuning, and heterogeneity in chip power profiles. The core of our optimizer is a mixed integer-linear programming (MILP) formulation that minimizes NoC power, including (i) laser source power due to propagation, bend and crossing losses; (ii) electrical and electrical-optical-electrical conversion power; and (iii) thermal tuning power. Our experiments vary numbers of cores, optical data rate per wavelength, number of waveguides and other parameters to investigate scalability and tradeoffs through a large design space. We demonstrate how the optimal floorplan changes with cross-layer awareness: metrics of interest such as optimal waveguide length or thermal tuning power change significantly (up to 4X) based on power and utilization levels of cores, chip and cluster aspect ratio, and laser source sharing mechanism. Exploration of a large solution space is achieved with reasonable runtimes, and is perfectly parallelizable. Our optimizer thus affords designers with more accurate, cross-layer chip planning decision support to accelerate adoption of PNoC-based solutions.

Chapter 6 describes a cross-layer co-optimization methodology for 2.5D interconnects. 2.5D integration technology is gaining attention and popularity in manycore computing system design. 2.5D systems integrate homogeneous or heterogeneous chiplets in a flexible and cost-effective way. The design choices of 2.5D systems impact overall system performance, manufacturing cost, and thermal feasibility. This chapter proposes a cross-layer co-optimization methodology for 2.5D systems. We jointly optimize the network topology and chiplet placement across logical, physical and circuit layers to improve system

performance, reduce manufacturing cost, and lower operating temperature, while ensuring thermal safety and routability. A key aspect of the co-optimization methodology is the inter-chiplet routing optimization based on an MILP that optimizes the link lengths and also accommodates a new type of link called a *gas-station*.

Chapter 7 considers 3D interconnect and clocking. 3D interconnect between two dies can span a wide range of bandwidths and region areas, depending on the application, partitioning of the dies, die size, and floorplan. We explore the concept of dividing such an interconnect into local *clusters*, each with a *cluster clock*. We combine such clustering with a choice of three clock synchronization schemes (synchronous, source-synchronous, asynchronous) and study impacts on power, area and timing of the clock tree, data path and 3DIO. We build a model for the power, area and timing as a function of key system requirements and constraints: total bandwidth, region area, number of clusters, clock synchronization scheme, and 3DIO frequency. Such a model enables architects to perform pathfinding exploration of clocking and IO power, area and bandwidth optimization for 3D integration.

Chapter 8 focuses on the PDN, which is a critical component in modern SoC designs. With the rapid development in applications, the quality of PDN, especially package (PKG) PDN, determines whether a sufficient amount of power can be delivered to critical computing blocks. In conventional PKG design, PDN design typically takes multiple weeks including many manual iterations for optimization. Also, there is a large discrepancy between (i) quick simulation tools used for quick PDN quality assessment during the design phase; and (ii) the golden extraction tool used for signoff. This discrepancy may introduce more iterations. In this work, we propose a learning-based methodology to perform PKG PDN quality assessment both *before layout* (when only bump/ball maps, but no package routing, are available) and *after layout* (when routing is completed but no signoff analysis has been launched). Our contributions include (i) identification of important parameters to estimate the *achievable* PKG PDN quality in terms of bump inductance; (ii) the avoidance of unnecessary manual trial and error overheads in PKG PDN design; and (iii) more accurate design-phase PKG PDN quality assessment. We validate accuracy of our predictive models on PKG designs from industry.

Chapter 9 summarizes the thesis and provides directions for future research enabled and motivated by this thesis.

Chapter 2

Tools that Enable Interconnect Search and Power-Area-Timing Optimization

Cross-layer pathfinding for off-chip interconnects requires interconnect exploration during the architecture definition of the design. The early exploration of off-chip interconnects is critical, as they are bottlenecks for key system metrics of performance, power and cost of the die, package and PCB. However, the performance, power and cost of off-chip interconnects are determined by circuit and physical layer analyses, which are often quite hard to comprehend by system architects. This makes it very important to build tools that enable architects to search over the off-chip interconnect solution space, and optimize interconnect power-area-timing tradeoffs. Such tools would need to expose the cross-layer analyses, while still maintaining the right level of abstraction suitable for pathfinding by architects.

In this chapter, we explore and present a tool that helps achieve this goal - namely, CACTI-IO, which has been integrated into the CACTI toolkit [171].

2.1 Introduction

The interface to the DRAM (Dynamic Random Access Memory), including the PHY, I/O circuit (IO), and interconnect, is becoming increasingly important for the performance and power of the memory subsystem [73, 112, 91, 108, 233, 196]. As capacities scale faster than memory densities [59], there

is an ever-increasing need to support a larger number of memory dies, especially for high-end server systems [195], often raising cooling costs. Mobile systems can afford to use multi-chip package (MCP) or stacked-die point-to-point memory configurations; by contrast, servers have traditionally relied on a dual-inline memory module (DIMM) to support larger capacities. With modern server memory sizes exceeding 1 TB, the contribution of memory power can reach 30-57% of total server power [196], with a sizable fraction (up to 50% in some systems) coming from the off-chip interconnect. The memory interface incurs performance bottlenecks due to challenges with interface bandwidth and latency. The bandwidth of the interface is limited by (i) the data rate, owing to the DRAM interface timing closure, signal integrity over the interconnect, and limitations of source-synchronous signaling [103, 50], and (ii) the width of the bus, which is often limited by size and the cost of package pins.

CACTI [171] is an analytical memory modeling tool which can calculate delay, power, area, and cycle time for various memory technologies. For a given set of input parameters, the tool performs a detailed design space exploration across different array organizations and on-chip interconnects, and outputs a design that meets the input constraints. CACTI-D [139] is an extension of CACTI with on-chip DRAM models.

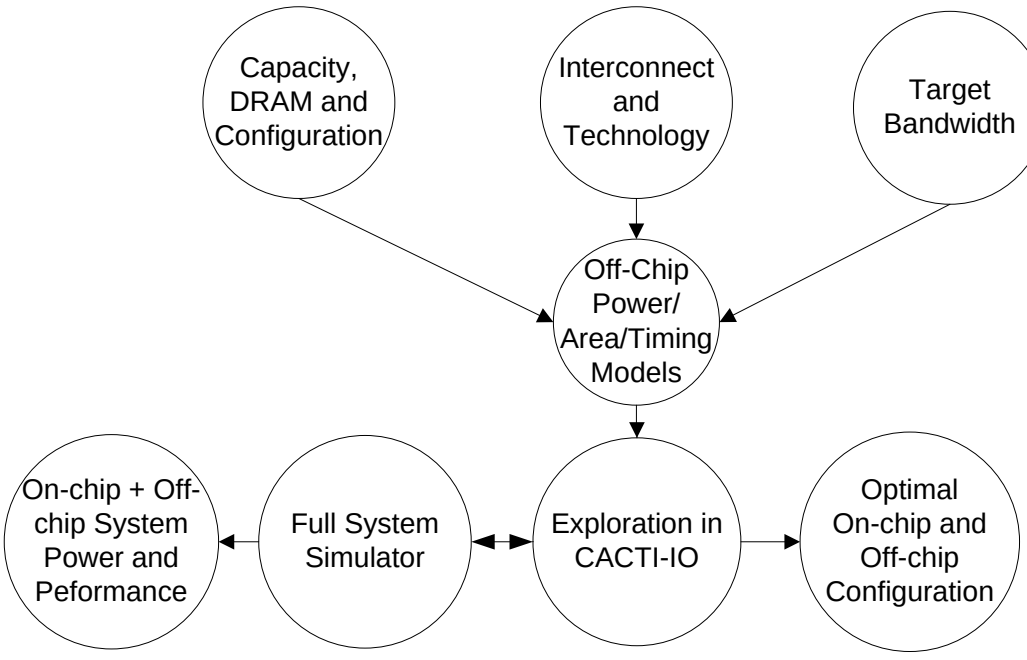


Figure 2.1: CACTI-IO: Off-chip modeling and exploration within CACTI.

In this chapter we describe CACTI-IO [58], an extension to CACTI, illustrated in Figure 2.1. CACTI-IO allows the user to describe the configuration(s) of interest, including the capacity and organization of the memory dies, target bandwidth, and interconnect parameters. CACTI-IO includes analytical models for the interface power, including suitable lookup tables for some of the analog components in the PHY. It also includes voltage and timing uncertainty models that help relate parameters that affect power and timing. Voltage and timing budgets are traditionally used by interface designers to begin building components of the interface [26, 103, 122, 101] and budget the eye diagram between the DRAM, interconnect, and the controller as shown in Figure 2.2. The *Eye Mask* represents the portion of the eye budgeted for the *Rx* (receiver). The setup/hold slacks and noise margins represent the budgets for the interconnect and the Tx (transmitter).

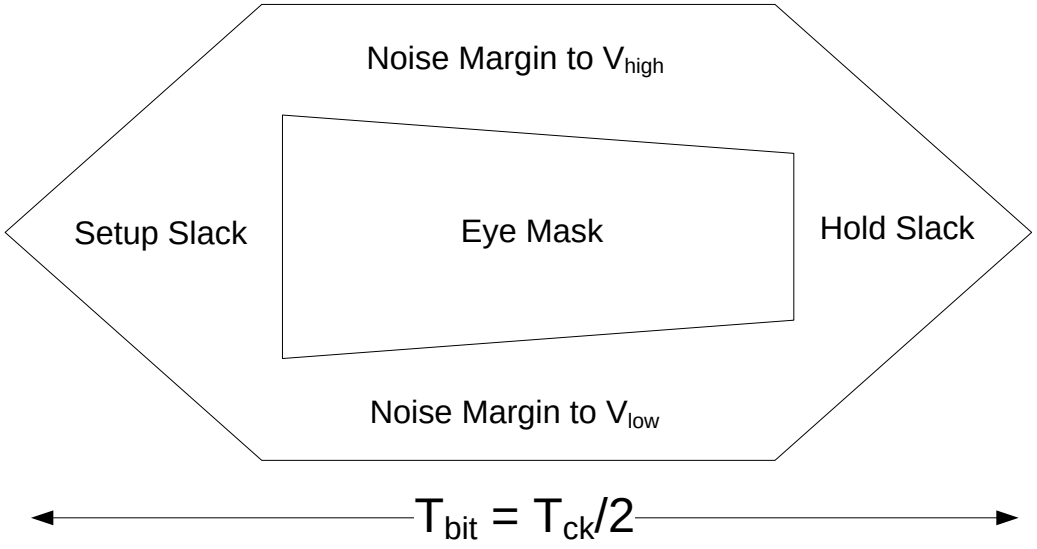


Figure 2.2: Memory interface eye diagram for voltage and noise budgets.

Final optimization of the IO circuit, off-chip configuration, and signaling parameters requires detailed design of circuits along with SPICE analysis, including detailed signal integrity and power integrity analyses; this can take months for a new design [103]. CACTI-IO is not a substitute for detailed analyses, but rather serves as a quick estimate for the system architect to enable the right tradeoffs between the large number of non-trivial IO and off-chip parameters. Up-front identification of the off-chip design space at an architectural level is crucial for driving next-generation memory interface design.

The main objectives for the CACTI-IO tool are as follows.

(1) Obtain IO power numbers for different topologies and modes of operation that can be fed into a full-system simulator. The tradeoffs between performance, power, and capacity in the memory subsystem are non-trivial [163, 139], but previous studies often do not explore alternatives to a standard DDR3 configuration for the memory interface. Furthermore, most modeling tools, including McPAT [78] and DRAMSIM [149], do not model the interface power and timing, and have no visibility into the details of the PHY and IO. CACTI-IO provides IO power numbers for Read, Write, Idle (only clock active), and Sleep modes that can easily be integrated into a system simulator. This enables architects to compare on-chip and off-chip sources of power across modes.

(2) Enable co-optimization of off-chip and on-chip power and performance, especially for new off-chip topologies. Historically, off-chip parameters (i.e., signaling properties and circuit parameters) have been limited to standardized configurations including DIMMs, with operating voltage, frequency, data rates, and IO parameters strictly governed by standards. A major drawback and design limiter – especially when operating at high frequencies – in this simplistic design context is the number of DIMMs that can be connected to a channel. This often limits memory capacity, creating a *memory wall*. Recent large enterprise servers and multicore processors instead use one or more intermediate buffers to expand capacity and alleviate signal integrity issues. Such a design still adheres to DRAM standards but has more flexibility with respect to the interconnect architecture that connects memory and compute modules, including serial interfaces between the buffer and the CPU. While current and future memory system capacity and performance greatly depend on various IO choices, to date there is no systematic way to identify the optimal off-chip topology that meets a specific design goal, including capacity and bandwidth. ***CACTI-IO provides a way for architects to systematically optimize IO choices in conjunction with the rest of the memory architecture.*** Below, we illustrate how CACTI-IO can help optimize a number of off-chip parameters – number of ranks (fanout on the data bus), memory data width, bus frequency, supply voltage, address bus fanout, and bus width – for given capacity and bandwidth requirements. CACTI-IO can also be used to evaluate the number of buffers needed in complex, high-end memory configurations, along with their associated overheads.

(3) Enable exploration of emerging memory technologies. With the advent of new interconnect

and memory technologies, including 3D TSS (through-silicon stacking) based interconnect being proposed for DRAM [66] as well as new memory technologies such as MRAM (magnetic RAM) and PCRAM (phase-change RAM) [113], architects are exploring novel memory architectures involving special off-chip caches and write buffers to filter writes or reduce write overhead. Most emerging alternatives to DRAM suffer from high write energy or low write endurance. The use of additional buffers plays a critical role in such off-chip caches, and there is a need to explore the changing on-chip and off-chip design space. When designing new off-chip configurations, many new tradeoffs arise based on the choice of off-chip interconnect, termination type, number of fanouts, operating frequency and interface type (serial vs. parallel). CACTI-IO provides flexible baseline IO models that can be easily tailored to new technologies and used to explore tradeoffs at a system-level.

In summary, the key contributions of this chapter are:

- Models for power, area, and timing of the IO, PHY, and interconnect for server and mobile configurations;
- CACTI-IO, an extension to CACTI that includes these models; and
- Four industry-driven case studies that use CACTI-IO to optimize parameters of the off-chip topology, including the number of ranks and memory data width.

In the remainder of this chapter, Section 2.2 describes the interface models, including those for power, voltage margins, timing margins, and area. Section 2.3 describes how the models can be ported for a different technology. Section 2.4 shows comparisons of the model against SPICE. Section 2.5 presents CACTI-IO using four case studies, showing a summary of the power and timing as well as optimal off-chip configurations. Section 2.6 summarizes our conclusions.

2.2 IO, PHY, and Interconnect Models

In this section, we give complete details of the IO, PHY, and interconnect models included in CACTI-IO. Power and timing models for interconnect and terminations have been well-documented and validated over the years [26, 4, 13]. Our goal here is to show the framework of the baseline models, which can then be adapted to any customized configuration needed, including new interconnect technologies.

2.2.1 Power Models

Power is calculated for four different modes: WRITE (peak activity during WRITE), READ (peak activity during READ), Idle (no data activity, but clock is enabled and terminations are on), and Sleep (clock and terminations are disabled, in addition to no data activity). The mode of the off-chip interconnect can be chosen by setting the *iostate* input parameter to W (WRITE), R (READ), I (IDLE) or S (SLEEP). CACTI-IO off-chip power models include the following.

(1) Dynamic IO Power. The switching power at the load capacitances is described in Equation (2.1), where N_{pins} is the number of signal pins; D_c is the duty cycle when the link is enabled; α is the activity factor for the signal switching (number of 0 to 1 transitions per clock period, i.e., $\alpha = 1$ for a clock signal); i denotes various nodes along the interconnect, with possibly different swings in a terminated or low-swing scheme; C_{Total_i} is the capacitance at node i; V_{sw_i} is the swing of the signal at node i; V_{dd} is the supply voltage; and f is the frequency of operation.

$$P_{dyn} = N_{pins} D_c \alpha (\sum_i C_{Total_i} V_{sw_i}) V_{dd} f \tag{2.1}$$

(2) Interconnect Power. The power dissipated on the interconnect ($P_{dyn\_interconnect}$) is given by Equation (2.2). Energy/bit ($E_{bit}^{interconnect}$) is given by Equation (2.3), where Z_0 is the characteristic impedance of the line, t_L is the flight time (time taken for the signal to traverse the line length), and t_b is the bit period. For high-end servers, generally $2t_L > t_b$ since the interconnect is long, while for mobile configurations, generally $2t_L < t_b$. For an FR-4 based interconnect used on printed circuit boards, t_L is approximately 180 ps/inch. The interconnect is generally modeled as a transmission line when $t_L > t_r/3$ (t_r is the rise-time of the signal), unlike an on-die RC network [4].

$$P_{dyn\_interconnect} = N_{pins} D_c \alpha E_{bit}^{interconnect} f \tag{2.2}$$

$$E_{bit}^{interconnect} = \begin{cases} \frac{t_L V_{sw} V_{dd}}{Z_0} & \text{if } 2t_L \leq t_b \\ \frac{t_b V_{sw} V_{dd}}{Z_0} & \text{if } 2t_L > t_b \end{cases} \tag{2.3}$$

(3) Termination Power. The IO termination power is provided for various termination options,

including unterminated (as used in LPDDR2 and Wide-IO), center-tap (as used in DDR3), VDDQ (as in DDR4), and differential terminations (as used in M-XDR). The voltage swing set by the terminations is fed into the dynamic power equation described in Equation (2.1).

The termination power is then calculated for source and far-end terminations. P_{term_ol} is the termination power when the line is driven to 0 (V_{ol}), and P_{term_oh} is the termination power when the line is driven to 1 (V_{oh}). The average power is reported assuming that 0 and 1 are equiprobable during peak activity. V_{dd} is the supply voltage, V_{TT} is the termination voltage and R_{TT} is the termination resistance.

$$P_{term\_oh} = (V_{dd} - V_{TT})(V_{oh} - V_{TT})/R_{TT} \tag{2.4}$$

$$P_{term\_ol} = V_{TT}(V_{TT} - V_{ol})/R_{TT} \tag{2.5}$$

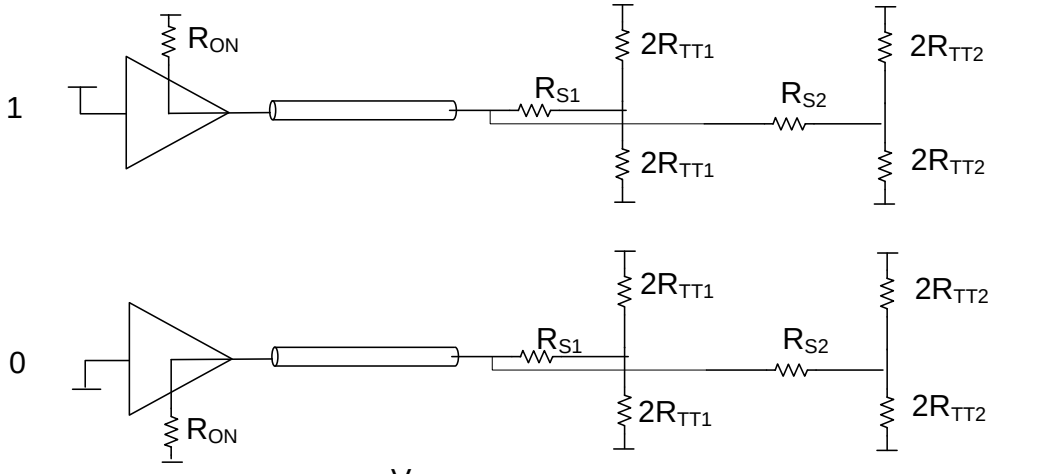
$$P_{avg} = (P_{term\_oh} + P_{term\_ol})/2 \tag{2.6}$$

$$P_{Totavg\_term} = \sum P_{avg} \tag{2.7}$$

Terminations are used to improve signal integrity and achieve higher speeds, and the values depend on the interconnect length as well as the frequency or timing requirements. Terminations on the DQ (data) bus typically use an ODT (on-die termination) scheme, while those on the CA (command-address) bus use a fly-by termination scheme to the multiple loads. Figures 2.3 and 2.4 show the DDR3 DQ and CA termination schemes along with the static current consumed by them as used in Micron's power calculator [220].

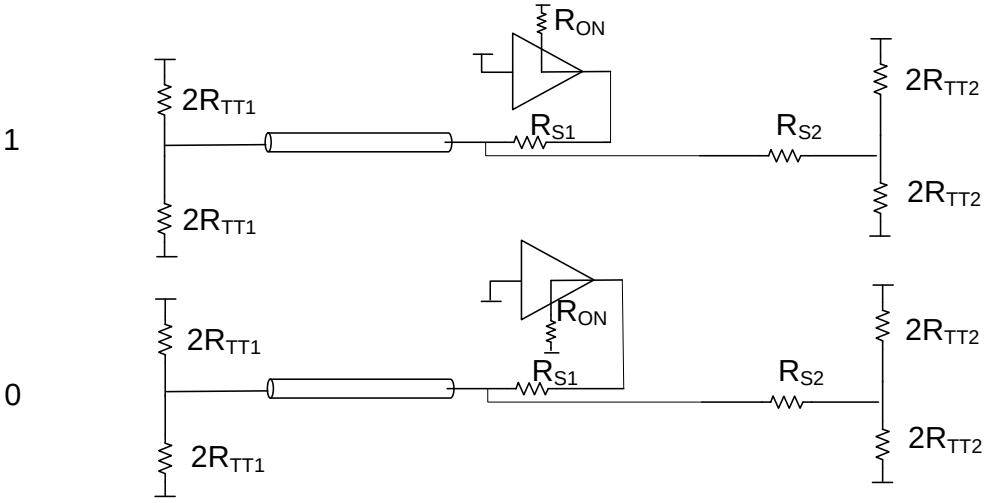
(i) Unterminated. No termination power.

(ii) Center-tap termination, as in DDR3. The DQ WRITE, DQ READ, and CA powers are described in Equations (2.8) - (2.10) respectively. R_{ON} is the driver impedance, R_{TT1} and R_{TT2} are the effective termination impedance of the used and unused ranks, respectively. $R_{||}$ is the effective impedance of both the ranks seen together. For the CA case, R_{TT} is the effective fly-by termination. R_{S1} and R_{S2} are the series resistors used for better signal integrity.



$$I_{Totavg} = \frac{V_{dd}}{4(R_{TT1} \,||\, R_{TT2} \,||\, (R_{ON}+R_{||}))} \qquad R_{||}=(R_{TT1}+R_{S1})||(R_{TT2}+R_{S2})$$

(a) WRITE



$$I_{Totavg} = \frac{V_{dd}}{4(R_{TT1} \,||\, R_{TT2} \,||\, (R_{ON}+R_{S1}+R_{||}))} \qquad R_{||}=(R_{TT1})||(R_{TT2}+R_{S2})$$

(b) READ

Figure 2.3: DDR3 DQ dual-rank termination.

$$P_{DQ\_Term} = \frac{V_{dd}^2}{4} \cdot \left(\frac{1}{R_{TT1}} + \frac{1}{R_{TT2}} + \frac{1}{R_{ON}+R_{||}} \right) \tag{2.8}$$

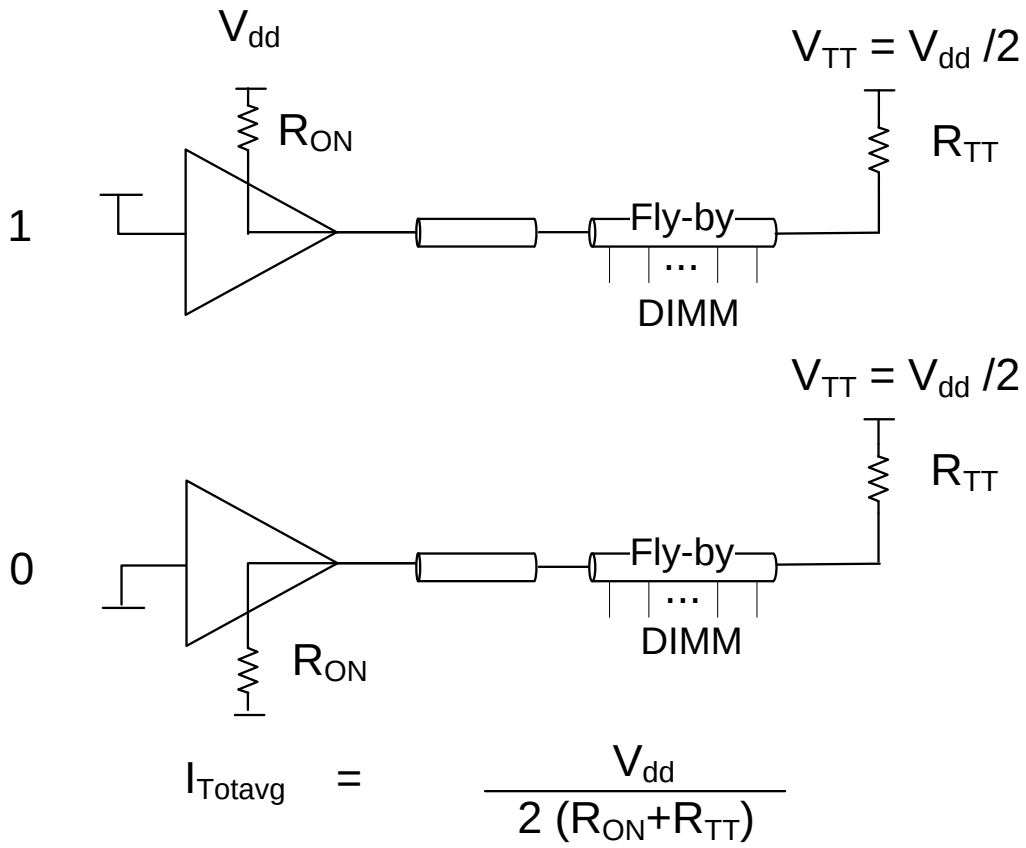


Figure 2.4: DDR3 CA termination.

$$P_{DQ\_Term} = \frac{V_{dd}^2}{4} \cdot \left(\frac{1}{R_{TT1}} + \frac{1}{R_{TT2}} + \frac{1}{R_{ON} + R_{S1} + R_{||}^{read}} \right) \tag{2.9}$$

$$P_{CA\_Term} = \frac{V_{dd}^2}{4} \cdot \left(\frac{1}{50 + R_{TT}} \right) \tag{2.10}$$

CACTI-IO calculates the voltage swing as follows. This calculation feeds into the dynamic power calculation of Equation (2.1). The swing is calculated at the two loads and on the line as shown in Figure 2.3 for both WRITE and READ modes.

WRITE:

$$V_{sw-line} = \frac{V_{dd} \cdot R_{||}}{(R_{ON} + R_{||})} \tag{2.11}$$

$$V_{sw-load1} = \frac{V_{dd} \cdot R_{TT1}(R_{S2} + R_{TT2})}{(R_{S1} + R_{TT1} + R_{S2} + R_{TT2})(R_{ON} + R_{||})} \tag{2.12}$$

$$V_{sw-load2} = \frac{V_{dd} \cdot R_{TT2}(R_{S1} + R_{TT1})}{(R_{S1} + R_{TT1} + R_{S2} + R_{TT2})(R_{ON} + R_{||})} \tag{2.13}$$

$$where\ R_{||} = (R_{TT1} + R_{S1})||(R_{TT2} + R_{S2}) \tag{2.14}$$

READ:

$$V_{sw-line} = \frac{V_{dd} \cdot R_{||}^{read}}{(R_{ON} + R_{S1} + R_{||}^{read})} \tag{2.15}$$

$$V_{sw-load1} = \frac{V_{dd} \cdot R_{TT1}(R_{S2} + R_{TT2})}{(R_{TT1} + R_{S2} + R_{TT2})(R_{ON} + R_{S1} + R_{||}^{read})} \tag{2.16}$$

$$V_{sw-load2} = \frac{V_{dd} \cdot R_{TT2} R_{TT1}}{(R_{TT1} + R_{S2} + R_{TT2})(R_{ON} + R_{S1} + R_{||}^{read})} \tag{2.17}$$

$$where\ R_{||}^{read} = (R_{TT1})||(R_{TT2} + R_{S2}) \tag{2.18}$$

(iii) Differential termination for low-swing differential interfaces. The power for a typical differential termination scheme is as follows.

$$P_{diff\_term} = 2 \cdot V_{dd} V_{sw} / R_{TT} \tag{2.19}$$

In some cases, differential low-swing transmitter circuits could use a small voltage-regulated supply to generate a voltage-mode output [233]. In such a situation, the termination power would be one half of the value given in Equation (2.19).

(iv) VDDQ and VSSQ terminations. We next present a power equation for a VDDQ-termination for DDR4 [178] and LPDDR3 [214]. The DDR4 and LPDDR3 specifications use a VDDQ termination scheme [207], i.e., a single termination resistor connected to the VDDQ supply. This is similar to other POD (pseudo-open-drain) schemes used by JEDEC [207]. The equations for the voltage swing for such a

termination scheme are the same as for DDR3 above in Equations (2.11) - (2.18). However, the signal is referenced to VDDQ rather than VDDQ/2, resulting in the power equation of Equation (2.20), where $R_{||}$ is calculated for WRITE and READ modes similar to the DDR3 DQ case (Equations (2.14) and (2.18)). The power shown in Equation (2.20) assumes 50% 0's and 50% 1's on the line. It must be noted that driving a 1 in this case results in no termination power. The CA termination would be similar to the DDR3 fly-by scheme.

$$P_{DQ\_Term} = 0.5 \cdot V_{dd}^2 \cdot \left(\frac{1}{R_{ON} + R_{||}} \right) \tag{2.20}$$

Termination schemes that are VDDQ or VSSQ terminated can benefit from significant idle power reductions by idling the bus at the same polarity of the termination. LPDDR3 supports the unterminated, full-swing interface as well.

(4) PHY Power. The PHY includes analog and digital components used to retime the IO signals on the interface. A wide range of implementations [73, 112, 91, 108, 33, 144] exist for the PHY that vary in power and are fine-tuned to specific design requirements. Currently, the user can change the inputs for the PHY power based on a specific implementation. Tables 2.1 and 2.2 respectively show the active dynamic power per bit and static power for the entire PHY of an example PHY implementation for a x128 3D configuration. The building blocks are representative of typical PHY components [73, 112, 91, 108, 33, 144]. Table 2.3 shows the dynamic and static power for an example DDR3-1600 PHY. At lower data rates, certain components are not required, indicated by N/A in Tables 2.1 and 2.2.

The building blocks listed include blocks that typically retime a source-synchronous interface using a forwarded clock scheme [26]. The *Datapath* refers to the data transmit path until the input to the IO Tx and the data receive path after the IO Rx. The *Phase Rotator* is a delay element used to generate a T/4 delay to center-align the data-strobe (DQS) with respect to the data (DQ) pins. It could be a DLL or any other delay element that meets the requirements on the edge placement error (T_{error}) described in Section 2.2. The *Clock Tree* is the local clock-tree within the PHY that distributes the clock to all the bit lanes. The *Rx* refers to the IO receiver, which typically consumes some static power for DDR3 SSTL (stub-series terminated logic), owing to a pseudo-differential V_{ref} based receiver first stage. Some PHY implementations have a *Duty Cycle Correction* that corrects duty-cycle distortion, *Deskewing* that reduces

static skew offsets, $Write/Read\ Leveling$ that lines up the various data byte lanes with the fly-by clock and a *PLL* dedicated for the memory interface. The static skew (T_{skew_setup}, T_{skew_hold}) on the interface and the duty-cycle distortion (T_{DCD}) can be reduced if the PHY implements a deskewing scheme and a duty-cycle corrector.

Specific implementations could have other blocks not listed here, but the framework supports easy definition of dynamic and static active and idle power for each of the building blocks. Each building block in the PHY has an idle and sleep state, similar to the IO. CACTI-IO provides these PHY parameters for a few standard configurations included within it. If a new PHY architecture is being investigated, the architect will have to work with the PHY datasheet or IP provider to obtain the model inputs. Frequency scaling can be implemented suitably by going into idle and sleep states for the various blocks based on the frequency of operation. These blocks often have wakeup times when entering active mode from idle and sleep modes, and these wakeup times can be modeled within CACTI-IO. Table 2.4 shows example wakeup times for the building blocks in the PHY. The wakeup times fall into a few broad categories:

- Closed loop blocks need large (order of μseconds) wakeup times to lock the loop. Sometimes designs try to optimize lock times but tradeoffs with loop dynamics and jitter performance need careful consideration [96].
- Mixed-signal or analog blocks may need bias setup times, which could range from μseconds to few nanoseconds, depending on the type of bias (e.g., a bandgap or a self-referenced receiver).
- Digital settings on mixed-signal blocks, e.g., delay line settings or voltage reference settings could change from active to idle and sleep modes. Changing these often requires settling time on the order of a few nanoseconds.
- Digital datapaths may need clock synchronization during frequency changes, and this could cause a wakeup time of a few clock cycles.

The wakeup time reported by CACTI-IO can be used by a system simulator to take into account the latency associated with such frequency scaling.

The above four components of the IO and PHY power are combined as follows, according to the mode of the interface.

Table 2.1: PHY active dynamic power per bit for 3D configurations.

Building Block	Dynamic Power (mW/Gbps)		
	500 Mbps	1 Gbps	2 Gbps
Datapath	0.1	0.2	0.5
Phase Rotator	N/A	0.1	0.2
Clock Tree	0.05	0.2	0.4
Duty Cycle Correction	N/A	N/A	0.05
Deskewing	N/A	N/A	0.1
PLL	N/A	N/A	0.1

Table 2.2: PHY static power for a x128 3D configuration.

Building Block	Static Power (mW)		
	500 Mbps	1 Gbps	2 Gbps
Phase Rotator	N/A	1	10
PLL	N/A	N/A	5

Table 2.3: PHY dynamic power per bit and static power for a x64 DDR3-1600.

Building Block	Dynamic Power (mW/Gbps)	Static Power (mW)
Datapath	0.5	0
Phase Rotator	0.01	10
Clock Tree	0.05	0
Rx	0.5	10
Duty Cycle Correction	0.05	0
Deskewing	0.1	0
Write/Read Leveling	0.05	0
PLL	0.05	10

Table 2.4: PHY wakeup times from sleep and idle modes.

Building Block	Sleep to Active	Idle to Active
PLL	10 μs	0
Phase Rotator	5 μs	0
Rx	2 μs	2 ns
Bandgap	10 μs	0
Deskewing	3 ns	0
Vref Generator	0.5 μs	0

WRITE or READ:

$$P_{Total\_Active} = P_{dyn} + P_{dyn\_interconnect} + P_{term} + P_{static/bias} \tag{2.21}$$

IDLE:

$$P_{Total\_Idle} = P_{term} + P_{static/bias} + P_{dyn\_clock} \tag{2.22}$$

SLEEP:

$$P_{Sleep} = P_{leakage} \tag{2.23}$$

The duty cycle spent in each mode can be specified using the *duty cycle* input parameter.

2.2.2 Voltage and Timing Margins

The minimum achievable clock period T_{ck} depends on the voltage and timing budgets (i.e., eye diagram and/or BER (bit error rate) compliance).

Traditionally, the memory interface budgets have been based on a worst-case analysis approach shown in Figure 2.2, where the budgets are divided between the DRAM, the interconnect, and the controller chip or SoC (system-on-chip). With increasing speeds there is a need for a statistical analysis approach similar to serial links [15] during detailed design analysis. However, for architectural exploration, we continue to use worst-case budgets in our initial framework, with the option of accounting for optimism or pessimism based on prior correlation between the two approaches, or with measurements. This correlation factor also helps address different BER requirements for server DIMM modules that include error correction (ECC) schemes [103, 195, 123].

(1) Timing budgets. The key interface timing equations are based on DRAM AC timing parameters in the JEDEC specification [177, 213]. There are nuances to the system timing based on the controller and PHY design, but most rely on measuring setup and hold slacks to ensure positive margins.

It is interesting to note that while the DQ bus is DDR in almost all DRAMs today, the CA bus is mostly SDR (single data rate), except for LPDDR2 and LPDDR3 where the CA bus is DDR [177, 213]. In addition, the CA bus provides an option for 2T (two clock cycles) and 3T (three clock cycles) timing to relax the requirements when heavily loaded. This is done since the CA bus is typically shared across all memories in the DIMM.

The jitter on the interface is the true limiter of the timing budget, and optimizing the interface for low jitter is the key challenge. The common sources of jitter include Tx jitter, ISI (inter-symbol interference), crosstalk, SSO (simultaneously switching outputs), supply noise, and Rx jitter [103].

Jitter can be estimated from various deterministic (DJ_i) and random (RJ_i) sources as follows [103]. Q_{BER} is a Q-function at the desired BER [103], and σ_i is the standard deviation of the random source. The user can calculate the jitter at the desired BER and enter it into Equations (2.27) - (2.38).

$$T_{jitter} = \sum_i DJ_i + \sqrt{\sum_i RJ_i^2} \tag{2.24}$$

$$RJ_i = 2 \cdot Q_{BER} \cdot \sigma_i \tag{2.25}$$

$$T_{jitter}(\mathbb{F}_0) = T_{jitter\_avg} + \sum_i (T_{jitter}(F_i = F_{i0}) - T_{jitter\_avg}) \tag{2.26}$$

Here, factor F_i is a parameter that affects T_{jitter} [103]. $\mathbb{F}_0$ is the value of a set of factors $F_i = F_{i0}$ for which we calculate the jitter, $T_{jitter}(\mathbb{F}_0)$, as an estimate assuming there is no interaction between the factors F_i [103]. This is done efficiently by running a Design of Experiments (DOE) for a set of orthogonal array experiments as defined by the Taguchi method [103, 137]. T_{jitter_avg} represents the average jitter from all the experiments in the orthogonal array, while $T_{jitter}(F_i = F_{i0})$ represents the average jitter from all experiments where $F_i = F_{i0}$. For cases where F_{i0} is not part of the orthogonal array, a piecewise linear approximation is employed.

The key interface timing equations are described below. T_{jitter_hold} (resp. T_{jitter_setup}) are the half-cycle jitter for hold (resp. setup) between DQ and DQS, and T_{jitter} is the full-cycle jitter. Depending on the implementation, either T_{jitter_setup} or T_{jitter_hold} may be quite small as the DQ and DQS track each other from a common source clock in a forwarded clock scheme, but the other edge of the eye would incur the half-cycle jitter. T_{error} is the edge placement error of the T/4 delay element, T_{skew} is the static skew in the interface, and $T_{rise/fall(V_{ref}->V_{IH/IL})}$ is the rise/fall time at the *Rx* input from V_{ref} (the switching reference voltage) to $V_{IH/IL}$ (the *Rx* thresholds). T_{SoC_hold} and T_{SoC_setup} are the hold and setup times at the SoC inputs during READ. $T_{DCD-SoC}$ is the DCD of the SoC clock output. T_{cor_margin} is a correlation term that allows the user to account for either a measurement correlation to the timing equations. The remaining parameters in the equations below are JEDEC DRAM parameters [177, 213].

(i) DQ-DQS WRITE:

$$\left(\frac{T_{ck}}{4}\right) - T_{DCD-SoC} - T_{error} - T_{jitter\_hold} - T_{skew\_hold} > T_{DHbase} + T_{rise/fall(V_{ref}->V_{IH/IL})} - T_{cor\_margin} \tag{2.27}$$

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_setup} - T_{skew\_setup} > T_{DSbase} + T_{rise/fall(V_{ref}->V_{IH/IL})} - T_{cor\_margin} \tag{2.28}$$

(ii) DQ-DQS READ:

$$T_{QSH/QSL} - T_{DCD-SoC} - T_{error} - T_{jitter\_hold} - T_{skew\_hold} > T_{SoC\_hold} - T_{cor\_margin} \tag{2.29}$$

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_setup} - T_{skew\_setup} - T_{DQSQ} > T_{SoC\_setup} - T_{cor\_margin} \tag{2.30}$$

(iii) CA-CLK (DDR for LPDDR2/3):

$$\left(\frac{T_{ck}}{4}\right) - T_{DCD} - T_{error} - T_{jitter\_hold} - T_{skew\_hold} > T_{IHbase} + T_{rise/fall(V_{ref}->V_{IH/IL})} - T_{cor\_margin} \tag{2.31}$$

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_setup} - T_{skew\_setup} > T_{ISbase} + T_{rise/fall(V_{ref}->V_{IH/IL})} - T_{cor\_margin} \tag{2.32}$$

For DDR3 the CA interface is SDR, and the above timing is relaxed to a half-cycle, as follows:

$$\left(\frac{T_{ck}}{2}\right) - T_{jitter\_hold} - T_{skew\_hold} > T_{IHbase} + T_{rise/fall(V_{ref}->V_{IH/IL})} - T_{cor\_margin} \tag{2.33}$$

The CA timing can be further relaxed if the 2T or 3T timing option is enabled in the DDR3 DRAM.

2T:

$$T_{ck} - T_{jitter\_hold} - T_{skew\_hold} > T_{IHbase} + T_{rise/fall(V_{ref}->V_{IH/IL})} - T_{cor\_margin} \tag{2.34}$$

3T:

$$\left(\frac{3 \cdot T_{ck}}{2}\right) - T_{jitter\_hold} - T_{skew\_hold} > T_{IHbase} + T_{rise/fall(V_{ref}->V_{IH/IL})} - T_{cor\_margin} \tag{2.35}$$

The setup equations are similarly relaxed.

(iv) CLK and DQS:

$$\left(\frac{T_{ck}}{2}\right) - T_{DCD} - T_{jitter\_setup/hold} + T_{cor\_margin} > T_{CH/CL\_abs} T_{jitter} < T_{JIT} + T_{cor\_margin} \tag{2.36}$$

$$T_{jitter\_hold} + T_{skew\_hold} + T_{DCD} < \left(\frac{T_{ck}}{2}\right) - T_{DSH} + T_{cor\_margin} \tag{2.37}$$

$$T_{jitter\_setup} + T_{skew\_setup} < \left(\frac{T_{ck}}{2}\right) - T_{DSS} + T_{cor\_margin} \tag{2.38}$$

(2) Voltage Budgets. A voltage budget can be developed for voltage margins as follows [26], which once again is based on a worst-case analysis, where V_N is the voltage noise, K_N is the proportionality coefficient for the proportional noise sources (that are proportional to the signal swing V_{sw}), V_{NI} is the noise due to independent noise sources and V_M is the voltage margin. Crosstalk, ISI (inter-symbol interference), and SSO (simultaneously switching outputs) are typical proportional noise sources [26], while the *Rx*-offset, sensitivity, and independent supply noise are typical independent noise sources.

$$V_N = K_N \cdot V_{sw} + V_{NI} \tag{2.39}$$

$$K_N = K_{xtalk} + K_{ISI} + K_{SSO} \tag{2.40}$$

$$V_{NI} = V_{Rx-offset} + V_{Rx-sens} + V_{supply} \tag{2.41}$$

$$V_M = \frac{V_{sw}}{2} - V_N \tag{2.42}$$

A DOE analysis for the voltage noise coefficient, K_N, can be performed in a similar manner as described above for T_{jitter}.

2.2.3 Area Models

The area of the IO is modeled as shown below in Equation (2.43), where N_{IO} is the number of signals, f is the frequency, and R_{ON} and R_{TT1} are the impedance of the IO driver and the on-die termination circuit respectively as shown in Figure 2.3. A_0, k_0, k_1, k_2, and k_3 are constants for a given technology and design. They need to be fitted based on data from the PHY IP provider or datasheet.

$$\begin{aligned} Area_{IO} = N_{IO} \cdot \left(A_0 + \frac{k_0}{min(R_{ON}, 2 \cdot R_{TT1})} \right) + \\ N_{IO} \cdot \left(\frac{1}{R_{ON}} \right) \cdot (k_1 * f + k_2 * f^2 + k_3 * f^3) \end{aligned} \tag{2.43}$$

The area of the last stage of the driver is proportional to $1/R_{ON}$ or the drive current, and the fanout in the IO for the predriver stages is proportional to f, the frequency of the interface, to reflect the proportional edge rates needed based on the frequency. In the event that the on-die termination ($2 \cdot R_{TT1}$) is smaller than R_{ON}, the driver size is determined by $1/(2 \cdot R_{TT1})$. A_0 is the fixed area of the rest of the IO, which includes ESD protection.

The case studies in Sections 2.5.3 and 2.5.5 discuss area results and the importance to keep area in mind when widening the bus. Further area tradeoffs of interest that can be explored using the tool can be found in the technical report [209].

2.3 Technology Portability

The models described in Section 2.2 above are dependent on on-die as well as off-chip technology. As with prior CACTI versions, the IO and off-chip parameters that scale with process technology are taken from ITRS [202]. The underlying assumption is that the DRAM technology scales to meet the speed bin that it supports [207], since if DRAM technology is scaled, the speed bin that the IO parameters belong to are suitably scaled as well, including load capacitances (DRAM DQ pin capacitance (C_{DQ}), DRAM CA

pin capacitance (C_{CA})), and AC timing parameters in Equations (2.27) - (2.38). LPDDRx use different technologies compared to DDRx to save leakage power, so their capacitances and timing parameters are different from a DDRx memory of the same speed bin. Voltage also scales with DRAM technology, typically when a DRAM standard changes, e.g., DDR2 used 1.8V IO supply voltage, while DDR3 uses 1.5V IO supply voltage [207]. Sometimes a lowered voltage specification is released as an addendum to a standard, e.g., DDR3-L [207]. Shown below in Table 2.5 are a subset of DDR3 DRAM parameters based on the speed bin.

If the user is interested in studying the impact of technology on a future memory standard, or a speed bin that is yet undefined, to first order the timing parameters can be assumed to scale down linearly with frequency.

Table 2.5: Technology scaling for DDR3.

Parameter	**Data rate (Mb/s)**		
	800	**1066**	**1600**
vdd_io (V)	1.5	1.5	1.5
c_data_max (pF)	3.0	3.0	2.3
c_addr_max (pF)	1.5	1.5	1.3
t_ds_base (ps)	75	25	10
t_dh_base (ps)	150	100	45
t_dqsq (ps)	200	150	100

The SoC PHY power and timing parameters scale with the technology node of the SoC, but are far more sensitive to the circuit architecture and analog components used to implement the design. It is hard to provide simplistic scaling trends for these parameters. For a given design and architecture, it would be possible to provide scaling power and timing for different technology nodes, but as speeds increase, the design and architecture for the PHY and IO are optimized and/or redesigned for the higher speed. Various design-specific trends for power and timing scaling with technology suggest around 20% scaling of analog power from one technology node to the next, or from one speed bin to the next [112].

The area of the IO directly scales mostly with the thick-oxide device of the technology. The scaling of the thick-oxide device typically does not keep pace with the core thin-oxide device as a consequence of supply voltages for external standards and reliability concerns. The constants k_0, k_1, k_2, and k_3 scale inversely with $I_{dsat}/\mu m$ of the thick-oxide device.

Besides the parameters that scale with technology, the topology impacts the models for timing and voltage noise. A suitable DOE is required to fit the jitter and voltage noise coefficients for a given topology that defines the number of loads and interconnect length. When defining a topology other than the three standard configurations, a DOE analysis (as shown in Section 2.4) needs to be performed to be able to port the timing models for the channel.

The user can also add a new configuration into CACTI-IO to evaluate a future standard. For every new technology, voltage and timing DOE will need to be run for the given loading, as described in Section 2.2.2. The IO area for a new technology can be obtained by curve fitting the constants in Equation (2.43) using an IO datasheet. Guidance on how to modify the power models can be found in the technical report [209].

2.4 Validation

We now discuss validation of the new analytical IO and off-chip models added in CACTI-IO. The analytical power models are verified to be within 1-15% of SPICE results. Models that are based on a lookup table, including the PHY power numbers, are valid by construction.

We first validate the power models for each DQ and CA bit line. Figures 2.5 and 2.6 show SPICE vs. CACTI-IO for the termination power and total IO power of a single lane of DQ DDR3. Figure 2.5 indicates that the worst case error between SPICE and CACTI-IO is less than 1% across different R_{TT1} values (R_{ON}= 34 Ω for these cases). The total IO power shown in Figure 2.6 for three different combinations of C_{DRAM}, R_{TT1} and T_{flight} shows a worst error of less than 14%.

Figures 2.7 and 2.8 show SPICE vs. model for the termination power and total IO power of a single lane of CA DDR3 using a fly-by termination scheme. Figure 2.7 shows the termination power for different R_{TT} values (the fly-by termination shown in Figure 2.4), while Figure 2.8 shows the total IO power for different numbers of loads or fly-by segments. Once again, the errors are similar to the DQ cases above, with the termination power within 1% and the total IO power within 15%.

Figures 2.9 shows SPICE vs. model for the switching power (dynamic IO and interconnect power) for DQ LPDDR2, where no terminations are used. In this scenario, the model is within 2% of the SPICE simulation.

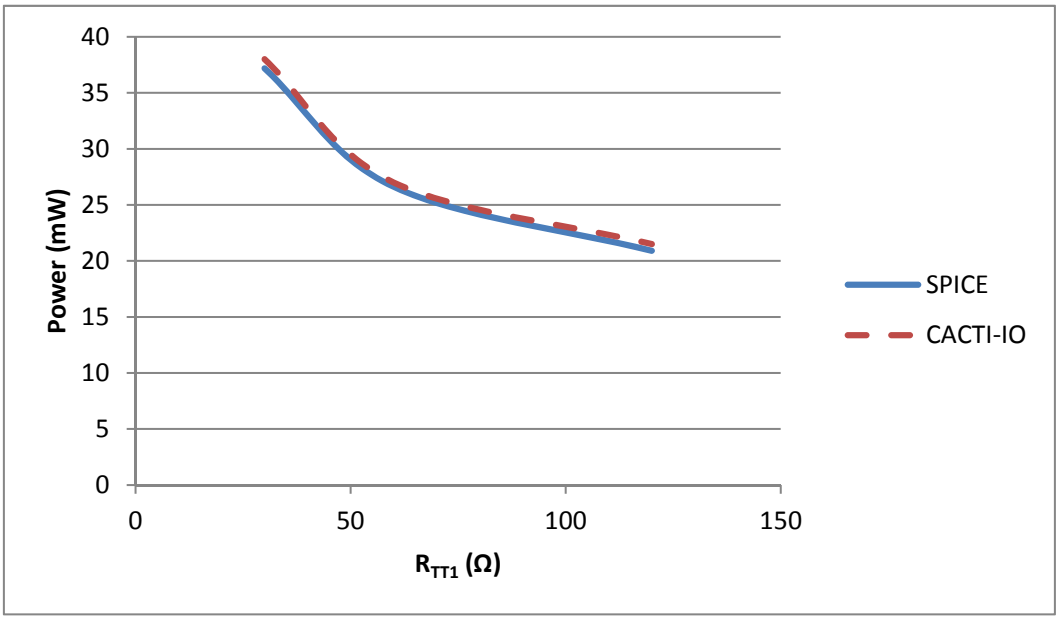


Figure 2.5: DQ single-lane DDR3 termination power.

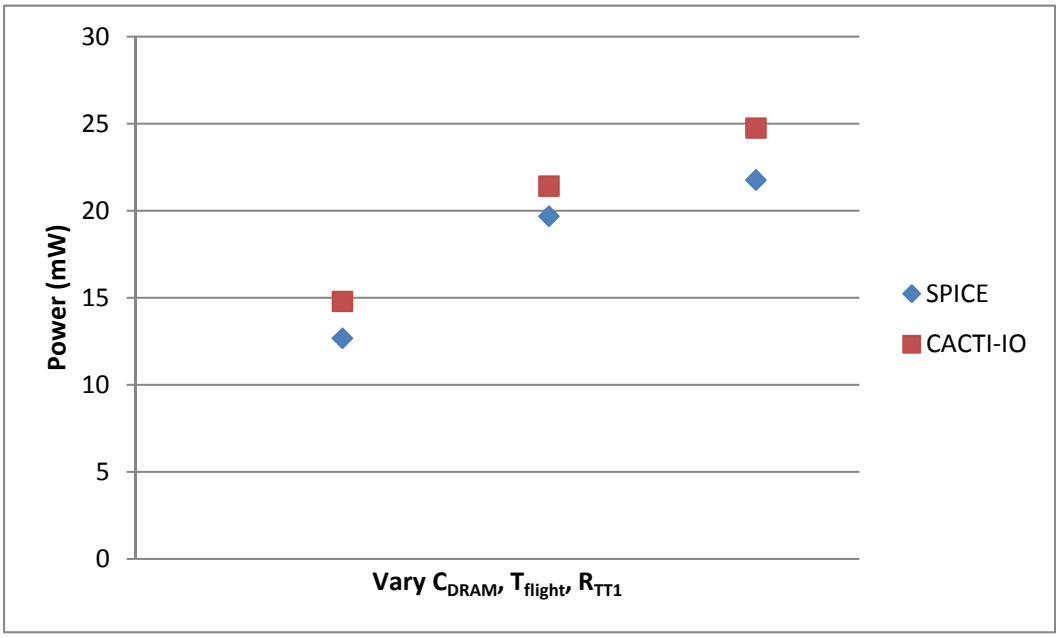


Figure 2.6: DQ single-lane DDR3 total IO power.

To validate the power model for the entire interface, we compare it against measurements. Shown in Figure 2.10 is measured vs. model power for LPDDR2 WRITE obtained from a typical memory interface configuration for a 32-wide bus using a x32 LPDDR2 dual-rank DRAM. As can be seen, the model is

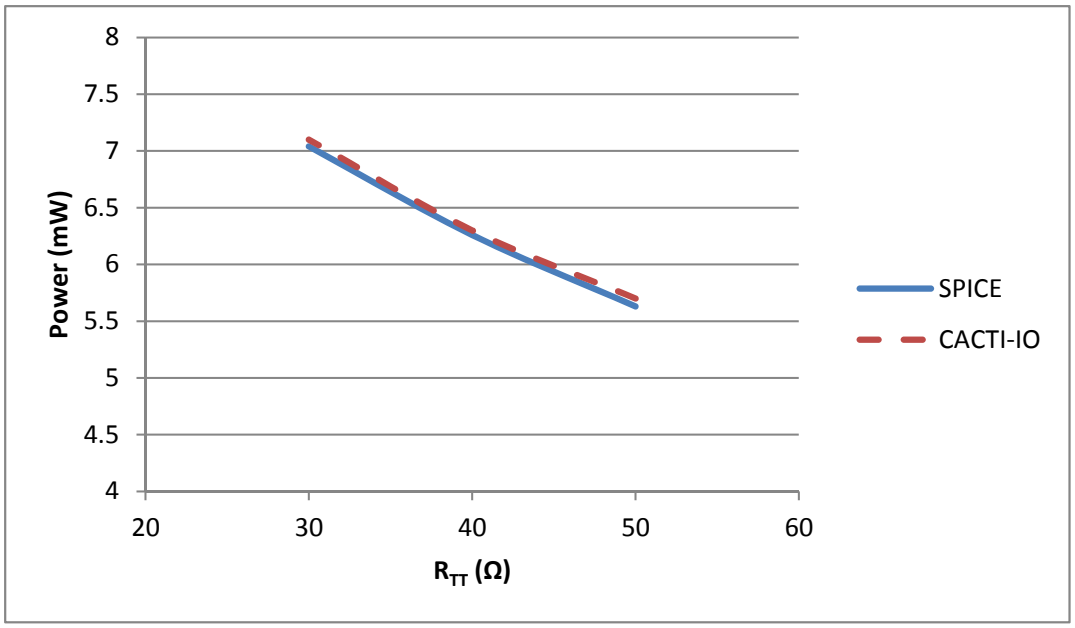


Figure 2.7: CA single-lane DDR3 termination power.

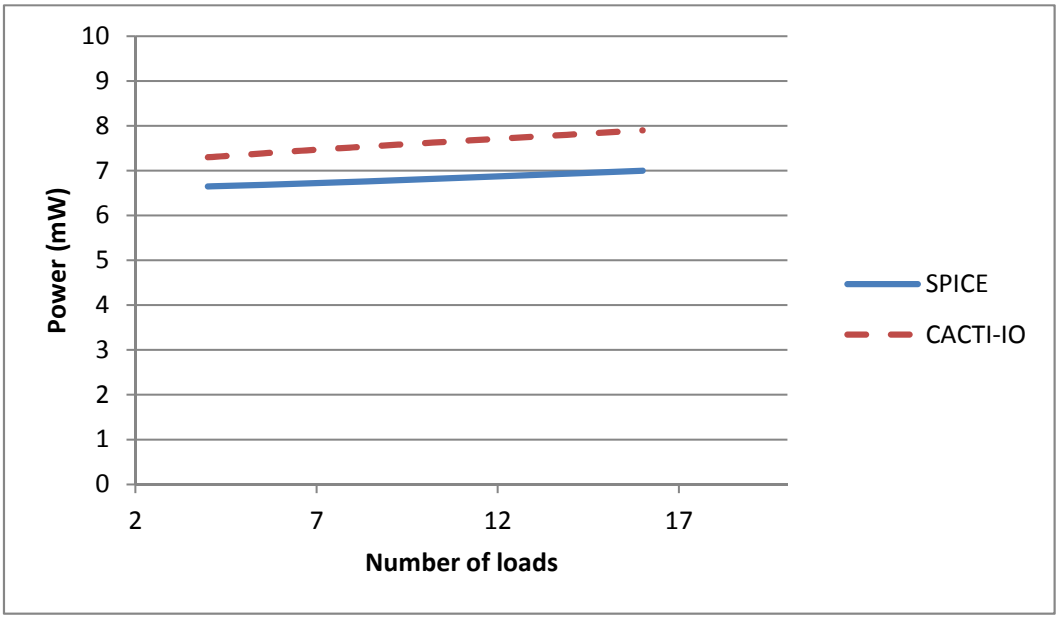


Figure 2.8: CA single-lane DDR3 total IO power.

within 5% of the measurement at the higher bandwidths. At lower bandwidths, power saving features make it harder to model the power as accurately since the duty cycle between the READ/WRITE/IDLE/SLEEP modes is harder to decipher. Here the error is within 15%.

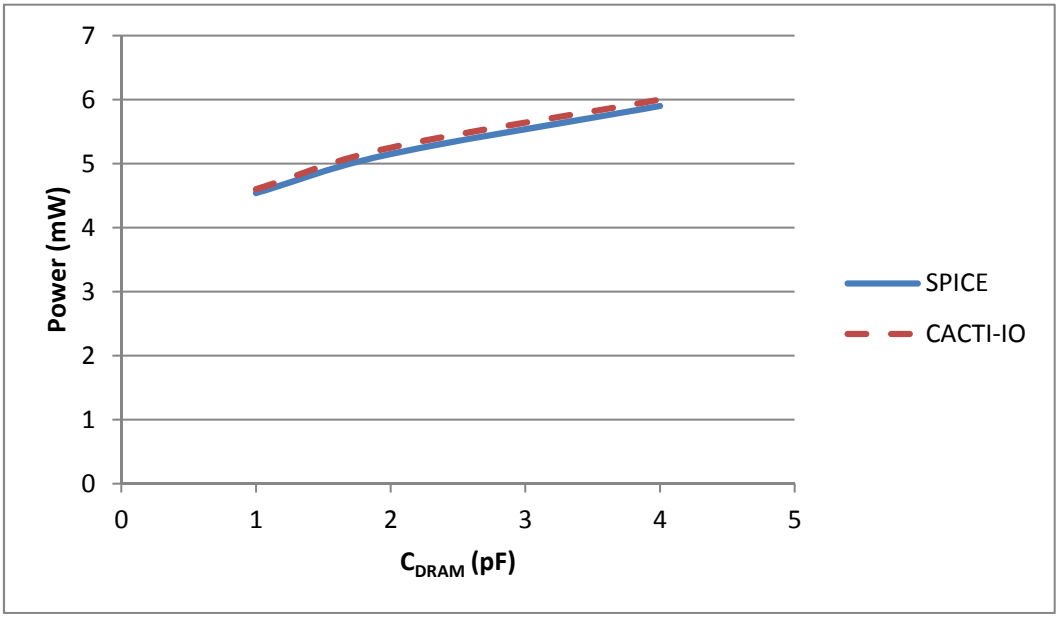


Figure 2.9: DQ single-lane LPDDR2 total IO power.

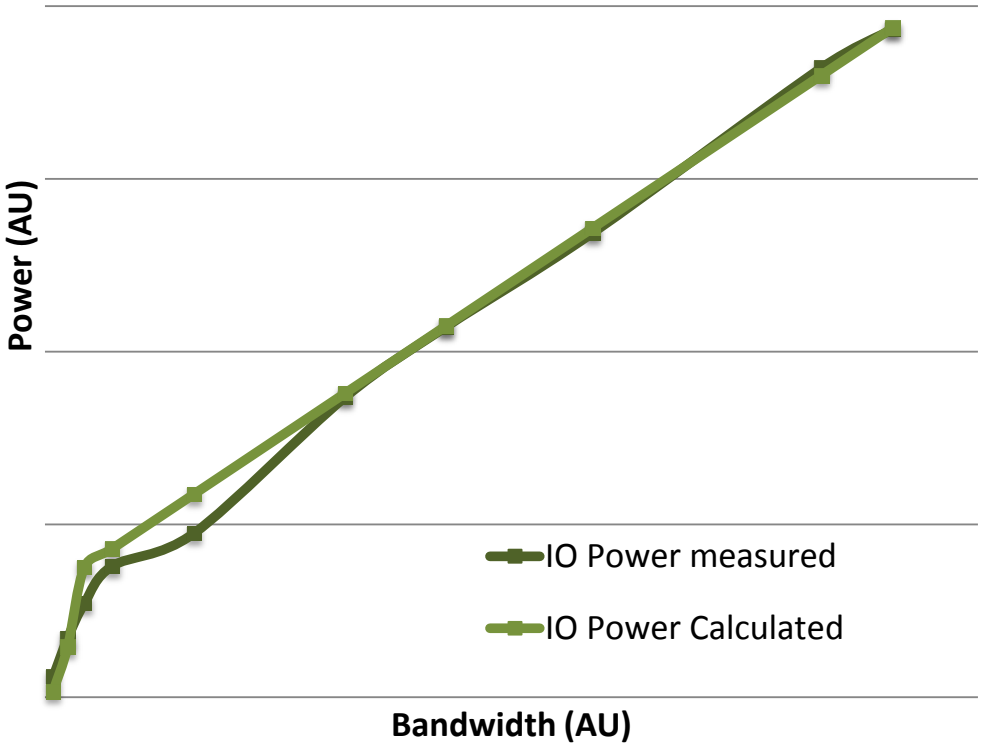


Figure 2.10: LPDDR2 WRITE measurement vs. model.

Shown in Figure 2.11 are the results of an example DOE analysis on a sample channel for T_{jitter}. The input factors (F_i in Equation 2.26) used here are R_{ON}, R_{TT1} and C_{DRAM_DQ}. The simulations are performed for 9 cases as indicated by the Taguchi array method explained in Section 2.2. JMP [208] is then used to create a sensitivity profile. The table of values used for the Taguchi array and the sensitivity profile are shown in Figure 2.11. The profile allows us to interpolate the input variables and predict T_{jitter}. CACTI-IO uses the sensitivity profile to perform the interpolation.

Ron (Ω)	Rtt (Ω)	Cdram (pF)	Pattern	Tjitter (ps)
34	30	1	−−−	45
34	40	1.5	−00	40
34	60	2	−++	66
40	30	1.5	0−0	56
40	40	2	00+	53
40	60	1	0+−	40
48	30	2	+−+	76
48	40	1	+0−	47
48	60	1.5	--0	80

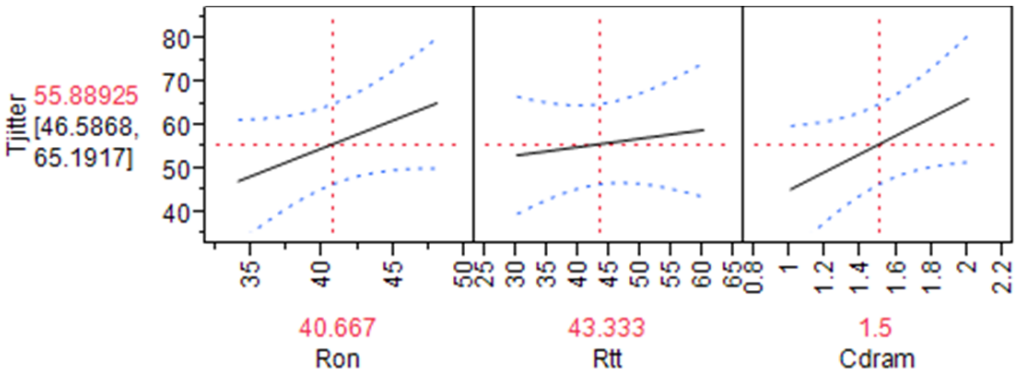


Figure 2.11: DOE analysis on a DDR3 channel.

2.5 CACTI-IO

CACTI-IO is an extended version of CACTI [171] that includes the models described in Section 2.2 above. CACTI-IO allows for a quick search of optimal IO configuration parameters that help optimize power and performance of the IO along with the DRAM and cache subsystem.

CACTI has analytical models for all the basic building blocks of a memory [139]: decoder, sense-amplifier, crossbar, on-chip wires, DRAM/SRAM cell, and latch. We extend it to include the off-chip

models presented in this chapter. This requires modifying CACTI's global on-chip interconnect to include buffers at the PHY and drivers at the bank edge to connect to the IO circuit. Since all calculations are based on the ITRS [202] technology parameters, the energy and delay values calculated by CACTI are guaranteed to be mutually consistent. When a user inputs memory parameters and energy/delay constraints into CACTI, the tool performs an exhaustive design space exploration involving different array sizes, degrees of multiplexing, and interconnect choices to identify an optimal configuration. CACTI-IO is capable of performing an additional search for off-chip parameters, including optimal number of ranks, memory data width (x4, x8, x16 or x32 DRAMs), off-chip bus frequency, and bus width. This allows for optimal tradeoffs between off-chip power, area, and timing.

We present four case studies: (i) high-capacity DDR3-based server configurations in Section 2.5.2; (ii) 3D memory configurations for high-bandwidth systems in Section 2.5.3; (iii) BOOM (Buffered Output On Module), a novel LPDDRx based configuration for servers [157] in Section 2.5.4; and (iv) a PCRAM (Phase Change RAM) study showing separate READ and WRITE buses in Section 2.5.7. All comparisons in the case studies are shown for one channel of the memory controller.

Table 2.6: Case Study 1: Summary of power for server configurations using x4 4Gbit DRAMs.

Configuration	Capacity (GB)	No. of DQ loads	BW (GB/s)	P_{IO} (W)	$P_{CPU-Buf}$ (W)	P_{PHY} (W)	Efficiency (GBps/W)	Efficiency·GB (GB·GBps/W)
2 RDIMMs dual-rank	32	4	12.8	4.7	0.55	0.6	2.19	70.1
3 RDIMMs dual-rank	48	6	12.8	6.2	0.55	0.8	1.70	81.6
3 LRDIMMs dual-rank	48	2	12.8	4.86	3.2	0.8	1.44	69.12
3 LRDIMMs quad-rank w/ 2-die stack	**96**	2x2d	12.8	5.1	3.2	0.8	1.41	135.4
BoB w/ 2 channels 2 dual-rank RDIMMs	64	4	25.6	10.8	0.34	1.2	2.07	132.5

Table 2.7: Case Study 2: Summary of power for 3D configurations.

Config.	Capacity (GB)	Freq. (MHz)	BW_{MAX} (GB/s)	IO Power (W)	PHY Power (W)	Efficiency (GBps/W)	IO Area (sq.mm.)	V_{ddmin} (V)	Eff.@V_{ddmin} (GBps/W)
3D 4-die	2	290	37	0.9	0.06	38	1.5	1	54
3D 8-die	4	290	37	1.2	0.06	30	1.7	1.2	30
4x64	2	533	34	0.74	0.14	38	0.9	1.2	38
4x32	2	1066	34	0.84	0.44	27	0.7	1.2	27
HMC	2	350	**176**	2.96	0.29	56	6.0	0.85	100

The IO power shown in the case studies is the peak power during activity, except in Section 2.5.4 for the BOOM case study, where we show how CACTI-IO can project the total system power as a sum of both IO and DRAM power and provide quick design-space exploration of both off-chip and on-chip

Table 2.8: Case Study 3: Summary of power for BOOM configurations.

Configuration	Capacity (GB)	Freq. (MHz)	No. of DQ loads	BW (GB/s)	P_{IO} (W)	$P_{CPU-Buf}$ (W)	P_{PHY} (W)	Efficiency (GBps/W)
x8 BOOM-N2-D-800	16	800	4	12.8	4.96	3.52	0.8	1.38
x8 BOOM-N4-L-400	32	400	4	12.8	2.51	3.52	0.4	2.0
x8 BOOM-N4-L-400 with serial bus to host	32	400	4	12.8	2.51	0.34	0.4	3.94

components together. The case studies show the variety of options the IO models provide, as well as the achievable range of capacities and power efficiencies, making for interesting tradeoffs for the architect.

To further highlight the utility of CACTI-IO, we study two tradeoffs in more detail for the BOOM designs: in Section 2.5.5 we discuss optimal fanout of the data bus, and in Section 2.5.6 we discuss the optimal fanout of the address bus.

2.5.1 Simulation Methodology

For studies of the high-capacity DDR3 configurations and 3D configurations, we run the CACTI-IO models stand-alone to provide IO power comparisons described in Sections 2.5.2 and 2.5.3 below. For the BOOM cases, we use a multi-core simulator [217] built on top of PIN [88] to provide the activity factor and idle-time information for multi-programmed workload mixes from SPLASH2 [152]. While different benchmarks will yield different results, we expect that overall trends for IO and DRAM power will remain stable. We model a 16-core processor with two memory controllers. Each controller has a dedicated memory channel and each channel has four ranks. Number of reads, writes, activates, idle cycles, and power down cycles from this simulation are fed into CACTI-IO to evaluate the DRAM as well as IO energy averaged over the SPLASH2 benchmarks for the different BOOM configurations described in Section 2.5.4.

2.5.2 High-capacity DDR3 Configurations

We compare several configurations shown in Table 2.6 for a x64 DDR3 memory channel; they all use a DIMM. RDIMM refers to a Registered DIMM, where the command and address signals are buffered to allow for increased capacity. A Load Reduced DIMM (LRDIMM) [215] has a buffer for both address and data signals, allowing further increase in capacity at the cost of some data latency due to the buffering. The quad-rank case shown for LRDIMM uses two dual-die packages (2x2d). The last configuration listed

uses a Buffer-on-Board (BoB) from Intel [169] shown in Figure 2.12. In this configuration, the buffer is not integrated into the DIMM, but is rather a stand-alone chip on the board. The buffer drives two RDIMMs and has two channels (4 RDIMMs in all). While the interface between the RDIMM or LRDIMM and the CPU remains a DDR3 bus, the interface between the BoB and CPU is a proprietary serial interface [169].

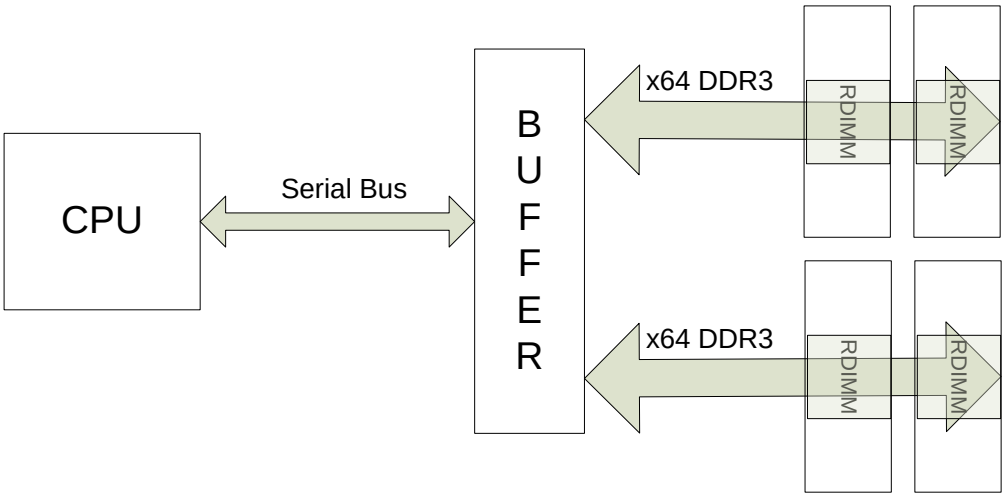


Figure 2.12: BoB (Buffer-on-Board) [169].

All configurations shown in Table 2.6 use x4 4Gb memory devices. We study the interface to the DRAM as the bottleneck in the system, and the timing on the interface between the buffer and the host CPU is assumed not to be the limiting factor in the study. The table lists the power consumed due to the IO on the DRAM interface (P_{IO}), the PHYs (P_{PHY}), and the IO on the interface between the CPU and the buffer ($P_{CPU-Buf}$). For signals that are buffered, although the P_{IO} reduces, $P_{CPU-Buf}$ goes up as it accounts for the buffered signal from the CPU to buffer. All configurations are assumed to operate at 800 MHz (DDR3-1600) and 1.5 V. As can be seen from the table, the LRDIMM offers a 50% increase in capacity (96 GB for a x64 channel) compared to the 3-RDIMM for a 17% decrease in efficiency. The product of capacity and efficiency is the highest for LRDIMM, at 135.4 GB·GBps/W. The BoB configuration offers a 30% increase in capacity and a 2X bandwidth improvement over the 3-RDIMM with 23% better power efficiency. Its product of capacity and efficiency is 132.5 GB·GBps/W.

This case study highlights the ability of CACTI-IO to calculate IO power numbers for various configurations under consideration, and search for an optimal solution based on either total capacity (3-LRDIMM with 2-die stack), or efficiency (2-RDIMM), or perhaps a very good balance between the two

(BoB). The BoB design presents a novel means of increasing capacity using a buffer on the board, while maintaining efficiency and low pin-count using a serial bus to the CPU with 2X the bandwidth (25.6 GB/s).

2.5.3 3D Stacking Using Wide-IO

In our second case study, we evaluate different 3D stacking configurations to maximize bandwidth. The configurations chosen include a 3D TSS (Through Silicon Stack) 4-die 4Gb stacked DRAM with 4x128 channels [66], an 8-die stack with 4x128 channels, and narrower buses (4x64 and 4x32 as opposed to 4x128) with same bandwidth, all of which connect to the CPU directly, exposing the die stack to the external pin loading. We also include the Hybrid Memory Cube (HMC) proposed by Micron [6], wherein the memory controller is included along with the DRAM stack, and connected by a 16x128 interconnect. A serial interface is used to connect the HMC to the CPU. The HMC 1.0 specification [197] supports various speeds (10, 12.5 and 15 Gb/s) for the serial link and supports a full-width (16 lanes) or half-width (8 lanes) option for the number of lanes. There are either 4 or 8 such links depending on what aggregate bandwidth is required. Since the serial interface can support up to 240 GB/s, it is assumed to not limit the bandwidth of the memory access, and focus is on the 16x128 interconnect within the HMC. All configurations operate at 1.2V [189]. The data-rate on the interface is limited by the DRAM timing and voltage parameters and data-rates proposed for Wide-IO [189], although CACTI-IO predicts some changes from the proposed data-rates based on the jitter sensitivity to loading and R_{ON}. Further, the HMC allows for opportunity to explore timing and voltage optimization of the interface to the DRAM, as this is within the DRAM cube. We explore this by relaxing the timing and voltage parameters by 20% for the HMC. This allows the HMC to achieve better power efficiency compared to 3D TSS.

Table 2.7 shows the results for these configurations calculated by CACTI-IO. As can be seen, the power efficiency varies by around 2X, with the HMC showing the highest efficiency (56 GBps/W), and a 3D stack using a 4x32 bus showing the lowest efficiency (27 GBps/W). A peak bandwidth of 176 GB/s for 16x128 channels is achieved for the HMC with a 4-die stack, a 4.76X improvement over the standard 3D TSS stack in an external connection using 4x128 channels. The isolation provided by the HMC to the CPU allows the bus to operate faster without the additional external loading.

The 4x64 and 4x32 cases shown in Table 2.7 represent narrower buses that achieve the same

bandwidth. The PHY power (taken from Tables 2.1 and 2.2) goes up considerably for the x32 case since the complexity increases at 1066 MHz; this leads to the poorest efficiency. CACTI-IO can furthermore predict V_{ddmin} based on the voltage noise parameters as described in Equations (2.39) - (2.42). The V_{ddmin} and the scaled efficiency at V_{ddmin} are shown in Table 2.7. CACTI-IO predicts that the HMC can further scale down to 0.85V and improve its efficiency to 100 GBps/W.

Table 2.7 also includes IO area comparison for the configurations shown, using the model discussed in Equation (2.43). Of interest to note, is that for the narrower buses (4x64 and 4x32), the area decreases, but not by a factor of 2x or 4x respectively. The additional area is to support the higher speed on the narrower bus. The 8-die TSS incurs an area overhead to support the larger load.

As described in Section 2.3, it is also important to note that for the comparisons of 3D configurations we modeled the voltage a timing budgets for a 3D interconnect based on running SPICE simulations to extract the timing and voltage noise coefficients described in Section 2.2.2. C_{Total}, the load capacitance and R_{ON}, the output impedance of the 3DIO are parameters that impact timing and voltage noise. They are used to model T_{jitter} and K_N as described in Section 2.2.2. The remaining timing parameters are from the Wide-IO specification [189].

An important design consideration in 3D DRAM configurations is to architect the internal banks to take best advantage of the improved off-chip bandwidth with the wider interface. Unlike traditional DDR or LPDDR DRAM chips, HMC and Wide-IO memories employ a number of smaller banks to improve the overall bandwidth. When modeling both on-chip and off-chip components in CACTI-IO, CACTI's on-chip design space exploration takes into account the latency of individual banks, and adjusts the internal bank count to match the off-chip IO bandwidth.

This case study highlights the ability of CACTI-IO to calculate IO power and timing for a new interconnect technology such as 3D, including the novel Hybrid Memory Cube. The baseline models included in CACTI-IO can be configured for DDR3-based signaling as well as for 3D interconnect. We see that CACTI-IO is able to identify the solution with the highest bandwidth and efficiency (HMC) and also predict how much the efficiency would be affected when going from 4x128 to 4x64 or 4x32 due to PHY power increase for the higher data rates. CACTI-IO is also able to calculate V_{ddmin} for a given frequency and loading, predicting a 1.8X improvement in power efficiency for the HMC.

2.5.4 BOOM: LPDDRx for Servers

The BOOM (Buffered Output On Module) architecture [157] from Hewlett-Packard relies on a buffer chip on the board that connects to lower-speed and lower-power LPDDRx memories. To match the channel bandwidth, BOOM uses a wider DIMM-internal bus (from the buffer to the DRAMs) as shown in Figure 2.13. Further, BOOM has the option of grouping multiple physical ranks into a single logical rank [157]. BOOM can use commodity LPDDRx DRAMs with lower power, but achieves high bandwidth and capacity through wider buses. As servers become more sensitive to memory subsystem power, BOOM provides a valuable means for use of mobile DRAM to achieve better power efficiency while still meeting server performance.

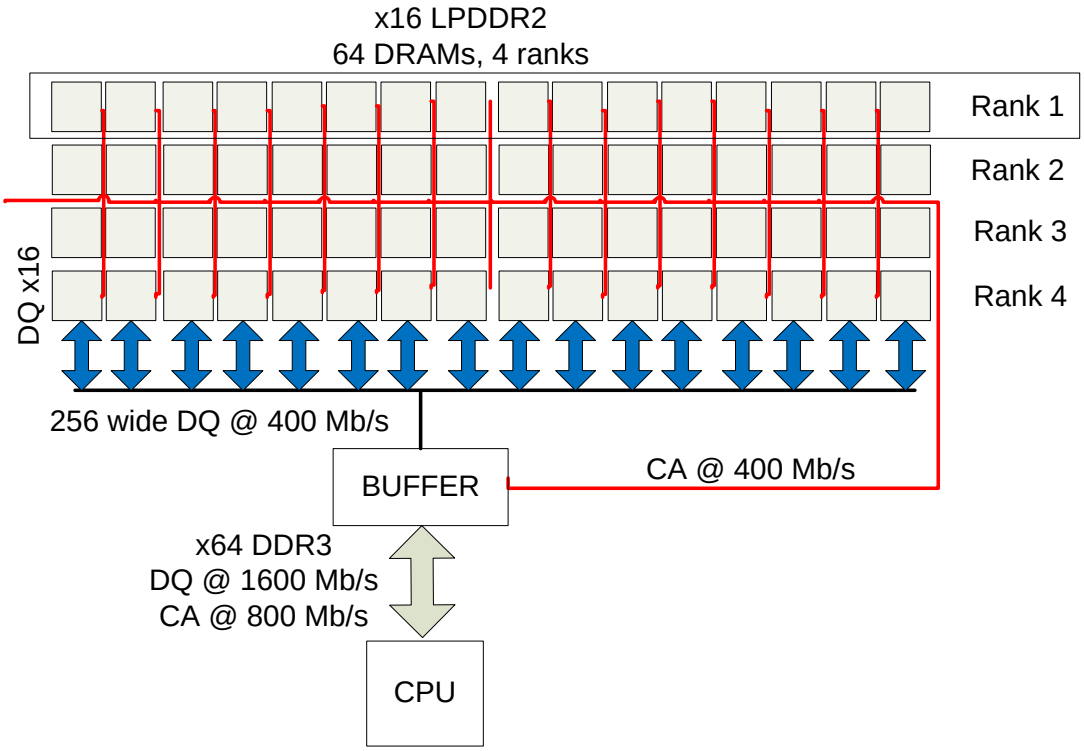


Figure 2.13: BOOM-N4-L-400 configuration with x16 devices [157].

Table 2.8 summarizes the IO peak power for three BOOM configurations [157]. The power is shown per memory channel (equivalent of a x64 DDR3 channel). A BOOM configuration is denoted as BOOM-N*n*-X-Y, where *n* is a ratio of the wider internal bus to the channel's x64 bus, X is DRAM type (D for DDR3 and L for LPDDR2) and Y is DRAM data rate (typically 1600/n Mb/s). All BOOM configurations shown use x8 memories.

Table 2.8 clearly shows a 2X improvement in IO power (P_{IO}) from buffer to DRAM using LPDDRx memories to achieve the same bandwidth when we compare BOOM-N2-D-800 (using DDR3 DRAM) and BOOM-N4-L-400 (using LPDDR2 DRAM).

Additionally, BOOM offers the advantage of using a custom interface between the CPU host and the buffer chip. Instead of a standard x64 DDR3 interface, a serial bus similar to the BoB [169] case in Section 2.5.2 above can be used. This further improves the total efficiency by 2X, achieving a 2.85X improvement in total power efficiency over a DDR3 design.

To highlight the ability of CACTI-IO to provide combined DRAM and IO power, we compare the three BOOM configurations with respect to normalized energy in Figure 2.14.

The simulation methodology used to obtain the normalized energy is described in Section 2.5.1. The total energy is broken down into the DRAM core power (Read, Write, Activate, Idle), the IO Active power (Read and Write), and the IO Idle power (mainly due to terminations and the active clock). The Precharge power is included in the Activate power. The total Idle power (also referred to as Background power [157]) is got from adding the DRAM core Idle power and the IO Idle power.

We make the following observations.

- The IO power is a significant portion of the combined power (DRAM+IO): 59% for the DDR3-based (BOOM-N2-D-800) configuration and 54% for the LPDDR2-based configuration (BOOM-N4-L-400). When using a serial bus from the buffer to the host, the IO power for BOOM-N4-L-400 reduces to 27% of the total power.
- The IO Idle power is a very significant contributor. The BOOM-N4-L-400 design reduces the IO Idle power by using LPDDR2 unterminated signaling, but since the BOOM configuration still relies on a DDR3 type bus from the buffer to the host as shown in Figure 2.13, the IO Idle power for the whole channel is still significant.
- Once the DRAM core becomes efficient, IO becomes a major contributor to the total power. Replacing DDR3 memories with LPDDR2 alone is not as efficient as further reducing the IO Idle power using a serial bus instead of a DDR3 style bus to the host. The BOOM-N4-L-400 design with a serial host provides a 3.4X energy savings (DRAM+IO) over the BOOM-N2-D-800 design. While Table

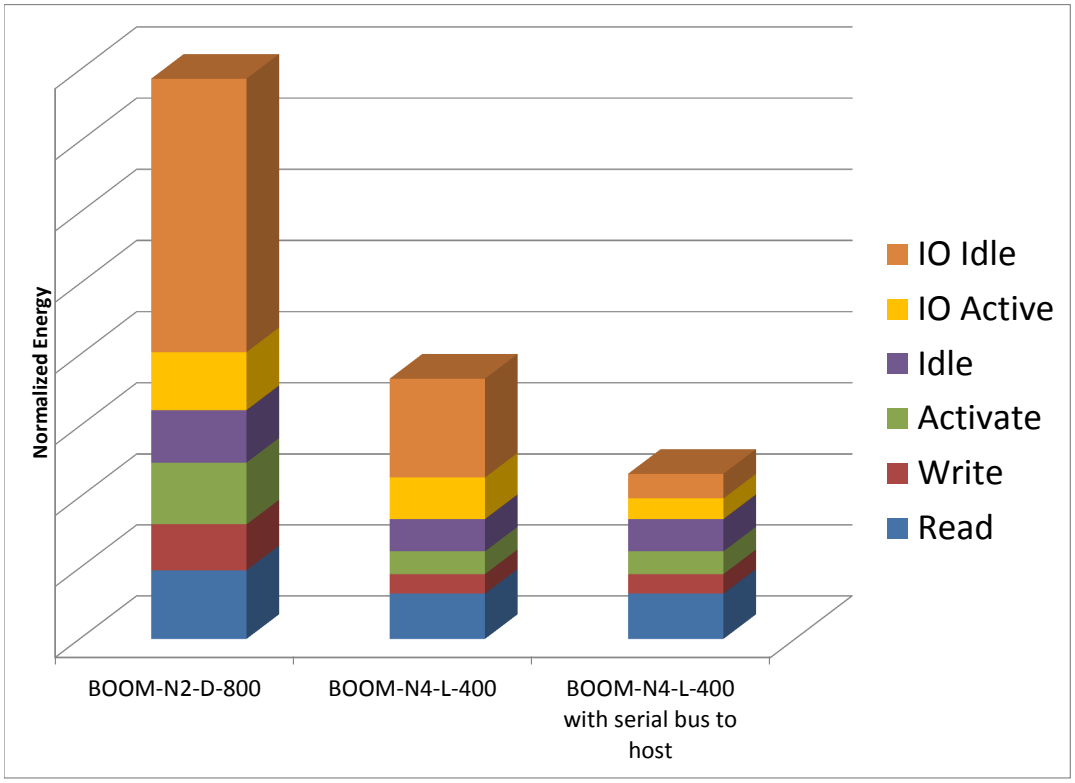


Figure 2.14: Normalized system (DRAM+IO) energy for BOOM configurations.

2.8 only compares the IO Active power, Figure 2.13 also accounts for IO Idle power and projects total energy based on active and idle times. While the serial bus only provides a 2.85X savings in IO Active power, it provides an 11X savings in IO Idle power when compared to the BOOM-N2-D-800 design.

- The number of power-down cycles is around 15% of the total cycles. More aggressive power-down will help reduce the IO Idle power. Supply scaling is also an option at lower frequencies in the case of BOOM-N4-L-400.

This case study highlights CACTI-IO's ability to provide IO power numbers to a system simulator, which can then provide valuable insight into total system power. Only combining the IO and DRAM power brings out the right tradeoffs needed to further improve efficiency. The study also highlights how CACTI-IO can be used to optimize a buffer-based topology such as BOOM, where IO choices including bus frequency and width can make a 2.85X difference in IO Active power and nearly an 11X difference in IO Idle power. Further, the need for aggressive power down depends on the off-chip configuration as well,

and IO Idle power is a key factor in determining how to address the power-down mode.

2.5.5 Optimizing Fanout for the Data Bus

We now illustrate how one can calculate the optimal number of physical ranks in a BOOM configuration to minimize IO power for a fixed capacity and bandwidth (BW). The number of physical ranks represents the fanout on the data bus. For this example, we assume that the memory density per DRAM die is fixed.

If N_R is the number of ranks, W_B the bus width, W_M the memory data-width, and f the data rate, then [59]:

$$N_R \cdot (W_B/W_M) = Capacity \tag{2.44}$$

$$W_B \cdot 2f = BW \tag{2.45}$$

Figure 2.15 shows the IO power as we vary the number of ranks to meet a capacity of 64 DRAMs and a bandwidth of 12.8 GB/s for an LPDDR2 bus. The IO power varies for different bus frequencies f, as the width of the bus and the memory data-widths vary to meet the conditions in Equations (2.44) - (2.45). The memory data-width is chosen to be x4, x8, x16 or x32 for the LPDDRx memories. The number of ranks is 1, 2, 4 or 8. The bus width is x64, x128, x256 or x512, and the bus frequency is 800 MHz, 400 MHz, 200 MHz or 100 MHz.

As can be seen from Figure 2.15, the wider and slower LPDDR2 bus provides the lowest power. A 512-wide bus using x8 memories in a single-rank configuration running at 100 MHz consumes the lowest power at 1.92 W, while a 64-wide bus using x8 memories in an eight-rank configuration running at 800 MHz consumes the highest power at 3.94 W. Also to be noted are the diminishing returns of scaling down to a lower speed once the bus is scaled to 200 MHz, owing to high-impedance terminations. This frequency at which termination is no longer needed depends on the interconnect length and the loading, which change based on the topology and technology as determined by the jitter DOE analysis.

One of the downsides to having a wider and slower bus is the cost of area on the die, package, and board. CACTI-IO predicts the impact on on-die area as we scale frequency and bus width to keep

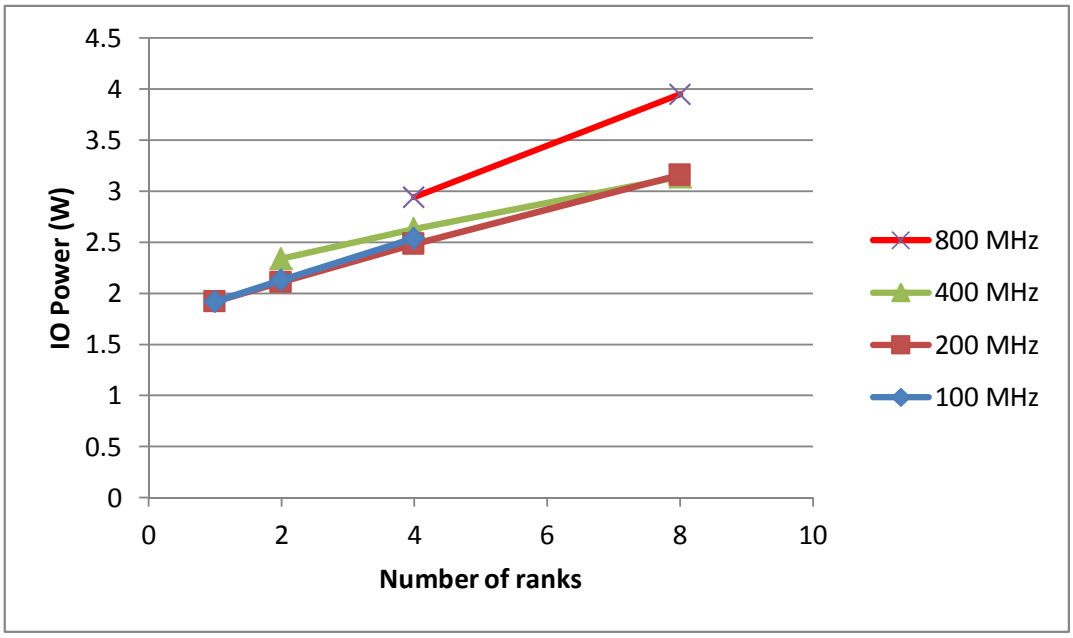


Figure 2.15: IO power vs. number of ranks for BOOM-LPDDR2.

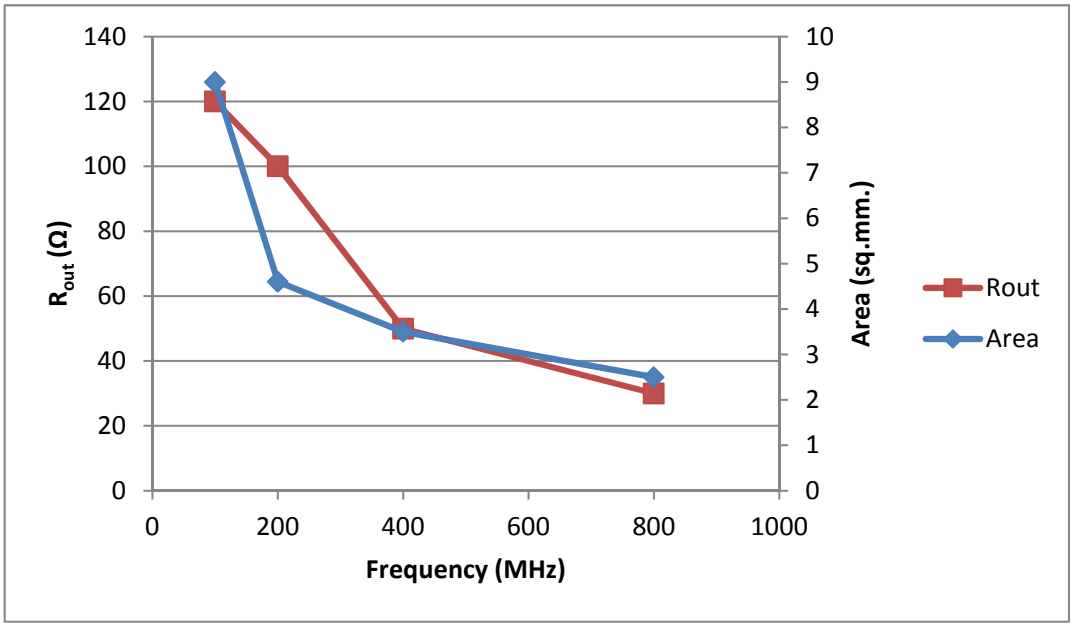


Figure 2.16: Area vs. frequency for a constant-bandwidth BOOM-LPDDR2.

the bandwidth constant. Shown in Figure 2.16 is the IO area vs. frequency for low fanouts (1 or 2 ranks) in 28nm technology, such that total bandwidth is kept constant. Also shown is the R_{out} that is used in Equation (2.43) to calculate the area. Wider buses result in a net increase in area even though they operate at lower frequencies. In a buffer chip this may be acceptable as there is less premium on area than on a

CPU or DRAM die. Since there is almost a 2X increase in area going from the 200 MHz to the 100 MHz solution, while there is hardly any difference in power, it may be prudent to choose the 200 MHz solution. The optimal solution would then be $N_R = 1$, $W_B = 256$, $W_M = 4$, and $f = 200$MHz. This example highlights CACTI-IO's ability to optimize the number of ranks based on IO power and any user-provided IO area, thus helping to optimize the IO configuration for a buffer-based design.

2.5.6 Optimizing Fanout for the Address Bus

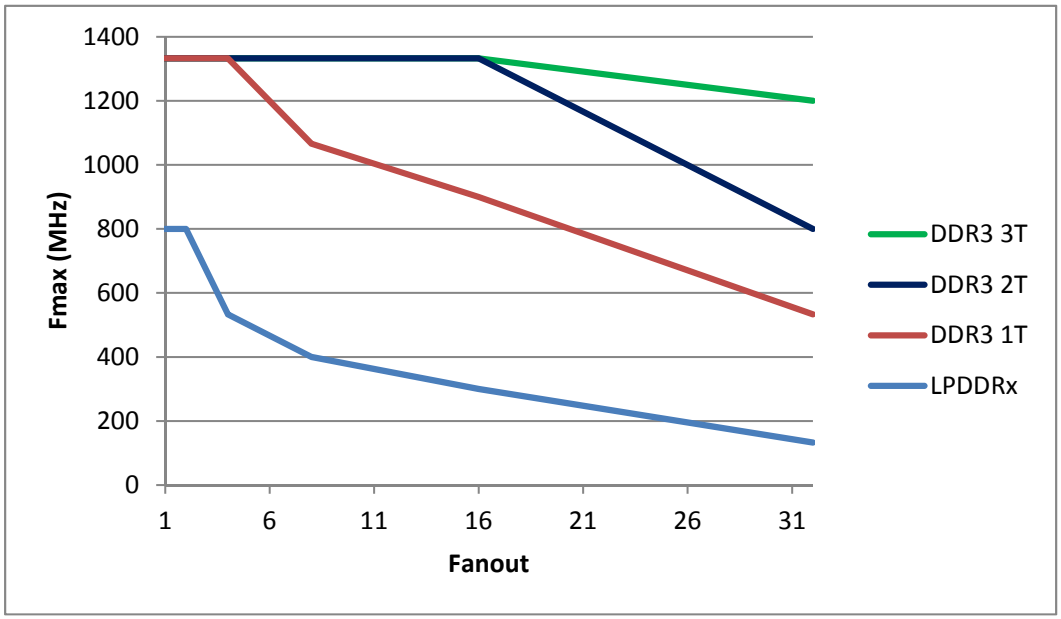


Figure 2.17: Fanout vs. F_{max} for a typical DDR3 CA bus.

As we increase capacity, the address bus incurs a penalty as all memories on the channel share a common address bus. The LPDDR2 and LPDDR3 standards [207] offer address buses at DDR speeds, with no option for 2T (2 clock cycle) timing [213]. This idiosyncrasy in the DRAM specification is not easily exposed to architects, but CACTI-IO allows for verified configurations to be systematically provided to architects.

To calculate the maximum achievable speed for a fly-by topology as shown in Figure 2.4, we need to define the sensitivity of the jitter on the CA (command-address) bus to the fanout of the bus as shown in Equation (2.26). Figure 2.17 shows the maximum achievable clock frequency on the CA bus for DDR3 and LPDDR2/3 as a function of the fanout for a representative channel. For DDR3, the 2T and 3T timing

options allow for relaxed timing on the CA bus [177].

Given the limitation for the LPDDR2 address fanout owing to the DDR speed requirement, multiple address buses may be needed to achieve higher capacities. For instance, based on the example in Figure 2.17, with a fanout of 16 we would need two LPDDR2 CA buses to support 400 MHz, while a single CA bus on DDR3 could support 1066 MHz with 2T timing.

With a buffer-based design, it is possible to have multiple address buses for a given channel between the buffer chip and the DRAMs. This would provide a means to limit the fanout on the address bus. Architects can optimize the design for a given address speed with optimal latency and burst requirements, including sub-ranking [157]. Understanding the limitations of the address bus allows architects to plan to overcome or minimize its impact on system performance.

2.5.7 PCRAM

Phase-Change Memory (PCRAM) is a type of non-volatile RAM [113]. The READ and WRITE latencies for a PCRAM are very different depending on whether the page is open or not. The READ or WRITE miss latency, when the page is closed, is an order of magnitude bigger than if it were a hit, with the page open. In addition, the WRITE miss latency is significantly larger than the READ miss latency.

In our case study, we evaluate the power and performance tradeoff of the IO bus to the PCRAM. We compare two configurations. The first configuration has a 64-wide bidirectional bus operating at 400 MHz DDR. In the second configuration the 64-wide bus is broken into two 32-wide unidirectional buses, one dedicated for READs, operating at 800 MHz, and the other for WRITE, operating at 200 MHz. This allows for the READ bandwidth to be maintained, while the WRITE bandwidth is much smaller owing to the significantly larger WRITE miss latency. The idea behind splitting the bus into a READ and a WRITE bus is to see if enabling READs independently can allow for the WRITE bus to be optimized for low power with higher latencies. The READ bus at 800 MHz needs terminations, while the WRITE bus at 200 MHz can be unterminated. Figure 2.18 shows a comparison of the power for the two configurations. The comparison assumes x8 devices and four ranks. For a 5% performance penalty, the split-bus configuration provides 10-40% savings in power depending on how the IO Idle power is managed.

There are various solutions to address IO Idle power reduction, and CACTI-IO can help the user

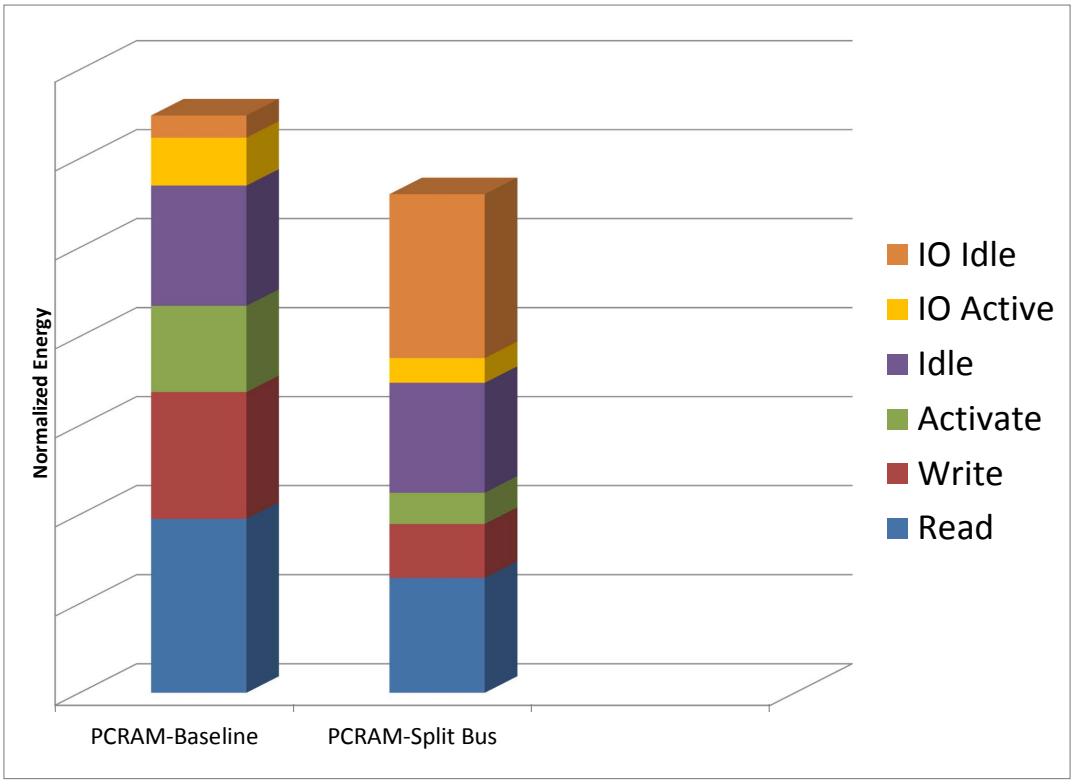


Figure 2.18: Normalized system (DRAM+IO) energy for PCRAM configurations.

evaluate these options.

- A suitable termination option to VDD (like DDR4 [178]) or VSS, rather than the DDR3-type mid-rail termination, could significantly help save IO Idle power as described in Section 2.2.1.
- Further, by scaling frequency appropriately, significant further reductions of power are possible. The wakeup time model in CACTI-IO described in Section 2.2.1 can help assess the latency impact of any such frequency scaling.
- Dynamic ODT [186] options could help optimize tradeoffs between idle power and signal integrity.

This example highlights the use of CACTI-IO to study not only an emerging memory technology, but the optimal IO and bus configuration as well. It helped identify the IO Idle power as significant for the split-bus configuration, which could be optimized by design choices described above. Having dedicated READ and WRITE buses (instead of a conventional bidirectional DDR bus) will require changes to the bank organization and interbank interconnects, which in turn will impact READ and WRITE latencies. For example, the off-chip write bus can be accommodated in two different ways. First, we can buffer writes internally in a PCRAM die using a small write buffer, and use a single shared inter-bank bus.

Alternatively, we can have a dedicated bus for read and write as shown in Figure 2.19. Both these scenarios can be modeled in CACTI. As these design choices are specific to a given architecture (in this case a dedicated read/write bus), the architect has to manually modify the internal bus specification to simulate an architecture like this.

Table 2.9: PCRAM IO Area.

Bus Config	**No. of IO**	**IO Area (sq. mm.)**
Bidirectional	64	1.75
Split	32+32	1.8

Table 2.9 compares the IO Area for the split-bus and bidirectional bus configurations. The IO area is nearly identical. This is because the split-bus benefits from a slower WRITE interface, while the READ interface is faster.

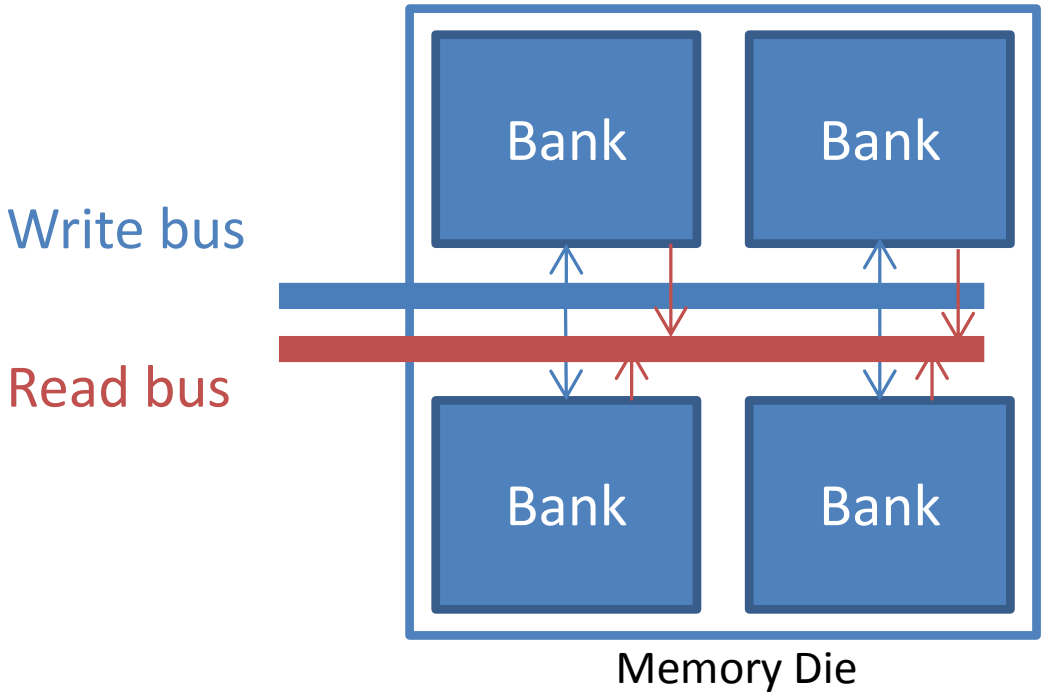


Figure 2.19: Split-bus PCRAM configuration.

2.6 Summary

We have presented CACTI-IO, a version of CACTI that models the off-chip memory interface for server and mobile configurations. Its models include off-chip power and IO area, as well as voltage and timing margins that help define the maximum achievable bandwidth. Our framework permits quick design space exploration with the rest of the memory subsystem and provides a systematic way for architects to explore the off-chip design space. It also exposes DRAM signaling standards and their idiosyncrasies to architects, while still providing an easily-extensible framework for customization of off-chip topologies and technologies.

Using CACTI-IO, we have also illustrated the tradeoffs between capacity, bandwidth, area and power of the memory interface through four industry-driven case studies. These clearly show the ability of CACTI-IO to calculate IO power for various configurations, including DIMMs, 3D interconnect, buffer-based designs such as BoB and BOOM, and new memory technologies like PCRAM. CACTI-IO helps determine the lowest-power off-chip configuration (bus width, memory data width, number of physical ranks, address bus fanout, minimum supply voltage, and bus frequency) for given capacity and bandwidth requirements.

Furthermore, we have highlighted the capability of CACTI-IO to combine IO and DRAM power, which shows the significant contribution of IO power to the total (DRAM+IO) memory power (up to 59% in some cases). We have observed the relative importance of IO Idle power by using CACTI-IO and a system simulator together to calculate system energy in various modes (Read, Write, Activate, Precharge, Idle). A combination of a wider and slower bus to the DRAM and a faster serial bus to the CPU provides the lowest IO Idle power.

CACTI-IO is publicly available online as part of the latest CACTI release [171]. We expect that the new capabilities provided by this tool will enable improved understanding of memory interface issues, allowing architects to evaluate customized off-chip buffer-based designs as well as the impact of new interconnect technologies on system power and performance.

2.7 Acknowledgments

Chapter 2 contains a reprint of N. P. Jouppi, A. B. Kahng, N. Muralimanohar and V. Srinivas, "CACTI-IO: CACTI With Off-Chip Power-Area-Timing Models", *IEEE Transactions on Very Large Scale Integration Systems* 23(7) (2015). The dissertation author is a main contributor to, and a primary author of this paper. I would like to thank my coauthors Norm P. Jouppi, Andrew B. Kahng and Naveen Muralimanohar.

Chapter 3

Early Exploration of Mobile Memory Interconnect

Included here is an early work focused on mobile memory interfaces that connect the SoC to the SDRAM. Despite some of this work predating CACTI-IO described in Chapter 2, we include it here in the present chapter for three reasons: (i) it provides a good summary of the unique mobile memory interconnect design space; (ii) it provides a glimpse into some of the early methods and approaches undertaken in this thesis, that would serve as a useful precursor to the more advanced methods in subsequent chapters; and (iii) it provides perspective not only on how the design space has changed since this work was undertaken, but also on how the system-level methods and models used are still relevant today albeit with the need for some updates. Mobile memory interconnect is quite unique, due to its phone form factor, low-power focus, and different capacity, bandwidth and cost requirements. The capacity requirements for mobile memory are much smaller than those seen in the server space, but are quickly catching up with the client and laptop markets. The bandwidth requirements for mobile memory interconnect have rapidly increased over the years, although they are still quite small compared to those required by server designs. The mobile memory interconnect is extremely cost- and power-sensitive, and there continues to be a strong impetus to optimize both of these attributes.

Some perspective on the applicability of this early work to the mobile memory interconnects of

today is as follows. We now have LPDDR5 memories in the market, but the decisions presented in Figure 3.1 are still relevant today although the thresholds have changed with respect to capacity. This is true because the general trajectory of LPDDR, DDR, wide-memory and custom memory continue to address specific market requirements driven by system-level metrics of capacity, bandwidth, power and cost that have not changed that much in relative priority. Additionally, the die density and I/O rate trends shown in Figures 3.2 and 3.3 have largely held from the 2009 prediction, with some downward (for density) and upward (for data rate) adjustment trend described below. Today, we have LPDDR5 DRAM dies with 16 Gb/die and 6 Gb/s data rate. The models used include closed-form analytical expressions as well as lookup tables. While some design and specification updates are needed for the closed-form expressions, they remain largely correct and relevant today. The lookup tables need regular updates based on changes to analog components and process technology. As described in Chapter 2, such models require porting with technology. While some scaling factors can be accounted for, some others need circuit simulations to be redone, and underlying design(s) of experiments revised and/or redone to fit a new model or lookup table.

3.1 Mobile System Considerations for SDRAM Interface Trends

SDRAM (Synchronous Dynamic Random Access Memory) is the mainstay main-memory solution for processors today. Many offerings exist in the market [20, 177, 213, 189, 190], spanning multiple packaging and interconnect options. Most SDRAMs available are DDR (Double Data Rate), such as DDR3 and LPDDR2 (Low-Power DDR2). While LPDDR2 and DDR3 memories exist in the market today, others are being discussed in JEDEC [207] or elsewhere as standards for the future, and will probably be available in the market soon based on the traction they receive. [1]

POP (Package-on-Package) and MCP (Multi-Chip Package) offer good point-to-point interconnection options, while DIMMs (Dual Inline Memory Module) are required for discrete parts. 3D stacking offers a new way to stack the die through a TSV (Through Silicon Via [32]) and promises to enable high

[1] JEDEC [35] memory standards and nascent standards being discussed in JEDEC <name, IO voltage, pin-width, data rate (Gbps), interconnect, signaling>: (i) <LPDDR2, 1.2V, x16/x32, 1.066, POP/MCP, single-ended LVCMOS output / SSTL input>, (ii) <DDR3, 1.5V, x4/x8/x16, 2.133, DIMM, SSTL>, (iii) <LPDDR3. 1.2V, x16/x32, 1.6, POP/MCP, being discussed>, (iv) <DDR4, 1.2V, x4/x8/x16, 3.2>, (v) <Mobile-XDR, 1.2V, x16/x32, 4.3, MCP, differential>, (vi) <Wide IO, 1.2V, x512, 0.2-0.3, 3D-stack, LVCMOS>, (vii) <Serial Memory, 1.2V, x16/x32, 4-8, MCP, SPMT or MIPI M-PHY>

bandwidth memory access through wide interfaces (up to x512 proposed [189]).

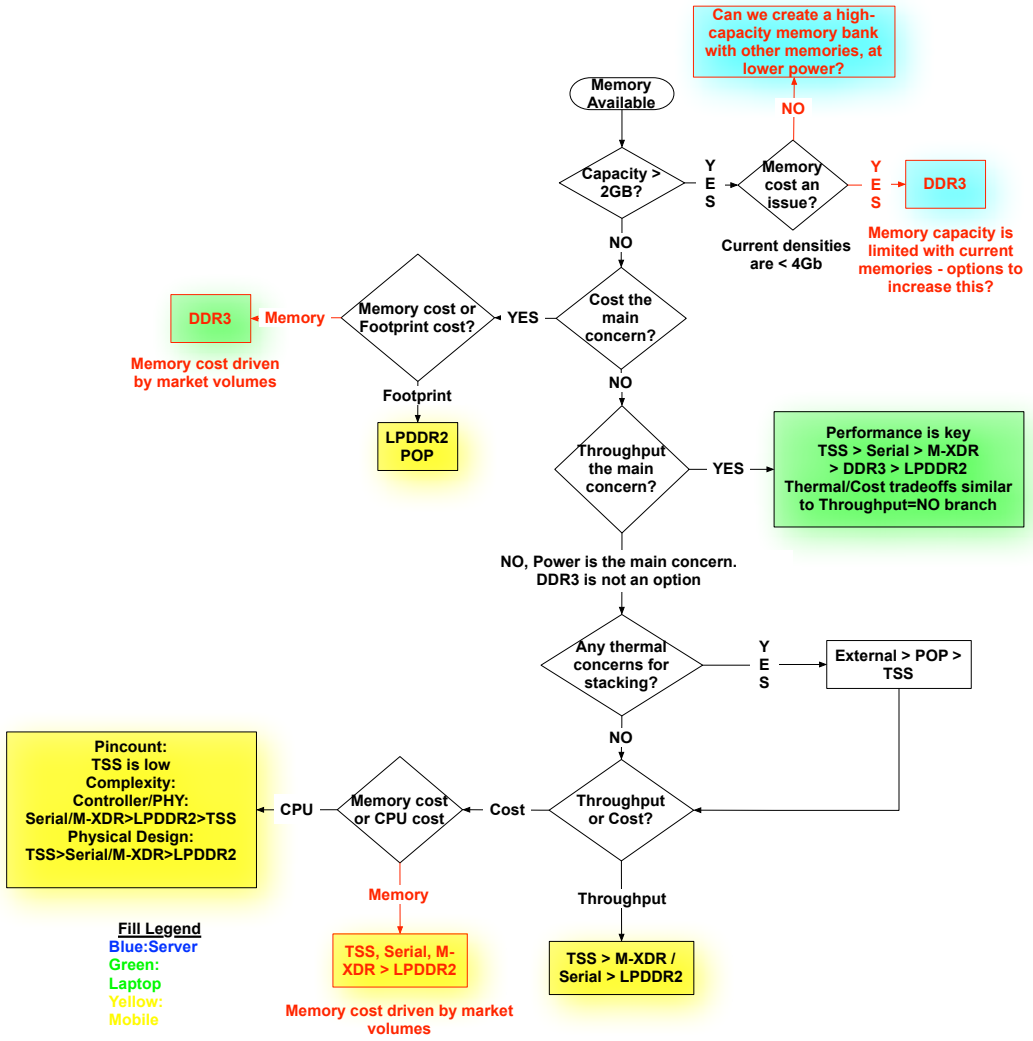


Figure 3.1: The decision tree with forks based on SDRAM capabilities and regions based on system requirements.

Currently, mobile controllers have been driven to LPDDR2 solutions by their need for lower power and point-to-point unterminated interconnect. As requirements on throughput and capacity scale upwards for mobile platforms, alternate solutions are needed since data rates for unterminated signaling do not scale easily beyond 1 Gbps [151]. Section 3.2 outlines some of the trends in the DRAM industry, including the competing standards available for future mobile memory solutions [189, 190, 225] and ITRS projections on DRAM densities and data rates [205, 206, 168]. It is shown that DRAM densities are expected to double

every three years, while DRAM data rates are expected to double every four years. Section 3.3 highlights the trends for mobile system requirements on the memory [52], including capacity and throughput. It is shown that capacity requirements are expected to scale 3-4x every three years, while throughputs are expected to double every three years [232].

Most of these competing DRAM options differ in their interface implementations. A memory-subsystem framework that includes the SDRAM interface metrics would be useful to compare them and help identify the best memory solution for a particular system. As the first step towards such a framework, we provide a *memory interface calculator*, described in Section 3.4, for the existing memory interface options available for a mobile system. We also describe how the calculator could be extended for future memory interface options based on the basic categories it supports. For each interface option, the calculator provides a framework to predict interface power and latency for a given throughput and capacity. Integration of this interface calculator into a memory-subsystem calculator such as CACTI [171] is described in Chapter 2.

The key metrics for the main memory are: availability, capacity, cost, throughput, power, latency and thermal considerations. These we call the *primary bounds* on the main memory design space. When choosing between SDRAM options, a priority order of the primary bounds is helpful to identify the best memory solution easily. One such priority order [36] is used to illustrate our framework. The most important bound is availability. Often availability and cost of the memory are tied to volume demand. During early adoption, enough traction is needed for demand to drive down cost and speed up availability. This normally happens through the process of standardization, where early adopters drive the co-development of the standard. The traction of a standard or solution depends on how it performs with respect to the primary bounds and intangible market forces that are beyond the scope of this chapter. What is worthwhile to note, though, is that there is a large transition period from one standard to the next. Often backward compatibility with respect to the interface and interconnect plays an important role in the success of a smooth transition to the new standard, as OEMs (Original Equipment Manufacturer) need time to slowly phase out the old memory type for the new one. This requires the same mobile controller IC to support both the old and new memory types over the transition period.

Shown in Figure 3.1 is an example *decision tree* with priority ordering of primary bounds to help

identify potential memory solutions. The tree is divided into three segments based on the requirements on these bounds – server (blue), laptop (green) and mobile (yellow). The priority of the primary bounds shown in Figure 3.1 is an example order of decisions by which a memory solution may be identified. We use this order in the chapter to illustrate how the calculator aids in determining the choice of memory interface. Other priority orders may be suited to different system requirements, and could serve just as well to identify a memory solution.

Each endpoint in the tree identifies one or more particular memory options based on the requirements, and taking into account market bottlenecks (availability, cost) or fundamental technical bottlenecks. The forks identified in red are those currently identified as market-driven. Currently, a high-capacity requirement (>2GB) would make DDR3 memories the only viable option. However, if throughput is not a key primary bound for a particular system, then other options may become viable based on MCP solutions (this is the focus of recent investigations elsewhere [216]). Such alternatives, if capable of increasing demand for a memory type across market segments, could further help reduce cost and improve availability. It is useful to note that while the memory options of the decision tree change over time due to the capabilities of the SDRAM options (both market-driven and fundamental) described in Section 3.2, the location of the market segments on the decision tree change based on the system requirements described in Section 3.3. The decision tree shown reflects the capabilities and requirements in today's market.

Once availability and cost have been established, the remaining driving questions (bounds) are capacity, throughput, power, latency and thermal considerations. The rest of the chapter focuses on the first four of these primary bounds and ignores availability and cost issues.

Section 3.4 describes how the interface considerations for each of the primary bounds impact the circuit and physical design of the mobile controller's IO logic and interconnect. This is shown through a set of equations that contain the requirements and capabilities that address each primary bound. The equations listed are part of the memory interface calculator.

Section 3.5 summarizes the chapter, while providing the motivation for integration into an optimization framework, especially:

(1) The need to integrate power and latency for IO and interconnect configurations of various memory types into a higher-level abstraction such as CACTI [171]. This will help provide memory

subsystem metrics of latency and energy for these different interconnect options [241, 142].

(2) Predict bottlenecks or gaps in the memory offerings that will provide clues to novel IO and interconnect solutions [108, 216] or help motivate a new memory standard for the future.

3.2 SDRAM Capability Trends

ITRS projections for DRAM density and DRAM data rate [205, 206, 168] are shown in Figures 3.2 and 3.3 respectively. The projections are shown from those made in 1999 till those made in 2009, each year being revised based on new ITRS data available. The data rate projections shown are for the DDR2/DDR3/DDR4 family.

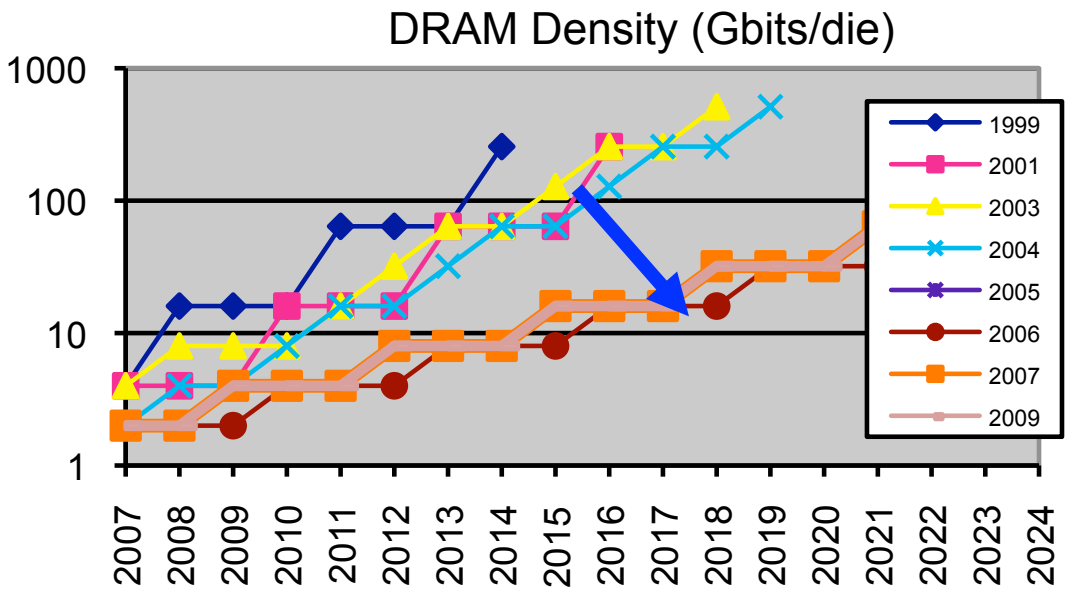


Figure 3.2: DRAM density projections [168].

Based on the latest projections (2009), it can be observed that DRAM densities are expected to double every three years, while DRAM data rates are expected to double every four years. Also interesting to note, are the blue arrows in Figures 3.2 and 3.3, which indicate that over the years, projections for DRAM densities have been revised downwards while DRAM data rates have been revised upwards.

Some of the standards being discussed in JEDEC today include DDR4, DDR3-ULV, LPDDR3,

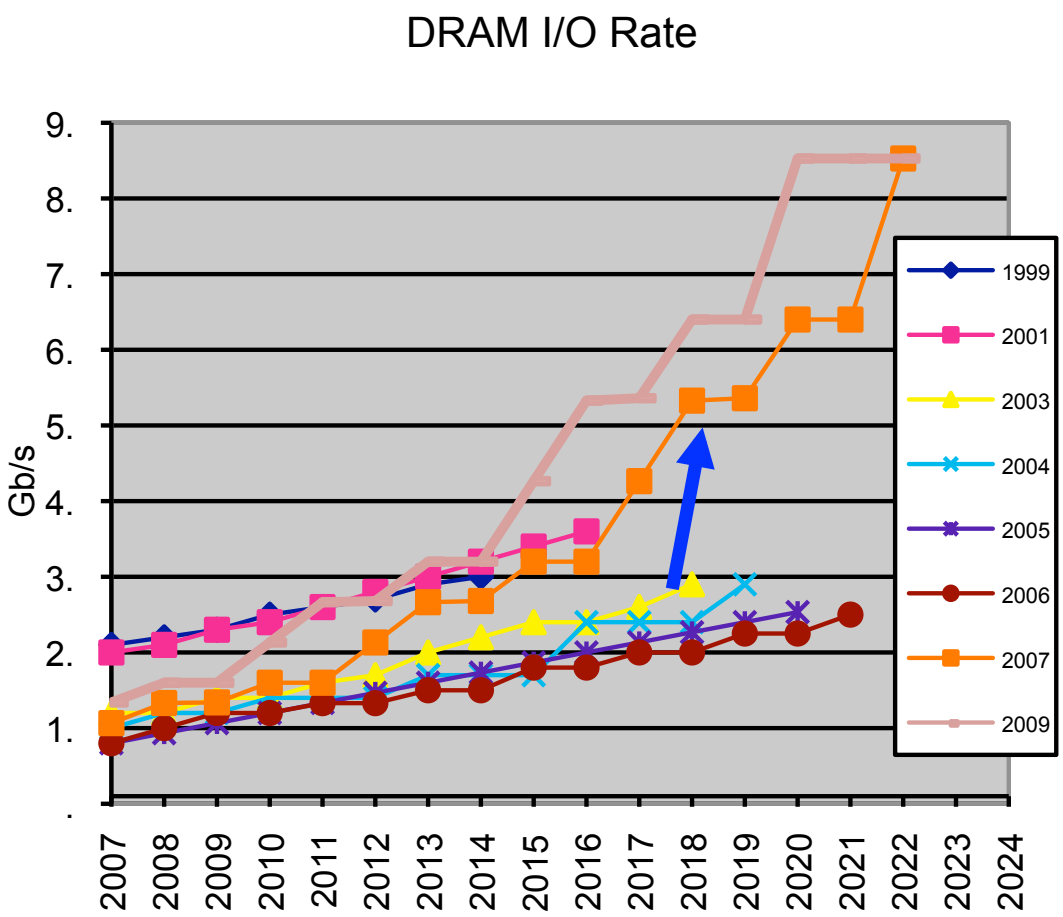


Figure 3.3: DRAM data rate projections [168].

Wide IO, Serial-MIPI and Serial-SPMT [189, 190, 119]. Mobile-XDR and XDR2 are signaling standards driven by Rambus [225]. Recent nascent standards also highlight the trend to either increase pin-width (using TSVs) or use differential signaling for higher throughputs on existing pin-limited interconnects. This is because data rates using single-ended signaling are hard to scale with good power efficiency and signal-integrity [108, 75, 49].

3.3 Mobile Requirements Trends

Mobile system requirements for peak throughput and capacity are shown in Figure 3.4 [232] and Table 3.1 respectively. Figure 3.4 shows throughput requirements of products based on volume production

timelines. Capacity requirements are expected to scale 3-4x every three years, while throughputs required are expected to double every three years. Other more aggressive throughput requirements for mobile application processors [19] suggest close to a doubling every year.

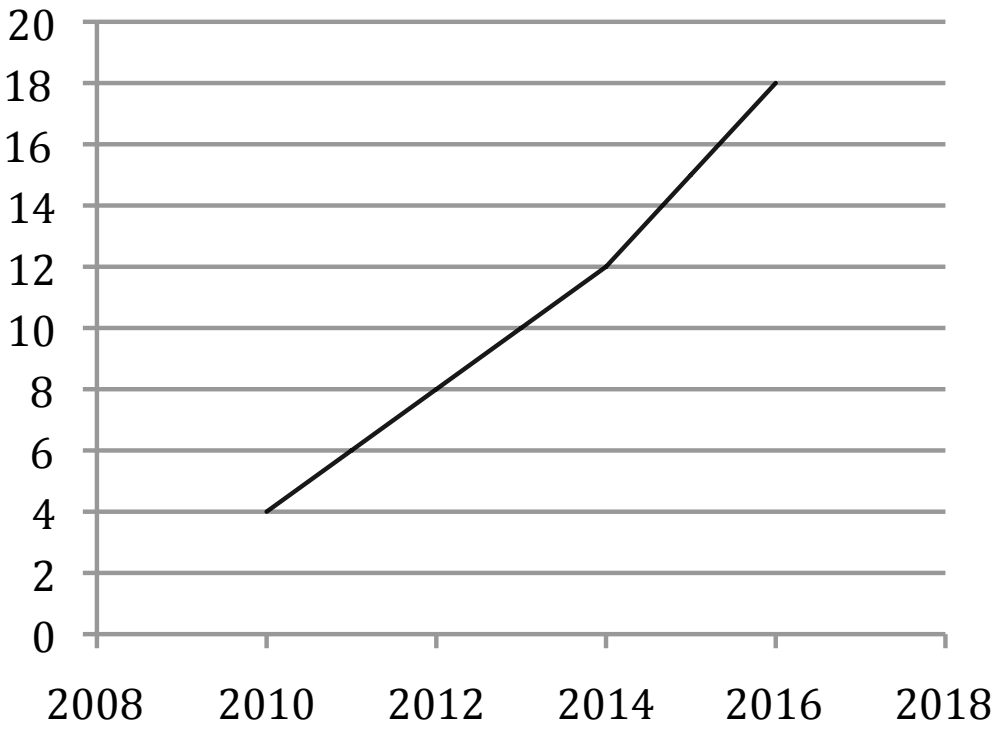


Figure 3.4: Peak throughput projections for mobile platforms (in GB/s) [232].

Table 3.1: Capacity projections for mobile and PC markets (source: IDC Market Survey).

Market	2010	2011	2012	2013	2014
Desktop	3.0	4.2	5.6	7.4	10.2
Laptop	2.0	3.3	4.6	6.3	8.0
Mobile	0.3	0.5	0.8	1.0	1.3

3.4 Interface Considerations

Sections 3.4.1 through 3.4.4 address the SDRAM interface considerations of a mobile system for capacity, throughput, power and latency as they appear in the decision tree. Each of these four

sections highlights the inferences it passes onto subsequent sections. The considerations are highlighted in Equations (3.1) - (3.31), which show the requirements and capabilities surrounding the circuit design stage of the mobile processor's design hierarchy. The memory interface calculator is based on these equations.

3.4.1 Capacity

Capacity is the total memory available, typically in GB. It is equal to the density of each memory die multiplied by the number of dies. The dies can be organized into multiple channels, each channel with multiple ranks comprised of x4, x8, x16 or x32 memories to support a required bus width of the mobile controller.

The capacity determines the number of memory dies needed for given density availability. This in turn determines the interconnect lengths and number of multi-drop loads on the bus. The interconnect structure is a key input for the signal-integrity analysis that determines data rate capability. For mobile systems with lower capacities (<2GB), this is typically one of three options: a stacked die, POP or a short trace to an MCP off-chip. Trace lengths within an inch or two are common for the SDRAM interface, keeping it unterminated. However, higher capacity requirements, especially for tablet products, could see the need to support DDR3 DIMMs or multi-drop structures.

Equations governing capacity are as follows.

$$Capacity = N_{channels} \cdot N_{ranks} \cdot \left(\frac{N_{bus\_width}}{N_{memory\_width}} \right) \cdot Density \tag{3.1}$$

$$N_{memories} = \left(\frac{Capacity}{Density} \right) \tag{3.2}$$

$$N_{memories/channel} = \left(\frac{N_{memories}}{N_{channels}} \right) \tag{3.3}$$

$$N_{ranks} = N_{memories/channel} \cdot \left(\frac{N_{memory\_width}}{N_{bus\_width}} \right) \tag{3.4}$$

$$C_{interconnect} = C_{RDL} + C_{TSV} + C_{PKG} + C_{PCB} \tag{3.5}$$

$$C_{ext\_DQ} = C_{interconnect} + C_{IO\_DRAM} \cdot N_{ranks} \tag{3.6}$$

$$C_{ext\_CA} = C_{interconnect} + C_{I\_DRAM} \cdot N_{memories/channel} \tag{3.7}$$

$$C_{Total} = C_{ext} + C_{self} \tag{3.8}$$

$$Delay_{interconnect} = f(N_{memories}) \tag{3.9}$$

3.4.2 Throughput

Throughput in GB/s is the total number of bytes of data transferred over the memory interface. The data rate is a multiple of the clock frequency; the multiple is 1 for SDR, 2 for DDR, and higher than 2 for multi-data rate.

Equations governing throughput are as follows.

$$Throughput = \left(\frac{2 \cdot N_{bus\_width}}{T_{ck}} \right) \tag{3.10}$$

DQ-DQS WRITE:

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_hold} - T_{skew\_hold} > T_{DH} \tag{3.11}$$

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_setup} - T_{skew\_setup} > T_{DS} \tag{3.12}$$

CA-CLK (DDR for LPDDR2/3):

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_hold} - T_{skew\_hold} > T_{IH} \tag{3.13}$$

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_setup} - T_{skew\_setup} > T_{IS} \tag{3.14}$$

For DDR3 the CA interface is SDR, and the above timing is relaxed to a half-cycle as opposed to a quarter-cycle. DQS-CLK:

$$T_{jitter\_hold} + T_{skew\_hold} < \left(\frac{T_{ck}}{2}\right) - T_{DSH} \tag{3.15}$$

$$T_{jitter\_setup} + T_{skew\_setup} < \left(\frac{T_{ck}}{2}\right) - T_{DSS} \tag{3.16}$$

DQ-DQS READ:

$$T_{QSH/QSL} - T_{error} - T_{jitter\_hold} - T_{skew\_hold} - T_{QHS} > T_{SoC\_hold} \tag{3.17}$$

$$\left(\frac{T_{ck}}{4}\right) - T_{error} - T_{jitter\_setup} - T_{skew\_setup} - T_{DQSQ} > T_{SoC\_setup} \tag{3.18}$$

Voltage Noise:

$$V_{noise}(crosstalk, SSN, reflection - ringbacks) < V_{NM} \tag{3.19}$$

$$V_{overshoot} < V_{MAX}(NBTI, HCI) \tag{3.20}$$

The data rate determined based on the throughput requirement forms a key input for the interface

design. Wider interfaces, such as Wide IO, have low data rates for equivalent throughputs, making timing closure easier. On the other hand, serialization increases the data rate and requires higher power to meet timing (due to power in the termination and the analog blocks that help recover and retime the signals).

The key consideration for throughput is the link timing budget based on the data rate. This is based on the implementation of the PHY and IO and the signal-integrity analysis of the link. Each of these factors reduces the data-valid window for the READ, WRITE and CA operations and limits the maximum achievable frequency.

For mobile systems, throughput considerations also include meeting timing with minimum power, and the ability to reduce power at low throughputs. This constrains the termination scheme (unterminated being preferred at lower throughputs) and the choice of retiming scheme. As mobile memories are designed with similar considerations, the power they consume is lower at the expense of timing. For example, LPDDR2 memories do not have a DLL, so although their power is lower than DDR3 memories, they have worse timing parameters [177, 213, 222] and thus the link timing budget is harder to close (LPDDR2 DRAM accounts for more than 33% of the Unit Interval (UI)). This puts considerable burden on the mobile controller to close timing on the SDRAM interface. Traditional link design for the SDRAM interface divides the UI into three equal (33%) buckets for the controller, channel and DRAM. Shorter interconnects (owing to smaller capacity requirements highlighted in Section 3.4.1) help keep the channel's contribution to the link timing budget to less than 33% at lower data rates. But as data rates get higher, the crosstalk and ISI increase, and lack of terminations for LPDDR2 interfaces impose greater challenges on the channel jitter. Mobile-XDR, Wide IO and serial memories offer different ways to solve these challenges while keeping the power efficiency low. While Wide IO proposes to use lower data rates to ease timing without increasing pin-count, Mobile-XDR and serial memories propose to use differential signaling to improve signal quality and reduce signal swing [108, 76, 102]. Figure 3.5 shows throughput projections for the various standards.

The support of multiple memory standards, especially during transitions from one standard to the next, poses challenges for the mobile processor's IO as the IO will most likely need to support multiple voltages with a single transistor device.

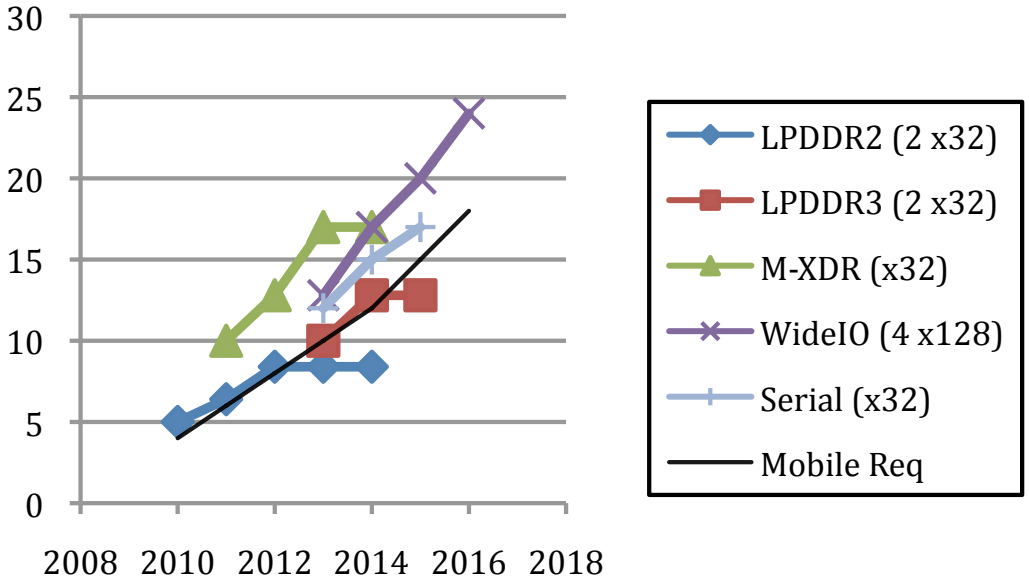


Figure 3.5: Throughput (GB/s) projections based on the memory interface calculator.

3.4.3 Power

The Micron DDR3 power calculator [5] provides a framework to calculate memory interface and core power for DDR3 configurations. Our memory interface calculator models power with a similar framework, but includes calculations for all of the mobile memory types of interest today – LPDDR2/3, Wide IO, Mobile-XDR and Serial Memory.

The interface power is mainly made up of the following sources:

1. Switching (dynamic) power: typically drawn when capacitors are charged, including self-loading.

2. Termination power: The static power drawn through the termination network. The termination power depends on the termination current, which in turn depends on whether the line is at a logic 0 or a logic 1. For DDR3, CA topology uses fly-by termination [177, 222]. If mobile systems use DDR3, sub-optimal termination values are used to save power. The Micron DDR3 power calculator [220] shows how the power would change for dual-rank configurations with different termination values.

3. Bias/static power: analog currents used to bias the IO circuits - transmitter as well as receiver,

adaptive circuits, DLLs and other timing circuits. The memory interface calculator provides a lookup for the power consumed for various implementations in literature [108, 75, 47] that relate to reported jitter and skew performance of recovery and retiming blocks along with adaptive schemes for calibrating the IO for the channel and the recovery logic for the eye opening. The ability to reduce such power based on data rate is very important for maintaining good power-efficiency at lower bandwidths [108, 75, 76]. SPMT provides a novel way to run a parallel interface at lower data rates and a serial interface at higher data rates [119], which provides good power efficiency at both ends of the throughput range.

Shown below are the considerations for power. Table 3.2 lists the power efficiencies of the various interfaces included in the memory interface calculator. Mobile processors use a variety of schemes to reduce power in low-throughput modes that include shutting off terminations, using low-power retiming schemes at lower throughputs, and power-gating schemes that turn off blocks not critical at lower throughputs. To achieve these power savings, transitions between various power states are enabled.

$$P_{Total} = \sum_{mode_i} (D_{c_i} \cdot (P_{Total\_Interface_i} + P_{DRAM\_Core_i})) \tag{3.21}$$

Interface Power:

$$P_{Total\_Active\_Peak} = P_{dynamic} + P_{term} + P_{static/bias} \tag{3.22}$$

$$P_{Total\_Active\_Idle} = P_{term} + P_{static/bias} \tag{3.23}$$

$$P_{Sleep} = P_{leakage} \tag{3.24}$$

Dynamic Power:

$$P_{dynamic} = N_{pins} D_c \alpha C_{Total} V_{sw} V_{dd} f \tag{3.25}$$

Single-ended termination:

$$P_{term\_oh} = (V_{dd} - V_{TT})(V_{oh} - V_{TT})/R_{TT} \tag{3.26}$$

$$P_{term\_ol} = V_{TT}(V_{TT} - V_{ol})/R_{TT} \tag{3.27}$$

DDR3 DQ:

$$P_{DQ\_Term} = 0.25 \cdot V_{dd}^2 \cdot \left(\frac{1}{R_{TT}} + \frac{1}{R_{on} + R_{stub} + R_{TT}} \right) \tag{3.28}$$

DDR3 CA:

$$P_{CA\_Term} = 0.25 \cdot V_{dd}^2 \cdot \left(\frac{1}{R_{TT}} + \frac{1}{50 + R_{TT}} \right) \tag{3.29}$$

Differential Termination:

$$P_{diff\_term} = V_{dd} V_{sw} / R_{TT} \tag{3.30}$$

Table 3.2: Summary of parameters of existing memory interfaces from the calculator.

Bound	**LPDDR2**	**TSS-Wide IO**	**DDR3**	**Serial**	**Mobile-XDR**
Clock Speed (MHz)	300-533, DDR	200-333, SDR	400-800, DDR	4-8 GHz, Serial	400-533, Octal
Throughput (GB/s)	3-4.3	12-24	6-13	12-17	12-17
Peak IO Power Efficiency (mW/Gbps)	˜40	˜10	˜120	˜60	˜20
Peak Core Power Efficiency (mW/Gbps)	˜50	˜35	˜100	˜50	˜50
Total Peak Power Efficiency (mW/Gbps)	˜90	˜45	˜220	˜110	˜70
Active Idle IO Power (mW)	6-10	2-4	˜500-600	˜450	˜200
Active Idle Core Power (mW)	˜20	˜20	˜150	˜20	˜20
Capacity (GB) (Current trends)	0.5-1 for x32 dual rank	0.5-2 through multi-die stacking	2-8 for dual-rank DIMM	0.5-1 0.5-1 GB	0.5-1 for x32 dual rank
Latency from MC-DRAM-MC	˜50ns	˜40ns	˜45ns, but penalty if DLL is off (˜512 Tck)	˜65ns, PLL lock penalty if off	˜60ns, DLL penalty if off
Data width	x32	x512	x64	x16-x32	x32
Pin count	60	640 TSVs (No ext.)	130	60-100	60
Form-factor	POP/MCP	Stacked TSS	DIMM	MCP	MCP/POP

3.4.4 Latency

Latency from the memory controller to main memory is a portion of the cache miss penalty during a load instruction from the CPU. Most of the mobile memories have similar access times (T_{RCD}, T_{RL}, T_{DQSCK}) as they depend on the DRAM core that they share. The key latency differentiator is the implementation of the PHY and IO on the mobile processor. Latency is not currently included in the decision tree as the various memory options have latencies of the same order (50-70ns from controller to DRAM and back), but this range could well be a differentiator in performance.

The memory interface calculator calculates latency during a load cache miss (a memory READ operation) from the controller CA path back to the DQ READ path. This includes the pipeline stages in the controller and PHY for CA and READ, the combinational output and input delay in the IO, the passive delay in the interconnect, T_{RCD} and the read latency (T_{RL} and T_{DQSCK}).

Apart from the latency during the load miss, the latency of waking up from sleep to full activity is an important consideration for mobile systems, since frequent mode changes are needed to enable power savings, and wakeup times impact performance of the CPU. DLL/PLL lock times and wakeup times for any of the adaptive schemes involved will impact the latency of the IO during wakeup from sleep modes. DLLs on DDR3 DRAMs take over 500 clock cycles to lock [177, 222], whereas LPDDR2 memories do not have DLLs and hence can exit self-refresh mode without a latency penalty. Periodic calibration for adaptive schemes, such as impedance and delay calibration, require periodic stalls to update the calibrated values. The stall frequency for such updates, and the stall time needed for propagating the updated settings, are other latency considerations. Eq. (31) shows the calculation for latency.

$$Latency = N_{pipes} \cdot T_{ck} + T_{del\_int} + T_{del\_comb} + T_{RCD} + T_{RL} + T_{DQSCK} \tag{3.31}$$

3.4.5 Examples

To further illustrate the calculator and the decision tree, we provide two examples that go through the steps described in Sections 3.4.1 through 3.4.4.

Example 1:

A particular system requires 4GB of DRAM and expects a peak throughput of 10 GB/s.

The 4GB capacity requirement, using 4Gb DRAM density, translates to 8 DRAM dies ($N_{memories} = 8$). This rules out an MCP or stacked solution, and an external interconnect through a DIMM. Hence, DDR3 is the memory of choice within the decision tree. Eight x8 DRAMs connected to a x64 bus width provides the needed single-rank memory configuration. The loading on the DQ pin is a single DRAM along with the interconnect. The loading on the CA pin is eight DRAMs along with the interconnect.

A throughput of 10 GB/s on a x64 bus translates to a data rate of 1.25 Gbps, or a clock frequency of 625MHz for the DDR3 DRAM. Use of a 667 MHz DDR3 speed bin is required. The interface timing parameters of interest for this DRAM are listed in Table 3.3. Plugging into the timing equations of Section 3.4.2 shows that the jitter and skews required over a DIMM interconnect require the use of terminations, and a fly-by channel for the CA bus. Based on this, outputs of the power and latency calculations from the calculator are summarized in Table 3.3.

Example 2:

Another system requires 1GB of DRAM and expects a peak throughput of 4 GB/s.

The 1GB capacity requirement, using 4Gb DRAM density, translates to 2 DRAM dies ($N_{memories} = 2$). POP, MCP or stacked solutions are possible. Given the lower throughput requirement, LPDDR2 POP is the preferred choice based on the decision tree (to reduce power and footprint cost). A dual-rank x32 POP is the suitable memory configuration. The loading on the DQ pin is two DRAMs along with a short POP interconnect. The loading on the CA pin is also two DRAMs along with the short POP interconnect.

A throughput of 4 GB/s on a x32 bus translates to a data rate of 1 Gbps, or a clock frequency of 500MHz for the LPDDR2 DRAM. Use of a 533 MHz DDR3 speed bin is required. The interface timing parameters of interest for this DRAM are also listed in Table 3.3. Plugging into the timing equations of Section 3.4.2 shows that the jitter and skews required over a POP interconnect require some careful retiming as the LPDDR2 interface does not support terminations. Based on this, the power and latency calculations are as summarized in Table 3.3.

Table 3.3: Inputs and outputs of the calculator for two design examples.

Parameter	Example 1	Example 2
$Capacity$	4GB	1GB
$Throughput$	10 GB/s	4 GB/s
$N_{memories}$	8	2
$Configuration$	x64 DDR3	x32 LPDDR2
N_{ranks}	1	2
$Clockrate$	625 MHz	500 MHz
T_{DH}	240 ps	210 ps
T_{DS}	205 ps	210 ps
T_{IH}	315 ps	220 ps
T_{IS}	240 ps	220 ps
T_{DSH}	300 ps	380 ps
T_{DSS}	300 ps	380 ps
$T_{QSH/QSL}$	570 ps	713 ps
T_{QHS}	180 ps	230 ps
T_{DQSQ}	125 ps	200 ps
$P_{Total\_Active\_Peak}$	˜1.5 W	˜170 mW
$P_{Total\_Active\_Idle}$	˜500 mW	˜5-10 mW
$Latency$	˜50 ns	˜50 ns

3.5 Summary

We have described a framework to compare mobile memory interfaces that includes a decision tree, a memory interface calculator and requirements/capabilities for the primary bounds. Considerations for a mobile system that concern the memory interface can be elaborated with the help of requirements and capabilities concerning each primary bound, and applied according to a priority order embodied in a decision tree. Our memory interface calculator currently implements Equations (3.1) - (3.31) in Excel to project power and latency for various interface options that meet a given capacity and throughput.

A summary of the outputs of the memory interface calculator for the competing mobile memory standards is shown in Table 3.2. This includes the power and latency of the mobile controller's IO logic. Table 3.2 clearly highlights that LPDDR2 needs to scale beyond 533 MHz if it is to maintain its place in the mobile market. LPDDR3, currently projected to support 800MHz would be a suitable choice until 12 GB/s. Given the large power consumption of DDR3, a suitable alternative is required for throughputs higher than 12 GB/s. Wide IO, Mobile-XDR and Serial Memory (SPMT, MIPI) provide such alternatives. Wide IO is clearly the most power-efficient, but uses TSS. Mobile-XDR and serial memories provide traditional

interconnect options with differential signaling. Efficiency of serial schemes at lower throughputs is a concern. Larger (>2GB) capacity requirements need to be addressed for all the mobile memory types, but Wide IO provides an option of a multi-die stack using TSS.

Projecting both capabilities and requirements can also help identify gaps in memory interface offerings or predict bottlenecks in capabilities. Shown below in Figure 3.6 is the commodity memory space for the mobile market, overlaid with the mobile requirements from Figure 3.4 but leaving out the LPDDR3 curve. The figure shows a clear gap in the commodity offerings between LPDDR2 (until 8 GB/s) and the future mobile offerings of serial and Wide IO (above 12 GB/s). LPDDR3 is being discussed in JEDEC, but was not on the roadmap a year ago. It is not expected to scale beyond 12.8 GB/s, which establishes the need for either Wide IO or serial memory beyond 2014 for the mobile market.

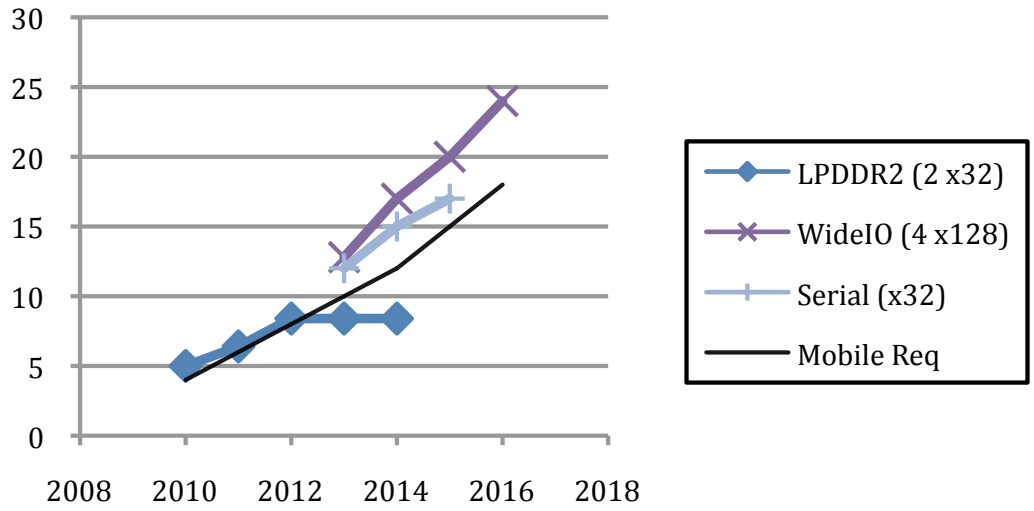


Figure 3.6: Gap in throughput (GB/s) before LPDDR3.

3.6 Acknowledgments

We thank Prof. Mark Horowitz and Prof. Dean Tullsen for illuminating discussions, Roger Barth for DRAM trends used by the ITRS Test ITWG, and Anand Srinivasan, Matt Nowak, Riko Radojcic, Michael Suh, Hari Rao and Jose Corleto for valuable inputs on mobile processor memory requirements.

Chapter 3 contains a reprint of A. B. Kahng and V. Srinivas, “Mobile System Considerations for SDRAM Interface Trends”, *Proc. ACM International Workshop on System-Level Interconnect Prediction*, 2011. The dissertation author is a main contributor to, and a primary author of this paper. I would like to thank my coauthor Andrew B. Kahng.

Chapter 4

Next-Generation Server Memory Interconnect

Server memory interconnect deserves a special discussion when it comes to off-chip electrical interconnect. The bandwidth and capacity requirements for server memory, make it not only one of the most challenging off-chip interconnect to architect and design, but also an expensive part of the server system. In this chapter, we extend the CACTI-IO framework described in the previous chapter to cover specialized server memory options, including cascaded and narrow channel options.

4.1 Introduction

Memory products have long been standardized and commoditized. Most server memory architectures have remained "traditional" – DDR channels emerge from a processor socket and support a small number of plug-in DDR DIMMs. The last few years have already seen the first signs of upheaval in the memory system. A number of new memory products have emerged recently or will soon be available, including buffer-on-board (BoB) [169], LR-DIMM [215], HC-DIMMs [192], NVDIMMs [243], HMC [111], NVMs [170, 134], and memory blades [82].

Server vendors are not viewing the memory system as a commodity any more – the memory system is now viewed as a differentiating feature, especially for customers that deal with big data workloads. There

are many example platforms and workloads that rely on processing of large in-memory datasets, e.g., SAP HANA [234], SAS in-memory analytics [235], RAMCloud [107], SPARK [158], and memory blades [82]. Memory system design choices dominate the performance, power, and cost metrics for such systems. For example, memory accounts for 50% of the power and 40% of the cost of 6 TB HP servers [194].

Vendors are therefore considering new approaches to design memory systems that best serve the needs of their customers. Further, future systems must efficiently support a combination of DRAM and NVM modules. We make the hypothesis that re-visiting the design of the basic DDR channel and introducing new memory interconnect topologies can yield large benefits.

To test the hypothesis, we first create a tool to precisely model interconnect power (referred to as I/O power) in the memory system. Instead of using the Micron power calculator's generic I/O model (as is done in most research evaluations today), we use SPICE simulations to model the effects of a number of parameters on interconnect power. These models build upon the ones in CACTI-IO [209], and a design space exploration has been added to identify the best design points. The overall tool has been integrated into version 7 of the popular CACTI package[2]. Our analysis with this tool shows that I/O power is indeed a significant contributor of power in large memory systems and is greatly affected by many parameters, e.g., the number of DIMMs per channel. The tool uses a simple API that allows users to define non-traditional interconnect topologies for the memory system.

Next, to make the case that memory interconnect specialization can yield significant benefit, and to test the value of our tool, we introduce and evaluate two novel interconnect architectures. We use the tool to carry out a simple design space exploration to identify the best design points for a given memory capacity requirement. Our analysis shows that higher bandwidth and lower cost can be achieved if the processor socket can somehow support a larger number of memory channels. This insight paves the way for the following two proposals.

1. A Cascaded Channel Architecture that is DDR compliant and a good fit for a DRAM/NVM hierarchy. By partitioning a DDR channel into multiple cascaded segments, it is able to support high memory capacity and bandwidth. It is also able to support a given memory capacity with a large number of

[2]CACTI 7 can be downloaded from Hewlett Packard Labs (`https://www.labs.hpe.com/downloads`) or mirror sites at UCSD (`http://vlsicad.ucsd.edu/CACTI7/`) or Utah (`http://arch.cs.utah.edu/cacti/`).

smaller-capacity DIMMs, thus lowering the overall cost for memory modules. DRAM DIMMs can be placed on high-frequency channels that emerge from the processor socket, while NVM DIMMs can be placed on lower-frequency channels that are further from the processor. The cascaded channels are enabled by the introduction of a new Relay-on-Board chip and a simple memory controller scheduler. We evaluate the new topologies in the context of a memory cartridge and show how our tool can be adapted to quantify the impact on performance, cost, and power.

2. A Narrow Channel Architecture that partitions a wide channel into parallel, independent, narrow, and higher-frequency channels. The corresponding channel and DIMMs are not DDR compliant. This opens up the possibility of defining new custom DIMM architectures. We therefore add new power-efficient features to the DIMM. Since a DIMM has lower external bandwidth, it can lower power by operating on-DIMM interconnects at lower frequencies. We also modify the error protection strategy to reduce data transfer overheads. Even though we consider DIMMs and channels that are not DDR-compatible, we assume that memory chips are unmodified and are always DDR-compliant. We again leverage our tool to demonstrate the benefits in memory bandwidth, cost, and power with this approach.

These new architectures thus make the case that re-thinking basic memory interconnect topologies can yield a rich space of very efficient memory architectures. While earlier versions of CACTI [97] have focused on cache hierarchies, the new version adds a new capability – the modeling of off-chip memory interconnects.

4.2 Tool Creation

4.2.1 Motivation

Emerging and future memory systems will use a combination of serial and parallel buses to connect a large number of memory modules to the processor and support high memory capacities. The memory modules usually adopt a DIMM form factor, and can be designed without on-DIMM buffers (an unbuffered DIMM or UDIMM), with on-DIMM buffers for command/address (a registered DIMM or RDIMM), or with on-DIMM buffers for all signals (a load-reduced DIMM or LRDIMM). A large number of memory organizations are therefore possible, each with varying memory capacity, bandwidth, power, performance,

and cost.

With each new memory generation, the energy per bit transfer is reduced, but at the same time, memory system bandwidth is also doubled. DDR4 DIMM designers are now targeting 3200 MT/s for multi-rank DIMMs [179]. Given this increase in bandwidth, and given the increase in memory capacity, memory power is increasing. I/O power is a significant contributor to overall memory power, especially when many ranks share a channel or when the channel operates at high frequencies.

Unfortunately, I/O power is overlooked in many research evaluations. For example, the Micron power calculator [220], the most popular memory power model, considers I/O power of a single dual-rank UDIMM for all cases, i.e., the Micron power calculator's output is oblivious to the DIMM and channel configuration.

The Micron power calculator reports that an 800 MHz channel operating at 80% utilization with a single dual-rank UDIMM, with a read/write ratio of 2 and a row buffer hit rate of 50%, dissipates 5.6 W of power, with 37% of that power being dissipated in I/O components (ODT, drivers). Since I/O power is such a significant contributor, we create and integrate a number of interconnect power models in a memory architecture exploration tool. As we show later, the I/O power strongly depends on technology (DDR3/DDR4), DIMM type (RDIMM, UDIMM, LRDIMM), the number of DIMMs per channel (DPC), the channel frequency, etc.

4.2.2 Tool Inputs and Outputs

An out-of-the-box version of our tool can receive a small number of high-level inputs from the user, explore the design space, and report the memory configurations that yield the best user-defined metrics. This may be used by researchers/practitioners to determine the properties of a baseline memory system. For novel non-standard memory architectures, the tool provides a simple API to represent the key interconnection features of the new architecture. A researcher can use this API to either define the properties of a custom memory network, or augment the tool's design space exploration to consider a wider range of network topologies. This chapter shows examples of all these use cases. The tool receives the following parameters as inputs.

1. **Memory capacity** in multiples of 4 GB and **if ECC support is provided**.

2. **Processor restrictions:** The following restrictions define how the memory is connected to the processor: (i) the number of memory channels, (ii) the number of wires in each memory channel, (iii) the memory type (i.e., DDR3 or DDR4), (iv) if the processor connects to a BoB and the number of DDR channels connected to the BoB.

3. **Minimum memory bandwidth requirement**.

4. **Memory access pattern:** the user can either specify a known memory traffic rate, row buffer hit rate, and read/write ratio, or it can allow the tool to iterate over a range of these values and report an average.

5. **Goodness metric:** This determines how the tool identifies the best memory organizations. The user can prioritize either power or cost or bandwidth, or provide his/her own metric that combines these metrics.

The new tool sweeps through all possible configurations that meet the input constraints, and identifies those that maximize the goodness metric, while also providing a power and cost breakdown.

4.2.3 Tool Modeling

High-Level Loops

The overall flowchart for the tool, written in C++, is described in Figure 4.1. Elements of this tool can be easily integrated in other architectural simulators so that workload characteristics can be used to estimate memory power.

Given the inputs specified in the previous subsection, the tool first identifies each required on-board channel (either parallel or serial). For the parallel DDR channels, it first runs through a loop that enumerates every possible allocation of the memory capacity across the DDR channels. For each channel, we then enumerate every combination of available DIMMs that can achieve that memory capacity, at both high-performance and low-power voltages. Based on these parameters, the channel frequency is determined (we provide more details on these steps shortly). We confirm that the total bandwidth of all specified channels is above the minimum specified requirement. For each valid configuration, we then estimate cost

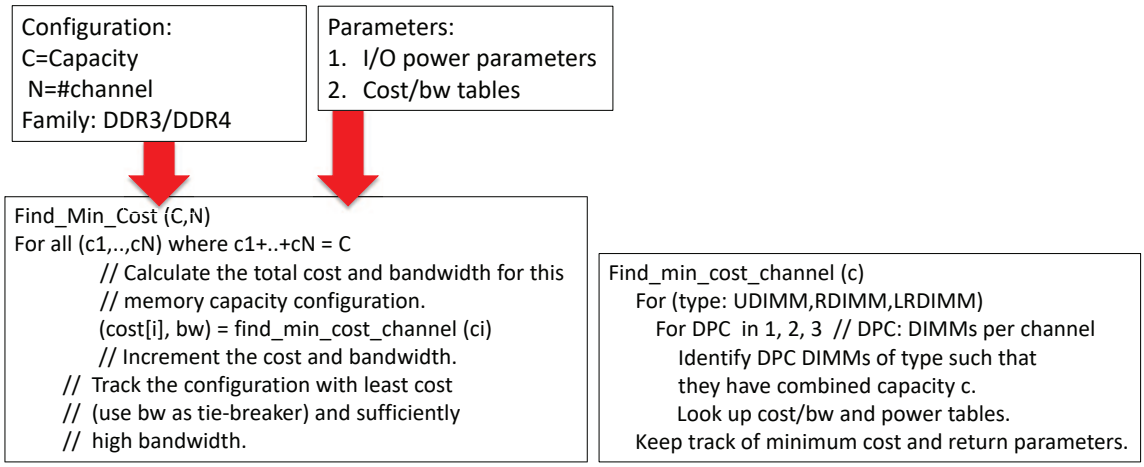


Figure 4.1: Basic flowchart for the tool's design space exploration for traditional DDR topologies.

to build a server, which is based on a lookup table that enumerates pricing for each type of DIMM. The power estimation is more non-trivial. We first use the Micron power calculator to estimate the non-I/O power consumed within the memory chips. For power consumed within each interconnect (on-DIMM, on-board, parallel, and serial), we develop our own methodology, which is described in Section 4.2.4. After estimating power and cost, we keep track of the memory configuration that maximizes the user-provided goodness metric (some combination of bandwidth, power, and cost).

Cost Model

To compute the cost to build a server, we need empirical pricing data for various DIMM types. We obtained DIMM prices from `www.newegg.com`, and averaged the price for the first ten products produced by our search. These prices are maintained by the tool in a lookup table. Some of this data is reproduced in Table 4.1 and shows prices for a variety of DIMM types and capacities for DDR3 and DDR4.

We must point out the obvious caveats in any pricing data. They are only meant to serve as guidelines because we can't capture the many considerations that might determine final price (for example, the pricing data might include some discount that may not be available a month later). That said, we feel the need to provide this pricing data because memory pricing considerations do drive the configurations of many server designs. A skeptical user can either use our pricing data to validate their own analytical

Table 4.1: Cost of DDR3 and DDR4 DIMMs.

DDR3					
DIMM	4GB	8GB	16GB	32GB	64GB
UDIMM	$40.4	$76.1	-	-	-
RDIMM	$42.2	$ 64.2	$ 122.6	$ 304.3	-
LRDIMM	-	-	$211.3	$ 287.5	$ 1079.5
DDR4					
DIMM	4GB	8GB	16GB	32GB	64GB
UDIMM	$ 26	$ 46	-	-	-
RDIMM	$ 33	$ 60.45	$ 126	$	-
LRDIMM	-	-	$ 279	$ 331.3	$ 1474.7

pricing model, or they can remove cost from the goodness metric altogether. The data in Table 4.1 is fairly representative of the memory market, where the highest-capacity DIMMs see a sharp rise in price-per-bit. To keep the tool up-to-date, we plan to periodically release updated pricing data.

Bandwidth Model

Memory channel frequency depends on the DIMM voltage, DIMM type, and the number of DIMMs per channel (DPC). This dependency is obtained from memory guideline documents of various server vendors. We obtained our specifications for frequency of DDR3 and DDR4 channels from Dell PowerEdge servers (12th generation) [181] and Super Micro's X10 series [239], respectively. Table 4.2 enumerates this frequency data.

Table 4.2: Typical frequencies of available DIMMs.

DDR3						
DIMM	1 DPC (MHz)		2 DPC (MHz)		3 DPC (MHz)	
type-ranking	1.35V	1.5V	1.35V	1.5V	1.35V	1.5V
RDIMM-DR	-	800	-	800	-	533
RDIMM-DR	667	667	667	667	-	533
UDIMM-DR	533	667	533	667	-	-
LRDIMM-QR	400	667	400	400	-	-
LRDIMM-QR	667	667	667	667	533	533

DDR4			
DIMM	1 DPC (MHz)	2 DPC (MHz)	3 DPC (MHz)
type-ranking	1.2V	1.2V	1.2V
RDIMM-DR	1066	933	800
RDIMM-QR	933	800	-
LRDIMM-QR	1066	1066	800

4.2.4 Power Modeling

We use CACTI-IO [209] as a starting point for our I/O models and introduce several extensions to enable a comprehensive exploration of memory systems:

1. DDR4 and SERDES models were added to the models already present in CACTI-IO (DDR3, WideIO, LPDDR3).

2. The DDR3 and DDR4 models are provided for three different types of segments:

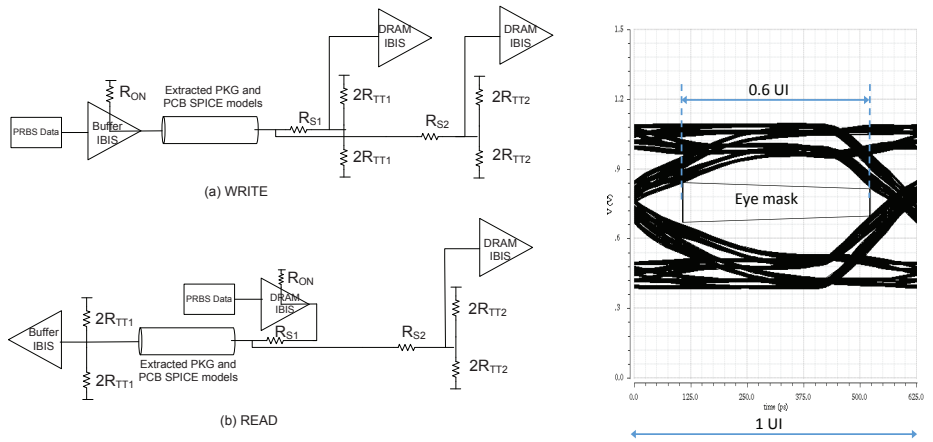
   - On-DIMM, i.e., from the DIMM buffer to the DRAM chip.
   - Main-board, for the processor or BoB to the DIMM buffer.
   - Both, for the host or BoB to the DRAM for an unbuffered signal.

3. Support has been added to compute termination power as a function of DDR3/DDR4 channel frequencies and channel loads (number of DIMMs/buffers on channel).

Each of the above models was constructed with detailed HSPICE simulations, following similar methodology as for CACTI-IO [209]. To construct the HSPICE models, we obtained IBIS [199] models for various components, we assumed an 8-bit datapath on a PCB metal layer to account for crosstalk, and we used a simple RLC model to approximate the DIMM connector [183]. As examples, Figure 4.2(i) shows the SPICE testbench used for WRITE and READ simulations for a DDR3 on-DIMM case.

For each interconnect end-point, a termination resistance is required to damp signal reflections. The value of the termination resistance determines interconnect power and signal quality. We therefore perform HSPICE time-domain analyses to plot eye diagrams for the data bus for each candidate memory configuration – different frequencies, different DIMMs per channel, and different topologies (on-DIMM, on-board, with/without a buffer). For each configuration, we sweep through different termination resistance values until the eye lines up to 0.6 UI quality, as shown in Figure 4.2(ii) (UI is unit interval, which is the ideal full eye opening).

With the above methodology, our tool is equipped with appropriate termination resistance values for a variety of parallel bus configurations. These termination resistance values are used by previously



(i) DDR3 SPICE testbench.

(ii) Eye diagram.

Figure 4.2: (i) DDR3 SPICE testbench. The testbench includes IBIS models for the buffer and DRAM, extracted SPICE models for the package and PCB, and a PRBS input stimulus pattern. We sweep the termination resistances R_{TT1} and R_{TT2} to identify a configuration with high signal integrity. (ii) DDR3 eye diagram.

validated I/O power equations in CACTI-IO to compute power for DDR3 interconnects. The DDR4 model has one significant change from that for DDR3: the termination resistors are referenced to VDDQ to allow for lower idle termination power. Equations (4.1) and (4.2) describe the WRITE and READ termination power for DDR4 DQ,

$$P_{DQ\_Term\_Write} = 0.5 \cdot V_{dd}^2 \cdot \left(\frac{1}{R_{ON} + R_{||\_Write}} \right) \tag{4.1}$$

$$P_{DQ\_Term\_Read} = 0.5 \cdot V_{dd}^2 \cdot \left(\frac{1}{R_{ON} + R_{S1} + R_{||\_Read}} \right) \tag{4.2}$$

where V_{dd} is the supply voltage, and R_{ON}, R_{S1}, $R_{||\_Write} = (R_{TT1} + R_{S1})||(R_{TT2} + R_{S2})$, and $R_{||\_Read} = R_{TT1}||(R_{TT2} + R_{S2})$ are DDR4 resistances similar to those shown in the DDR3 testbench in Figure 4.2(i). In addition, we allow technology scaling, borrowing the same methodology as CACTI-IO [209].

The timing budgets in CACTI-IO [209] are based off bit error rate (BER) models [65]; when termination resistance values are extracted for a given topology, frequency, and voltage, the eye mask feeds into the timing budget that is based on a specified BER. Our tool does not allow the user to modify the interconnect design while tolerating lower/higher BER.

For serial buses, the SERDES I/O power is modeled as a lookup table based on the length of the interconnect and the frequency. Because of the high variation in SERDES link architectures, it is difficult to

capture them with general analytical models. Figure 4.3 shows the typical components of a SERDES link, including the transmitter, termination, and the receiver. The power numbers for the various components are derived from a survey [73, 112, 91, 108, 111, 169], and divided into three types based on the length of the interconnect: short ($<$ 2 inches), mid ($<$ 10 inches), and long ($>$ 10 inches). We further provide power numbers for each type at three different frequencies: slow ($<$ 1 Gbps), medium ($<$ 5 Gbps), and fast (upto 10 Gbps). We provide these parameters for static, dynamic, and clock power. This allows for scaling with bandwidth, and also to investigate amortization of clock power over different numbers of data lanes, as shown in Equation (4.3),

$$P_{component} = P_{clock} + N_{lanes} \cdot (P_{static} + BW \cdot P_{dynamic}) \tag{4.3}$$

where $P_{component}$ is the power of the component of the SERDES link shown in Figure 4.3, P_{clock} is the clock power of that component, P_{static} is its static power, $P_{dynamic}$ is its dynamic energy/bit, BW is the bandwidth of the whole link, and N_{lanes} is the number of data lanes.

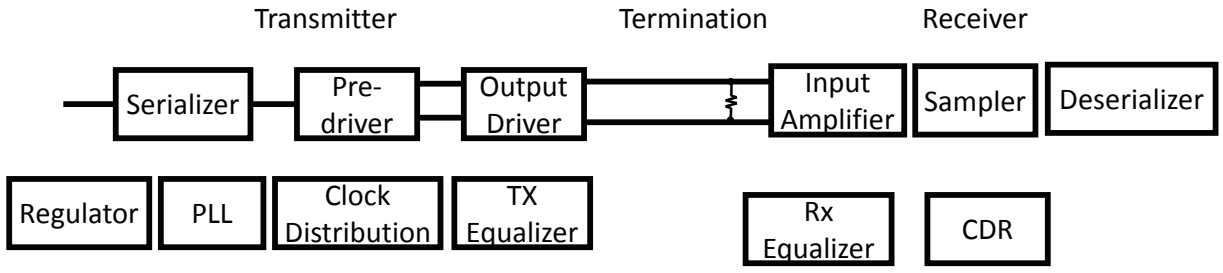


Figure 4.3: Typical components in a SERDES link.

To summarize this subsection, we have created lookup tables in our tool that can help capture the I/O power for a very large design space of memory interconnects. Some of these tables capture termination resistances that are obtained with detailed HSPICE simulations (DDR3 and DDR4) – analytical equations are then used to compute final I/O power. Other lookup tables (SERDES) directly capture I/O power from literature surveys.

4.2.5 An API to Define New Interconnects

The above discussion describes how our tool evaluates the design space of commercially available commodity memory products. As new products emerge, the data in the above tables and the power calculator equations can be augmented to widen the design sweep. In addition to new memory products, we expect that researchers will likely experiment with new memory network topologies and new memory hierarchies that combine multiple different memory devices. To facilitate such new models, we introduce the following API to define new links in the memory network. The new ideas explored in the second half of the chapter leverage these APIs for their evaluations.

For each interconnect, the API requires us to define the type of link and the width of that link. Our tool categorizes links into three types: serial link, parallel DDR (double data rate) link, and parallel SDR (single data rate) link. Serial links are used for high-speed point-to-point connections, e.g., the SERDES links used by the HMC or by the FBDIMM [148]. DDR links are used for the data bus in multi-drop memory channels. SDR links are used for the command/address bus in multi-drop memory channels.

Each link type has a few key knobs as input parameters, that allow the user to define the configuration and physical location of the segments of the interconnect. These knobs include the following, and are summarized in Table 4.3[3].

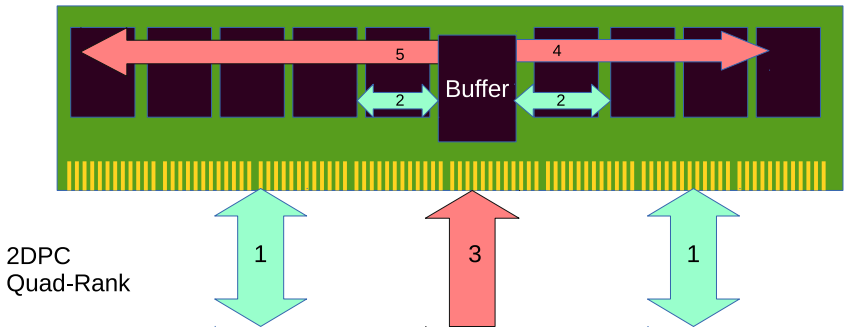
Table 4.3: Configuration parameters of different link types.

Type	Parameters
SERDES (Serial)	range, frequency, num_wire range: short or long
DDR	range, frequency, num_wire, num_drop, type, connection type:DDR3, DDR4, LPDDR2 connection: on_dimm, on_main_board
SDR	range, frequency, num_wire, num_drop, type, connection type:DDR3, DDR4, LPDDR2 connection: on_dimm, on_main_board

Figure 4.4 provides a detailed example of how the above API is used to describe an LRDIMM. A similar process would be used to define a new DIMM that (say) contains customized links between an

[3] *Range* refers to the length of the interconnect (as described previously). *Frequency* refers to the clock frequency on the bus. *Num_wire* refers to the number of wires or the bus width. *Num_drop* refers to the number of other devices (typically DIMMs and/or buffers) in a multi-drop interconnect. *Type* refers to the memory technology (DDR3, DDR4, WideIO, LPDDR3, etc.). *Connection* refers to the location of the segment, i.e., on-dimm or main-board or both (as in the unbuffered DIMM described before).

accelerator on the DIMM and specific memory chips, or a new memory topology where a BoB is connected to an HMC and a DDR channel.



1. DDR(long_range, frequency, 2(dimms), DDR3, 72, on-main-bus)
2. DDR(short_range, frequency, 4(ranks), DDR3, 72, on-dimm)
3. SDR(long_range, frequency, 2(dimms), DDR3, 23, on-main-bus)
4. SDR(short_range, frequency, 4(dies), DDR3, 23, on-dimm)
5. SDR(short_range, frequency, 5(dies), DDR3, 23, on-dimm)

Figure 4.4: Link description for a DDR3 LRDIMM

4.2.6 Validation

The key to precise I/O power calculations is the estimation of termination resistances that yield sufficiently open eye diagrams; as described earlier, that step is being performed with detailed SPICE simulations. These resistance values are then fed into our tool's equations to obtain the I/O power. The DDR3 equations have already been validated by CACTI-IO [209]. Here, we validate the DDR4 equations against SPICE DC simulations. As shown in Table 4.4, we compare the termination powers of one DQ lane driving low (driving a high result in close to 0 termination power). We assume V_{dd}=1.2, R_{on}=34, R_s=10 for DDR4 reads. This comparison is performed for a range of R_{TT1} and R_{TT2} termination values. Table 4.4 shows that the analytical equations used by our tool for the DDR4 termination power (as shown in Equations (4.1) and (4.2)) are aligned with SPICE simulations. It should be noted that our power models further include dynamic switching power (of the loads and the interconnect) and PHY power similar to the DDR3 case, as described and validated in [209].

Table 4.4: Validation of termination power.

Rtt1 (Ω)	Rtt2 (Ω)	CACTI 7 termination power (mW)	SPICE termination power (mW)
120	120	13.53	13.6
120	60	16.32	16.4
120	40	18.16	18.1
60	120	16.93	17.1
60	60	18.87	19.1
60	40	20.20	20.1
40	120	19.30	19.4
40	60	20.73	20.8
40	40	21.74	21.5

4.2.7 Contributions

We have thus created a tool and API, integrated into CACTI 7, that makes it easy to carry out design space exploration for modern and emerging memory topologies, while correctly modeling I/O power as a function of various parameters (not done by the Micron power calculator), and correctly considering DDR4, SerDes, cost, and bandwidth (not done by CACTI-IO).

4.3 Tool Analysis

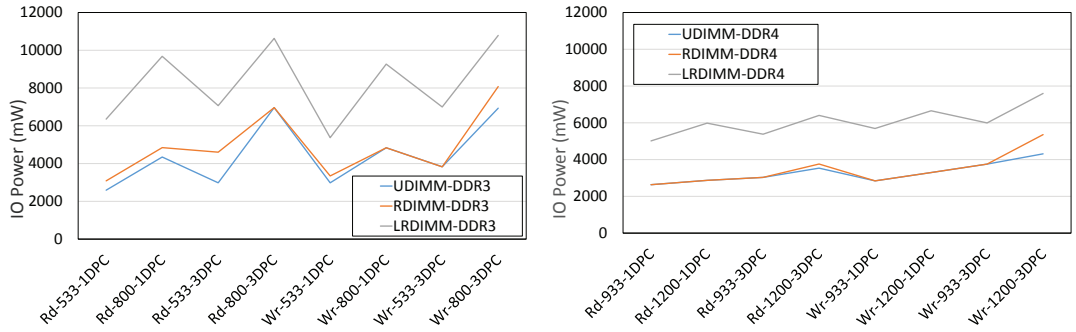
4.3.1 Contribution of Memory I/O Power

The previous section has described (i) how CACTI-IO power models have been augmented to handle a much larger interconnect design space, and (ii) how a large memory organization design space can be evaluated in terms of power, cost, and bandwidth.

To test our initial hypothesis that design choices can significantly impact I/O power and overall memory power, we use our tool to evaluate a number of memory configurations that differ in terms of DIMM types, read/write intensity, channel frequency, and DIMMs per channel (DPC). The power values are reported in Figure 4.5 for DDR3 and DDR4.

First, this paragraph compares I/O power to the DRAM chip power without any I/O components (obtained from the Micron power calculator). The configurations shown in Figure 4.5 dissipate I/O power between 2.6 W and 10.8 W for fully utilized channels. Even if we assume a low row buffer hit rate of 10%, eight ranks sharing the channel would collectively dissipate only 10.3 W in non-I/O DRAM power.

Similarly, two-rank and one-rank configurations would dissipate only 5.6 W and 4.8 W respectively. This highlights the significant role of I/O power in accessing the memory system.



(a) DDR3 configurations.

(b) DDR4 configurations.

Figure 4.5: Memory I/O power (mW) for a number of DDR3/DDR4 design points that vary DIMM type, read/write intensity, frequency, and DIMMs per channel (DPC).

In looking at each graph in Figure 4.5, we see that RDIMMs and UDIMMs dissipate similar amounts of power, but LRDIMMs dissipate nearly 2× more power. We also observe that at least in DDR4, there is a clear trend where write-intensive traffic (the right half of the figures) consumes more power than read-intensive traffic (the left half of the figures). Not surprisingly, power increases with channel frequency, and power increases as more DIMMs are added to the channel. To a large extent, I/O power is influenced by the following five factors in decreasing order of importance: DIMM type, technology, frequency, read/write intensity, and DIMMs per channel.

To better understand the differences between DDR3 and DDR4, we show the power breakdown for various benchmarks and frequencies in Figure 4.6. These graphs assume 3 LRDIMMs per channel. DDR4 consumes less power within DRAM and within interconnects, primarily because of its lower voltage and its lower idle termination power. This is true in spite of DDR4 operating at a significantly higher frequency than DDR3. But as a percentage of total memory power, the average contribution of I/O increases from 21% in DDR3 to 24% in DDR4.

Next, to show the impact of IO power in future large memory systems, including those that incorporate new memory devices such as Micron's Hybrid Memory Cube (HMC) [55], we evaluate the memory organizations described in Figure 4.7. Both organizations connect the processor to a low-latency HMC device that caches the most popular pages. The remaining pages are either scattered uniformly across four LRDIMMs on two DDR4 channels (a), or across an iso-capacity network of 32 HMCs (b). HMC

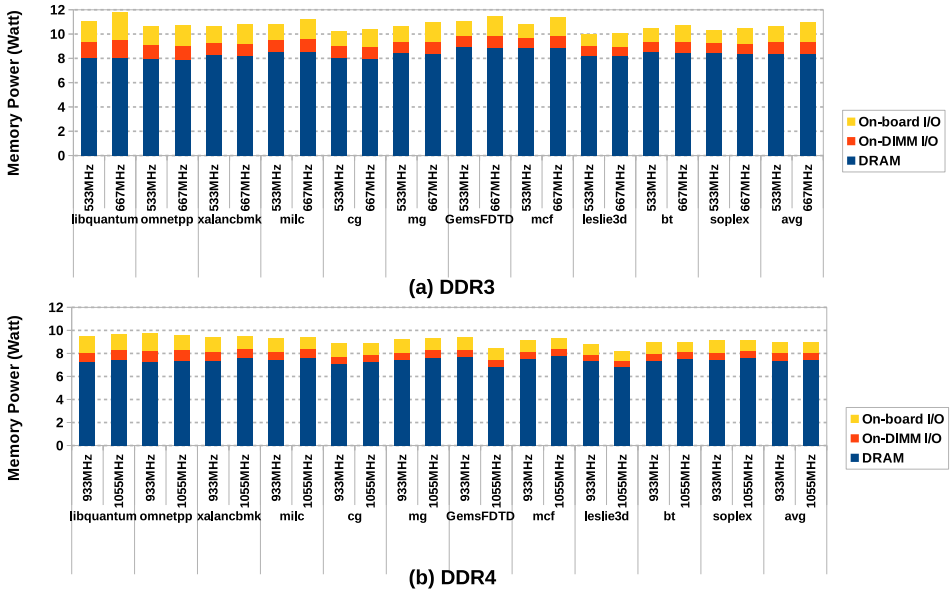


Figure 4.6: Breakdown of memory power (mW) for DDR3/DDR4 design points for each benchmark. We assume 3 LRDIMMs per channel.

memory access energy is based on the empirical data of Jeddeloh et al. [55]. We assume the same amount of background DRAM power per bit for both cases. The graph shows memory power dissipation as the hit rate in the HMC cache is varied. Given the many HMC SerDes links that are always active, we see that the HMC-only organization consumes significantly more power. This analysis showcases (i) the importance of IO power, and (ii) the relevance of DDR channels in future memory eco-systems.

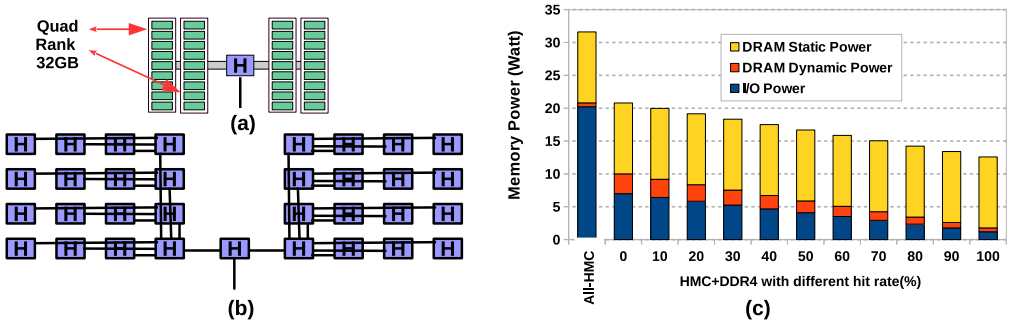


Figure 4.7: (a) A memory organization with one HMC "cache" that is backed by two DDR4 channels and four quad-rank LRDIMMs. (b) An HMC-only organization with one HMC "cache" backed by 32 other 4GB HMCs. (c) Memory power for the two organizations, as a function of the hit rate in the HMC "cache".

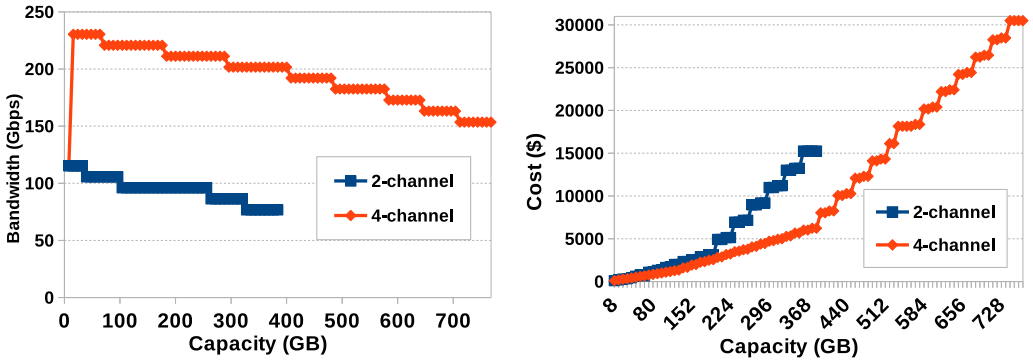


Figure 4.8: Identifying the best bandwidth (left) and cost (right) design points for each memory capacity requirement.

4.3.2 Motivation for Cascaded Channels and Narrow Channels

Next, to show the power of our tool's design space explorations, and to motivate the ideas in the second half of the chapter, we identify the best cost and bandwidth design points for a range of memory capacities. This is shown in Figure 4.8. We consider processor models that support two channels, and processor models that support four channels. The data shows that for a given memory capacity requirement, the use of four channels can improve overall bandwidth by more than the expected $2\times$. This is because the use of more channels helps spread the DIMMs, thus lowering per-channel load, and boosting per-channel bandwidth. Similarly, the use of more channels can also yield DIMM configurations that cost a lot less. This is because with more channels, we can populate each channel with different DIMM types and frequency, thus providing a richer design space, and avoiding the need for expensive high-capacity DIMMs.

With these observations in mind, we set out to create new memory channel architectures that can grow the number of memory channels without growing the pin count on the processor. Our first approach creates a daisy chain of channels with a lightweight buffer chip. Our second approach partitions a DDR channel into narrow sub channels. By decoupling memory capacity and processor pin count, we can further reduce cost by enabling lower-end processors and motherboards for certain platforms/workloads. In Figure 4.9, we show that Intel Xeon processors with fewer channels are available for a much lower price range.

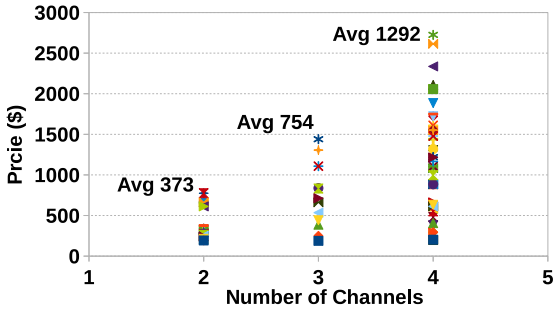


Figure 4.9: Prices for Intel Xeon (E3 and E5) processors as a function of channel count [201].

4.4 The Cascaded Channel Architecture

The cascaded channel architecture can simultaneously improve bandwidth and cost, while having a minimal impact on server volume or complexity or power. It can also be leveraged to implement a hybrid memory system, thus serving as an important enabling technology for future systems that may integrate DRAM and NVM.

The key tool insights that motivate this proposal are: (i) low load on a parallel bus leads to higher frequency, and (ii) cost is lowered by using many low-capacity DIMMs. We therefore partition a single DDR channel into multiple shorter DDR channels with a simple on-board relay chip.

4.4.1 Proposed Design

Baseline Memory Cartridge

We use a state-of-the-art memory cartridge as the evaluation platform and baseline in this work. Both HP and Dell have servers that can accommodate eight 12-DIMM memory cartridges to yield servers with 6 TB memory capacity [226, 193, 182].

Figure 4.10(a) shows the overall configuration of an HP ProLiant DL580 Gen8 Server [193]. Four processor sockets in the 4U server connect to 8 memory cartridges. Each processor socket has four memory links that connect to four BoB chips (Intel C104 Scalable Memory Buffers [200]). A memory cartridge is composed of two BoBs and their four DDR memory channels. Each memory channel can support up to three 64GB LRDIMMs at a channel frequency of 533 MHz. Figure 4.10(b) shows a logical view of a single memory cartridge; Figure 4.10(d) shows an actual image of an open cartridge populated with 12 DIMMs. The two PCBs close up to form a dense enclosure, as shown in Figure 4.10(c) (a side view). Each

BoB chip is on a separate PCB and the two emerging DDR channels are connected to the six interleaved DIMMs. We refer to the six DIMMs on the bottom PCB as *stalagmites* and the six DIMMs on the top PCB as *stalactites*. The DIMMs are arranged so that stalactites lie above a BoB and do not interfere with stalagmites (and vice versa).

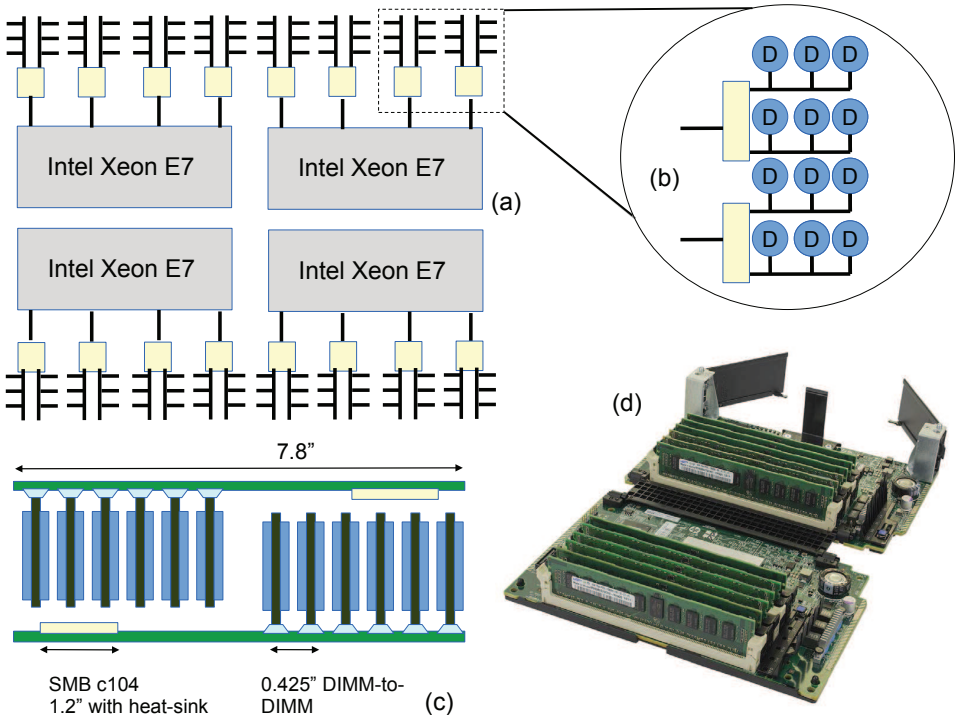


Figure 4.10: Organization of the HP ProLiant DL580 Gen8 server (a). The memory cartridge is represented in (b), (c), and (d).

Cascaded Channel Architecture

Our proposal uses a new Relay-on-Board (RoB) chip to create a daisy chain of DDR channels. As shown in Figure 4.11, the DDR channel emerging from the (yellow) BoB chip terminates in a (pink) RoB chip. Another DDR channel emerges from the other end of the RoB chip. What distinguishes this RoB chip from all other known BoB chips is that it has traditional DDR channels on either end that are populated with DIMMs.

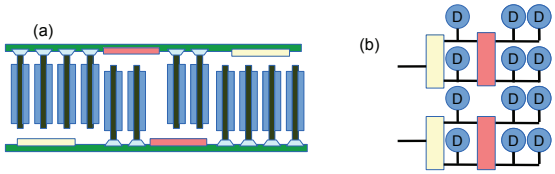


Figure 4.11: The Cascaded Channel Architecture. RoB chips are used to partition each channel.

From the memory channel's perspective, the RoB chip appears similar to an LRDIMM buffer chip, i.e., in terms of handling ODT and rank-to-rank switching delays, the RoB chip is simply handled as another rank. The RoB chip has a minimal amount of logic to receive data and (if necessary) drive the same signal on the next channel in the daisy chain (with appropriate re-timing and skew); the area of this chip is largely determined by its pin count. If we assume that the channels connected to the RoB operate at the same frequency, no data buffering is required on the RoB, and the signals on one channel are propagated to the next channel with a delay of one cycle. At boot-up time, the memory controller must go through a few additional steps for system initialization.

On a memory read or write, depending on the address, the request is serviced by DIMMs on the first channel or by DIMMs on the second cascaded channel. In the baseline (Figure 4.10), a single channel supports three LRDIMMs at a frequency of 533 MHz. In the cascaded channel design (Figure 4.11), the first channel segment supports the memory controller, a single LRDIMM, and a RoB chip. Such a channel is equivalent to a DDR channel populated with two LRDIMMs. Based on server datasheets [180], such a channel can safely operate at a frequency of 667 MHz. The second channel segment is similar – it supports the RoB chip (equivalent to a memory controller) and two LRDIMMs. Therefore, it too operates at a frequency of 667 MHz. The introduction of a RoB chip in a memory channel is similar to the introduction of a latch in the middle of a CMOS circuit to create a pipeline. The primary benefit is a boost in frequency and parallelism.

Figure 4.11(a) also shows how the cartridge can be re-designed in a volume-neutral way. The LRDIMMs are now interspersed with 1-inch wide RoB packages, again ensuring non-colliding stalactites and stalagmites. Assuming that cartridges can be designed with longer dimensions, we can continue to grow the daisy chain to further boost the number of DIMMs per cartridge.

While the proposed design bears a similarity to the FB-DIMM approach of daisy-chained buffer chips, it has been carefully designed to not suffer from the pitfalls that doomed FB-DIMM. First, we continue to use standard DDR channels and the RoB chips are on the cartridge rather than on the DIMM. This enables the use of commodity DIMMs. Second, the RoBs simply propagate signals and do not include power-hungry circuits for buffering, protocol conversion, and SerDes. Third, as described next, we introduce collision avoidance logic to simplify the memory controller scheduler.

A Scalable Memory Controller Scheduler

A many-rank system accessed by a cascaded channel requires an adapted memory controller scheduler. In essence, the memory controller must be careful to avoid collisions on the cascaded memory channels. Depending on the ranks involved in already scheduled data transfers, the timing for the next column-read/write must be adjusted. This is done by maintaining a small table with latency constraints. We have synthesized this collision avoidance circuit to timing-correct gate-level netlists and confirmed that the circuit adds area and power overheads under 10% to the memory controller. Since the design of this scheduler is not central to memory interconnect analysis, we do not delve into more details here.

Hybrid DRAM/NVM Hierarchy

It is expected that future off-chip memory systems will support a combination of DRAM and NVM. A number of papers have explored caching policies for such hybrid DRAM/NVM systems, e.g., [113, 117, 156]. Most of these studies assume that the processor socket has a channel populated with DRAM DIMMs and a separate channel populated with NVM DIMMs. However, such an organization suffers from three weaknesses. First, the processor pins are statically partitioned between DRAM and NVM; as a result, the entire processor-memory bandwidth is not available for the faster DRAM region. Second, some of these studies operate the NVM pins at a frequency lower than that of the DRAM pins, further lowering the overall processor-memory bandwidth. Third, data migration between DRAM and NVM involves the processor and consumes bandwidth on both the DRAM and NVM pins.

Given the expected popularity of future DRAM/NVM hierarchies, it is important to define memory interconnects that can address the above problems. Ham et al. [43] address the third problem above by introducing BoB chips on the DRAM and NVM channels, and a new link between these BoB chips; pages can therefore be copied between DRAM and NVM without involving the processor pins. We note that the Cascaded Channel architecture addresses all three weaknesses listed above.

The RoB chip decouples the characteristics of cascaded channels. Not only can each channel support different voltages and frequencies, they can also support different memory technologies. Figure 4.12 describes a number of possible DRAM/NVM configurations for baseline channels, as well as for RoB-based cascaded channels where the first channel represents a lower-capacity lower-latency region of memory, and the second channel represents a higher-capacity higher-latency region. While the baseline

cases allocate some processor pins for DRAM channels and some for slower NVM channels, the RoB-based designs boost processor pin bandwidth by connecting all the processor memory pins to fast DRAM that can contain hot pages. The proposed design also allows the copying of data between DRAM and NVM without occupying the processor pins twice. For example, when copying data from NVM to DRAM, the processor first issues a Read to the distant NVM; the data transfer on the distant channel is buffered on the RoB; the processor then issues a Write to the nearby DRAM; the data for that write is driven on the near channel by the RoB; the write into DRAM thus leverages the already scheduled data transfer from the distant NVM.

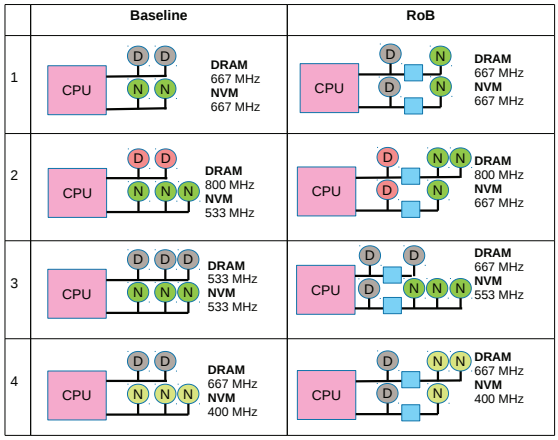


Figure 4.12: Four different cases that use DRAM and NVM. The Baseline organizations use separate channels for DRAM and NVM. The corresponding RoB architecture implements NVM on a distant cascaded channel. The 800 MHz and 400 MHz channels work at 1.5V and 1.2V, respectively, while other channels operate at 1.35V.

4.4.2 Cascaded Channel Evaluation Methodology

To model the power and cost of RoB-based designs, we modified our tool's outer for loops to not only partition the required capacity across channels, but also across cascaded channels. Every time a cascaded channel is used, we estimate IO power correctly, treating each RoB chip similar to an LRDIMM buffer. We assume that the second cascaded channel has half the utilization of the first channel. Since the RoB chip partitions a channel into two sub-channels, a larger design space is explored since we consider different options for every channel.

For our architectural evaluation, we consider a number of memory-intensive workloads from SPEC2k6 (libquantum, omnetpp, xalancbmk, milc, GemsFDTD, mcf, leslieeD, and soplex) and NPB [3]

(cg, mg, and bt). We generate memory access traces for these workloads with Windriver Simics [236]. Two (staggered) copies of these 8-core traces are then fed to the USIMM cycle-accurate memory system simulator [174] to create a 16-core workload sharing a single memory channel. This enables tractable simulations of future throughput-oriented architectures. Simics and USIMM parameters are summarized in Table 4.5.

Table 4.5: Simics and USIMM parameters.

Processor	
ISA	UltraSPARC III ISA
size and freq.	8-core, 3.2 GHz
ROB	64 entry
Fetch, Dispatch, Execute, and Retire	Maximum 4 per cycle
Cache Hierarchy	
L1 I-cache	32KB/2-way, private, 1-cycle
L1 D-cache	32KB/2-way, private, 1-cycle
L2 Cache	8MB/8-way, shared, 10-cycle
Protocol	Snooping MESI
DRAM Parameters	
DDR3	Micron DDR3-1600 [223],
DRAM Configuration	Single BoB with 2 Channels 3 ECC DIMMs/Channel LRDIMM 4 Ranks/DIMM
Mem. Capacity	192 GB (half a cartridge)
Mem. Frequency	533MHz
Mem. Rd Queue	48 entries per channel
Mem. Wr Queue Size	48 entries per channel

4.4.3 Cascaded Channel Results

For our DRAM-only analysis, we compare against a baseline memory cartridge with two 533 MHz DDR3 channels per BoB with 3 DDR3L quad-rank LRDIMMs per channel. The RoB-based cartridge has the same capacity as the baseline, but each DDR channel can now operate at 667 MHz.

Power Analysis

We use the modified version of our tool to estimate the power of the memory cartridge under high utilization. The baseline has a DDR channel utilization of 70%, while with RoB-based cascaded channels, the first channel has a 70% utilization and the second has a 35% utilization. We also assume a read/write ratio of 2.0 and a row buffer hit rate of 50%. The power breakdown is summarized in Table 4.6. At higher

frequencies, the DRAM power is higher because of higher background power. I/O power in the cascaded channel design is also higher because of the increase in channel segments and the increase in frequency. The result is an 8.1% increase in total cartridge power, but lower energy per bit (power/bandwidth). The I/O power is influenced by bus utilization. If we assume 50% and 80% utilization, the cascaded model consumes 4.6% and 7.1% more power than the baseline, respectively.

Table 4.6: Power breakdown of baseline and cascaded channels.

	DIMM Power (W)	BoB Power (W)	I/O Power (W)	Total Power (W)	Power/ BW (nJ/B)
Baseline	23.2	5.5	9.4	38.1	7.94
Cascaded	22.6	6.4	12.2	41.2	6.86

Performance Analysis

Figure 4.13 compares execution times for the cascaded design, relative to the baseline cartridge with the same memory capacity. For DDR3 design points, even though RoB traversal adds a few cycles to the memory latency, it enables a 25% higher bandwidth and lower queuing delays. The net effect is a 22% performance improvement. For the DDR4 design point, the cascaded channels enable a bandwidth increase of 13%, and a performance improvement of 12%.

To further boost memory capacity, additional RoB chips can be used to extend the daisy chain and add more DIMMs, without impacting the bandwidth into the processor.

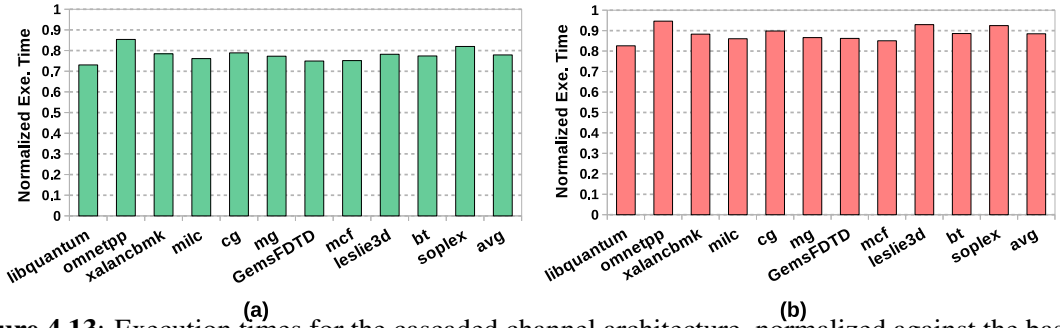


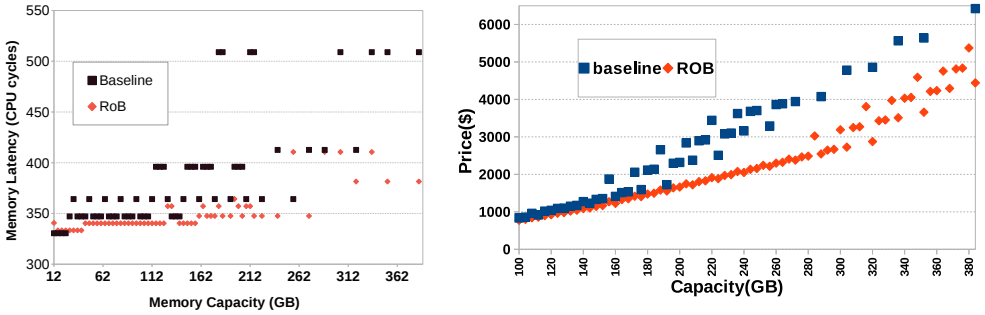
Figure 4.13: Execution times for the cascaded channel architecture, normalized against the baseline cartridge (DDR3 (a) and DDR4 (b)).

Design Space Explorations

With our extended tool, we evaluate performance as we sweep the design space. Figure 4.14(a) shows memory latency for all considered design points. Again, the RoB-based designs are superior in

nearly every case. Note that the baseline shows lower latency in a few cases – in these cases, the baseline and RoB-based designs end up with similar channel bandwidths and the RoB designs suffer from the extra hop latency.

Similarly, Figure 4.14(b) shows the cost of constructing the cheapest memory system for a range of memory capacity requirements. RoB-based designs offer more options when configuring a memory system. As a result, for example, a 256 GB memory system can be configured in the baseline with two 32 GB and one 64 GB DIMM per channel, while a RoB-based design can use two segments per channel, each with two 32 GB DIMMs. For this example, by avoiding expensive 64 GB DIMMs, the RoB-based approach yields a 48.5% reduction in cost. We see that in all cases, the cascaded approach is superior, with the improvements growing as capacity increases.



(a) Memory latency comparison.

(b) Cost comparison.

Figure 4.14: Memory latency and cost comparison with baseline and RoB approaches, as the memory capacity requirement is varied.

This analysis introduces the RoB as an additional knob in our design space exploration and helps identify the best way to configure a memory system for a given capacity requirement. Our extended tool makes the case that cost and performance can be improved by partitioning a standard DDR channel into multiple cascaded segments.

DRAM/NVM Hierarchies

We now turn our attention to the RoB chip's ability to implement hybrid DRAM/NVM hierarchies on cascaded channels. We consider iso-capacity comparisons in four cases, depicted in Figure 4.12 for baseline and cascaded approaches. The baseline in each case isolates the NVMs to a separate channel. The parameters for the NVM are based on PCM parameters assumed by Lee et al. [74]. In the first case,

the two designs have the same total bandwidth; the cascaded approach has the potential to do better if most requests are steered to DRAM because of hot page caching. The second case is similar, but uses lower-capacity dual-rank 1.5V RDIMMs for higher bandwidth, while increasing the number of NVM DIMMs. The third case increases capacity further; the baseline suffers from low memory bandwidth while the RoB-based design isolates the high capacity and low bandwidth to a single distant channel. The fourth case is a low-power version of the second case.

A DRAM/NVM hierarchy must be complemented by OS/hw policies that can move hot pages to low-latency DRAM, e.g., [156, 117, 82]. We view such policies as an orthogonal effort that is beyond the scope of this work. Here, we show results for a number of assumptions, ranging from 50% to 90% of all accesses being serviced by DRAM. Figure 4.15 shows the normalized execution time averaged across all benchmarks for the four cases, and for varying levels of DRAM activity.

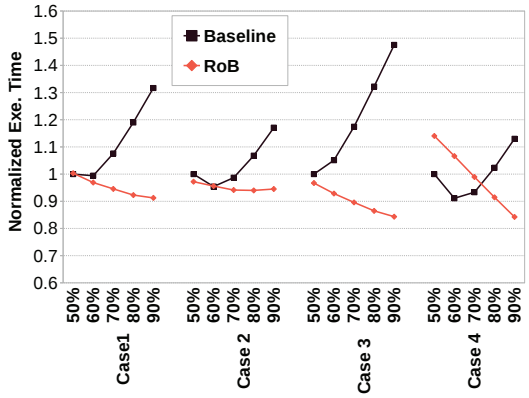


Figure 4.15: Normalized execution time (averaged across all benchmarks) for baseline and RoB cases, as the fraction of DRAM accesses is varied from 50% to 90%.

A common theme in these results is that when the DRAM is responsible for more than 60% of all memory traffic, performance starts to degrade in the baselines. This is because a single DRAM channel is being over-subscribed. The cascaded approaches allocate more processor pins for DRAM and can better handle the higher load on DRAM. In nearly all cases, the cascaded approach is a better physical match for the logical hierarchy in DRAM/NVM access. In the fourth power-optimized case, the RoB-based NVM is disadvantaged, relative to the baseline NVM, so it performs worse than the baseline at high NVM traffic rates. The energy trends are similar to the performance trends.

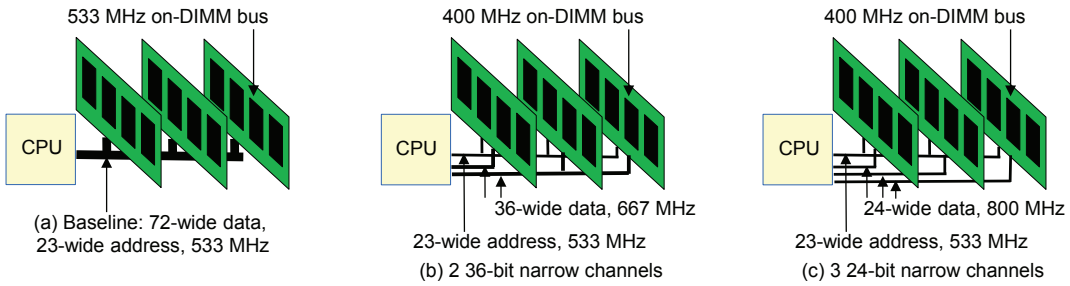


Figure 4.16: The baseline (a) and two narrow channel organizations (b and c).

4.5 The Narrow Channel Architecture

Finally, we introduce a new interconnect topology that is not constrained by the DDR standard. This is also an example of how our tool is useful for on-DIMM evaluations.

4.5.1 Proposal

Example Narrow Channels

Figure 4.16(a) shows a standard 72-bit DDR channel supporting 3 DIMMs at 533 MHz. In order to support high memory capacity without growing the load on the data bus, we propose to use narrow parallel data channels – two options are shown in Figures 4.16(b) and 4.16(c). The first option in Figure 4.16(b) implements two 36-bit wide parallel channels, both operating at 667 MHz. For an iso-capacity comparison, we assume that one channel supports two DIMMs and the other supports a single DIMM. These DIMMs use a buffer chip for address and data, similar to the design style of an LRDIMM. The buses between the buffer chip and the DRAM chips on the DIMM are similar to that of a baseline DIMM, but they can operate at a lower frequency and still keep up with the bandwidth demands of the external link. In the example in Figure 4.16(b), the on-DIMM 72-bit bus conservatively operates at 400 MHz and supports an external 36-bit link at 667 MHz.

In the example in Figure 4.16(c), the channel is split into 3 narrow channels, each 24 bits wide, and each supporting a single DIMM at 800 MHz. Again, the on-DIMM 72-bit bus conservatively operates at 400 MHz and supports the 24-bit external link at 800 MHz.

In both of these designs, we assume that only the data bus is partitioned across the narrow channels.

The address/command bus remains the same as before, i.e., a single address/command bus is shared by all three channels and all three DIMMs. This ensures that the new architecture does not result in a large increase in pin count for the processor. Because the address/command bus is driving the same load as the baseline, it continues to operate at the slower 533 MHz frequency. Since address/command buses are utilized far less than the data bus, this lower frequency does not make the address/command bus a new bottleneck.

Advantages and Disadvantages

Both of these new narrow channel designs have four primary advantages. (i) They support a faster aggregate bandwidth into the processor. (ii) They have the potential to reduce DIMM power by operating DIMM components at a lower clock speed. (iii) This approach can also be used to grow memory capacity without a corresponding steep penalty in bandwidth. (iv) Just as we saw for the RoB-based cascaded channels, there is a potential to reduce cost by implementing a given memory capacity with many low-capacity DIMMs instead of a few high-capacity DIMMs.

There are three disadvantages as well. (i) The primary disadvantage of course is that non-standard non-DDR DIMMs will likely be more expensive because they are produced at lower volume. In spite of this, we believe that this approach is worth exploring in the era of memory specialization, e.g., similar to how IBM produces custom DIMMs for their Power8 line of processors [238]. (ii) A longer transfer time per cache line is incurred. (iii) There is limited rank-level parallelism within each narrow channel. The second and third disadvantages are captured in our simulations and turn out to be relatively minor.

A Two-Tier Error Protection Approach

Since cache lines are aggregated on the buffer chip before returning to the processor, we take this opportunity to also improve error protection. We assume that the DIMM supports some form of error protection (say, SECDED or chipkill), but the processor is kept oblivious of this protection. The buffer chip inspects the bits read from DRAM, performs error detection and correction, and sends just the cache line back to the processor. Since ECC typically introduces a 12.5% overhead on bandwidth, this strategy eliminates or reduces that overhead. To deal with potential link transmission errors, we add a few CRC bits to every data packet returned to the processor. The CRC is used only for error detection. When an error is detected, the buffer chip simply re-transmits the data packet. To keep the protocol relatively unchanged, the

re-transmission can be triggered by the processor requesting the same cache line again. With this two-tier protection (SECDED or chipkill within DRAM and CRC for the link), we are still maintaining data in DRAM with strong protection, but the processor and link are subject to the overheads of a lightweight error detection scheme.

The reverse is done on a write, where the buffer chip on the DIMM receives a CRC-protected packet, computes the SECDED or chipkill code, and performs the write into memory chips.

If we assume the 36-bit narrow channel in Figure 4.16(b), a minimum of 15 transfers are required to communicate a 512-bit cache line. This leaves room for a 28-bit CRC code. For a well-constructed 28-bit CRC code, the probability of a multi-bit transmission error going undetected is very low (2^{-28}, the probability that a random 28-bit string matches the CRC for a new data block). By comparison, DDR4 has support for an 8-bit CRC [224] and HMC has support for a 32-bit CRC [197]. The CRC code can be made even stronger by sharing a code across multiple cache lines. For example, a group of four cache lines can share a 112-bit CRC code. As the size of the CRC code grows, the probability of multi-bit errors going undetected is exponentially lower.

Our two-tier coding approach further reduces bandwidth requirements by reducing the ECC bits sent back to the processor. In the above example, we are sending 540 bits on every cache line transfer instead of the usual 576 bits.

This error handling strategy moves the memory system towards an abstracted memory interface [228], where the processor simply asks for and receives data, while the details of memory access and memory errors are entirely handled by the DIMM or memory product.

4.5.2 Evaluation

To evaluate the proposed narrow channel architecture, we use the same simulation infrastructure as in Section 4.4.2.

Modeling this architecture with our tool is relatively trivial because of the convenient API provided. For the model in Figure 4.16(c), power was estimated with the following queries:

1. DDR(long-range, 800, 1, ddr3, 24, on-main-board): the 24-wide DDR3 data bus on the board connecting to 1 DIMM at 800 MHz.

2. DDR(short-range, 400, 4, ddr3, 72, on-dimm): the 72-wide DDR3 data bus on the DIMM connecting to 4 ranks at 400 MHz.

3. SDR(long-range, 533, 3, ddr3, 23, on-main-board): the 23-wide DDR3 address/command bus on the board connecting to 3 DIMMs at 533 MHz.

4. SDR(short-range, 400, 4, ddr3, 23, on-dimm): the 23-wide DDR3 address/command bus on the DIMM connecting the buffer chip to 4 DRAM dies on its left at 400 MHz.

5. SDR(short-range, 400, 5, ddr3, 23, on-dimm): the 23-wide DDR3 address/command bus on the DIMM connecting the buffer chip to 5 DRAM dies on its right at 400 MHz.

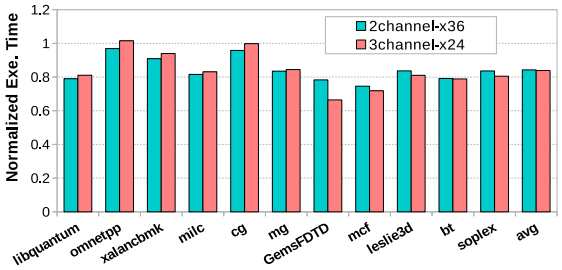


Figure 4.17: Execution time for the two narrow channel designs in Figure 4.16, normalized against the baseline.

Figure 4.17 shows normalized execution time for the two narrow channel architectures shown in Figure 4.16. The longer data transfer time and the reduced rank level parallelism per channel introduce second-order negative impacts on performance. But to a large extent, performance is impacted by the higher bandwidth enabled in the narrow channel designs. The new error handling strategy also contributes to the improvement in bandwidth, so the 24-bit and 36-bit channels increase peak bandwidth by 64% and 33% respectively (without the new error handling strategy, the bandwidth increase would have been 50% and 25%). Overall, the 24-bit and 36-bit channels yield performance that is respectively 17% and 18% higher than the baseline.

The power results are shown in Figure 4.18, which also shows a breakdown across the different components. The on-board I/O power is higher because of the higher frequency for the on-board interconnects; meanwhile, the on-DIMM interconnects and DRAM chips consume less power than the baseline because of their lower frequency. The net result is an overall memory power reduction of 23%. Again, we

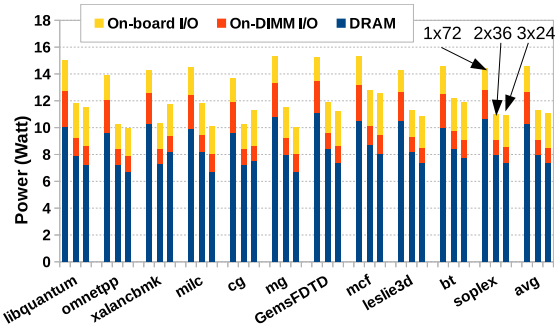


Figure 4.18: Memory power for the two narrow channel designs in Figure 4.16, normalized against the baseline (left).

see that I/O power is a significant contributor to overall memory power, highlighting the importance of precise I/O models as we explore specialized memory architectures.

4.6 Related Work

Research on memory systems has gained significant traction in the last few years. This is attributed to both interest in emerging non-volatile memories and relatively modest evolution of memory architecture in recent decades. As a result, the past few years have seen a flurry of work in the area of memory simulators and tools to facilitate research on disruptive memories with non-traditional fabric. CACTI-IO [209] is an IO modeling tool with its standard model limited to DDR3 configurations with unbuffered DIMMs. While our tool also focuses on IO, it is a comprehensive framework that considers cost, power, and noise (jitter and skew), and performs exhaustive search within the tool to find an optimal DIMM configuration for a given memory capacity and bandwidth. In addition to providing support for both on-dimm and main-board buffers for both DDR3 and DDR4, it supports a wide range of frequencies without any modification to the tool. CACTI 7 is also the first tool to support serial-io with different data rates.

Modeling tools such as Micron power model [221], DRAMpower [185], NVSIM [184], and DRAM energy models by Vogelsang [147] are primarily targeted at either microarchitecture of memory or DRAM die.

NVSIM is based on the CACTI tool that focuses on emerging non-volatile memories such as STTRAM, PCRAM, ReRAM, NAND Flash, and Floating Body Dynamic RAM.

Memory simulators such as DRAMSim [149], USIMM [174], and Ramulator [210] are perfor-

mance simulators that model DRAM timing in a cycle accurate manner. These tools take input from power models described earlier to calculate memory system power and can benefit from the proposed tool.

Memory DIMMs have been optimized for bandwidth, capacity, and power. The fully-buffered DIMM was Intel's solution for extending DDR2 memory capacity [38]. FB-DIMM uses narrow serial links to connect many buffer chips in a daisy chain. Hyper-Cloud DIMM (HC-DIMM) optimized LRDIMM by distributing LRDIMM's memory buffer functionality across multiple smaller buffer chips that are closer to the corresponding DRAM chips on the DIMM [192]. IBM has a custom DIMM for its AMB BoB, that supports up to 160 devices [145]. Apart from these industrial solutions, recent academic papers also suggest better DIMM and network organizations. Ham et al. [43] design a hierarchical tree topology for the memory network to support DRAM and NVMs, and Kim et al. [70] design a network of HMCs. Decoupled-DIMM decouples channel frequency from DIMM frequency using a custom buffer on DIMM [164]. BOOM [157] and Malladi et al. [92] use different approaches to integrate mobile LPDDR chips to reduce idle power. These related works not only highlight the importance of special non-standard DIMMs but also the need for a proper IO modeling framework to speed up research in this area.

4.7 Summary

In the future, we expect that parts of the memory system will move towards specialization. Much of that specialization will likely revolve around new interconnect topologies to connect different types of memory products. This chapter makes the case that I/O power is a significant fraction of memory power. We therefore develop a tool that models a variety of memory interconnects and provides a framework for design space exploration. With the insights gained from the tool, we devise two new memory network architectures that improve a number of metrics. We show that partitioning a memory channel into multiple channels (either cascaded or narrow) has a first-order effect on bandwidth and cost. The CACTI 7 tool was also modified and used to characterize power and cost for the proposed architectures. We observed performance improvements of 18% and energy reduction of 23% for the narrow channel architecture, and cost reductions of up to 65% for the cascaded channel architecture.

4.8 Acknowledgments

Chapter 4 contains a reprint of R. Balasubramonian, A. B. Kahng, N. Muralimanohar, A. Shafiee and V. Srinivas, “CACTI 7: New Tools for Interconnect Exploration in Innovative Off-Chip Memories”, *ACM Transactions on Architecture and Code Optimization* 14(2) (2017). The dissertation author is a main contributor to this paper. I would like to thank my coauthors Rajeev Balasubramonian, Andrew B. Kahng, Naveen Muralimanohar and Ali Shafiee.

Chapter 5

Interconnect for Silicon-Photonic NoCs

Apart from electrical interconnect, a growing area of research is silicon-photonic interconnect. A key cross-layer aspect of silicon-photonic interconnect for networks on chip (NoCs) is floorplan optimization, which includes floorplanning of the photonic and electrical elements of the NoC.

In this chapter, we describe our floorplan optimizer for silicon-photonic NoCs. The key differentiating aspect of our optimizer is that it performs P&R for PNoC using a cross-layer approach, which simultaneously considers system design choices, network design choices, optical device choices and application-dependent factors.

5.1 Introduction and Motivation

Over the past decade, the computing industry has regularly increased the number of cores per die [28], using parallelism to help sustain historical performance scaling. As core count continues to increase, both individual core performance and performance of the Network-on-Chip (NoC) fabric will determine the overall performance of a many-core system. It is preferable to use high-radix, low-diameter NoCs that are easier to program and provide more predictable communication. Such NoC topologies imply long global links that can be power-hungry when implemented with traditional electrical signaling circuits. Global silicon-photonic link designs provide noticeably higher bandwidth density, as well as lower data-dependent energy consumption, than electrical counterparts [125, 57]. In recent years, many efforts

have explored various types of high-radix, low-diameter photonic NoC (PNoC) topologies, including bus [71, 146], butterfly/Clos [42, 57, 109] topologies. A common aspect of these studies is that only one or two underlying link designs are considered as the basis of NoC architecture design decisions. Moreover, physical (layout) implementations (used for energy, performance and area evaluation) are fairly coarse. This potentially yields suboptimal PNoCs, as well as only limited comparisons versus electrical NoCs (ENoCs).

The placement and routing (P&R) solution of the various silicon-photonic link components bring several concerns beyond what is commonly seen for electrical links. These concerns include high thermal sensitivity, large propagation loss in CMOS-compatible link designs, large crossing losses, and laser source inefficiencies.[4] Indeed, thorough evaluation of the PNoC design space requires true *cross-layer*[5] optimization that considers (i) the rich design space of silicon-photonic devices, (ii) a range of network topologies, and (iii) detailed P&R solutions for PNoCs, particularly at the level of core cluster and PNoC floorplanning. Several recent efforts have provided flavors of cross-layer design [7, 48, 80].

In this paper, we develop a cross-layer approach to floorplan optimization in many-core systems with PNoC. At the level of system organization, our optimization considers chip aspect ratio, number of cores, and clustering of the cores. With respect to network design, our optimization considers PNoC logical topology, router design and placement, number of wavelengths to be multiplexed in a waveguide, and number of waveguides. Application-dependent factors (e.g., required PNoC bandwidth and different thermal profiles of the chip) are also considered.[6] Our simultaneous cluster placement and PNoC routing algorithm, based on a mixed integer-linear programming (MILP) formulation, provides a thermally-aware cross-layer global NoC (electrical and optical) optimization for a cost function of power and area. Specifically, our algorithm finds the optimal core cluster size and shape, router group placement, waveguide routing, and chip aspect ratio that together minimize total PNoC power. The main contributions of our

[4]High sensitivity of optical components to manufacturing variation is an additional issue, and can be influenced by the die planning and P&R solution.

[5]We use the term "cross-layer" in the usual way, to connote information flow across multiple layers of the system stack. Examples: (i) considering photonic device characteristics in P&R optimizations, or (ii) optimizing the floorplan based on architecture- and application-dependent power and thermal profiles.

[6]As detailed in Section 5.4.1, some of these parameters (physical dimensions of routers, laser source sharing solution, thermal sensitivity coefficients for photonic devices, etc.) are fixed in the experiments that we report. However, it is straightforward to explore these additional axes in many-core chip optimization, e.g., using an "outer loop" around the optimization that we describe.

work are as follows.

- We formulate an MILP that outputs P&R solutions for Clos PNoC with minimum power consumption, area, or some weighted combination of both. Our MILP formulation takes thermal effects of cores on photonic components into consideration and integrates all sources of power consumption, including laser power, electrical-optical-electrical (EOE) power, and thermal tuning power, required to reliably operate photonic devices.

- We develop a flow that uses a 3D extension of HotSpot [94] to precalculate thermal impacts of all cores on all potential router group locations; these are used in an optimization flow that is thermally-aware of a mix of heterogeneous power profiles.

- We propose the notion of a *power weight*, which represents the temperature impact of a core with unit power consumption on a router group, that allows us to efficiently consider a mix of high- and low-end core clusters within the router group placement optimization. This opens the door to the study of heterogeneous-core designs in the cross-layer optimization.

- We identify trends in experimental data – e.g., dominant sources of power, and impact of levers such as chip and cluster aspect ratios – that suggest future heuristic approaches to PNoC designs.

In the following, Section 5.2 reviews relevant previous work, and Section 5.3 describes our floorplan optimization approach including details of the MILP formulation. Section 5.4 gives experimental results of floorplan optimization, and we conclude in Section 5.5.

5.2 Previous Work

Floorplanning and P&R approaches for NoC designs have attracted significant research attention during the past years. The design decisions on these objects greatly impact the performance and energy efficiency of the overall many-core system. However, due to the vast design space and complexity, it is very challenging to consider all constraints during the design stage optimization of the NoC. To clearly present the large span of design constraints in our work, we make a comparison between our proposed method and representative previous work in this field, as in Table 5.1. In this table, we show the optimization goal of

each work in the last column. With the exception of the first four papers, all papers in the table focus on PNoCs.

Table 5.1: Classification of previous work and our work. OR–Optical Routing; OP–Optical Placement; ER–Electrical Routing; EP–Electrical Placement; TA–Thermally-Aware; 3D–3D-Related; NoC: NoC Topology.

Work	OR	OP	ER	EP	TA	3D	NoC	Opt.
Jafari et al. [53]			✓	✓				Fault-tolerant
Yan et al. [155]				✓	✓	✓		Wire Length
Ou et al. [106]			✓	✓	✓	✓		Wire Length
Dubois et al. [31]			✓	✓		✓	✓	Fault-tolerant
2-Sided Swap [22]	✓							Signal Loss
O-Router [28]	✓							Total Power
SNAKE [116]	✓					✓	✓	Total Power
Chen et al. [17]		✓				✓	✓	Total Power
GLOW [29]	✓	✓			✓			Total Power
PROTON [8]	✓	✓				✓		Laser Power
VANDAL [48]	✓	✓				✓		Signal Loss
Our Work	✓	✓			✓	✓	✓	Total Power

In the area of ENoCs, floorplanning and P&R approaches have been widely explored, with previous work chiefly focusing on reduction of total wirelength and/or maximum on-chip temperature. Yan et al. [155] propose a hierarchical algorithm for 3D placement to achieve the above goals with fixed routing input. Dubois et al. [31] provide a 3D-NoC floorplanning method that reduces the number of vertical-link connections in 3D layout. Ou et al. [106] propose a P&R method that minimizes chip area and routing wirelength, while satisfying current-flow and current-density constraints. Jafari et al. [53] propose an algorithm allowing a simultaneous P&R search. We use a similar MILP-based formulation to simultaneously find an optimal P&R solution.

In contrast to traditional ENoCs, PNoCs have more design-space constraints stemming from use of optical devices. For example, photonic devices' P&R solution directly impacts the attenuation of the optical signal, and in turn, the laser source power consumption. Previous work [22, 116, 28] propose routing algorithms that minimize the optical losses in the PNoC given a fixed netlist. However, although the optimization in [28] provides 50% optical power reduction, it does not consider the thermal tuning power, which is a significant contributor to the total PNoC power. Chen et al. [17] investigate the impact of on-chip

laser source placement and sharing on laser source power consumption; however, they do not propose a method to optimize the placement under different thermal profiles, and their work does not consider optimal routing of the PNoC. Hendry et al. [48] implement a tool flow that provides an environment to place-and-route the PNoC. A waveguide routing tool is provided in this flow to find the waveguide route with minimum optical signal loss. However, there are two common drawbacks of the above techniques: (1) they do not take into consideration thermal profiles, which can significantly impact thermal tuning power; and (2) the router placement is fixed before routing, which limits the design space for optimization. Li et al. [81] introduce a methodology for evaluating the thermal impact of chip designs on PNoC with VCSEL-based laser sources, but do not combine it with a comprehensive floorplan optimization. PROTON [8] and GLOW [29] both provide P&R algorithms for many-core systems. While GLOW takes thermal profile into account, it does not optimize router placement or account for thermal tuning power of the router groups, and the PNoC routing is performed with respect to a single thermal map. Typically, for many-core chips, thermal maps vary constantly depending on the system workloads and thread scheduling methods, hence it is important that the generated solution works for as many thermal maps as possible.

The key differentiating aspect of our optimizer is that it performs P&R for PNoC using a *cross-layer* approach, which simultaneously considers system design choices, network design choices, optical device choices and application-dependent factors. The optimizer solves for the best placement of on-chip optical devices *and* routing of waveguides so as to minimize total PNoC power (including EOE conversion and thermal tuning power). Our optimizer can optimize for one or more thermal profiles computed based on common workload profiles.

5.3 MILP-Based Floorplan Optimization

Our floorplan optimization comprehends the many-core chip and the PNoC as follows (see Figure 5.1). In the chip, *cores* are grouped together to form *tiles*. All communication within a tile is local (i.e., does not go through the PNoC) and electrical. In the studies reported below, we assume a fixed bandwidth of 512 GB/s for the PNoC [17]. We also assume an *8-ary 3-stage Clos* logical topology of the PNoC. The PNoC consists of *router groups*, each assigned to a set of tiles that constitute a *cluster*. All communication

between tiles within a given cluster and between routers in the same router group goes through electrical links. Two router groups across clusters communicate with each other using an optical link. The connection from one router group to another is called a *net*, which we must route legally within the routing graph.[7]

5.3.1 Notation Used in the MILP

Table 5.2 gives parameters and notations that we use in formalizing our MILP. The PNoC is defined by the locations of each router group in the set C, the orientations of their corresponding clusters, and the specific waveguides used to connect the router groups according to the topology implied by the set of nets N. As shown in Figure 5.1, each router group is associated with a rectangular cluster of tiles around it. The cluster can be oriented vertically or horizontally, with the router group itself at the cluster's geometric center. A is the set of all available edges in the routing graph, where a_{vrq} (resp. a_{hrq}) denotes a vertical edge from vertex (r,q) to $(r+1,q)$ (resp. a horizontal edge from vertex (r,q) to $(r,q+1)$). N is the predefined set of nets connecting the router groups according to the logical topology of the PNoC. Each net n has a given source cluster s_n and sink cluster t_n, where $s_n, t_n \in C$.

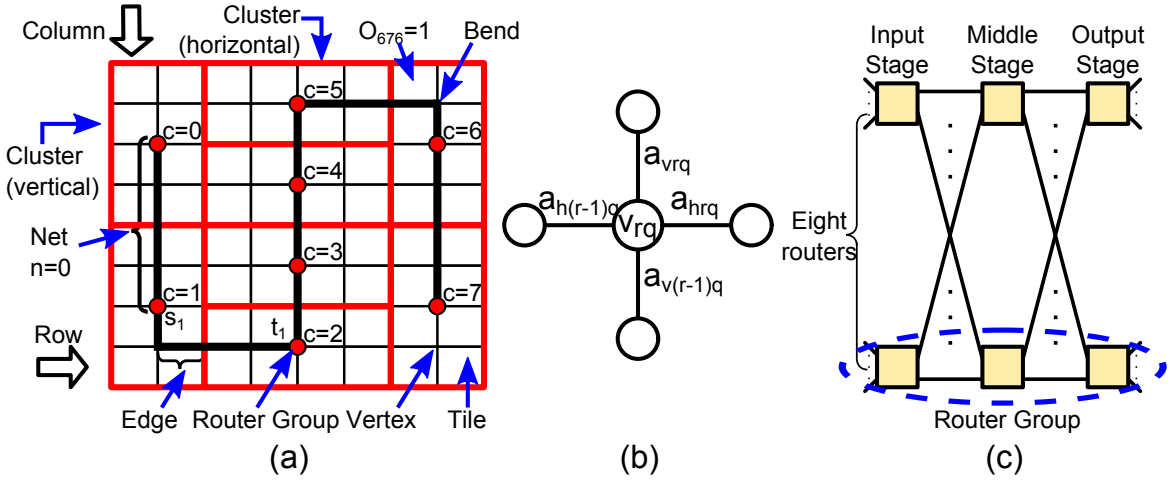


Figure 5.1: (a) Example of chip floorplan to illustrate our terminology. (b) A vertex and its surrounding edges in the routing graph. (c) 3-stage Clos topology with 8 router groups per stage.

[7]Implicitly, the studies reported below consider monolithic integration [104, 39] (as opposed to TSV-based stacked-die integration) of the photonic components with serpentine routing of all waveguides together (due to the cost of the trenches on the die). We assume on-chip laser sources are placed next to the router groups on a separate layer [17] where the link begins and ends.

Table 5.2: Notations used in the MILP.

Notation	Meaning
C, R, Q	sets of router groups, rows and columns, respectively
$G(V, A)$	routing graph that defines all possible locations and connections of the PNoC
V	set of vertices in the routing graph (= router group locations)
A	set of edges connecting pairs of vertices in the routing graph
N	set of 2-pin nets connecting the router groups
r, q	indices of rows and columns of tiles (defining possible vertex locations)
c	a router group $\in C$
n	a net $\in N$ with one source and one sink
s_n (t_n)	source (sink) router group of net n
r_c (q_c)	tile row r (column q) coordinate of router group c
γ^c_{frq}	0-1 indicator of whether router group c occupies vertex (r,q) with orientation f
a_{vrq} (a_{hrq})	denoting a vertical edge from vertex $(r-1,q)$ to (r,q) (horizontal edge from vertex (r,q) to $(r,q+1)$)
f_c	0-1 indicator of whether cluster of router group c is horizontal (0) or vertical (1)
x_{cr} (y_{cq})	0-1 indicator of whether cluster of router group c occupies row r (column q)
o_{crq}	0-1 indicator of whether router group c occupies tile (r,q)
$o_{fr'q'}(r,q)$	precalculated two-dimensional array capturing whether a cluster placed at (r',q') with orientation f would occupy tile (r,q); the array has entry 1 at each location that is occupied; with other entries 0
v^n_{rq}	0-1 indicator of whether vertex (r,q) is used in the route of net n
e^n_{hrq} (e^n_{vrq})	0-1 indicator that edge a_{hrq} (a_{vrq}) is used in the route of net n
d^n_{hrq} (d^n_{vrq})	cost of using a_{hrq} (a_{vrq}) in the route of net n
$used_r$ ($used_q$)	0-1 indicator of whether row r (column q) contains any router groups
H, W	height and width of the chip
H_T, W_T	height and width for each tile
H_C, W_C	height and width for each router group
P_{PNoC}, $AREA_{PNoC}$	power and area of the PNoC
P_{laser}, P_{tuning}, $P_{electrical}$	static optical laser power, thermal tuning power and electrical/EOE power of the PNoC, respectively
P_{prop}, P_{bend}, P_{cross}	propagation power per unit length, power per bend and power per crossing, respectively
$P_{constant}$	losses that are not affected by the MILP solution, including through loss, coupling loss, etc.
P_{loss}	sum of all optical loss terms, which then defines the laser power needed
$w_{r'q'}(r,q)$	thermal weight for vertex (r,q) due to tile (r',q')
θ_c	thermal impact at router group c due to the thermal weights and the power profiles
θ_{max}	maximum thermal impact among all router group locations
$p_{r'q'}$	power level of tile (r',q') (power profile)
p_c	power weight of cluster c (to allow for heterogeneous clusters)
SV_{rq} (SH_{rq})	0-1 indicator for a straight vertical (horizontal) route at vertex (r,q)
SV^n_{rq} (SH^n_{rq})	0-1 indicator for a straight vertical (horizontal) route on net n at vertex (r,q)
B_{rq}	0-1 indicator for a bend at vertex (r,q)
$\hat{B}_{rq}$	0-1 indicator for no bend at vertex (r,q)
CR_{rq}	0-1 indicator for a cross at vertex (r,q)
n_{bend}	total number of bends
n_{cross}	total number of crossings
P^0_{tuning}	thermal tuning power per degree Kelvin
$P_{modulator}$, $P_{detector}$, P_{SERDES}, $P_{cluster}$	DSENT-calculated power of modulator, detector, SERDES and ENoC within a cluster

5.3.2 Formal MILP Statement

We minimize a bicriterion objective function (Equation (5.1)) that is a weighted combination of the PNoC area and power. In the objective, α and β are user-specified scaling factors.

$$\textbf{Minimize:} \quad \alpha \cdot P_{PNoC} + \beta \cdot AREA_{PNoC} \tag{5.1}$$

Subject to:

$$\sum_{r \in R, q \in Q, f \in \{0,1\}} \gamma^c_{frq} = 1, \qquad \forall c \in C, \ \gamma^c_{frq} \in \{0,1\} \tag{5.2}$$

$$o_{crq} = \sum_{r' \in R, q' \in Q, f \in 0,1} o_{fr'q'}(r,q)\gamma^c_{fr'q'}, \quad \forall c \in C \tag{5.3}$$

$$\sum_{c \in C} o_{crq} \leq 1, \qquad \forall q \in Q, r \in R \tag{5.4}$$

$$\begin{aligned} &2v^n_{rq} - e^n_{hrq-1} - e^n_{vr-1q} - e^n_{hrq} - e^n_{vrq} - \sum_{f \in 0,1} \gamma^{s_n}_{frq} - \sum_{f \in 0,1} \gamma^{t_n}_{frq} = 0, \\ &\forall n \in N, r \in R, q \in Q \end{aligned} \tag{5.5}$$

$$r_c = \sum_{r \in R, q \in Q, f \in 0,1} r \cdot \gamma^c_{frq}, \ \ q_c = \sum_{r \in R, q \in Q, f \in 0,1} q \cdot \gamma^c_{frq}, \ \ \forall c \in C \tag{5.6}$$

$$f_c = \sum_{r \in R, q \in Q, f \in 0,1} f \cdot \gamma^c_{frq}, \quad \forall c \in C \tag{5.7}$$

Structural Constraints

A number of constraints enforce proper structure of the cluster placement and the PNoC routing. Using the 0-1 indicator variable γ^c_{frq}, Equation (5.2) ensures that exactly one vertex (r,q) and one orientation (horizontal or vertical) are chosen for each router group c. Equation (5.6) captures the vertex (r_c, q_c) in the routing graph where the router group c is placed, and Equation (5.7) captures the orientation f_c of the cluster of router group c.

Equation (5.3) captures which tiles on a chip are occupied by which cluster. A given o_{crq} indicates whether tile (r,q) is occupied by the cluster of router group c. $o_{fr'q'}(r,q)$ is a precalculated two-dimensional array that indicates whether tile (r,q) would be occupied by a cluster of a router group placed at (r',q') with orientation f. The array has an entry of one at each location that is occupied, and zero everywhere else. Equation (5.4) enforces the constraint that no tile on the chip can belong to more than one cluster. This ensures legal placement of clusters. If a tile is not in the footprint of any placed cluster (implying whitespace in the floorplan, e.g., for components other than the cores that communicate through the PNoC), then for that tile we will have $\sum_{c \in C} o_{crq} = 0$. Equation (5.5) [53] imposes flow conservation, i.e., a well-formed path of routing graph edges for each net n from its source s_n to its sink t_n. The 0-1 indicator variable v^n_{rq} captures the use of vertex (r,q) in the routing of net n; $e^n_{h/vrq}$ is a 0-1 indicator of whether edge $a_{h/vrq}$ is used in the routing of net n.

Equations for Area and Power

The area component of our objective function is determined by the following constraints. Equation (5.8) uses γ^c_{frq} to identify a binary indicator for the row (x_{cr}) and column (y_{cq}) the router group c is in. There is only one γ^c_{frq} that can be non-zero, and there is only one value in an array of all rows and an array of columns that is non-zero for each router group. Equations (5.9) and (5.10) indicate which rows and columns have router groups assigned to them. Router group locations cause extra area to be taken up in the chip, so by counting the number of rows and columns that are occupied we can obtain a figure of merit for

how much area is required for photonic components.

$$x_{cr} = \sum_{q \in Q, f \in 0,1} \gamma^{c}_{frq}, \;\; y_{cq} = \sum_{r \in R, f \in 0,1} \gamma^{c}_{frq} \;\; \forall c \in C \tag{5.8}$$

$$used_r = \begin{cases} 1 & \text{if } \sum_{c \in C} x_{cr} \geq 1, \forall r \in R \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

$$used_q = \begin{cases} 1 & \text{if } \sum_{c \in C} y_{cq} \geq 1, \forall q \in Q \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

$$\Delta H = H_C \cdot \sum_{r \in R} used_r \tag{5.11}$$

$$\Delta W = W_C \cdot \sum_{q \in Q} used_q \tag{5.12}$$

$$AREA_{PNoC} = (H + \Delta H) \cdot (W + \Delta W) - H \cdot W \tag{5.13}$$

The power component of the objective is determined by the following constraints. We convert P_{loss} (dbM) to P_{laser} (mW) using a piecewise-linear approximation and the laser's wall plug efficiency (WPE) to obtain the electrical input power required to operate the laser. P_{tuning} is the thermal tuning power needed to keep the ring groups at a similar temperature. $P_{electrical}$ is the power required for EOE conversion. $P_{modulator}$, $P_{detector}$, P_{SERDES} and $P_{cluster}$ values are obtained from code extracted from DSENT [135].

$$P_{PNoC} = P_{laser} + P_{tuning} + P_{electrical} \tag{5.14}$$

$$P_{loss} = P_{prop} \sum_{n \in N} \sum_{a_{h/vrq} \in A} d^{n}_{h/vrq} e^{n}_{h/vrq} + P_{cross} \cdot n_{cross} + P_{bend} \cdot n_{bend} + P_{constant} \tag{5.15}$$

$$P_{electrical} = P_{modulator} + P_{detector} + P_{SERDES} + P_{cluster} \tag{5.16}$$

Thermal tuning power is proportional to the difference between the thermal impact of a given router group (θ_c) and the maximum thermal impact (θ_{max}) over all router groups. Equation (5.17) calculates the thermal impact of each router group using the power profile of the system, with each tile's power level contributing a thermal weight $w_{r'q'}(r,q)$ to the router group at (r,q). Given that θ_c is a product of two binary variables, we must linearize it using the following technique (reprised in how we handle bends and crossings below).

$$\theta_c = \sum_{r\in R, q\in Q, f\in 0,1, r'\in R, q'\in Q} \gamma^c_{frq} \cdot w_{r'q'}(r,q) \cdot p_{r'q'}, \ \ \forall c \in C \tag{5.17}$$

$$p_{r'q'} = \sum_{c\in C} o_{cr'q'} \cdot p_c, \ \ \forall r' \in R, q' \in Q, p_c \ is \ fixed \tag{5.18}$$

$$P_{tuning} = P^0_{tuning} \sum_{c\in C} (\theta_{max} - \theta_c) \tag{5.19}$$

Accounting for Optical Bends and Crossings

We include the number of bends and crossings that exist in any given routing solution. The 0-1 indicator variable SV^n_{rq} (respectively, SH^n_{rq}) captures the existence of a straight vertical (respectively, horizontal) route through vertex (r,q) for net n. We derive SV^n_{rq} and SH^n_{rq} from $e^n_{h/vrq}$.

$$SV^n_{rq} \leq e^n_{vr-1q};\ SV^n_{rq} \leq e^n_{vrq};\ SV^n_{rq} \geq e^n_{vr-1q} + e^n_{vrq} - 1,\ \forall n \in N, r \in R, q \in Q \tag{5.20}$$

$$SH^n_{rq} \leq e^n_{hrq-1};\ SH^n_{rq} \leq e^n_{hrq};\ SH^n_{rq} \geq e^n_{hrq-1} + e^n_{hrq} - 1,\ \forall n \in N, r \in R, q \in Q \tag{5.21}$$

To account properly for all bends in the routing solution, we define a 0-1 indicator B_{rq} to capture the existence of a bend at vertex (r,q), and $\hat{B}_{rq}$ as a binary indicator for a vertex used with no bends. SH_{rq}, SV_{rq}, and v_{rq} respectively indicate straight vertical routes, straight horizontal routes, and vertex used at each (r,q) coordinate, for the superposition of all routed nets $n \in N$. Finally, we add the number of bends across all (r,q) to obtain the total number of bends in the routing solution.

$$\hat{B}_{rq} \leq SH_{rq} + SV_{rq};\ \hat{B}_{rq} \geq SH_{rq};\ \hat{B}_{rq} \geq SV_{rq}; B_{rq} + \hat{B}_{rq} - v_{rq} + \sum_{c \in C, f \in 0,1} \gamma^{c}_{frq} = 0,\ \forall r \in R, q \in Q \quad (5.22)$$

$$n_{bend} = \sum_{q \in Q, r \in R} B_{rq} \quad (5.23)$$

We also include all straight-straight crossings in our power loss equation, using some of the same variables. The 0-1 indicator variable CR_{rq} captures the existence of a crossing at vertex (r,q), enabling us to obtain the total number of crossings across all (r,q).

$$CR_{rq} \geq SH_{rq} + SV_{rq} - 1;\ CR_{rq} \leq SH_{rq};\ CR_{rq} \leq SV_{rq},\ \forall r \in R, q \in Q \quad (5.24)$$

$$n_{cross} = \sum_{q \in Q, r \in R} CR_{rq} \quad (5.25)$$

5.3.3 MILP Instance Complexity and Scalability

Using the notation and from the formulation given above, the complexity of an instance of our MILP is as follows.

- The number of variables: $8NRQ + 3CRQ + 4RQ + C + CR^2Q^2$.
- The number of constraints: $3CR^2Q^2 + NRQ + 14RQ + 5C + 1$.

For a typical instance that we study in the experiments reported below, $C = 8$, $R = 8$, $Q = 8$ and $N = 7$, implying 38152 variables and 99689 constraints. Both the number of variables and the number constraints

have terms that scale (i) linearly with the number of router groups, and (ii) quadratically with the number of tiles (RQ). If we assume that the number of cores per tile is fixed, then these parameters respectively translate to (i) the number of cores, and (ii) the size of the chip. For instances of this complexity, runtimes of ILOG CPLEX v12.5.1 [175] range from 10 seconds to several minutes on a 2.8 GHz Xeon server.

5.3.4 Optimization Flow Details and HotSpot Approximation

We conclude this section by pointing out several key details of our optimization flow and setup.
(1) Our floorplan optimizer takes as input a *.param file* with the following contents: (i) CoreParams (N_{cores}, W_{core}, H_{core}, Core Power); (ii) AspectRatio (AR_{min}, AR_{max}); and (iii) OpticalParams (loss mechanisms, waveguide dimensions and spacing, ring dimensions and spacing, and photodetector sensitivity). The .param file is used during setup of the MILP.
(2) Quite importantly, while we use HotSpot-3D [94] (embedded in HotSpot v6.0) as our thermal simulator, it is not practical to run HotSpot inside a high-dimensional optimization of floorplan, placement, routing and other solution attributes. Even more, the MILP approach is fundamentally incompatible with running a thermal simulator "in the loop", as might be contemplated with annealing or other iterative optimization frameworks. We work around this issue by precharacterizing a *core impact matrix* that captures the steady-state temperature impact of each running core on each possible router group location. The core impact matrix contains the thermal impact in K/W due to a 1 W core at each core location. We assume a linear superposition of core impacts due to all cores to calculate a final temperature at each vertex. We compare the temperature profile based on superposition with the data from HotSpot (with all the cores active simultaneously) and confirm less than 3% error. (Briefly: (i) a script generates potential HotSpot-compatible floorplan files (.flp) based on the core dimensions, number of cores, and router group dimensions; (ii) a dummy router group is placed at every valid vertex for each .flp file; and (iii) a *core impact matrix* is generated that consists of $(N_{rows}-2)\times(N_{columns}-2)$ values. Figure 5.2 illustrates the concept of core impact matrix generation. For the simple floorplan shown in part (a), two example core impact calculations are shown in parts (b) and (c) (for the two router group locations (1,3) and (2,2), respectively). The core impacts for each router group location (r,q) are summarized in the array of core impacts (d).)

(3) We extract code from the current DSENT [135] distribution to calculate the EOE power (modulator, detector, SERDES) and the electrical power for the NoC within the clusters. We would like to note that for the three link bandwidths considered in our analysis (see Table 5.3), we leverage DSENT's capability to perform datapath power optimization by balancing insertion loss and extinction ratio with modulator/receiver and laser power.

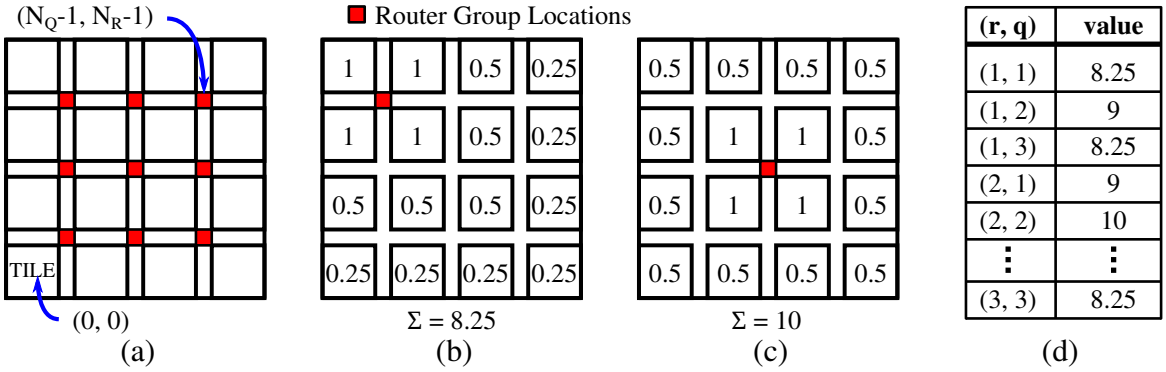


Figure 5.2: Core impact matrix generation: (a) illustrative floorplan with 16 tiles (64 cores) and nine potential router group positions; (b) sample core impact calculation for router group (1,3); (c) sample core impact calculation for router group (2,2); (d) a 1x9 core impact array generated for the floorplan.

5.4 Experimental Results and Discussion

5.4.1 Simulation Infrastructure

To test our optimization model, we use technology parameters corresponding to a 22 *nm* SOI CMOS process, and cores whose architecture is similar to the IA-32 in Intel Single-Chip Cloud Computer (SCC) [51], for our many-core systems. Each core has a 16 *KB* I/D L1 cache along with a 256 *KB* private L2 cache. After scaling the IA-32 core to 22 *nm*, each core (including caches) is reduced to a square shape with a side of 1.129 *mm*. We use HotSpot's default configuration file settings, but scale the heat spreader and heat sink lengths to be 2X and 4X the longest chip side length, respectively, for each floorplan. We also modify the configuration file in DSENT to match our experiments as follows: 22 *nm* technology; 1 *GHz* operating frequency; 2, 4, 8 *Gbps* link data rates for all the test cases; 3-stage 8-ary Clos or 3-stage 16-ary Clos topology according to different test cases; 64, 128, or 256 cores according to different test cases.

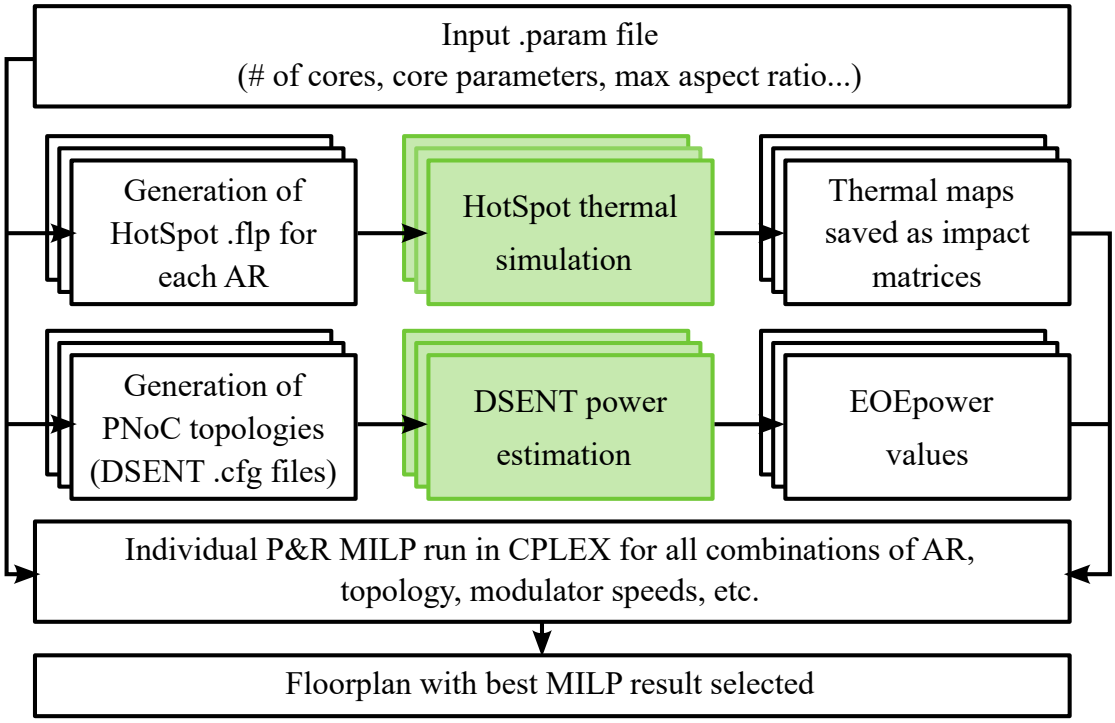


Figure 5.3: The floorplan optimization flow.

5.4.2 Design of Experiments

From above, our floorplan optimizer finds the optimal packing of clusters and routing of waveguides, based on given design inputs. To validate our optimization approach over a large experiment space, we use the set of configurations shown in Table 5.3. We consider WPE values of 5% and 15%, considering WPEs of current and future on-chip laser sources.

Workloads are intrinsically different from each other, which results in potential variations in their power profiles. Especially in a many-core system, it is common to have multi-program workloads and thus imbalanced power profiles [87, 24]. Furthermore, the emergence of heterogeneous systems exacerbates the imbalance within the power profiles. Thus, optimizing for known imbalances in power profiles may work as a viable goal for many real-life systems. Our experiments consider the power profiles in Figure 5.4(a)-(f).[8] We include these power profiles in our design of experiments to demonstrate that the optimal

[8]Our power profiles capture possibilities arising from both floorplanning of high- vs. low-power cores and thermally-aware task allocation. Profile (f) is a "Pringle's chip" that mimics systematic variation (slow cores on die edge requiring boosted supply voltage) seen in large, full reticle field ICs starting around the 65 *nm* node [45, 212].

Table 5.3: Experimental configurations studied.

#cores	Clos size	(chip AR, cluster AR)	optical datarate (Gbps)	#waveguides
64	8-ary (1 core/tile)	(1:1,1:2), (1:4,1:2)	8	8,16,32,64,128
			4	16,32,64,128
			2	32,64,128
128	8-ary (2 core/tile)	(1:2,1:1), (1:2,1:4)	8	8,16,32,64,128
			4	16,32,64,128
			2	32,64,128
256	8-ary (4 core/tile)	(1:1,1:2), (1:1,1:8)	8	8,16,32,64,128
			4	16,32,64,128
			2	32,64,128
	16-ary (1 core/tile)	(1:1,1:1), (1:1,1:4), (1:4,1:1), (1:4,1:4)	8	32,64,128
			4	64,128
			2	128

floorplan is sensitive to the power profile, and that designers can potentially determine the floorplan based on a power profile of a use case or combination of use cases (average, weighted-average, or worst-case). We assume the optical loss coefficients listed in Table 5.4.

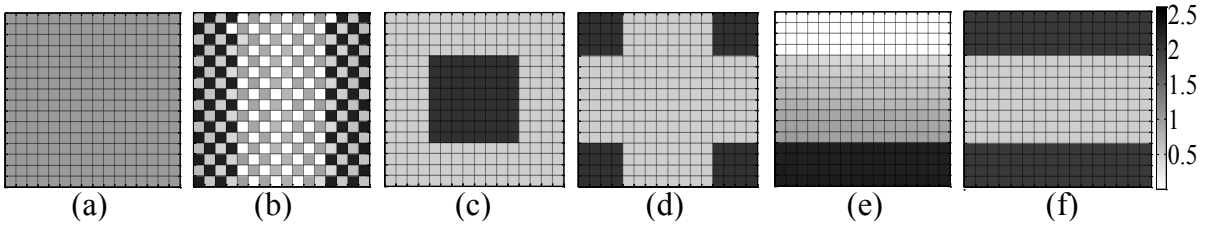


Figure 5.4: Six power profiles studied. Darker tiles indicate higher-power cores.

5.4.3 Results and Discussion

We now present our experimental results. In all cases that we consider, the logical topology is a chain from router group $c = 0$ to router group $c = |C|$, with $|N| = |C| - 1$ nets. Figure 5.5 shows how the accumulated thermal weight profile and the optimal floorplan vary with change in N_{cores} for a given NoC topology, optical data rate and number of waveguides. We see that although waveguide lengths increase with N_{cores}, the thermal tuning power (which depends on the thermal weight profile as highlighted in Figure 5.5(a)) tends to flatten out in larger chips due to more symmetry.

Table 5.4: Losses in PNoCs [57].

Loss Mechanism	Loss Contribution
Splitter Through Loss	0.2 dB per split
Waveguide Propagation Loss	2 dB per cm
Waveguide Crossing Loss	0.05 dB per crossing
Ring Drop Loss	1.5 dB per wavelength per ring
Ring Insertion Loss	0.1 dB per wavelength per ring
Ring Through Loss	0.01 dB per wavelength per ring
Photodetector Loss	0.1 dB per photodetector
Merge Loss	5 dB per merge

Figure 5.6 shows how the thermal weight profile and floorplan vary with the aspect ratio (AR) of the chip. In general, a skewed chip AR leads to a larger periphery, creating more asymmetry in the thermal weight profile as shown in Figure 5.6(a), but at the same time allowing for a shorter waveguide length.

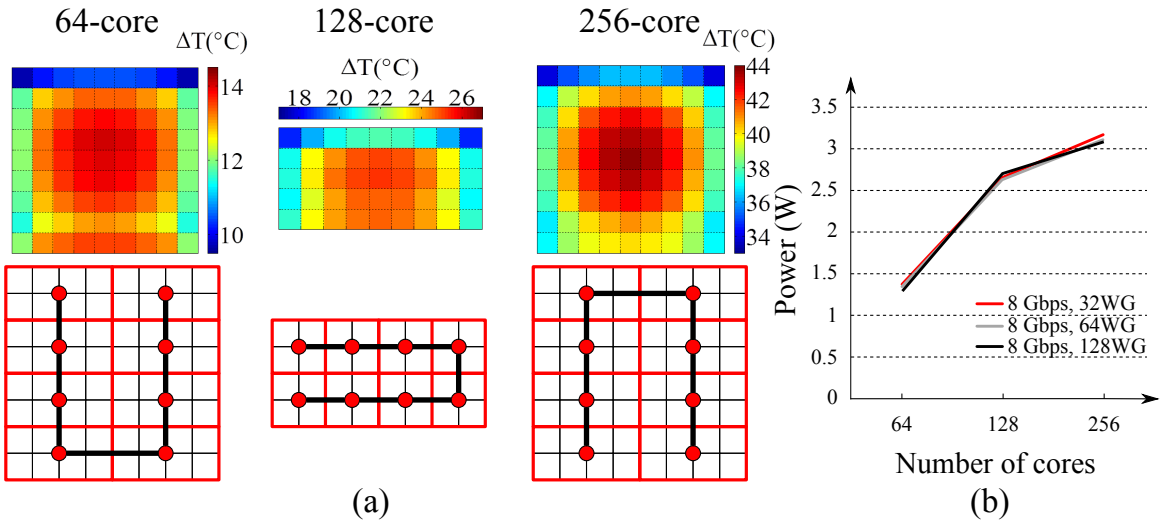


Figure 5.5: Accumulated thermal weight profile and optimal floorplan vs. N_{cores}.

Figure 5.7 shows the thermal weight profiles and floorplans for the different power profiles described in Figure 5.4. We note that the PNoC power varies by nearly 1.7X across the different power profiles. We also note that the optimal floorplans vary when we change WPE from 5% to 15%. A poorer

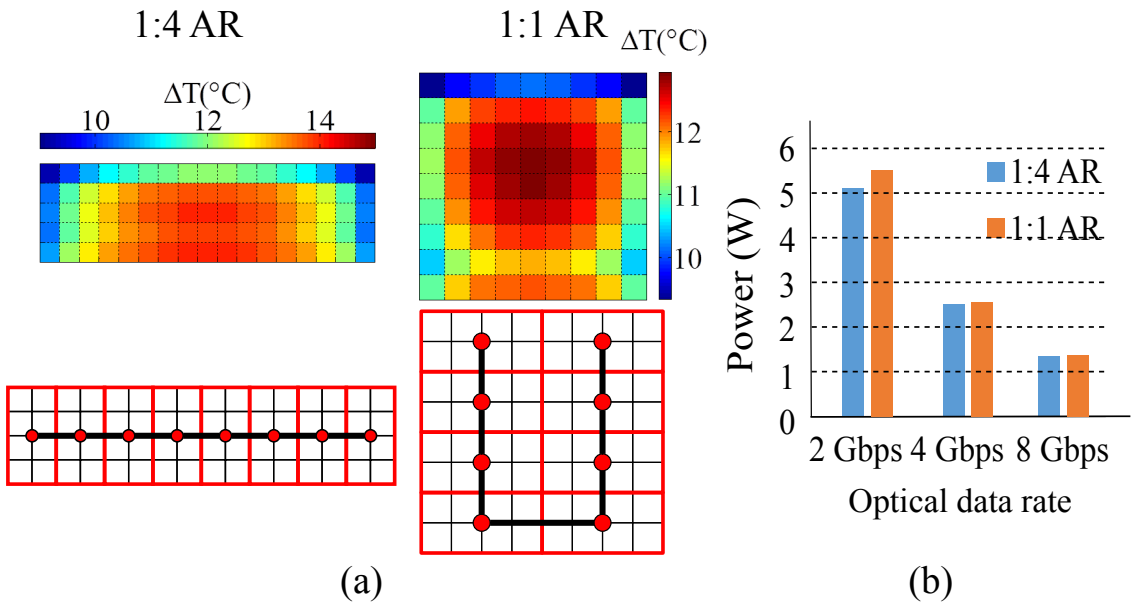


Figure 5.6: Accumulated thermal weight profile and optimal floorplan vs. AR.

laser source efficiency tends to favor the U-shaped floorplan. In comparison to a baseline vertical U-shaped floorplan, the floorplan in Figure 5.7(e) saves up to 15% power under the heterogeneous power profile in Figure 5.4(e).

Figure 5.8 addresses the impact of optical data rate on thermal weight profile, floorplan, and PNoC power consumption for a given NoC topology, N_{cores} and number of waveguides. We note that the power increases as optical data rate decreases, owing to the larger number of wavelengths needed to maintain the PNoC bandwidth.

Figure 5.9 shows how the optimal floorplan varies when we assign a power weight (p_c) to each cluster. This allows us to evaluate a mix of heterogeneous clusters. Part (a) of the figure shows the optimal floorplan for the case where the first half of router groups in the logical topology have low-power clusters (indicated by a blue router group), and the second half (red) have high-power clusters (indicated by a red router group). Part (b) shows the case where the low-power and high-power clusters alternate. The optimal floorplan attempts to pack as many high-power clusters as possible into regions of relatively similar thermal weight. Such heterogeneous clusters could also allow us to study best options for "dark silicon" during thermal throttling.

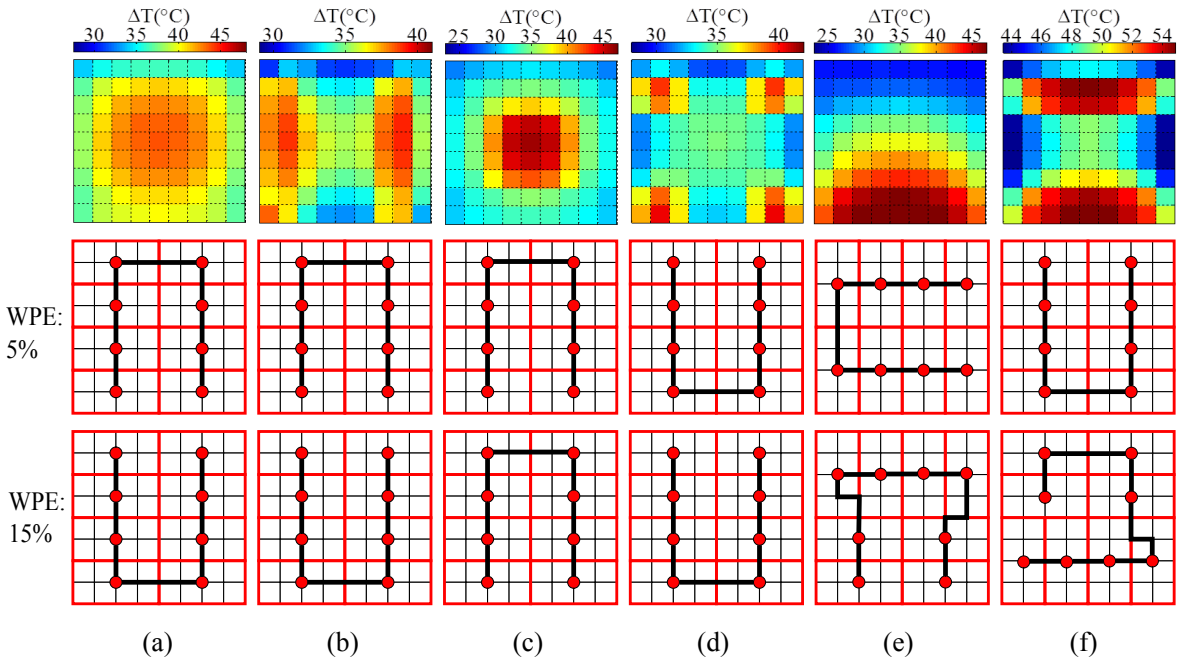


Figure 5.7: Accumulated thermal weight profile on the first row, and optimal floorplan with WPE of 5% and 15% on the second and third row respectively for power profiles (a) - (f) in Figure 5.4.

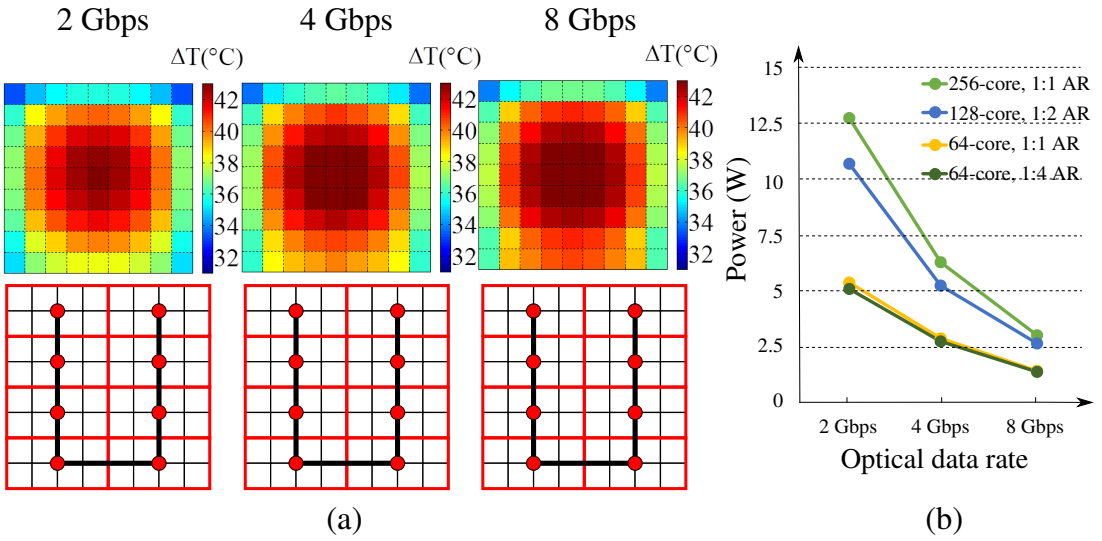
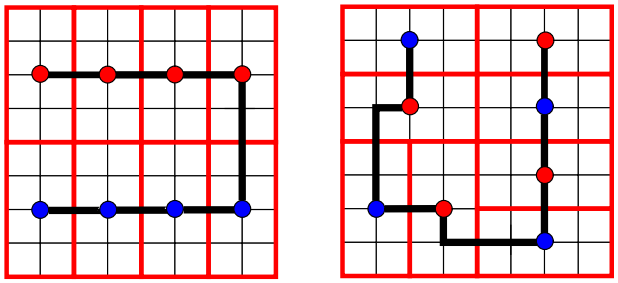


Figure 5.8: Accumulated thermal weight profile and optimal floorplan vs. optical data rate.

From our experiments, we arrive at the following general conclusions.

- Both thermal tuning power and laser power are important sources of power in the PNoC. Sensitivity to thermal weight profiles is especially important for cases with better WPE.



(a) LLLLHHHH (b) LHLHLHLH

Figure 5.9: Optimal floorplan for different cluster power weights; blue (red) router group indicates a low- (high-) power cluster respectively.

- Larger chips present an economy of scale for the PNoC power due to the more symmetric thermal weight profiles of larger chips.
- Skewed chip aspect ratios provide larger periphery and create asymmetry in the thermal weight profiles.
- The maximum achievable optical data rate is always preferred.
- It is important to consider different power profiles during the design time, since heterogeneous power profiles expose inherent weaknesses to certain router group locations. Being thermally aware of runtime management issues during floorplan optimization provides a key cross-layer advantage to such an optimization. Weighting the power profiles based on duty cycle and benchmarking metrics could provide a way to choose an optimal floorplan that is aware of the heterogeneous runtime power profiles.
- Allowing for power weights associated with clusters provides an additional knob to investigate the best mix and locations for high- and low-performance clusters, and the impact of dark silicon considerations on the optimal floorplan.

5.5 Summary

This paper has proposed a cross-layer, thermally-aware optimizer for floorplanning of PNoCs. Our simultaneous placement of router groups and core clusters, along with routing of waveguides, comprehends

scheduling policy, thermal tuning of photonic devices, and heterogeneity in application-dependent chip power profiles. Our optimizer uses an MILP formulation that minimizes PNoC power by explicitly considering laser source power (aware of propagation and other losses), EOE conversion power, and thermal tuning power. We also introduce the concept of a power weight, associated with each core cluster, which allows optimal placement of heterogeneous clusters, accounting for designs with heterogeneous cores. We achieve very reasonable running times for optimization of large many-core chips (on the order of a few minutes), which gives users the capability to make quick design-time decisions. We verify scalability and accuracy of the optimizer over a large design space. The results show that the optimal PNoC power is sensitive to thermal weight profiles and power profiles (1.7X variation), optical data rate (3-4X variation), number of cores (with chip edge for laser power and inversely for tuning power) and chip aspect ratios (up to 10%). Also, compared to thermally-agnostic solutions, our optimizer saves up to 15% PNoC power. We expect to integrate the floorplan optimizer with more extensive EDA flows in the future.

5.6 Acknowledgments

Chapter 5 contains a reprint of A. Coskun, A. Gu, W. Jin, A. J. Joshi, A. B. Kahng, J. Klamkin, Y. Ma, J. Recchio, V. Srinivas and T. Zhang, "Cross-Layer Floorplan Optimization For Silicon Photonic NoCs In Many-Core Systems", *Proc. Design, Automation and Test in Europe*, 2016, pp. 1309-1314. The dissertation author is a main contributor to this paper. I would like to thank my coauthors Ayse Coskun, Anjun Gu, Warren Jin, Ajay Joshi, Andrew B. Kahng, Jonathan Klamkin, Yenai Ma, John Recchio and Tiansheng Zhang.

Chapter 6

Interconnect for 2.5D NoCs

2.5D integration technology is gaining attention and popularity in manycore computing system design. 2.5D systems integrate homogeneous or heterogeneous chiplets in a flexible and cost-effective way. Inter-chiplet interconnect for 2.5D designs include microbumps on the chiplets and routing between them on a silicon interposer. The interposer could be passive or active, and the interconnect could be repeaterless, repeatered or pipelined.

In this chapter, we build a cross-layer framework to optimize 2.5D networks and inter-chiplet interconnect. We jointly optimize the network topology and chiplet placement across logical, physical and circuit layers to improve system performance, reduce manufacturing cost, and lower operating temperature, while ensuring thermal safety and routability.

6.1 Introduction

CMOS technology scaling has been slowing down over the past decade. It is getting increasingly difficult to continue technology scaling; hence, the industry has started to seek alternative solutions in the '*More Than Moore*' direction. Instead of putting more transistors in a monolithic chip, one approach is to pack multiple dies in a package [198, 203, 176]. This approach enables flexible integration of homogeneous or heterogeneous dies, and speeds up the design and manufacturing of semiconductor systems. Therefore, die-stacking technologies like 2.5D and 3D integration have gained traction.

These multi-die systems are cost-effective alternatives to single-chip systems (also called 2D systems), as breaking down a chip into multiple chiplets alleviates the manufacturing yield drop suffered in a large 2D chip. 3D integration stacks chiplets vertically to increase memory bandwidth and reduce system footprint [62], but aggravates thermal challenges [86]. 2.5D integration places multiple chiplets on a silicon interposer, which can be either passive or active. The chiplets communicate with each other through high-density fine-grained μbumps and interconnects in the interposer. Both 2.5D and 3D integration technologies enable designing high-bandwidth, low-latency networks, which could be utilized to handle the growing data traffic requirements of today's applications [198, 203, 176]. Compared to 2D systems, 2.5D systems have better thermally-safe system performance [34], enable integration of heterogeneous technologies [10, 176], and have lower cost [62]. Compared to 3D systems, 2.5D systems have better thermal dissipation capability, provide additional routing resources, and are more cost effective [62, 132].

Therefore, 2.5D systems are gaining attention and popularity as competitive candidates to sustain the performance and cost scaling in computing systems [176, 62, 40, 14, 16, 244]. There are already commercial 2.5D products in the market, such as Xilinx Virtex 7 [244], AMD Fiji [89], Nvidia Tesla [227], and Intel Foveros [191]. These existing products typically place the chiplets adjacent to each other on an interposer to embrace the benefits of low communication latency due to short inter-chiplet links and low manufacturing cost resulting from small interposer sizes. However, the design and optimization of 2.5D systems, including chiplet placement, inter-chiplet network architecture, design of inter-chiplet links and μbump assignment, need to be thoroughly explored to maximize the benefits of 2.5D integration [204].

In this chapter, we perform a cross-layer co-optimization of 2.5D inter-chiplet network design and chiplet placement across logical, physical, and circuit layers. Our methodology jointly optimizes network topologies, link circuit and routing options, μbump assignment, and chiplet placement. Consider the following two cases that highlight the need for such a cross-layer approach. (1) If we adopt a top-down approach, an architecture-level analysis of network topologies indicates that high-radix, low-diameter networks provide the best overall system performance (in instructions per cycle) for inter-chiplet networks. However, in the physical layer, such networks usually require long wires, which would limit the network performance, and hence, the overall system performance. In the circuit layer, such long wires require repeaters and/or need to be pipelined to achieve high performance, which necessitate active

(rather than passive) interposer technology. Since active interposers are 10× more expensive than passive interposers [110], the system cost becomes expensive and so the top-down approach does not provide a desirable solution. (2) A bottom-up, cost-centric approach prefers to use passive interposers, which can only support repeaterless links in the circuit layer, thus degrading link performance and limiting maximum link length. This leads to the adoption of low-radix, high-diameter inter-chiplet networks, which lowers overall system performance. Our cross-layer methodology comprehends logical layer, physical layer, and circuit layer together, leading to a better system solution compared to using solely top-down or bottom-up approaches as in previous works.

Our cross-layer methodology fills a significant gap in the literature on 2.5D system optimization by including inter-chiplet network design and chiplet placement together. Cross-layer co-optimization allows for simultaneous consideration of thermal behavior of chiplets, multiple potential network topologies, and multiple inter-chiplet link options, including their circuit designs, physical design constraints and routing costs. Previous works have explored limited tradeoffs among cost, power, thermal feasibility and performance of 2.5D systems due to the lack of such a cross-layer co-optimization methodology. For example, our prior work [34] describes a chiplet placement method that results in high-performance, low-cost, and thermally-safe 2.5D systems. However, that method lacks a true cross-layer co-optimization as it considers only a Unified-Mesh network topology in the logical layer, determines the physical design of inter-chiplet links without accounting for the μbump overhead in the physical layer, and uses only a repeaterless link in the circuit layer. Our latest work [25] improves on our prior work [34] by jointly accounting for network topologies, μbump overhead, and inter-chiplet circuit designs across the three layers, but it covers a limited set of chiplet placement options.

As shown in the rest of this chapter, our proposed cross-layer co-optimization methodology achieves better performance-cost tradeoffs of 2.5D systems. Our methodology explores a rich solution space. Specifically, in the logical layer, we consider a variety of network topologies, including Mesh, Concentrated-Mesh (Cmesh), Butterfly, Butterdonut [62], and Ring. In the physical layer, we search for the chiplet placement that minimizes operating temperature and meets the routing constraints. In the circuit layer, we explore inter-chiplet link designs. We co-optimize network topology, chiplet placement and routing, as well as inter-chiplet link design and provide a solution that achieves 88% iso-cost performance

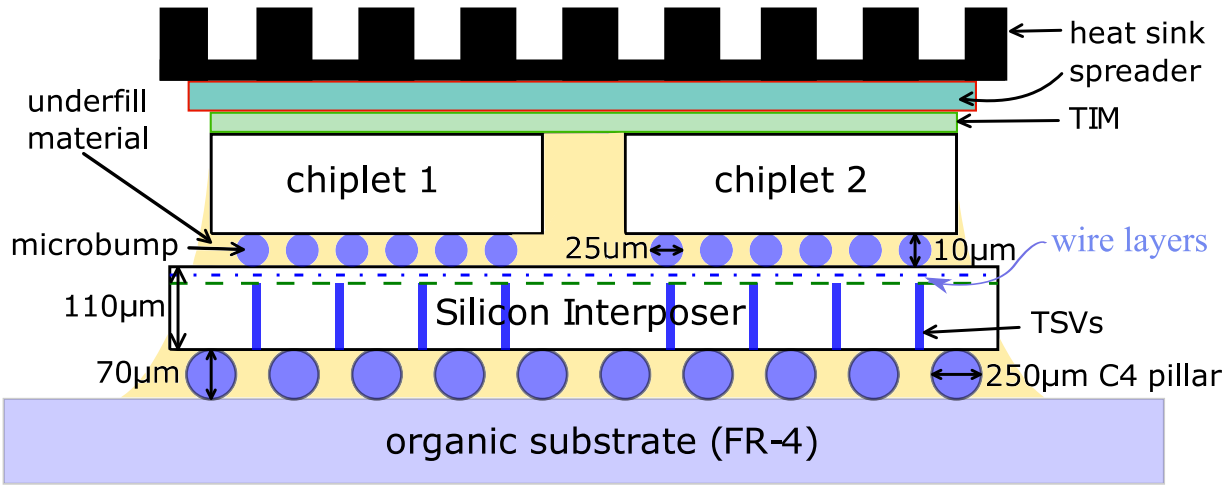


Figure 6.1: Cross-section view of a 2.5D system.

improvement and 29% iso-performance cost reduction compared to a single-chip design. Compared to our prior work [25], we achieve 40-68% (49% on average) iso-cost performance improvement and 30-38% (32% on average) iso-performance cost savings. The main contributions of this chapter are as follows.

- We develop a cross-layer co-optimization methodology that jointly optimizes 2.5D systems across logical, physical, and circuit layers. The outcome of our methodology includes network topology, chiplet placement, inter-chiplet link design and routing.
- Our methodology maximizes performance, minimizes manufacturing cost, and minimizes operating temperature. We use a soft constraint for peak temperature in the optimization problem to achieve better overall performance gain or cost reduction by allowing a small amount of thermal violation.
- We develop a simulated annealing algorithm to search the high-dimensional placement solution space. Our placer supports arbitrary placements that consider non-matrix and asymmetric chiplet organizations. We enhance a 2.5D cost model [133] to incorporate a comprehensive μbump overhead analysis on chiplet area and yield. We use *gas-station* link design [25] to enable pipelining in a passive interposer.

6.2 Background

2.5D integration is a promising technology that enables the integration of homogeneous or heterogeneous sets of chiplets onto a carrier. The carrier provides additional wiring resources that can be

leveraged to increase communication bandwidth between the chiplets and improve system performance [56]. Furthermore, 2.5D integration is more cost effective than large 2D chips and is more thermally efficient than 3D systems [133]. Currently, 2.5D integration technology is being widely explored by both academia [56, 62, 40] and industry [244, 16, 89, 143, 227, 191].

Embedded Multi-die Interconnect Bridge (EMIB) [188] and interposer [244] are two commonly used carrier options for 2.5D integration technology. EMIB is a novel integration method, which embeds small pieces of silicon interconnect bridges in the organic package substrate to connect the edges of adjacent chiplets for die-to-die communication. Silicon interposer technology uses a relatively large silicon interposer to house all chiplets. It is more mature and has been used in commercial products [244, 89]. Both EMIB and interposer can provide high density die-to-bridge and die-to-interposer connections, respectively, and correspondingly, high-density die-to-die connections [187]. EMIB-based approach requires less silicon area than silicon interposer-based approach and thus has lower silicon cost [187]. However, the number of die-to-die connections per layer of EMIB is limited by bridge interface length [115], and EMIB increases organic substrate manufacturing complexity [90]. Furthermore, EMIB can only hook up adjacent chiplets. When two chiplets that are far apart are logically connected, they cannot have direct links and need multi-hop communication using EMIB technology. Interposer-based integration provides more flexibility in chiplet placement, network design and interconnect routing, and thus, has better thermal dissipation capability as it does not require chiplets to be placed close to each other. Therefore, we focus on interposer-based 2.5D integration in this chapter.

A 2.5D-integrated system consists of three main layers: an organic substrate, a silicon interposer, and a chiplet layer. μbumps connect the chiplets and the silicon interposer. Through-silicon vias (TSVs) connect the top and the bottom of the interposer, and C4 bumps connect the interposer and the organic substrate. Epoxy resin is often used to underfill the connection layers (C4 bumps layer and μbumps layer) and the empty spaces between chiplets [160]. Figure 6.1 shows the cross-section view of a 2.5D system in our study.

6.3 Related Work

2.5D integration of smaller chiplets on a large interposer has been demonstrated to achieve a higher compute throughput per watt (or volume) than a single large die [133, 72]. Several related studies have explored the design and optimization of 2.5D systems, with primary focus being placed on individual design layers: logical, physical, and circuit.

At the logical layer, Jerger *et al.* [56] present a hybrid network topology between the cores and memory. They account for different coherence and memory traffic characteristics across applications, and design a hybrid network-on-chip (NoC) that has low latency and high throughput. In their follow-up work, Kannan *et al.* [62] evaluate the impact of different network topologies on 2.5D systems, and demonstrate that disintegration of a large 2D chip into multiple chiplets improves manufacturing yield and lowers costs. However, their work overlooks the μbump overhead. Ahmed *et al.* [1] identify that interposer's routing resources are highly under-utilized due to the high interconnect pitch in 2.5D systems. To maximize performance, they propose a hierarchical mesh network for inter-chiplet communication. Akgun *et al.* [2] perform a design space exploration of different memory-to-core network topologies and routing algorithms. However, a static placement of chiplets in their work limits a complete cross-layer exploration that leaves much of the performance benefits in 2.5D systems untapped. While these works aim to maximize the system performance under different traffic conditions, they do not account for the thermal impact and a complete manufacturing cost model in the NoC design and optimization. In addition, these works do not consider different chiplet placement and link routing options.

At the physical layer, there have been several optimization-based approaches aimed at providing routing and placement solutions for 2.5D systems. Placing chiplets closer to each other results in lower manufacturing cost and higher performance (reduced wirelength), but higher temperature. Therefore, finding a thermally-aware placement and routing solution that maximizes performance and/or minimizes cost is essential in 2.5D systems. Osmolovskyi *et al.* [105] optimize the chiplet placement to reduce the interconnect length using pruning techniques. Ravishankar *et al.* [118] determine the quality of different placement options in a 2D grid using a stochastic model and implement a placer for 2.5D FPGAs. Seemuth *et al.* [124] consider the increased design solution space in 2.5D systems due to flexible I/Os in their

chiplet placement problem. They present a method for die placement and pin assignment using simulated annealing to minimize the total wirelength. Much of the focus of routing in 2.5D systems has been placed on minimizing IR drops and total wirelength in inter-chiplet links [35] and minimizing the number of metal layers [85]. None of these physical layer optimization solutions consider thermal effects.

Prior research at the circuit layer of 2.5D systems generally focuses on link optimization techniques to improve the network and system throughput. Karim *et al.* [63] evaluate the power efficiency of electrical links with and without electrostatic discharge (ESD) capacitance. Stow *et al.* [133] evaluate both repeater and repeaterless links to explore the benefits of active and passive interposers respectively. There have also been efforts on using emerging technologies like wireless links [126] and silicon-photonic links for communication in 2.5D systems [41, 69, 99].

A common drawback among these previous works is that their design and optimization only focus on a single design layer. In contrast, we optimize the cost, performance and temperature by jointly considering the logical, physical and circuit layers of the inter-chiplet network. We evaluate various logical topologies and their feasibilities at the physical and circuit layer. At the physical layer, we design an overlap-free and thermally-safe routing and placement solution that results in the lowest cost and operating temperature. The circuit layer provides us with multiple circuit design options for inter-chiplet links. Our cross-layer methodology, thus, presents a rich solution space to evaluate a variety of network options at different design layers for 2.5D systems, thus enabling accurate and complete modeling of such systems.

6.4 Cross-layer Co-Optimization of Network Design and Chiplet Placement in 2.5D Systems

The ultimate goal of our cross-layer co-optimization methodology is to jointly maximize performance, minimize manufacturing cost, and minimize peak operating temperature. Our methodology comprehends a wide design space across logical, physical and circuit layers, and integrates multiple simulation tools and analytical models that evaluate aspects of system performance, manufacturing cost, interconnect performance, temperature, and routing.

In this section, Section 6.4.1 first introduces the cross-layer co-optimization problem formulation

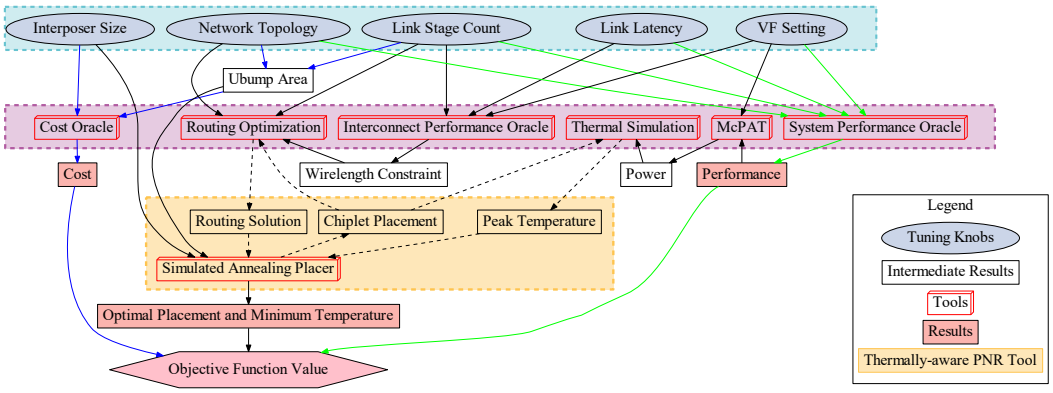


Figure 6.2: Cross-layer co-optimization methodology.

and the methodology we use to solve it. Figure 6.2 shows our cross-layer methodology and provides an outline of upcoming subsections. Section 6.4.2 describes the optimization knobs in the design space across the logical, physical and circuit layers. These knobs form the basis for modeling the 2.5D network and chiplet placement, and enable cross-layer optimization. Section 6.4.3 presents the tools and evaluation framework that models the 2.5D system and evaluates the system metrics of performance, power, temperature and cost. We present five tools that work within the framework to evaluate these system metrics: (1) System Performance Oracle that uses Sniper [11] and McPAT [78]; (2) Cost Oracle that computes the manufacturing cost of the 2.5D system; (3) Interconnect Performance Oracle that uses HSPICE [219] simulations to evaluate the interconnect circuit timing; (4) Thermal Analysis Tool that uses HotSpot [246] to evaluate the temperature; and (5) Routing Optimizer that uses an MILP to solve for the optimal routing solution and the corresponding maximum wirelength. Section 6.4.4 demonstrates the thermally-aware place-and-route (P&R) tool that is based on simulated annealing and interactively uses the oracles described in Section 6.4.3 to explore the chiplet placement solution space to minimize operating temperature and meet routing constraints.

6.4.1 Optimization Problem Formulation and Methodology

Our objective is to jointly maximize performance, minimize manufacturing cost, and minimize peak operating temperature. While minimizing temperature for longer system lifetime, we also maintain

Table 6.1: Notations used in the cross-layer co-optimization methodology.

Notation	Meaning
α, β, γ	Coefficients for the cross-layer objective function.
η	Penalty function weight.
IPS	Instructions per nanosecond as a performance metric.
$Cost$	Manufacturing cost of the 2.5D system.
T	Peak operating temperature of the 2.5D system.
T_{th}	Peak temperature threshold of 85^oC.
L	Maximum wirelength in the routing solution.
L_{th}	Maximum wirelength threshold to meet transmission timing.
w_{int}	Interposer edge width.
w_{2D}	Width of the 2D chip: $18mm$.
w_{ubump}	μbump stretch-out width from original chiplets. Stretch-out width corresponds to the necessary increase of chiplet's dimensions to accommodate the μbumps needed for the off-chiplet communication.
w_{gap}	Minimum gap width between two adjacent chiplets.
X_i, Y_i	Left bottom x- and y-coordinates for chiplet i.

the peak temperature below a threshold to avoid failures. We explore various network topologies, link options (stage count and latency), interposer sizes, frequency and voltage settings, and chiplet placements to find an optimal solution that is routable and thermally-safe. Ensuring that timing is met across the inter-chiplet links is crucial for the design, and the placement and routing have a dramatic impact on closing timing. The temperature threshold is relatively negotiable, as there is usually some headroom between the threshold and the actual temperature that causes rapid failures. Exceeding the temperature threshold (85^oC in our case) by a few degrees would not immediately burn the system, and the impact on system lifetime could be alleviated by applying reliability management techniques that stress different parts of a chip over time. Thus, in the objective function we apply a soft constraint for peak temperature instead of a hard constraint. We use the notations listed in Table 6.1 to formulate our optimization problem

as follows:

Minimize:

$$\alpha \times (\frac{1}{IPS})_{norm} + \beta \times Cost_{norm} + \gamma \times T_{norm} + \eta \times g(T, T_{th}) \tag{6.1}$$

Subject to:

$$g(T, T_{th}) = \frac{1}{10}(max(T - T_{th}, 0))^2 \tag{6.2}$$

$$L \leq L_{th} \tag{6.3}$$

$$w_{int} \leq 50 \tag{6.4}$$

$$max(|X_i - X_j|, |Y_i - Y_j|) \geq \frac{w_{2D}}{4} + 2 \times w_{ubump} + w_{gap}, \forall i, j, i \neq j \tag{6.5}$$

Equation (6.1) is the cross-layer objective function, which jointly maximizes performance (*IPS*) while minimizing manufacturing cost (*Cost*) and peak operating temperature (T). We normalize each term using Min-Max Scaling ($X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$) to reduce the impact of imbalanced ranges and values of raw data. α, β, and γ are user-specified weights having no units, and we set the sum of α, β, and γ to 1. The last term $g(T, T_{th})$ is the penalty function for peak temperature, and η is the penalty weight. It is important to pick an appropriate value for η for a soft-temperature-constrained problem. If η is too small, the optimization problem has no thermal constraint, but if η is too large, the optimization problem effectively becomes a hard-temperature-constrained problem. In our case, we explore a range of η from 0.001 to 1 and pick η to be 0.01, which gives a good balance between not having any constraint and having a hard temperature constraint. Equation (6.2) describes the penalty function. The penalty term is zero when T meets the threshold T_{th}, and positive otherwise. We use a quadratic function instead of a linear function to suppress the penalty for a small violation and highlight the penalty for a large violation. Equation (6.3) is the routing constraint, where the wirelength must be shorter than the reachable length for a given voltage-frequency setting and target latency (see Figure 6.6). Equation (6.4) constrains the interposer size

to be no larger than $50mm \times 50mm$, which is within the exposure field size of 2X JetStep Wafer Stepper [120] and avoids extra stitching cost. Equation (6.5) ensures there is no overlap between chiplets.

To solve the optimization problem, we integrate simulation tools and analytic models discussed in Section 6.4.3. We first generate a complete table of all the combinations of network topologies, inter-chiplet link stage counts and latencies, voltage-frequency settings, and interposer sizes (see Section 6.4.2). We precompute system performance, power, allowable inter-chiplet link length, and manufacturing cost for each entry in the table. We normalize the performance as well as the cost, and compute the weighted sum of the first two terms in the objective function ($\alpha \times (1/IPS)_{norm} + \beta \times Cost_{norm}$), and denote it as $Obj2$, where 2 indicates the number of terms. We then sort the table entries based on the values of $Obj2$ in ascending order. To get the temperature term for each table entry, we build a thermally-aware P&R tool to determine the chiplet placement that minimizes the system operating temperature while meeting the routability requirement (see Section 6.4.4). For our design-time optimization, we assign the worst-case power, which is the highest core power among 256 cores of high-power application *Cholesky*, to all the cores while determining the optimal chiplet placement using our thermally-aware P&R tool. Then, we run real applications on top of the optimal chiplet placement to get the actual application temperature. Our thermally-aware P&R tool iterates chiplet placement, and interactively evaluates peak operating temperature and maximum inter-chiplet wirelength of each placement. Each temperature simulation takes approximately 30 seconds and each routing optimization takes a few seconds to 10 minutes. For manageable simulation time, for each table entry we limit the number of placement iterations to 1000, while determining the minimum peak temperature.

To speed up the simulation, we progressively reduce the number of table entries for which we need to complete the thermally-aware P&R process, which determines the minimum peak temperature and the corresponding chiplet placement for each table entry. Once the process completes for a table entry, all the terms (performance, cost, temperature, and penalty) in the objective function for that table entry become available. We add up the four terms to get the objective function value of the entry, and denote it as $Obj4$, where 4 indicates the number of terms. We keep track of the minimum of the available $Obj4$ values using $Obj4_{min}$. For the entries whose $Obj2$ value is greater than $Obj4_{min}$, there is no need to run the thermally-aware P&R tool, since the tool cannot find a solution whose $Obj4$ value is less than $Obj4_{min}$.

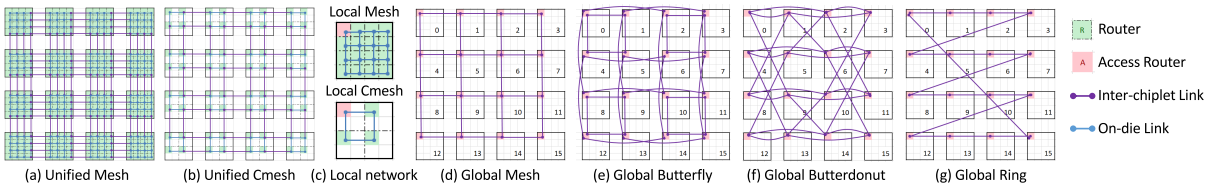


Figure 6.3: Logical view of network topologies. (a)-(b) are unified networks, while (c)-(g) are used to form hierarchical networks.

We start the thermally-aware P&R process with the entries in the sorted order based on $Obj2$ values, progressively removing the entries that have no chance to be optimal, and stop when all the remaining entries have available temperature and $Obj4$ values. Using this technique of progressively reducing solution space, we achieve $6\times$ speedup for the performance-focused case ($(\alpha,\beta,\gamma) = (0.8,0.1,0.1)$), $7.8\times$ speedup for the cost-focused case ($(\alpha,\beta,\gamma) = (0.1,0.8,0.1)$), and $1.5\times$ speedup for the case that jointly focuses on performance, cost, and temperature ($(\alpha,\beta,\gamma) = (0.333,0.333,0.333)$). For the temperature-focused case ($(\alpha,\beta,\gamma) = (0.1,0.1,0.8)$), we only achieve $1.02\times$ speedup because the temperature term dominates, and thus, we can barely rule out any of the table entries using the $Obj2$ and $Obj4_{min}$ comparison. In this chapter, our experiments are based on the performance-focused case.

6.4.2 Cross-layer Optimization Knobs

Logical Layer

One of the main questions in 2.5D logical design is how to connect multiple chiplets using the interposer. In the logical layer, we explore two types of network topologies for 2.5D systems. In Figure 6.3, we show the logical views of network topologies. These views only illustrate the logical connections and not the actual chiplet placement. The first type is a unified network, which directly maps a NoC topology designed for a 2D system onto a 2.5D system to preserve the same logical connections and routing paths. We explore Unified-Mesh (U-M), where each core has a router, and Unified-Cmesh (U-CM), where four cores share a router, as shown in Figure 6.3(a)-(b). Unlike single-chip NoCs, the source and the destination of a logical channel in 2.5D systems may not reside on the same chiplet. The inter-chiplet link has to travel through the silicon interposer, which may not always meet the single-cycle latency due to long physical wires. In our evaluation, we consider inter-chiplet links with latencies varying from single cycle to five

Table 6.2: μbump count, stretch-out width of μbump region (w_{ubump}), and μbump area (A_{ubump}) overhead per chiplet for different network topologies designed using repeaterless links, 2-stage and 3-stage *gas-station* links.

		Unified Mesh	*Unified Cmesh*	*Global Mesh*	*Global Butterfly*	*Global Butterdonut*	*Global Ring*	*Global Clos*
#bidirectional inter-chiplet channels		16	8	4	4	4	2	32
repeaterless links	**#μbumps**	4916	2458	1229	1229	1229	615	9831
	w_{ubump} (mm)	0.54	0.27	0.135	0.135	0.135	0.09	0.945
	A_{ubump} Overhead (%)	53.8	25.4	12.4	12.4	12.4	8.2	101.6
2-stage *gas station*	**#μbumps**	9831	4916	2458	2458	2458	1229	19661
	w_{ubump} (mm)	0.945	0.54	0.27	0.27	0.27	0.135	1.665
	A_{ubump} Overhead (%)	101.6	53.8	25.4	25.4	25.4	12.4	202.8
3-stage *gas station*	**#μbumps**	14746	7373	3687	3687	3687	1844	29492
	w_{ubump} (mm)	1.305	0.72	0.405	0.405	0.405	0.225	2.25
	A_{ubump} Overhead (%)	149.6	74.2	39.2	39.2	39.2	21.0	300.0

cycles.

The second type is a hierarchical network, which breaks down the overall network into two levels: one level has multiple disjoint local networks and the other level has a global network. In 2.5D systems, each chiplet has an on-chip local network and an access router. The global network hooks up all the access routers using inter-chiplet links embedded in the interposer. Intra-chiplet packets travel through the local network, while inter-chiplet packets first travel through the local network to the access router of the source chiplet, then use the global network to reach the access router of the destination chiplet, and finally use the local network of the destination chiplet to reach the destination. The local network and the global network can be designed independently. For local networks, we explore Mesh (M) and Cmesh (CM) topologies (Figure 6.3(c)); while for global networks, we explore Mesh (M), Butterfly (BF), Butterdonut (BD) [62] and Ring (R) topologies, (see Figure 6.3(d)-(g)). We use *G-X-L-Y* notation to denote a hierarchical network, where *X* and *Y* correspond to the global and local network topologies, respectively.

Physical Layer

Physical design of 2.5D systems determines the chiplet placement and a routing solution, subject to the chosen network topology. The placement of chiplets not only impacts the system temperature profile, but also affects the inter-chiplet link lengths. The routing solution affects the μbump assignment and circuit choice of inter-chiplet links. In our approach, we explicitly evaluate the area overhead of μbumps and the inter-chiplet link transceivers that are placed along the peripheral regions of the chiplets.

μbumps connect chiplets and the interposer. Inter-chiplet signals first exit the source chiplet

through μbumps, travel along the wires in the interposer, and then pass through μbumps again to reach the destination chiplet. μbumps are typically placed along the periphery of the chiplet, for the purpose of signal escaping [114]. The μbump area overhead is determined by the number of inter-chiplet channels, channel bandwidth, and μbump pitch. We list the μbump area overhead for various network topologies in Table 6.2, where we use a 128-*bit* wide bus for each channel, $45\mu m$ μbump pitch, and $4.5mm \times 4.5mm$ chiplet size, and assume 20% additional μbumps are reserved for power delivery and signal shielding [114]. Here, w_{ubump} indicates the stretch-out width from the chiplet edge to accommodate the μbumps, as shown in Figure 6.4. In Table 6.2, we also include Global Clos topology [57], which is a commonly used low-diameter-high-radix network. However, the area overhead is too high to make Clos a feasible inter-chiplet network option.

Inter-chiplet links can be routed on either a passive interposer or an active interposer. An active interposer enables better link bandwidth and latency because repeaters and flip-flops (for pipelining) can be inserted in the interposer [110]. However, an active interposer is expensive due to FEOL (front-end-of-line) process and yield loss. A passive interposer is a cost-effective alternative. The passive interposer is transistor-free, can be fabricated with BEOL (back-end-of-line) process, and inherently has high yield [110]. We conducted a study of the performance benefit of an active interposer over a passive interposer. We observed $2\times$ to $3\times$ latency improvement for the same link length, or 50% longer maximum allowed link length for the same throughput, but these benefits come at a $10\times$ cost overhead ($500 per wafer for passive interposer vs. $5000 per wafer for active interposer [110]). Due to this cost overhead, we focus on the passive interposer in our present study. Active interposers, however, are currently being considered for 2.5D systems [56, 62]. Our methodology can be easily extended to active interposers, and we leave this as future work.

Circuit Layer

In the circuit layer, we explore multiple circuit designs for inter-chiplet links. Due to the high cost of an active interposer, we do not consider repeatered links. A link on a passive interposer is naturally repeaterless and non-pipelined. Such a link has limited performance, especially in 2.5D systems, where inter-chiplet links could reach a few *cm*. Essentially, a passive interposer cannot always ensure single-cycle

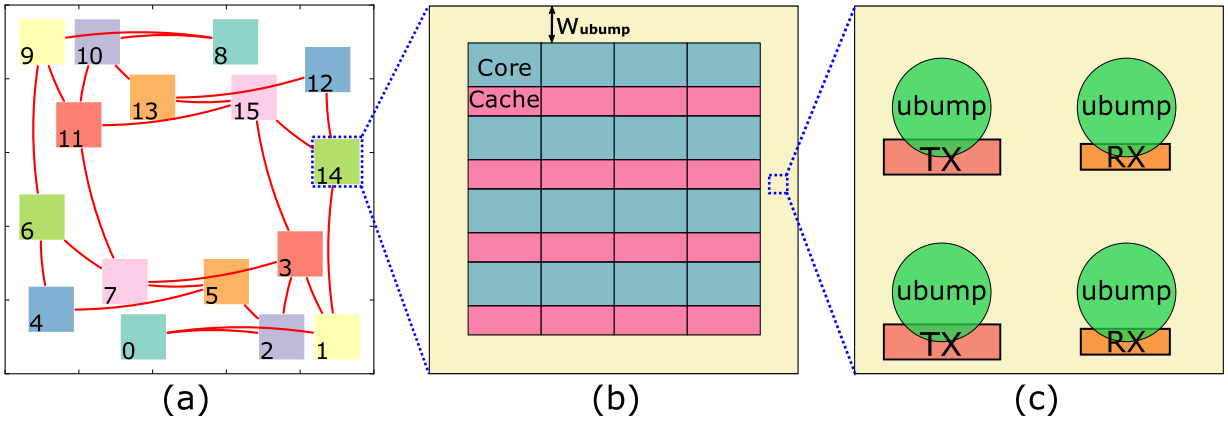


Figure 6.4: Illustration of (a) chiplet placement on an interposer with logical connections, (b) a chiplet with μbump overhead, and (c) μbumps with TX/RX regions (not to scale).

communication latency due to signal degradation and rise-/fall-time constraints. Hence, we explore a range of repeaterless inter-chiplet link (Path 1 in Figure 6.5) latencies from single cycle to five cycles, which corresponds to a variety of inter-chiplet link lengths (see Figure 6.6). This provides sufficient flexibility in chiplet placement. In addition, we use a novel '*gas-station*' link design [25], which enables pipelining in a passive interposer, to overcome the performance loss. Our '*gas-station*' link leverages flip-flops placed on other chiplets along the way to 'refuel' a passive link. As shown in Figure 6.5, Chiplet #2 is a *gas station* for Path 2 from Chiplet #1 to Chiplet #3, where signals first enter Chiplet #2 through μbumps, get repeated or retimed, and then return to the passive interposer through μbumps. Here we trade off μbump area overhead computed in Table 6.2 for performance. It is important to note the differences between an inter-chiplet repeaterless pipelined link and a *gas-station* link [25]. A repeaterless pipelined link requires an active interposer to house flip-flops and these flip-flops are designed using the active interposer's technology node. A *gas-station* link only needs a passive interposer and inserts active elements in the intermediate chiplets. Thus, the active elements are designed using the chiplets' technology node (22*nm* in our case). In our analysis, we set t_{rise}/t_{cycle} upper bound to be 0.5 and ensure full voltage swing at all nodes in the inter-chiplet link to account for non-idealities such as supply noise and jitter. We also explore t_{rise}/t_{cycle} of 0.8, which allows signals to go longer distances without repeaters. Relaxing the clock period or allowing for multi-cycle bit-periods permits us to use longer inter-chiplet links.

Figure 6.5(c) and (d) show the distributed circuit models in a passive interposer for repeaterless

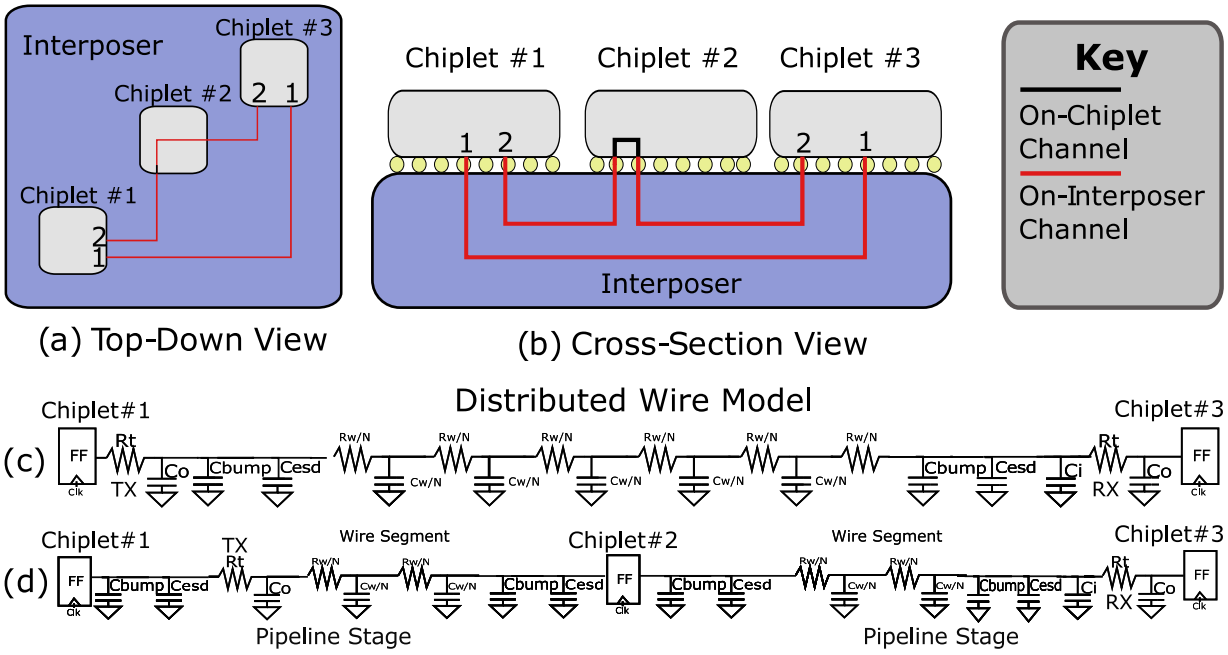


Figure 6.5: Illustration of (a) top-down view and (b) cross-section view of inter-chiplet link implementation, and distributed wire models for (c) repeaterless link (Path 1 in (a)-(b)) and (d) *gas-station* link (Path 2 in (a)-(b)).

Table 6.3: Technology node parameters.

Technology Node	**22nm**	**65nm**
Wire Thickness	$300nm$	$1.5\mu m$
Dielectric Height	$300nm$	$0.9\mu m$ [63]
Wire Width	$200nm$	$1\mu m$ [114]
C_{bump}	$4.5fF$	$4.5fF$ [63]
C_{esd}	$50fF$	$50fF$ [63]
$C_{g\perp}$ (Gate Cap)	$1.08fF/\mu m$	$1.05fF/\mu m$
$C_{d\perp}$ (Drain Cap)	$1.5 \times C_g$	$1.5 \times C_g$
R_t (Inverter resistance)	$450\Omega \cdot \mu m$	$170\Omega \cdot \mu m$
Driver NMOS Sizing	$22nm \times 100$	$65nm \times 100$
Wire Pitch	$0.4\mu m$	$2\mu m$ [114]
Flip-Flop Energy per Bit	$14fJ/bit$ [18]	$28fJ/bit$ [211]
Flip-Flop $t_{c-q} + t_{setup}$	$49ps$ [18]	$70.9ps$ [23]

link and *gas-station* link, respectively. We model wire parasitics using a distributed, multi-segment π model. We use $22nm$ technology parameters for intra-chiplet components (drivers, receivers, repeaters, and flip-flops) and $65nm$ parameters for the inter-chiplet wires. Table 6.3 shows technology parameters used in our experiments. We calculate capacitance and resistance based on the model in Wong *et al.* [150], and we

calibrate our stage and path delay estimates based on extraction from layout and Synopsys PrimeTime timing reports. Figure 6.6 shows maximum reachable wirelengths that meet both the propagation time constraint and the rise-time constraint for various frequencies and cycles. For a given rise time constraint, as the inter-chiplet link latency constraint increases, the distance that a signal can travel in a single cycle increases. In a single cycle, a signal can travel more than 10*mm* owing to the relaxed rise time constraint as well as low interconnect RC parasitics (i.e., due to using an older technology node for the interposer).

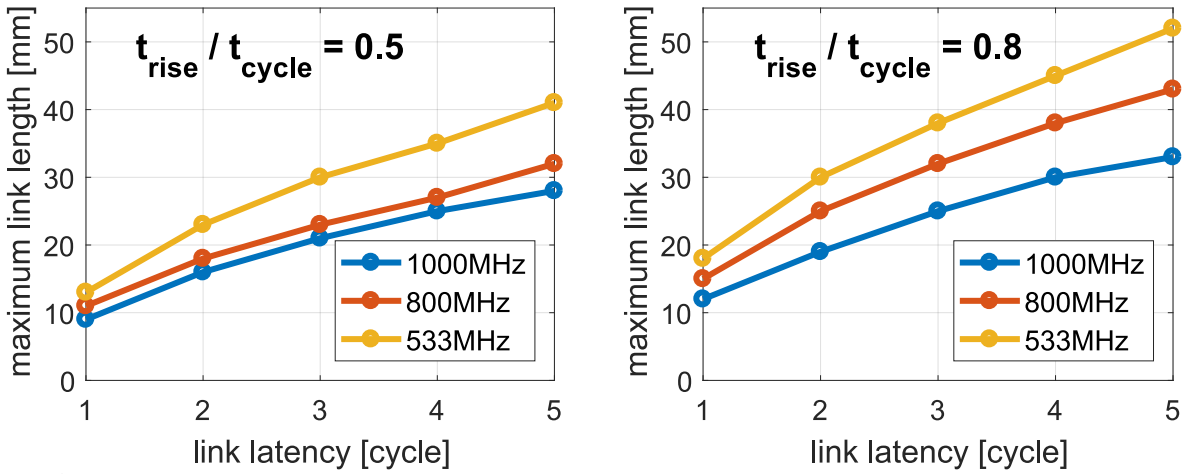


Figure 6.6: Maximum reachable inter-chiplet link length with respect to clock cycles for various frequencies and rise-time constraints.

6.4.3 Evaluation Framework

System Performance Oracle

We construct a manycore system performance oracle that tells us the manycore system performance and core power for a given choice of network topology, voltage-frequency setting, link type, and link latency. We use Sniper [11] to precompute system performance. Our target system has 256 homogeneous cores, whose architecture is based on the IA-32 core from the Intel Single-Chip Cloud Computer (SCC) [51], with size and power scaled to 22*nm* technology [159]. We divide the 256-core system into 16 identical chiplets.[9] In Sniper, we implement the unified and hierarchical network models described in Section 6.4.2. For inter-chiplet links, we use either passive links or *gas-station* links (see Section 6.4.2). We vary link latency from one to five cycles for passive links and explore 2-stage and 3-stage pipelines for *gas-station* links. We

[9]Our methodology is applicable to any system with even number of chiplets, each with aspect ratio of 1.

explore three voltage-frequency settings: $(0.9V, 1GHz)$, $(0.89V, 800MHz)$, and $(0.71V, 533MHz)$. We use multi-threaded benchmarks that cover high-power applications (*Cholesky* from SPLASH-2 suite [152]), medium-power applications (*Streamcluster* and *Blackscholes* from PARSEC suite [110]), and low-power applications (*Lu.cont* from SPLASH-2 suite). We fast-forward the sequential initialization region and simulate 10 billion instructions in the parallel region with all cores active to collect performance statistics. Then, we feed the performance results to McPAT [78] to compute the core power. We calibrate the McPAT power output with the measured power dissipation data of Intel SCC [51], scaled to $22nm$.

Cost Oracle

We construct a cost oracle that computes the manufacturing cost of 2.5D systems for a given choice of network topology, chiplet size and count, link type and stage count, and interposer size. We adopt the 2.5D manufacturing cost model published by Stow [133], which takes into account the cost and yield of CMOS chiplets, μbump bonding, and the interposer. The model assumes known-good-dies. We enhance the cost model to account for the impact of μbump overhead on the dies per wafer count and yield.

$$A_{chiplet} = (\frac{w_{2D}}{4})^2 \tag{6.6}$$

$$A_{ubump} = (\frac{w_{2D}}{4} + 2 \times w_{ubump})^2 - A_{chiplet} \tag{6.7}$$

$$N_{int} = \frac{\pi \times (\phi_{wafer_{int}}/2)^2}{A_{int}} - \frac{\pi \times \phi_{wafer_{int}}}{\sqrt{2 \times A_{int}}} \tag{6.8}$$

$$N_{chiplet} = \frac{\pi \times (\phi_{wafer}/2)^2}{A_{chiplet} + A_{ubump}} - \frac{\pi \times \phi_{wafer}}{\sqrt{2 \times (A_{chiplet} + A_{ubump})}} \tag{6.9}$$

$$Y_{chiplet} = (1 + (A_{chiplet} + A_{TXRX}) \times D_0/\varepsilon)^{-\varepsilon} \tag{6.10}$$

$$C_{int} = C_{wafer_{int}}/N_{int}/Y_{int} \tag{6.11}$$

$$C_{chiplet} = C_{wafer}/N_{chiplet}/Y_{chiplet} \tag{6.12}$$

$$C_{2.5D} = \frac{C_{int} + \Sigma_1^{16}(C_{chiplet} + C_{bond})}{Y_{bond}^{15}} \tag{6.13}$$

Equation (6.6) (see Table 6.4 for all notations) computes the equivalent functional area of chiplets

Table 6.4: Notations used in the cost oracle.

Notation	Meaning
A_{int}	Area of interposer.
$A_{chiplet}$	Chiplet area without μbump overhead.
A_{ubump}	Area of μbump region in a chiplet.
A_{TXRX}	Critical transceiver area in μbump region.
ϕ_{wafer}	Diameter of CMOS wafer: 300mm.
$\phi_{wafer_{int}}$	Diameter of interposer wafer: 300mm.
N_{int}	Number of interposer dies per wafer.
$N_{chiplet}$	Number of CMOS dies per wafer.
D_0	Defect density: $0.25/cm^2$ [132].
ε	Defect clustering parameter: 3 [132].
$Y_{chiplet}$	Yield of a CMOS chiplet.
Y_{int}	Yield of an interposer: 98% [242].
Y_{bond}	Chiplet bonding yield: 99% [132].
C_{wafer}	Cost of CMOS wafer.
$C_{wafer_{int}}$	Cost of passive interposer wafer.
$C_{chiplet}$	Cost of a chiplet.
C_{int}	Cost of an interposer.
C_{bond}	Cost of chiplet bonding.
$C_{2.5D}$	Manufacturing cost of a 2.5D system.

generated by dividing a 2D chip. Equation (6.7) evaluates the μbump area overhead. Equations (6.8) and (6.9) determine the number of interposer dies and the number of CMOS dies, respectively, that can be cut from a wafer [133]. Here the first term counts the number of dies purely based on the wafer area and the die area, and the second subtraction term compensates for incomplete dies along the wafer periphery. In Equation (6.9), we take into account the μbump area overhead A_{ubump}. Equation (6.10) is the negative binomial yield model, where D_0 is the defect density and $\varepsilon = 3$ indicates moderate defect clustering [133]. Unlike the center area of chiplets that has high transistor density, the μbump regions have very limited active regions that contain inter-chiplet link transmitters (TXs) and receivers (RXs). Only the defects occurring in the active regions would cause a failure, while the rest of the passive region is non-critical. Hence, our yield calculation (Equation (6.10)) uses only the critical active area. The yield of a passive interposer is as high as 98% [242] because it does not have any active components. Equations (6.11) and (6.12) calculate the per-die cost of the interposer and the chiplets, respectively. Equation (6.13) estimates the overall manufacturing cost of the 2.5D system by adding up the costs of the chiplets, the interposer,

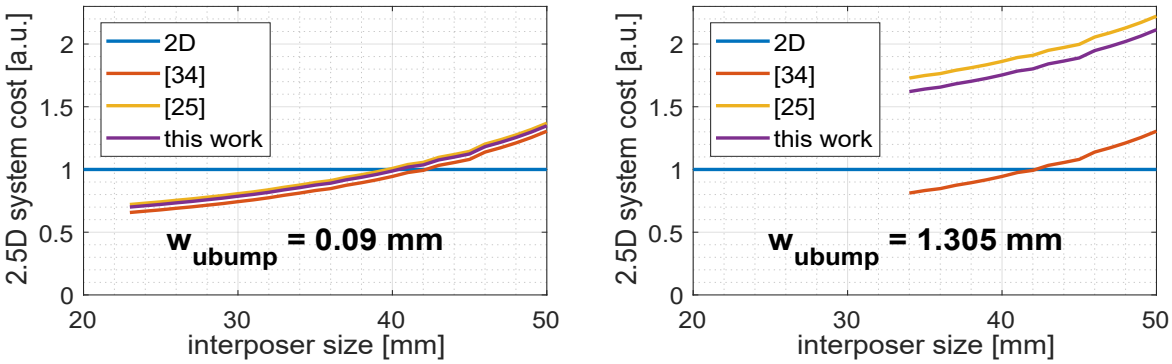


Figure 6.7: Comparison between the cost of a 2D system, and the cost of a 2.5D system estimated using prior cost models [34, 25] and our enhanced cost model for interposer sizes from 20*mm* to 50*mm* and μbump stretch-out widths (w_{ubump}) of 0.09*mm* and 1.305*mm*, which correspond to the lower and upper limits of w_{ubump} in our analysis, respectively.

and bonding.

Figure 6.7 shows the manufacturing cost of 2.5D systems with respect to interposer sizes from 20*mm* to 50*mm* for two different μbump stretch-out widths, which correspond to the minimum value (for *G-R-L-M/CM* topology without *gas stations*) and maximum value (for *U-M* topology with 3-stage *gas-station* links) in our experiments. The 2.5D system costs are normalized to the cost of 2D system. The 2.5D system cost increases with the interposer size. The cost model in our prior work [34] did not consider μbump overhead and thus, the 2.5D system cost is independent of w_{ubump}. The cost model in our latest work [25] overestimated the yield drop due to μbump regions and thus, overestimated the overall cost. This error of this cost model [25] is trivial with a small w_{ubump}, but with a large w_{ubump}, the error is not negligible (up to 10% of the 2D system cost in our example). With a small w_{ubump}, the predicted cost of a 2.5D system using our enhanced model is cheaper than the cost of a 2D system, when the interposer is smaller than $40mm \times 40mm$. With a large w_{ubump}, the predicted cost of a 2.5D system using our enhanced model is always higher than that of a 2D system. This eliminates some network topologies, such as Clos, that require large w_{ubump}.

Interconnect Performance Oracle

We build an interconnect performance oracle that analyzes the maximum reachable length of inter-chiplet link for a given operating voltage and frequency, rise-time constraint, and propagation time constraint in the unit of cycles. We use HSPICE [219] to simulate the link models discussed in Section 6.4.2. The TX circuit is designed using up to six (the exact number depends on the wirelength) cascaded inverters with standard fan-out of 4, and the RX circuit consists of two cascaded inverters of the minimum size. We estimate the TX and RX area using the physical layout of the standard inverter cell in NanGate 45nm Open Cell Library [211], and scale it down to 22nm technology. The area of TX and RX logic (A_{TXRX}) takes up less than 1% of the μbump area. The interposer wire resistance is $14.666 \times 10^{-3} \Omega/\mu m$ and the capacitance is $114.726 \times 10^{-3} fF/\mu m$, for the wire dimensions provided in Table 6.3 for 65nm technology. Since the inter-chiplet link latency is wire dominated, we set a sizing upper limit of 100$\times$ the minimum size for the last inverter in the set of cascaded inverters of TX in 22nm technology since the drivers are placed in chiplets instead of the interposer. We do not increase the size beyond 100$\times$ because we do not observe latency improvement. For the workloads that we have considered, the inter-chiplet link power is up to 22W, which is insignificant compared to the total average system power of 508W. Hence, inter-chiplet link power has negligible influence on chiplet placement.[10]

Thermal Simulation

We use HotSpot [246] to simulate thermal profiles for given chiplet placement choices and core power values. We use an extension of HotSpot [94] that provides detailed heterogeneous 3D modeling features. To model our 2.5D system, we stack several layers of different thickness and heterogeneous materials on top of each other and model each layer with a separate floorplan on a 64×64 grid. Our 2.5D system model follows the properties (such as layer thickness, materials, dimensions of bumps and TSVs) of real systems [16, 14]. We use the HotSpot default conventions for the thermal interface material properties, the ambient temperature of 45^oC, and the sizing of the spreader and the heatsink such that the spreader edge size is 2$\times$ the interposer edge size and the heatsink edge size is 2$\times$ the spreader edge size. To keep

[10] If link power were to increase substantially, this would affect the system temperature, which in turn would affect the chiplet placement.

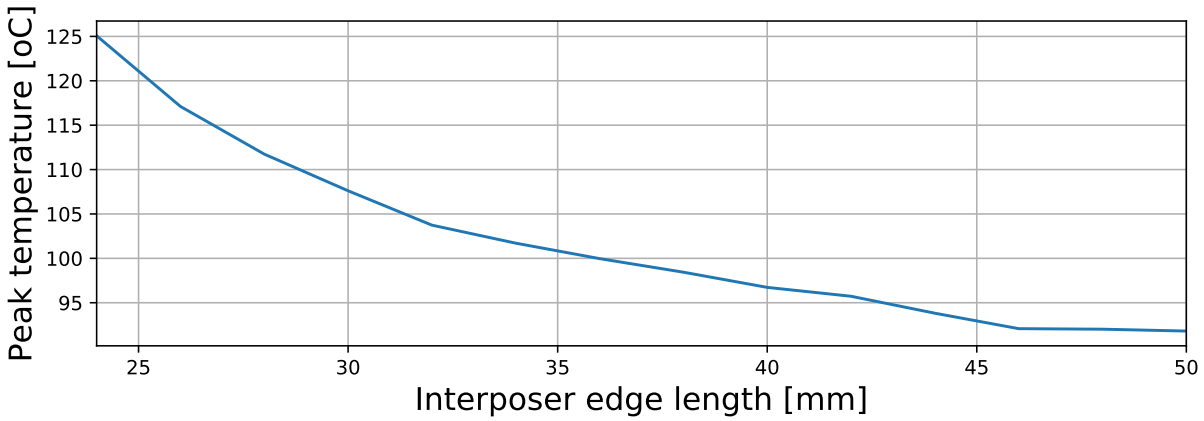


Figure 6.8: Temperature of best chiplet placement for each interposer size, running *Cholesky* with *Mesh* network using single-cycle link without *gas stations*.

the heat transfer coefficient consistent across all simulations, we adjust the convective resistance of the heatsink.

We implement a linear model of temperature-dependent leakage power based on published data of Intel 22*nm* processors [245]. We assume 30% of power is due to leakage at 60^oC [159]. We update the core power to include the leakage power based on initial temperature obtained from HotSpot and iterate the thermal simulation. In all of our studies, the leakage-dependent temperature quickly converges after two iterations.

Figure 6.8 shows the temperature of the best chiplet placement for each interposer size, while running *Cholesky* benchmark with *Mesh* network using single-cycle links without *gas stations*. As the interposer size increases, the peak temperature decreases due to the increasing flexibility of chiplet placement. Although the main direction of heat dissipation is vertical through the heatsink on top of the system and the lateral heat transfer is relatively weak, the effect of lateral heat flow is sufficient to motivate thermally-aware chiplet placement [161]. The temperature benefit shown in Figure 6.8 comes at the cost of a larger interposer. The cost of the interposer has been accounted in our cost model and the user can adjust the cost weight in the objective function for different design needs.

Table 6.5: Notations used in routing optimization.

Notation	Meaning
C	Set of chiplets.
P	Set of pin clumps.
N	Set of nets.
c, i, j	Index of a chiplet $\in C$.
p, h, k	Index of a pin clump $\in P$.
n	A net $\in N$.
s_n	Source chiplet of net n.
t_n	Sink chiplet of net n.
x_p, y_p	x- and y-offsets from left bottom of the chiplet for pin clump p.
d_{ihjk}	Distance from pin clump h on chiplet i to pin clump k on chiplet j. Note that $d_{ihjk} = d_{jkih}$.
P_{ih}^{max}	Pin capacity for a pin clump h on chiplet i.
R_{ij}	Input requirement on the wire count between chiplet i and chiplet j.
f_{ihjk}^{n}	Flow variable. Number of wires from pin clump h of chiplet i to pin clump k of chiplet j that belong to net n.
λ_{ihjk}^{n}	Binary indicator for a route between pin clump h on chiplet i to pin clump k on chiplet j belonging to net n.
S_{max}	Maximum permissible segment count allowed for any route; a segment is defined as a route between chiplets. For the case where no *gas stations* are permitted, $S_{max} = 1$. Permitted values of S_{max} include 1, 2 or 3.
θ, φ	Coefficients for the objective function of routing optimization.

Routing Optimization

We build an MILP to solve for the optimal routing solution and the corresponding maximum wirelength given the logical network topology, chiplet placement, link stage count, and μbump resources. The MILP objective is a weighted function of the maximum length of a route on the interposer and the total routing area overhead. We group the μbumps along the chiplet periphery into pin clumps to limit the problem size and the MILP runtime. We use 4 pin clumps per chiplet in our experiments. We frame the delivery of required number of wires between chiplets as multi-commodity flow, and formulate the MILP to find optimal routing solutions that comprehend the finite availability of μbumps in each pin clump.

Table 6.5 describes the notations used in the MILP. We use ILOG CPLEX v12.5.1 to implement and run the MILP. The number of variables and constraints in the MILP instance are both bounded by $O(|C|^2 \cdot |P|^2 \cdot |N|)$. For our 16-chiplet design, $|N|$ is 48 for Mesh/Cmesh, 56 for Butterdonut, 64 for Butterfly and 32 for Ring networks. The outputs of our MILP implementation are the optimal value of the

objective function and the values of the variables f^n_{ihjk}, which describe the routing solution and μbump assignment to pin clumps.

Based on the inputs to the routing optimization step (see Table 6.6), we precompute d_{ihjk}, the routing distance (assuming Manhattan routing) from pin clump h on chiplet i to pin clump k on chiplet j, using Equation (6.14). Equation (6.15) is the objective function for the MILP that includes the maximum length L, and the total length of the routes. In all reported experiments, we set $\theta = 1$ and $\varphi = 0$. Equation (6.16) ensures that the flow variable f^n_{ihjk} is a non-negative number. Equation (6.17) is the flow constraint governing the flow variables f^n_{ihjk}. It guarantees the sum of all flows for a net n, over all pin clumps from chiplet s_n to chiplet t_n, meets the R_{ij} requirement. It also makes sure that net flow is 0 for all other (non-source, non-sink) chiplets for the given net. $\sum_{h\in P, j\in C, k\in P} f^n_{ihjk}$ is the outgoing flow of chiplet i, while $\sum_{h\in P, j\in C, k\in P} f^n_{jkih}$ is the incoming flow of chiplet i. Equation (6.18) assures that there is no input flow (for net n) for any pin clump in the source chiplet s_n from any other chiplet's pin clump. Similarly, Equation (6.19) ascertains that there is no output flow (for net n) for any pin clump in the sink chiplet t_n to any other chiplet's pin clump. Equation (6.20) maintains that the sum of input and output flows from a given pin clump is always less than or equal to the capacity of the pin clump. This ensures that all routes have available pins. Equation (6.21) defines λ^n_{ihjk} as a boolean value based on f^n_{ihjk}. This helps identify the maximum route length L, as shown in Equation (6.22). Equation (6.23) constrains the maximum number of segments (S_{max}) to be either 1, 2 or 3. A segment is defined as a portion of the net connecting two chiplets. If $S_{max} = 1$, then the net connects s_n and t_n directly, and no *gas stations* are permitted, while if $S_{max} = 2$ or $S_{max} = 3$, then *gas stations* are permitted, where the net connects s_n and t_n *through* 1 or 2 other chiplets respectively, i.e., *gas station* hops.

$$d_{ihjk} = \left|X_i + x_h - X_j - x_k\right| + \left|Y_i + y_h - Y_j - y_k\right| \tag{6.14}$$

Table 6.6: Inputs to routing optimization.

Input	Properties
Chiplets	$\|C\|$ Chiplet instances, at $\{X_c, Y_c\}$ left bottom, $c \in C$. The locations provided for the chiplets are assumed to be legal.
Pin Clumps	$\|P\|$ Pin clump instances of pin capacity P_{ih}^{max} each. Each pin clump p has a predetermined location $\{x_p, y_p\}$ relative to the left bottom of the chiplet.
Required Connections	R_{ij} between every pair of chiplets $\{i, j\}$ indicating the number of wires that need to go between the pair of chiplets. If $R_{ij} > 0$ then a net n exists between chiplet i and chiplet j with source $s_n = i$ and sink $t_n = j$.
Routing Rules	Maximum number of segments, S_{max} equal to 1, 2 or 3. $S_{max} \leq 3$ to limit impact on latency.

Minimize:

$$\theta \cdot L + \varphi \cdot \sum_{i \in C, h \in P, j \in C, k \in P, n \in N} d_{ihjk} \cdot f_{ihjk}^{n} \tag{6.15}$$

Subject to:

$$f_{ihjk}^{n} \geq 0, \qquad \forall i \in C, h \in P, j \in C, k \in P, n \in N \tag{6.16}$$

$$\sum_{h \in P, j \in C, k \in P} f_{ihjk}^{n} - \sum_{h \in P, j \in C, k \in P} f_{jkih}^{n} = \begin{cases} R_{s_n t_n}, & \text{if } i = s_n, \forall n \in N \\ -R_{s_n t_n}, & \text{if } i = t_n, \forall n \in N \\ 0, \forall i \neq s_n || t_n, \forall n \in N \end{cases} \tag{6.17}$$

$$f^n_{jks_nh} = 0, \qquad \forall n \in N, \forall h \in P, \forall j \in C, \forall k \in P \tag{6.18}$$

$$f^n_{t_nhjk} = 0, \qquad \forall n \in N, \forall h \in P, \forall j \in C, \forall k \in P \tag{6.19}$$

$$\sum_{j\in C,k\in P,n\in N} f^n_{ihjk} + \sum_{j\in C,k\in P,n\in N} f^n_{jkih} \leq P^{max}_{ih}, \;\; \forall i \in C, h \in P \tag{6.20}$$

$$\lambda^n_{ihjk} = \begin{cases} 1 & \text{if } f^n_{ihjk} > 0, \forall i \in C, h \in P, j \in C, k \in P, n \in N \\ 0 & \text{otherwise}, \forall i \in C, h \in P, j \in C, k \in P, n \in N \end{cases} \tag{6.21}$$

$$L \geq d_{ihjk} \cdot \lambda^n_{ihjk}, \forall i \in C,\ h \in P,\ j \in C,\ k \in P,\ n \in N \tag{6.22}$$

$$\sum_{i\in C,h\in P,j\in C,k\in P} f^n_{ihjk} \leq \begin{cases} R_{s_nt_n}, \text{ if } S_{max} = 1 \\ 2 \cdot R_{s_nt_n} - \sum_{h\in P,k\in P} f^n_{s_nht_nk}, \text{ if } S_{max} = 2 \\ 3 \cdot R_{s_nt_n} - 2 \cdot \sum_{h\in P,k\in P} f^n_{s_nht_nk} - \\ \sum_{i\in C|i\neq s_n||t_n} min(\sum_{h\in P,k\in P} f^n_{s_nhik}, \\ \sum_{h\in P,k\in P} f^n_{ikt_nh}), \text{if } S_{max} = 3 \end{cases} \tag{6.23}$$

6.4.4 Thermally-Aware Placement Algorithm

Our thermally-aware P&R tool supports arbitrary chiplet placements that consider non-matrix and asymmetric chiplet organization styles while searching for the optimal placement for each table entry. Including arbitrary placements, the solution space explodes to quadrillions (10^{15}) placement options with $1mm$ granularity. It is impractical to exhaustively search such a vast space. In addition, the solution space is non-convex. Approaches like gradient descent or greedy search [34] can easily get trapped in a local

minima. Therefore, we use simulated annealing to explore chiplet placement and find the optimal placement solution that gives lowest peak temperature while meeting the maximum wirelength. Simulated annealing is a probabilistic technique to approximate the global optimum. We introduce the key components of our algorithm below.

Placement Description. Prior works [34, 25] only consider 4×4 matrix-style chiplet placement, which covers a small portion of the overall solution space and the chiplets have limited freedom to move. For example, the corner chiplets cannot move, the edge chiplets can only slide along the periphery of the interposer, and the center chiplets can only slide along the interposer diagonal. Thus, the previous approach of matrix-style chiplet placement cannot cover the cases where the four chiplets along an edge of the interposer do not align or the cases where the first row does not always have four chiplets. In addition, the previous assumption of 4-fold rotational symmetry does not allow us to ever find the optimal placement for some topologies. For Butterdonut and Butterfly networks, because of the 4-fold rotational symmetry, the maximum wirelength cannot be shortened with chiplet movement due to the connection between a chiplet and its reflection in any one of the remaining quadrants. Therefore, we enhance our cross-layer co-optimization methodology to support arbitrary placement and relax our symmetry assumption to 2-fold rotational symmetry. We use x- and y-coordinates to specify the locations of the first eight chiplets, and the coordinates of the remaining eight chiplets are based on the rotational image of the first eight. We assume $1mm$ granularity for placement, such that the coordinates of the center of each chiplet has to be positive integer numbers. The chiplets cannot overlap with each other and there is a $1mm$ guardband along the interposer periphery. The minimum gap between two chiplets is $0.1mm$ [16, 98].

Neighbor Placement. A neighbor placement is the placement obtained by either moving a chiplet by the minimum step size in any of the 8 directions (N, S, E, W, NE, NW, SE, SW) or swapping a pair of chiplets from a current placement. Without swapping, it is likely to have a 'sliding tile puzzle' issue. For instance, a chiplet cannot move in some directions because other chiplets block the way, especially when the interposer size is small.

Acceptance Probability. The decision of whether a neighbor placement is accepted or not depends on the $delta$ calculated using Equation (6.24). Here T_{curr}, L_{curr}, T_{nei}, L_{nei} are the peak temperature of current placement, the longest wirelength of current placement, the peak temperature of neighbor

placement, and the longest wirelength of neighbor placement, respectively. When both the current placement and the neighbor placement meet the wirelength constraint, we emphasize the temperature difference when calculating $delta$. Similarly, when either the neighbor or the current placement violates the wirelength constraint, we emphasize the wirelength difference while calculating $delta$ as there is no point in considering temperature because we do not have a viable solution. We compute the acceptance probability AP using Equation (6.25), where K is the annealing temperature. Here K decays from 1 to 0.01 with a factor of 0.8 every v iterations, where v is proportional to the interposer edge width w_{int}. We accept the neighbor placement if AP is greater than a random number between 0 and 1. In the case that a neighbor placement is better ($delta > 0$), AP evaluates to greater than 1 and we are forced to accept the neighbor placement. In the case that a neighbor placement is worse ($delta < 0$ and $0 < AP < 1$), there is still a nonzero probability of accepting the worse neighbor placement to avoid being trapped in a local minima. The worse a neighbor placement is, the lower is the probability of accepting it. As the annealing temperature K decays, the solution converges since the probability of accepting a worse neighbor placement decreases.

$$delta = \begin{cases} 0.9 \times (T_{curr} - T_{nei}) + 0.1 \times (L_{curr} - L_{nei}), \\ \qquad if\ L_{curr} \leq L_{th}\ and\ L_{nei} \leq L_{th} \\ 0.1 \times (T_{curr} - T_{nei}) + 0.9 \times (L_{curr} - L_{nei}), \\ \qquad if\ L_{curr} > L_{th}\ or\ L_{nei} > L_{th} \end{cases} \tag{6.24}$$

$$AP = e^{\frac{delta}{K}},\ accept\ if\ AP > rand(0,1) \tag{6.25}$$

Multi-Start and Multi-Phase Techniques. As a probabilistic algorithm, simulated annealing approximates the global minimum but provides no guarantee to find it. It is also challenging to find a good enough solution due to the astronomical non-convex solution space (up to quadrillions of placement options) and the limited simulation time (up to a thousand moves). In order to improve the solution quality of simulated annealing, we adopt multi-start and multi-phase techniques. For multi-start, we repeat the thermally-aware P&R process ten times for each table entry and pick the placement solution which has the

lowest peak temperature and meets the routing constraint. Given the probabilistic nature of the simulated annealing algorithm, the multi-start technique is helpful in reducing the chance of getting a poor solution. We can run the multiple starts of the multi-start technique in parallel, so as not to increase the time required to arrive at the solution. For multi-phase, we map an existing placement solution of a smaller interposer to a larger interposer (while keeping all the other tuning knobs the same) and use it as the initial starting placement to find the placement solution for the larger interposer. This improves the quality of the final placement solution for a table entry without increasing the simulation time or the electricity bill. The multi-phase step size must be a multiple of $2mm$ since we assume $1mm$ placement granularity. A smaller step size yields better solution quality, but requires longer actual simulation time. In our case, we set the multi-phase step size to $4mm$, which provides a good balance between the simulation time and the solution quality.

6.5 Evaluation Results

In this section, we first provide the maximum performance and the optimal chiplet placement for various networks. We compare the maximum performance using our new approach against the prior work [25], with and without *gas stations*. Next, we present the iso-cost performance improvement, the iso-performance cost reduction using our new approach, and the Pareto Frontier curve of performance and cost. We then show the thermal maps for high-power, medium-power, and low-power applications on their respective optimal chiplet placement solution. In addition, we evaluate the running of medium-power and low-power applications on the optimal chiplet solution for a high-power application. Lastly, we conduct a sensitivity analysis to show the optimal combinations of performance, cost and peak temperature with respect to different temperature thresholds and different choices of constraints.

6.5.1 Optimal Chiplet Placement Analyses

Figure 6.9 shows the maximum performance, the corresponding cost and the corresponding peak operating temperature for various networks and link designs running the high-power *Cholesky* benchmark for three different approaches. Here the focus is on performance. The first approach corresponds to our

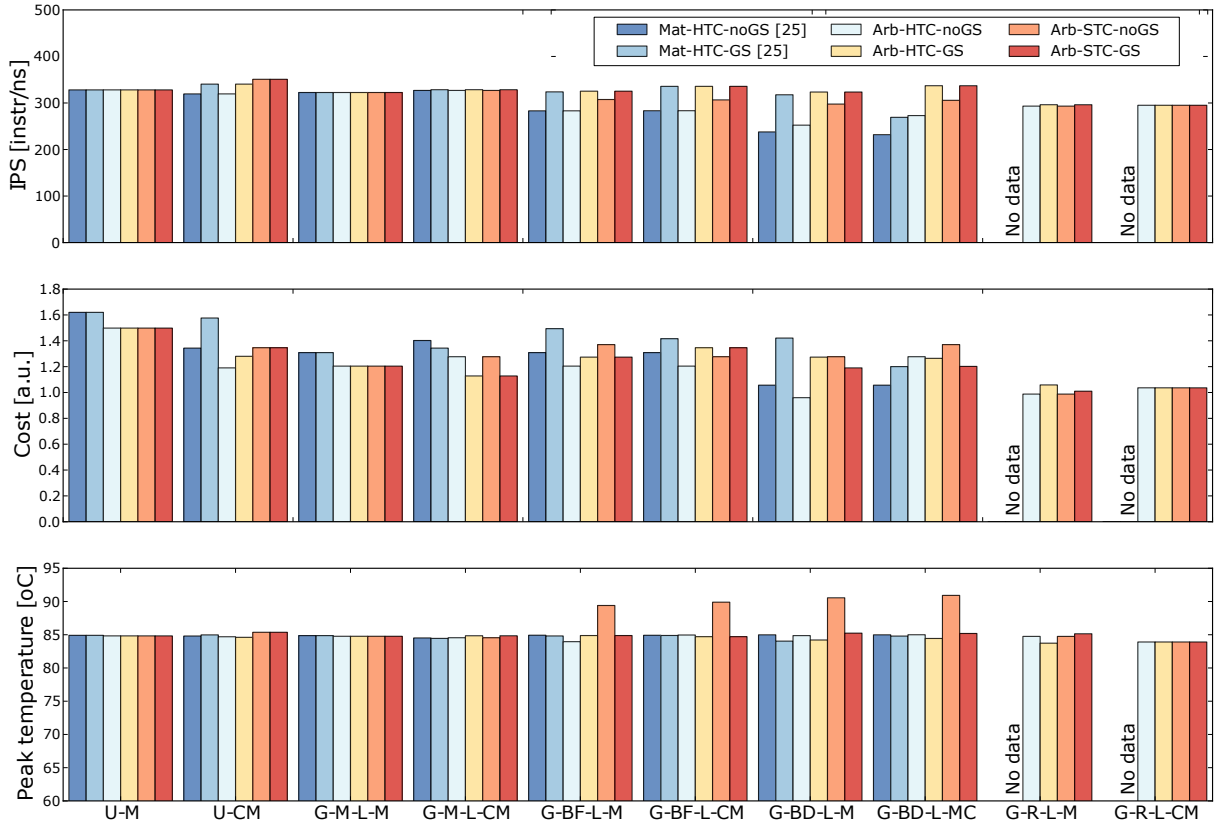


Figure 6.9: Maximum performance, the corresponding cost and the corresponding peak temperature for various networks with and without *gas-station* links when running *Cholesky* benchmark. Here the optimization goal is to maximize performance; the cost values are normalized to the cost of a 2D system.

prior work [25] that only considers matrix-style chiplet placement (*Mat*) and a hard temperature constraint (*HTC*) of 85^oC, with and without *gas stations*. We use *Mat-HTC-GS* and *Mat-HTC-noGS* to denote these cases. The second approach uses the same *HTC* of 85^oC but allows arbitrary placement of chiplets (*Arb*). We use *Arb-HTC-GS* and *Arb-HTC-noGS* to denote these cases. The third approach uses a soft temperature constraint (*STC*) of 85^oC and arbitrary placement, as described in Section 6.4.4. We use *Arb-STC-GS* and *Arb-STC-noGS* to denote these cases.

For the mesh-like networks (*G-M-L-M*, *G-M-L-CM*, *U-M*, and *U-CM*), our *Arb-HTC* approach does not improve the performance over the previous *Mat-HTC* approach [25]. This is because the previous approach already achieves the maximum performance for *G-M-L-M*, *G-M-L-CM*, and *U-M*, while for *U-CM*, there is not much room for improvement with arbitrary placement since the optimal placement also follows a matrix style. However, we achieve a 8-19% (11% on average) reduction in cost. The *Arb-STC*

approach achieves the highest performance (10% improvement) with *U-CM* network at a manufacturing cost which is equal to the *Mat-HTC-noGS* case, while exceeding the temperature threshold by less than 0.5^oC. For the remaining three mesh-style networks, the *Arb-STC* approach does not improve performance but it does reduce cost in some cases. Even when using our thermally-aware P&R tool with the option of arbitrary placement, the optimal chiplet placements are matrix style. Since these four mesh-like networks have similar optimal placement patterns, we just show the logical connection and thermal map of *U-CM* network in Figure 6.10(a).

For Butterfly networks, the *Arb-STC-GS* approach achieves the same maximum performance as achieved using *Mat-HTC-GS* approach [25] and reduces the cost by 5% (see Figure 6.9). The optimal placement for Butterfly network is shown in Figure 6.10(b). Note in the top subfigure, we only show the logical connections instead of actual routing path of *gas-station* links. For Butterdonut networks, the *Arb-STC-GS* approach improves the performance by 25% without increasing the cost (see Figure 6.9). Figure 6.10(c) shows the optimal placement for Butterdonut network. The Ring networks (*G-R-L-M/CM*) are not included in the prior work [25], thus we do not show the comparison. The chiplets are distributed along the periphery of the interposer in the optimal placement for the Ring topology (see Figure 6.10(d)), which is good for heat dissipation. Thus, the performance of the Ring topology saturates at a relatively small interposer size, and we observe lower cost and temperature than those of other networks (see Figure 6.9).

6.5.2 Iso-cost and Iso-performance Analyses

Figure 6.11 shows the iso-cost performance for various networks running *Cholesky* benchmark, while not exceeding the cost of a 2D system. In general, our *Arb-HTC* approach improves the iso-cost performance by 13-37% (20% on average), and our *Arb-STC* approach improves the iso-cost performance by 40-68% (49% on average), compared to our prior *Mat-HTC* approach [25]. The previous work [25] shows that the *U-M* network cannot be implemented feasibly due to the large μbump area overhead and the incorrectly estimated yield drop. Using our more accurate cost model, it is actually feasible to implement the *U-M* network within the cost budget.

Figure 6.12 shows the iso-performance cost and the corresponding peak temperature for each network. Here, for each network, we match the performance of the 2.5D system designed using our

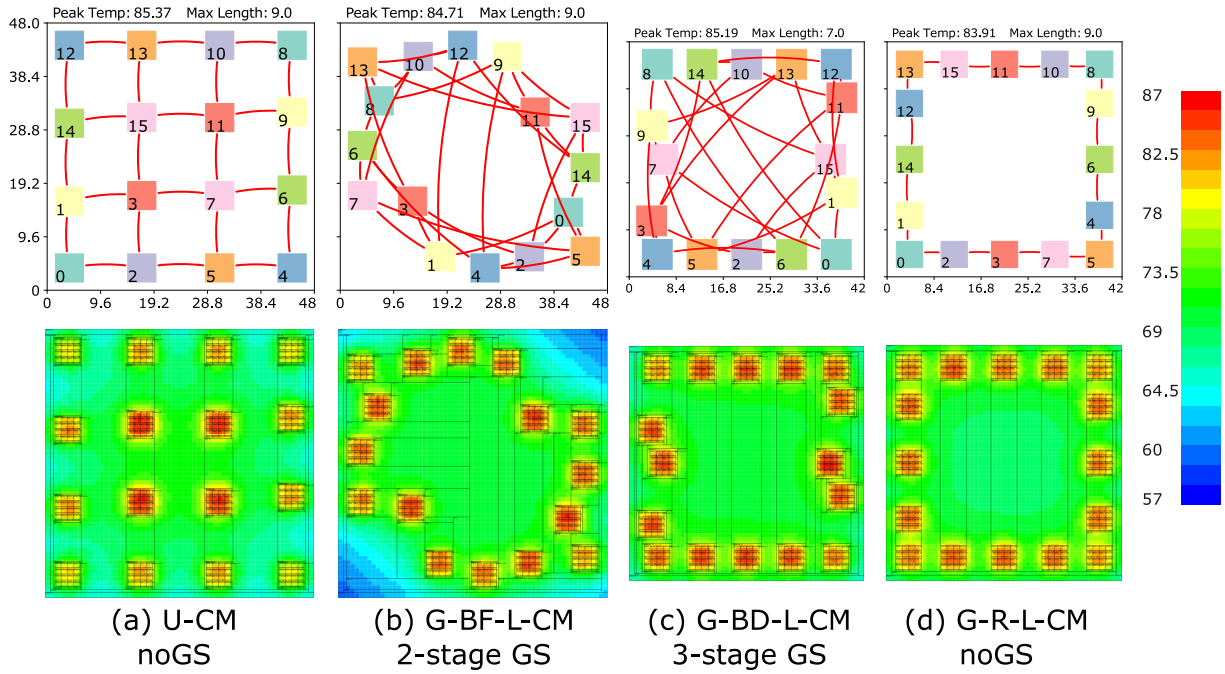


Figure 6.10: Optimal chiplet placement for maximum performance and corresponding thermal maps when running the *Cholesky* benchmark in 2.5D systems with different network topologies. The figures are scaled to the interposer sizes.

proposed approach with the corresponding maximum performance of the 2.5D system designed using prior *Mat-HTC* approach [25] when running *Cholesky* benchmark. The cost values are normalized to the cost of a 2D system. Under the same hard temperature constraint as the prior work [25], our *Arb-HTC* approach reduces manufacturing cost by 5-20% (14% on average) without lowering the performance. Using the *Arb-STC* approach, we can push the iso-performance cost saving to 30-38% (32% on average) with up to 91^oC overall system peak temperature.

Figure 6.13 shows the Pareto Frontier Curve of normalized performance ($1/IPS$) and normalized cost using *Mat-HTC* approach [25], *Arb-HTC* approach, and *Arb-STC* approach. Our arbitrary placement pushes the Pareto frontier curve towards higher performance and lower cost, and the soft temperature constraint approach pushes the frontier further.

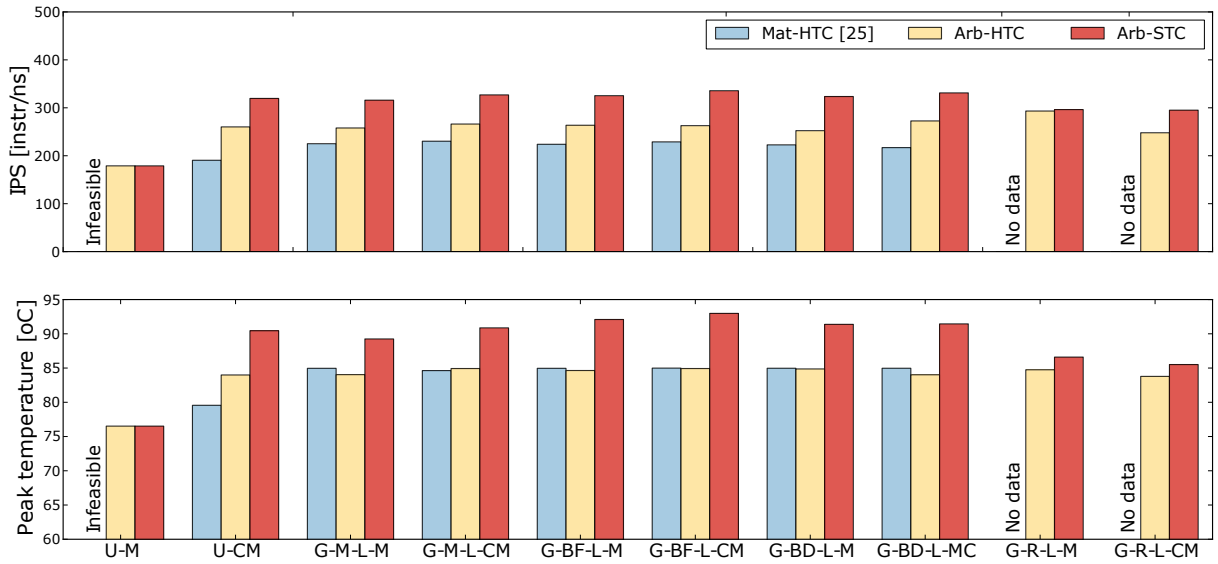


Figure 6.11: Iso-cost performance and the corresponding peak temperature when running *Cholesky* benchmark for various networks, while not exceeding the cost budget of a 2D system.

6.5.3 Analyses of Different Types of Applications

Figure 6.14 shows the thermal maps of 2.5D systems designed for high-power (*Cholesky*), medium-power (*Streamcluster*), and low-power (*Lu.cont*) applications using *Mat-HTC* [25], *Arb-HTC* and *Arb-STC* approaches. For comparison, we choose the same optimization objective as in the prior work [25], which focuses on performance ($(\alpha, \beta, \gamma) = (0.999, 0.001, 0)$). With the *Arb-HTC* approach, we can achieve the same performance as using the prior *Mat-HTC* approach [25] and reduce the manufacturing cost by 19%, 14%, and 3% for high-power, medium-power, and low-power applications, respectively. The equivalent performance is achieved at a smaller interposer size where the chiplets are pushed to the periphery of the interposer to ease the heat dissipation. For high-power and medium-power applications, 2-stage *gas-station* links are used, which provides flexibility in chiplet placement to form a ring shape for mesh-like networks, while for low-power application, such a ring-shape placement is not feasible as we need to provide routability of single-cycle links.

Using *Arb-STC* approach, for high-power application, we can achieve the maximum possible performance (3% higher than both *Mat-HTC* approach [25] and *Arb-HTC* approach) and 15% lower cost. The improvement is achieved by violating the temperature threshold by 0.5^oC and using single-cycle

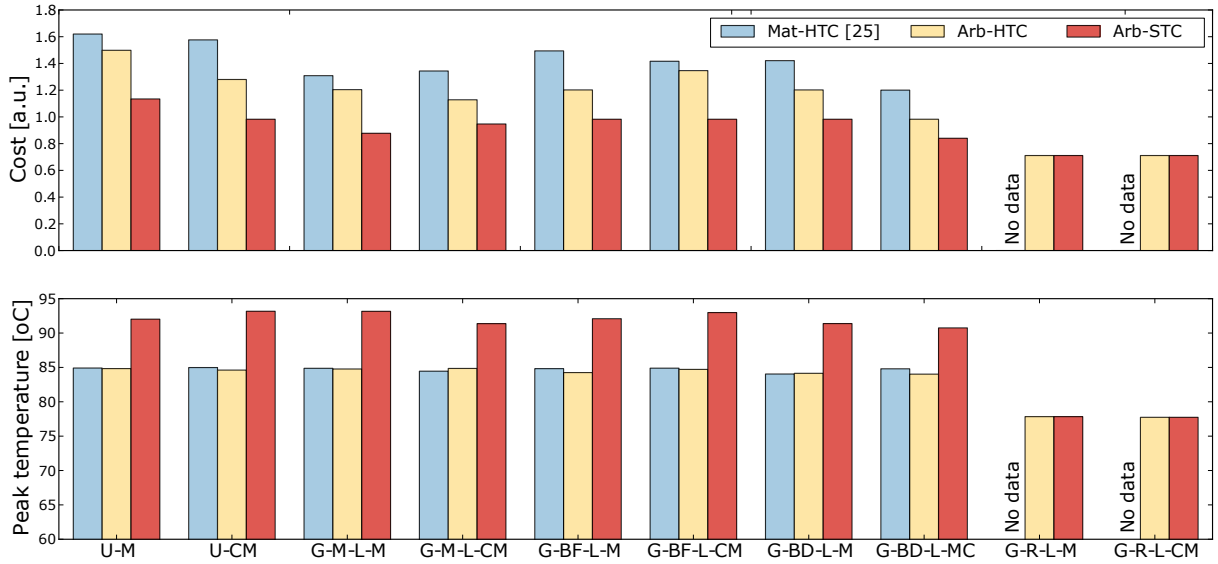


Figure 6.12: Iso-performance cost and the corresponding peak temperature for each network. Here the performance is equal to the maximum performance achieved using *Mat-HC-GS* [25] when running *Cholesky* benchmark. The cost values are normalized to the cost of a 2D system.

inter-chiplet links without *gas stations*, which constrains distance between chiplets and forms a matrix-style placement. For medium-power application, we get identical network choices and placement solutions using *Arb-STC* and *Arb-HTC* approaches. For low-power application, our *Arb-STC* approach achieves the maximum possible performance while violating the temperature threshold by 1.4^oC. This improvement also comes with 40% cost overhead, but in this example, cost is not our concern. The chiplets cluster in the center of the interposer to meet single-cycle latency constraint for a butterfly topology, and leave large empty space on the edges of the interposer to help heat dissipation.

It should be noted that the results we show in Figure 6.14 assume that we know what application will be running at the design time, and we optimize for each application. For unknown target applications or a mix of known and unknown applications, we optimize for the worst-case (highest power application) scenario at the design time, and run the target application on the optimized organization (including network topology, interposer size, chiplet placement, and inter-chiplet link design). For example, if a system is expected to run high-power (*Cholesky*), medium-power (*Streamcluster*), and low-power (*Lu.cont*) applications, we design and optimize the system using the high-power application. When running medium-power application on the system optimized for high-power application, we observe the

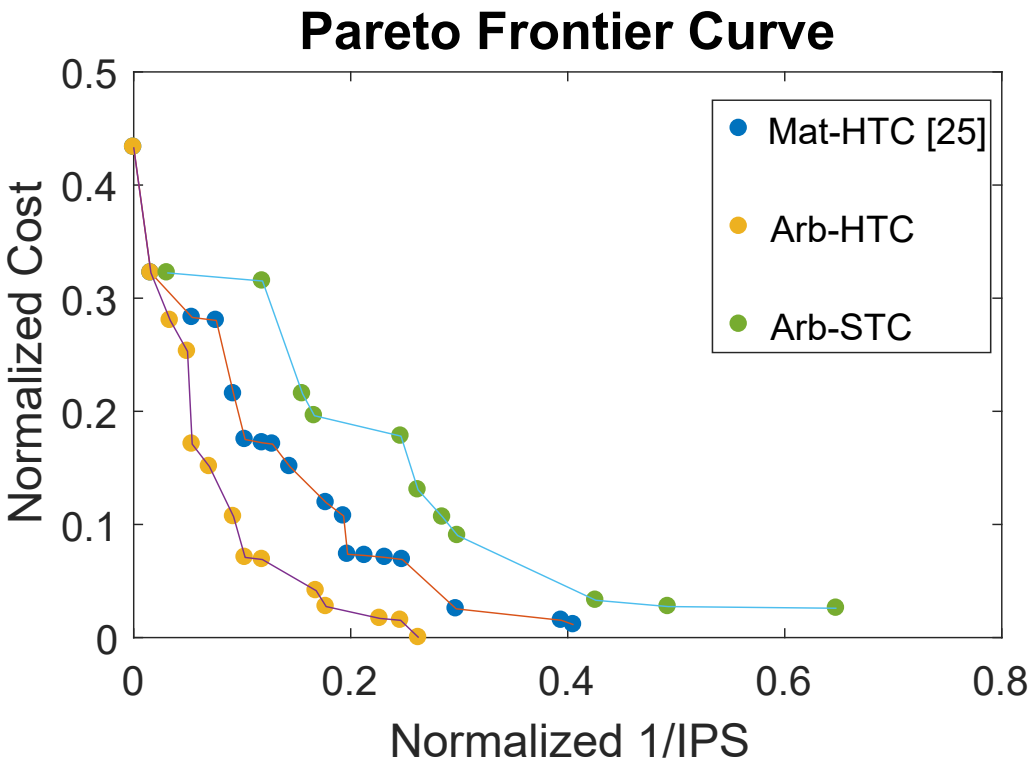


Figure 6.13: Pareto Frontier Curve of normalized performance ($1/IPS$) and normalized cost using *Mat-HTC* approach [25], *Arb-HTC* approach, and *Arb-STC* approach.

same performance, 23% higher cost, and 6^oC lower temperature compared to that of a system custom designed for medium-power application. When running low-power application on the system designed for high-power application, we observe 5% lower performance, 5% higher cost, and 12^oC lower temperature compared to that of a system custom designed for low-power application.

6.5.4 Analyses of Cross-layer Co-optimization Benefits

To understand the benefits of co-optimizing across multiple design layers simultaneously, we conduct a comparison between cross-layer and single-layer methodologies while running the *Blackscholes* benchmark. We compare multiple cases in Table 6.7. The baseline is the optimal solution of our cross-layer co-optimization methodology. We use three letters to represent the choices at each of the logical, physical,

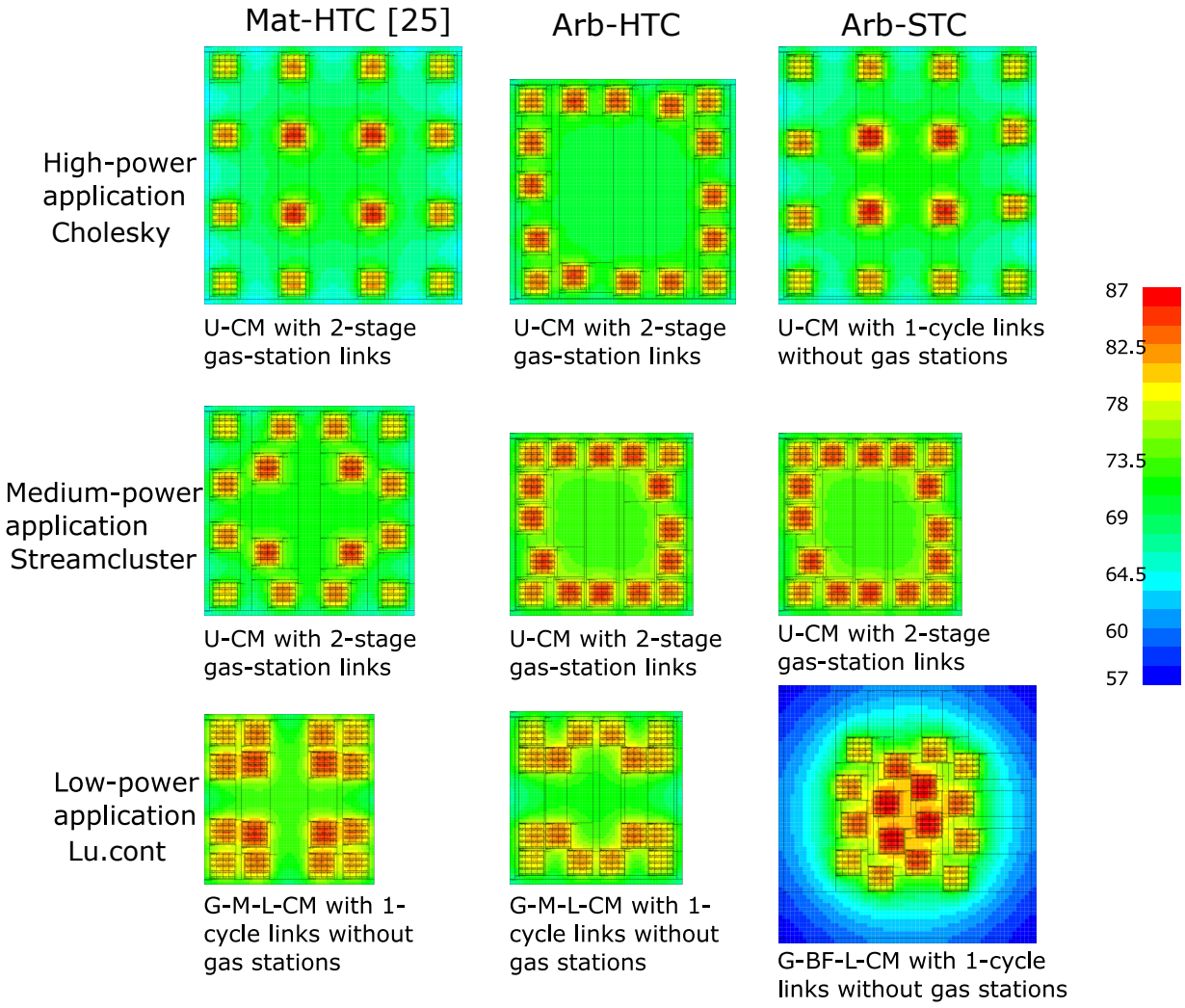


Figure 6.14: Thermal maps of 2.5D systems designed for high-power, medium-power, and low-power applications using *Mat-HTC* [25], *Arb-HTC*, *Arb-STC* approaches. The figures are scaled to the interposer sizes.

and circuit layers, for the remaining nine cases. Here O means optimal, W means worst possible, F means prefixed, B means best possible. So for example, the OOW case corresponds to the use of the same design choices as the optimal cross-layer solution at the logical and physical layers, and use of the worst possible choice at the circuit layer. This case shows the contribution of the circuit layer in our cross-layer co-optimization methodology. In the FFB case, we fix the design choices at the logical and physical layers, and only optimize the circuit layer. We report performance improvement, cost increase, and temperature for each case. To better compare the different cases, we use the *Performance/Unit Cost* metric. For the

OOW and OWO cases, we observed a cost reduction and/or slight performance improvement, but at a high infeasible peak temperature. For the case of WOO, the temperature is acceptable but we get 20% lower performance and 50% higher cost. For the cases of FFB, FBF, BFF, and BFB, we get either higher performance at higher cost or lower performance at lower cost, but the temperature becomes infeasibly high. For the cases of FBB and BBF, the temperature is safe, while performance and cost offset each other. In terms of the *Performance/Unit Cost* metric, our cross-layer co-optimization approach performs better than all cases except OOW, OWO and BFF, but these cases have high infeasible temperature.

Table 6.7: Cross-layer vs. single-layer optimization.

Cases	Perf Improvement	Cost Increase	Temperature [^oC]	Perf/Unit Cost
Cross-layer	0%	0%	86	3.10
OOW	4%	-8%	99.9	3.50
OWO	0%	-22%	108.0	3.97
WOO	-20%	56%	84.2	1.59
FFB	-39%	-34%	100.9	2.88
FBF	4%	11%	102.5	2.92
BFF	-16%	-36%	103.4	4.09
FBB	-9%	-4%	85.8	2.94
BFB	-35%	-34%	100	3.09
BBF	2%	3%	86.2	3.06

6.5.5 Sensitivity Analysis

We conduct a sensitivity analysis (see Figure 6.15) to show the optimal combinations of performance, cost and peak temperature, and the corresponding objective function values with respect to different temperature thresholds from 75^oC to 95^oC and different temperature constraint choices (including hard temperature constraint, soft temperature constraint with linear and square penalty functions, and no temperature constraint). We choose the weights to be $((\alpha,\beta,\gamma) = (0.8,0.1,0.1))$ as an example for a performance-focused objective function. With no temperature constraint, we can always achieve the maximum performance and the lowest cost, at a temperature of 93.2^oC. Thus, with a temperature threshold of 94^oC or higher, the optimal performance, cost, and temperature combinations with different constraint choices are the same. With a hard temperature constraint, any case that exceeds the temperature threshold

is considered as infeasible, thus, the peak temperature is close to, but below the temperature threshold. As the temperature threshold increases, there are more feasible design choices and the objective function value decreases. A soft temperature constraint allows violating the temperature threshold and translates the violation into a penalty in the objective function. The soft temperature constraint approach provides more choices and thus is guaranteed to have a solution that better or equal to that obtained using hard temperature constraint approach. For the soft temperature constraint approach with a linear penalty function, we are allowed to violate the temperature threshold only slightly to find a solution that has higher performance and/or lower cost than the hard temperature constraint approach. A square penalty function suppresses the penalty for a small violation and highlights the penalty for a large violation of the temperature threshold. Thus, with a soft temperature constraint approach with the square penalty function, we can achieve higher performance and lower cost compared to the case with the linear penalty function. For example, with a temperature threshold of 80^oC, the result with the hard temperature constraint has lowest performance. With the soft temperature constraint with the linear penalty function, we violate the temperature threshold by 0.59^oC and achieve 6% higher performance but at 5% higher cost compared to the hard temperature constraint approach. With the soft temperature constraint with the square penalty function, we violate the temperature threshold by 0.93^oC and achieve 5% higher performance at the same cost compared to the hard temperature constraint approach.

6.6 Summary

In this chapter, we have introduced a cross-layer co-optimization methodology for network design and chiplet placement in 2.5D systems. Our methodology optimizes network topology design, inter-chiplet link design, and chiplet placement across logical, physical, and circuit layers to jointly improve performance, lower manufacturing cost, and reduce operating temperature. Compared to our prior work [25], we improve the optimization methodology by enhancing the cost model, including operating temperature in the optimization goal, applying a soft temperature constraint, and improving the optimization algorithm to enable arbitrary chiplet placement. Our new methodology shifts the performance-cost Pareto tradeoff curve for 2.5D systems substantially. Our approach improves thermally-constrained performance by 88% at the same manufacturing cost and reduces the cost by 29% at the same performance in comparison to

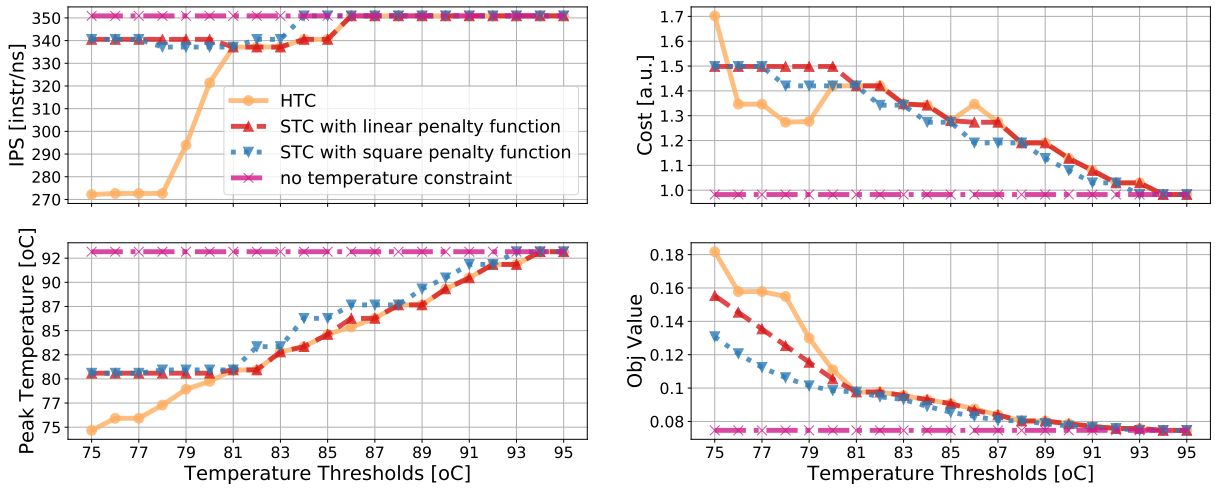


Figure 6.15: Sensitivity analysis comparing hard temperature constraint, soft temperature constraints with linear function and square function, and no temperature constraint for various temperature thresholds from 75-95^oC.

2D systems. Compared to our prior work [25], for the same hard temperature constraint our enhanced placement algorithm with arbitrary placement improves iso-cost performance by 13-37% (20% on average) and reduces iso-performance cost by 5-20% (14% on average). Overall, our new optimization methodology with a soft temperature constraint and arbitrary placement achieves 40-68% (49% on average) higher iso-cost performance and 30-38% (32% on average) lower iso-performance cost over our prior work [25].

6.7 Acknowledgments

Chapter 6 contains a reprint of A. Coskun, F. Eris, A. Joshi, A. B. Kahng, Y. Ma, A. Narayan and V. Srinivas, “Cross-Layer Co-Optimization of Network Design and Chiplet Placement in 2.5D Systems”, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, in revision. The dissertation author is a main contributor to this paper. I would like to thank my coauthors Ayse Coskun, Furkan Eris, Ajay Joshi, Andrew B. Kahng, Yenai Ma and Aditya Narayan.

Chapter 7

IO and Clock Tree Optimizations for 3D Integration

A pathfinding framework for off-chip interconnects would not be complete without 3D interconnects. 3D interconnects are emerging as an attractive option as their manufacturing cost and yield improve. They provide unique design spaces to partition traditional SoCs, and integrate heterogenous technologies into a single package, e.g., DRAM and logic processes, or longer channel technologies for analog/RF while shorter channel technologies for digital blocks. A critical aspect of 3D interconnect design and integration is the design choices for the IO and the clocking across dies in the 3D stack. Both these are tightly tied to the synchronization scheme used across the interconnect.

In this chapter, we explore IO and clock tree optimizations for 3D integration, including synchronization schemes suitable for 3D interconnect.

7.1 Introduction

3D interconnect can span a wide range of bandwidths (few GB/s to hundreds of GB/s) and region area (few mm^2 to the full die area). The 3DIO *frequency* and *synchronization scheme* are two key choices that play into the power and area for such an interconnect. Clock and data power for such interconnect can often be a significant component of total chip power. The synchronization scheme for the 3DIO

clocking could be (i) *synchronous* (traditional clock tree), (ii) *source-synchronous* (forwarded clock), or (iii) *asynchronous* (separate clock trees). Based on the 3DIO frequency and synchronization scheme, the design could lend itself to dividing the clock into local *clusters* as shown in Figure 7.1, which are tightly skew-balanced within themselves but have looser skew requirements between clusters. This allows for a tighter timing budget for the clock and data paths and also lends itself to use of low-power 3DIO for source-synchronous and asynchronous schemes at the expense of overhead in the clock generation and distribution for every cluster. The 3DIO frequency that can be achieved depends on the timing budget, which in turn depends on the synchronization scheme and clustering.

In this work, we investigate the optimal choice of 3DIO frequency, synchronization scheme and clustering for 3D interconnect power, area and bandwidth optimization. The optimal 3DIO frequency depends on the cost function of power and area. While wider and slower buses generally consume less power due to lower clock tree power and lower 3DIO power, they do take up more area, especially due to limitations in the micro-bump pitch and 3DIO electrostatic discharge (ESD) requirements.

The default clocking methodology for 3D is likely still based on to be a synchronous clocking scheme. Previous research has sought to improve CTS outcomes and 3D clock tree structures for such a clocking scheme [84], [68], [162], [100], [95]. These studies on 3D clock trees focus on optimization of the synchronous clock tree to minimize skew, power and area, but do not consider alternative synchronization schemes. In this chapter, we study different synchronization and clustering schemes, seeking to identify and model the optimal choice of 3DIO frequency (number of 3DIOs), clock synchronization scheme, and clustering for a given cost function of power, area and bandwidth.

To find these optimal choices, we build several analytical models to estimate the power, area and timing of the clock tree, data path and 3DIO. The models that we develop encompass on-die clock tree and data path models, and the 3DIO models from [58], adapted for the three synchronization schemes. We combine analytical models [21], [77], [138] and metamodeling techniques [60] to fit models against data from a Design of Experiments (DOE).

Our contributions in this chapter are as follows.

- We propose a new view of the design space for 3DIO frequency, clock synchronization scheme and clustering based on bandwidth and region area of the interconnect.

- We develop accurate power and area model for the 3D interconnect based on bandwidth, region area, number of clusters, cluster clock frequency, and clock synchronization scheme.
- We demonstrate how the space of design requirements and constraints is partitioned according to the choice of optimal synchronization scheme.

In the following, Section 7.2 describes the concept of clustered clocking, elaborates on the differences between the clocking schemes, and provides hypotheses on the design space partitioning based on these options. In Section 7.3, we propose a methodology to build power and area models for the design space based on DOE results, using metamodeling techniques [60]. This modeling can then enable a flow that chooses the optimal choice of 3DIO frequency, synchronization scheme and clustering as described in Section 7.2. The P&R and timing flow used to implement our design of experiments (DOE) is described in Section 7.4. Section 7.5 describes our DOE and the results, including the fitted models versus the DOE data. We summarize the chapter in Section 7.6.

7.2 3DIO Clustering and Clocking

One way to localize the clock tree of the 3D interconnect is to divide it into clusters. Such a clustered interconnect has a *cluster clock*, with skew constraints that could be tighter than those of the *global clock*. Figure 7.1 shows the design divided into four clusters. The global clock is distributed to the clusters, and once in the cluster, a *cluster clock tree* distributes the clock within the cluster. The skew constraints for the 3D interconnect are limited to the cluster clock tree; hence, use of the cluster clock tree allows for smaller skew. Further, the 3DIO array is now divided per cluster as well, which means that the data paths, going from the uniformly distributed sinks to the 3DIO at the center of the cluster, are much shorter. On the other hand, clustering the design has multiple overheads: synchronizing data between clusters; the clock 3DIO per cluster to the top die; and the blockages caused by the 3DIO array being distributed per cluster.

The cluster clock between the two dies must be synchronized. There are three basic methods used to synchronize the interconnect.

(i) Synchronous. In the synchronous method, the cluster parts in both dies share a common cluster

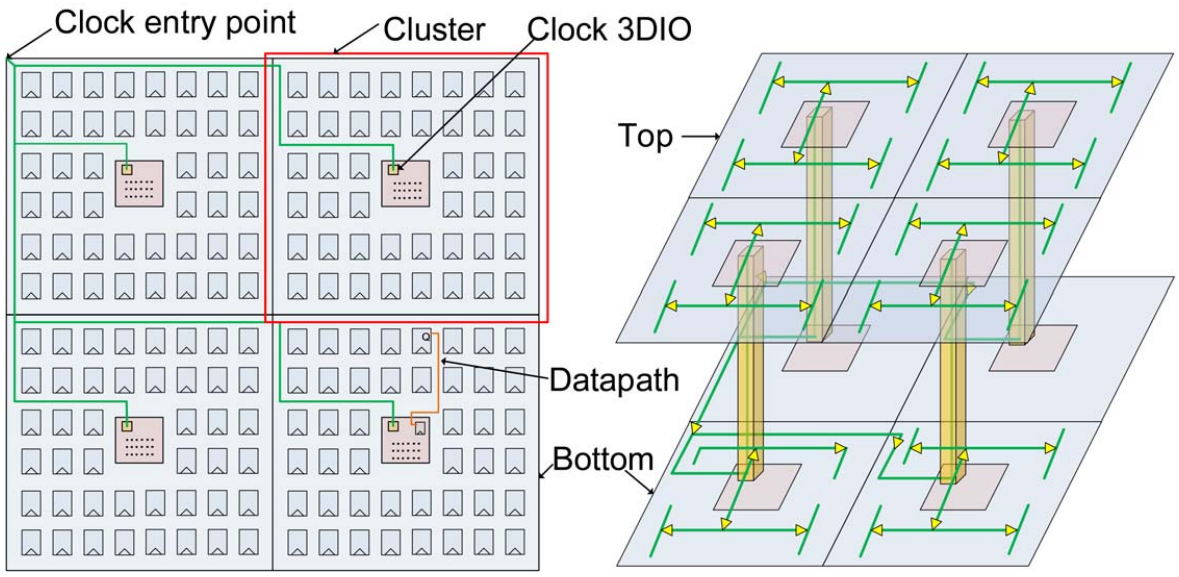


Figure 7.1: Clustering the clock domains.

clock tree that is balanced to end points in both dies. The clock tree on each die will vary according to the process variation of that die, so inter-die skew is large for the inter-die tree. But, at lower frequencies, this scheme provides a simple approach that is consistent with low-power implementation.

(ii) Source-synchronous. In the source-synchronous method, a clock is forwarded from the bottom die to the top die (or vice-versa). This forwarded clock is then built into a tree for the sinks on the top die. Such a scheme does not require skew balancing across two dies, but requires *balance delays* (T_b) within each die on the data path to match the clock insertion delay and enable source-synchronous data capture. There are power and area overheads associated with such balancing, but the improvement in the timing budget means that higher speeds can be achieved over the 3DIO. Both synchronous and source-synchronous data paths can enable a double data rate (DDR, both edges of the clock) design without any overhead in clock generation. We assume such a DDR design since it provides a way to minimize the number of 3DIOs.

(iii) Asynchronous. The asynchronous scheme has a FIFO that helps clock domain crossing between the two dies. Since the latency impact can be large, this is often combined with a high-speed serial IO that enables lower latency. As a consequence of the serial IO, we also obtain a much smaller IO footprint (smaller number of 3DIOs) at the expense of power and latency. We assume a 1:8 serialization

for the asynchronous scheme. Although further serialization may be possible, the latency impact for much larger serialization ratios could be considerable, given the need to deserialize as well as the much slower cluster clock needed for a given 3DIO maximum frequency.

We observe that the source-synchronous clocking scheme allows for higher 3DIO frequencies than the synchronous clocking scheme, as it matches the delays in each die and does not expose the 3DIO timing to inter-die variations in the clock tree. But, such matching delays come with an area and power cost. Asynchronous clocking schemes provide a unique way of improving power for higher 3DIO frequencies. As they serialize the bus before off-chip transmission, such clocking schemes allow the cluster clock to operate at a lower speed while achieving a higher 3DIO bandwidth. This enables cluster clock tree power reduction at higher 3DIO speeds, offsetting the increased 3DIO power incurred by a narrower and faster bus. Furthermore, the narrower bus itself allows for 3DIO area reduction. When asynchronous clocking is combined with clustering, the 3DIO timing budget enables higher 3DIO frequencies, allowing for interesting tradeoffs between power and area. Exposing how these choices affect power and area based on the bandwidth and region area can enable system architects to make better 3DIO and clocking choices.

The three synchronization schemes are outlined in Figure 7.2. The figure shows the DDR (double data rate) implementation for the synchronous and source-synchronous schemes, which effectively doubles the 3DIO frequency and implies a half-cycle timing path for setup (as opposed to a full-cycle timing path as in a traditional single data rate design). The data path is shown in red, while the clock path in blue. For the asynchronous case, the clock and data path is common, shown in purple. The bottom die clock tree is shown in yellow, while the top die clock tree is shown in light blue. As can be seen, the synchronous case exposes an inter-die skew between the yellow and light-blue clock trees, while the source-synchronous case does not. Further description of Figure 7.2 can be found in Section 7.3.

For a given 3DIO topology (defined by the synchronization scheme and number of clusters) there is a maximum frequency that can be achieved by the 3DIO. This maximum frequency is determined by two factors: (i) the 3DIO timing budget for retiming across the dies, and (ii) the on-die timing budget for retiming from the launch sinks to the 3DIO (or vice-versa from the 3DIO to the capture sinks).

In general, we expect the maximum 3DIO frequency to be the highest for asynchronous/highly-clustered and lowest for synchronous/lightly-clustered topologies. This is because on-die and 3DIO timing

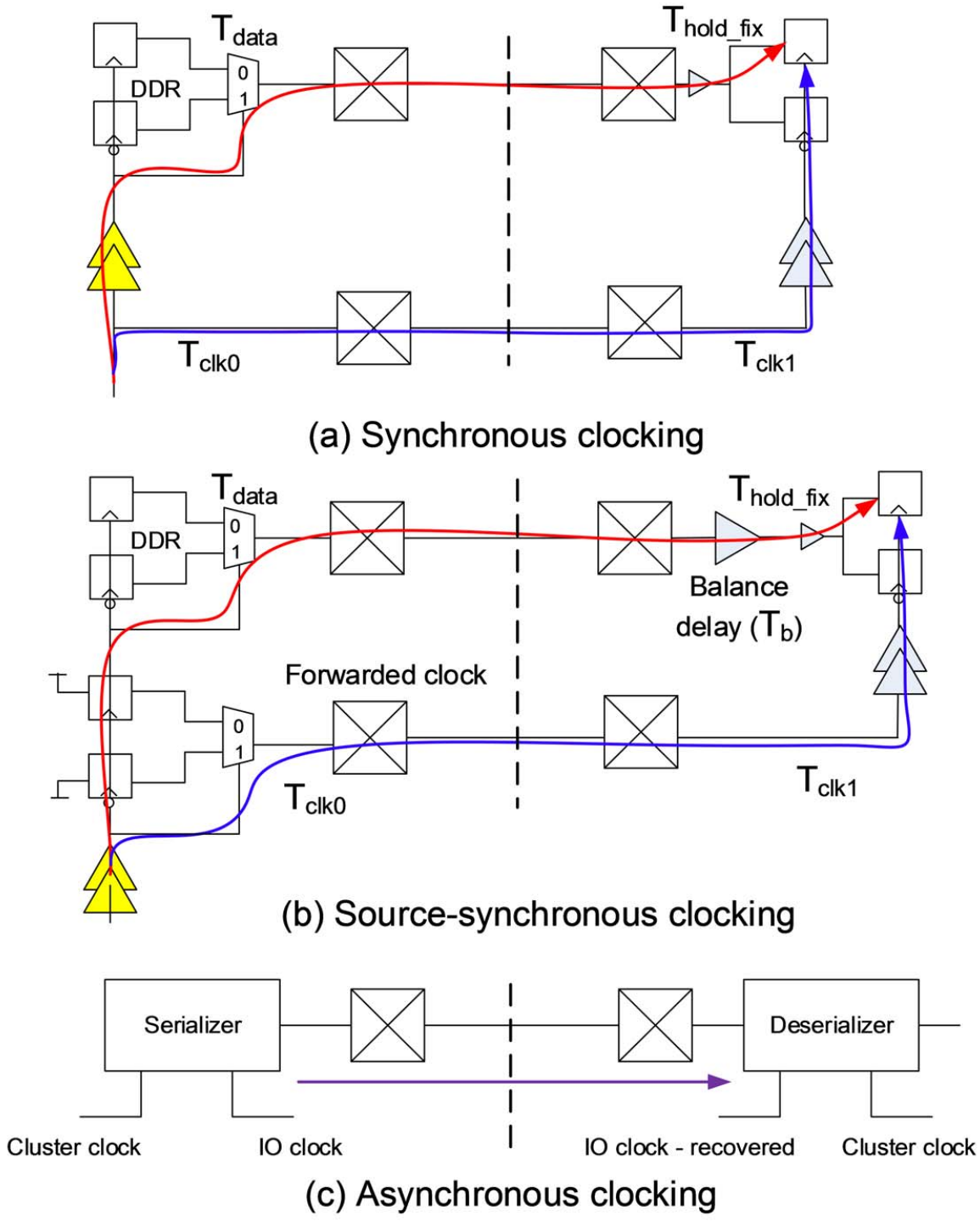


Figure 7.2: Clock synchronization schemes.

budgets worsen as we progress from the former end of the design space to the latter end. The 3DIO timing budget is best for asynchronous schemes due to embedded clocks not having tight skew requirements,

and much better jitter tolerance. The on-die timing budget is also better for asynchronous schemes due to lower cluster clock frequency (a consequence of serial 3DIO), and properties of the local sinks (together, highly-clustered).

In the following sections, we build our model and review the results in the context of three problem statements:

1. For a given BW and region area, find a clock synchronization scheme, number of clusters and clock frequency that minimizes power, given an area constraint.

2. For a given BW and region area, find a clock synchronization scheme, number of clusters and clock frequency that minimizes area, given a power constraint.

3. For a given region area, find a clock synchronization scheme, number of clusters and clock frequency that maximizes BW for a given power and area constraint.

7.3 Power, Area and Timing Model

In this section, we describe our model which accurately estimates total power, area and timing (worst negative slack (WNS)) for a given tree topology (synchronization scheme and clusters), bandwidth and region area. Figure 7.3 shows the *3DIO/CTS directed graph* of our modeling approach. The primary inputs (indicated by circles) are bandwidth, region area, cluster clock frequency, synchronization scheme and number of clusters. Based on these inputs, we calculate the number of sinks/FFs ($= 2 \cdot BW/f$), 3DIO frequency ($f_{IO} = 2 \cdot f$ for synchronous and source-synchronous schemes; $f_{IO} = 8 \cdot f$ for asynchronous scheme), number of 3DIOs ($= BW/f_{IO}$), and maximum skew and transition requirements ($= k/f$).

The analytical expressions given below are often hard to fit across large ranges of many input parameters. We employ metamodeling techniques to improve fitting across the large ranges and number of input variables. We choose to use an ANN (Artificial Neural Network) model for our fit [60]. Feeding just the primary inputs in Figure 7.3 does not, in our experience, yield very good results; however, with the help of the directed graph, a systematic approach to propagate the inputs through the graph aids the ANN model in finding fits for power, area and WNS within +/-15% across our design of experiments (DOE). In

the following subsections, we describe the details of our model. Section 7.5 then gives fitting results. Note that to estimate the final outcomes (i.e., power, area, and WNS), we need buffer and wirelength models for clock and data path as our foundations.

7.3.1 Clock and Data Wirelength

The work of Snyder et al. [128] and Steele and Snyder [130] show that the expected total tree length of a Euclidean Steiner minimal tree over n points uniformly distributed within a bounded region A_{reg} is proportional to $\sqrt{A_{reg} \cdot N}$ for sufficiently large N. The constant of proportionality, denoted by k, is dependent on the functional of the pointset. Further, we empirically observe that the constant is not a single value, but is itself a function of region area A_{reg}, number of clusters N_c, and maximum transition time constraint T_{tran}^{max} ($k = f(A_{reg}, T_{tran}^{max}, N_c)$).

Clock wirelength significantly changes according to the clock tree structure (tall and narrow tree vs. short and wide tree). To analyze the structure of the clock tree generated by commercial tools, we extract the average fanout and driven wirelength of clock buffers at each depth (l) of clock trees synthesized with different region areas and/or number of FFs (N_{ff}). The depth of clock source is zero and the maximum depth is L, which is the number of buffer stages on the longest path from the clock source to any sink FF.

From the extracted data, we observe that the average fanout changes depending on the region area and the number of FFs, but the average wirelength of driven nets at depths up to L is bounded. This implies that the tool increases N_g^l, which is the number of buffers at depth $l-1$, and the maximum depth of clock tree L, as A_{reg} increases, and as N_{ff} and T_{tran}^{max} decrease. Therefore, it is very important to accurately estimate both L and N_g^l at each clock depth. We estimate the clock wirelength using the following method. First, we determine the N_e^l, which is the number of cells driven by a buffer at depth $l-1$, and calculate N_g^l (i.e., N_g^{l+1}/N_e^l). Then, using the equation $\sqrt{A_{reg} \cdot N}$ from [130], we obtain Equation (7.1):

$$w^l = N_g^l \cdot \sqrt{A_{reg} \cdot \ N_e^l / N_g^l} \tag{7.1}$$

Extending Equation (7.1) to total clock wirelength, we obtain

$$
\begin{aligned}
W_{clk} &= \sum_{i=0}^{L} w^l = \sqrt{A_{reg}} \cdot (\sqrt{N_{ff}} + \sum_{i=0}^{L-1} \sqrt{N_e^i \cdot N_g^i}) \\
&= \sqrt{A_{reg} \cdot N_{ff}} \cdot (1 + \sum_{i=1}^{L} \sqrt{N_g^i}) = k \cdot \sqrt{A_{reg} \cdot N_{ff}}
\end{aligned}
\tag{7.2}
$$

where k is a coefficient factor. However, Equation (7.2) does not account for the global clock tree that distributes the clock to the clusters. We extend Equation (7.2) as

$$
W_{clk} = k_c \cdot \sqrt{A_{reg} \cdot N_{ff}} + k_g \cdot \sqrt{A_{reg} \cdot N_c} \tag{7.3}
$$

where k_c and k_g are fitted coefficients for cluster and global clock trees. Note that both k_c and k_g are sensitive to A_{reg}, T_{tran}^{max}, N_{ff} and N_c.

Data path wirelength is proportional to the number of data wires and the cluster dimension.

$$
W_{data} = k_0 \cdot N_{ff} \cdot \sqrt{A_{reg}/N_c} \tag{7.4}
$$

For a large number of clusters, when the number of sinks per cluster is small, the data wirelength deviates from the above based on the placement tool runscript, since the sinks are not evenly distributed in the cluster. We do not consider such cases since they incur large power and area overheads due to over-clustering.

7.3.2 Clock and Data Buffer Area

Tellez and Sarrafzadeh [138] give a method to insert the minimum number of buffers under a given transition time (T_{tran}^{max}) constraint. They formulate a nonlinear constrained buffer insertion problem that minimizes the number of stages in a given rooted clock tree, such that for each stage, the stage rise time does not exceed T_{tran}^{max}. They then linearize this problem by using the concept of maximum capacitance (C_{max}). Maximum wirelength W_{max} is computed such that the rise time at the end of the wire is T_{tran}^{max}. Using the relationship $C_{max} = C_0 \cdot W_{max} + C_g$, where C_0 is the capacitance per unit length and C_g is the gate input capacitance, they conjecture that any buffer stage i with stage capacitance $C_i \leq C_{max}$ will have $T_{tran}^i \leq T_{tran}^{max}$. Based on this approach, we estimate the number of clock buffers (N_{cbuf}) as

$$N_{cbuf} = \frac{W_{clk} \cdot C_0 + N_{ff} \cdot C_g^{ff}}{C_{max} - C_g^{buf}} \tag{7.5}$$

where C_g^{buf} and C_g^{ff} are the input gate capacitance of a buffer and a flip-flop, respectively.

C_{max} is related to T_{tran}^{max} as follows, which is a consequence of the slew degradation discussed in [64]:

$$\begin{aligned} T_{tran}^{max} &= \sqrt{(T_0)^2 + (k_1 \cdot C_{max} \cdot R_{max})^2} \\ C_{max} &= k_2 \cdot \sqrt{(T_{tran}^{max})^2 - (T_0)^2} \end{aligned} \tag{7.6}$$

where T_0 is the transition time at the input to the line and R_{max} is the maximum resistance of W_{max}. The total buffer area (A_{clk}) is proportional to N_{cbuf}, so we have

$$A_{clk} = k_3 \cdot \frac{W_{clk} \cdot C_0 + N_{ff} \cdot C_g^{ff}}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} \tag{7.7}$$

We again extend this expression to a clustered design, and add the clock buffers for the global tree and cluster tree:

$$A_{clk} = k_1 \cdot \frac{N_{ff} \cdot C_g^{ff} + W_{clus} \cdot C_0}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} + k_2 \cdot \frac{N_c \cdot C_g^{buf} + W_{glob} \cdot C_0}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} \tag{7.8}$$

where k_1 and k_2 are fitted coefficients. C_g^{ff} and C_g^{buf} are the flip-flop and buffer input gate capacitances, respectively, and C_0 is the wire capacitance per unit length.

The data buffer area is calculated as follows. Equation (7.7) can be used to calculate the number of buffers per data wire. Since some of the data wires could be quite small, we need to use a ceiling function to find an integral number of buffers per data wire as shown in Equation (7.9). Also, we need a minimum number of data buffers, n_{hold}, to ensure that hold time is not violated:

$$n_{dbuf}(w_{data}^i) = MAX\left(\left\lceil k_3 \cdot \frac{w_{data}^i \cdot C_0 + C_g^{ff}}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}} \right\rceil , n_{hold} \right) \tag{7.9}$$

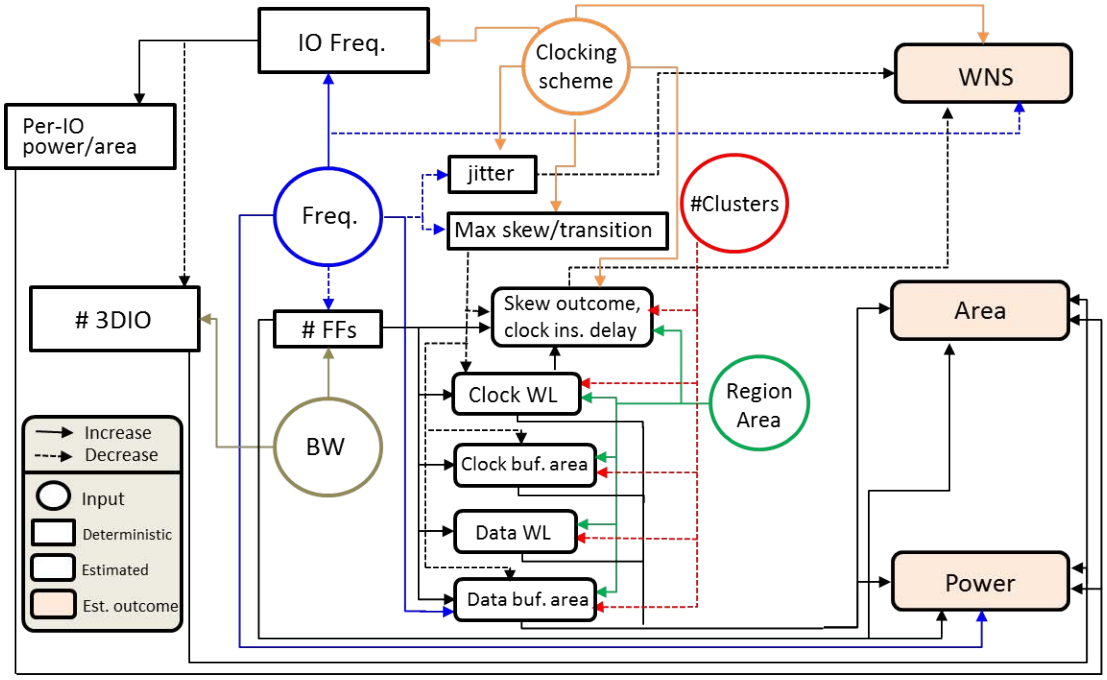


Figure 7.3: 3DIO/CTS directed graph.

where w^i_{data} is the wirelength of i^{th} data path. For each cluster, the data wires begin at the sinks that are distributed uniformly across the cluster and end at the 3DIO array at the center of the cluster, as shown in Figure 7.4.

Figure 7.4 shows how for a square of side d, and a pitch p of the placed sinks, there are concentric octagons with the same Manhattan wirelengths from the 3DIO. If we index each octagon i (starting from the center), then the number of sinks per octagon increases as $4 \cdot i$ until $i = \frac{d}{2p}$, beyond which the number of sinks per octagon decreases as $4 \cdot (\frac{d}{p} - i)$. We thus obtain the total number of data buffers in Equation (7.10), and consequently the data buffer area (A_{data}). Note that we ignore the area of the 3DIO array and the other cases in which the placement of sinks does not exactly follow the octagon shape, since this equation is only to guide our metamodel for better estimation.

For the synchronous clocking case, the inter-die variation is large, which leads to a large number of hold buffers added in the design, and to a larger n_{hold} value for the 3DIO timing path.

$$A_{data} \propto N_{dbuf} = \sum_{i=1}^{\frac{d}{2p}} 4i \cdot n_{dbuf}(i \cdot p) + \sum_{i=\frac{d}{2p}+1}^{\frac{d}{p}-1} 4(\frac{d}{p} - i) \cdot n_{dbuf}(i \cdot p)$$
$$p = \sqrt{\frac{A_{reg}}{N_{ff}}}, \quad d = \sqrt{\frac{A_{reg}}{N_c}} \tag{7.10}$$

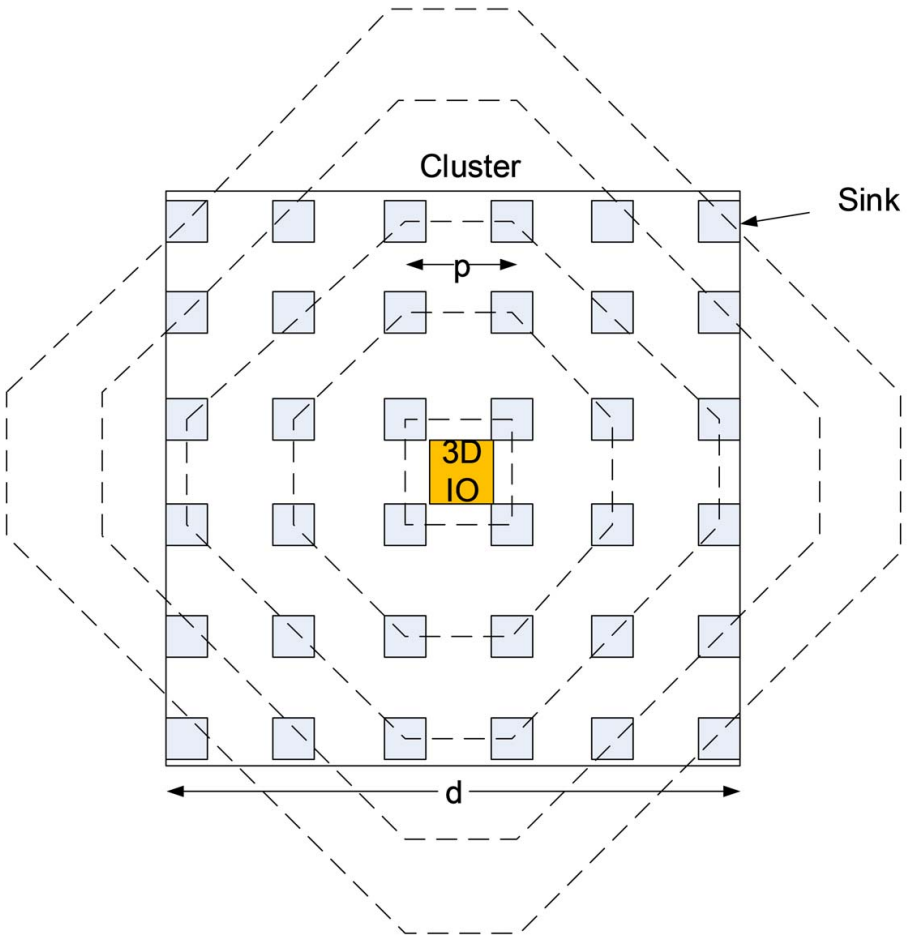


Figure 7.4: Data wirelength distribution.

7.3.3 3DIO Area and Power

The 3DIO power and area models are based on CACTI-IO [58]. The IO area equation is shown below in Equation (7.11). The fitting constants used are for a foundry 65nm technology to align with the CTS results.

$$A_{IO} = N_{IO} \cdot \left(A_0 + \frac{k_4}{min(R_{on}, 2 \cdot R_{TT1})} \right) + N_{IO} \cdot \left(\frac{1}{R_{on}} \right) \cdot k_5 \cdot (f_{IO}) + k_6 \cdot (f_{IO})^2 + k_7 \cdot (f_{IO})^3) \tag{7.11}$$

The IO power is a sum of the dynamic switching power, termination power and bias power as described in [58]. The load capacitance and termination frequency based threshold values have been nominally set based on a 3D interconnect. We use 750 MHz as the threshold for a pseudo-differential receiver consuming 100uA per IO, and 1500 MHz as the threshold for a termination consuming 1mA per IO. We also assume a static power overhead of 2.5mA for the PLL required in the asynchronous serial design per cluster. There are a number of energy-efficient serial IO designs that have been developed in the literature [73, 112, 91, 108, 33, 144], which could be applied to 3D interconnect as well, but we do not include all these schemes. Instead, we use a simple model for the asynchronous IO. The user using our flow can easily replace the 3DIO models with models from the library choices available and repeat the design space search that we illustrate below. Extending our model to include serial link design optimization [131], [129] may be interesting if the nature of these tradeoffs is to be studied in the context of the 3D clock clustering and 3DIO frequency.

7.3.4 Total Power and Area

The total area is calculated as a sum of clock and data wirelength, buffer area, and 3DIO area, as shown in Equation (7.12). We treat wire area and buffer area with equal weight for the purpose of our design space study, which might lead to large area values relative to the region area of the design, since multiple metal layers exist. One may wish to weight these unequally, e.g., if buffer area is a bigger concern than wire area, or vice-versa. The total power for the wire and buffers is proportional to their area (capacitance) and frequency, but also includes leakage and any static power which is independent of frequency but proportional to the number of buffers (see Equation (7.13)).

$$A_{total} = k_8 \cdot W_{clk} + k_9 \cdot W_{data} + A_{clk} + A_{data} + A_{IO} \tag{7.12}$$

$$P_{total} = (k_{10} \cdot W_{clk} + k_{11} \cdot W_{data} + k_{12} \cdot A_{clk} + k_{13} \cdot A_{data}) \cdot F_{clk} + k_{14} \cdot (A_{clk} + A_{data}) + P_{IO} \tag{7.13}$$

7.3.5 WNS and F_{max} for On-die and 3DIO

WNS is calculated for two paths, the first being the on-die path from launch in the cluster to capture in the 3DIO, and the second being the 3DIO die to die path.

For the on-die worst-case negative slack, all three synchronization schemes follow a synchronous full-cycle reg-to-reg timing, with the setup (WNS_{setup}) and hold (WNS_{hold}) slacks described as follows:

$$\begin{aligned} WNS_{setup} &= T_{per} - T_{data} - T_{hold\_fix} - T_{skew} - T_{jit\_setup} - T_{setup} \\ WNS_{hold} &= T_{hold\_fix} - T_{skew} - T_{jit\_hold} - T_{hold} \end{aligned} \tag{7.14}$$

where T_{per} is the clock period, T_{data} is the delay of the data as shown in Figure 7.2(a), T_{hold_fix} is the inserted delay to fix hold time, T_{skew} is the skew in the clock tree, T_{jit_setup} is the clock jitter, T_{jit_hold} is the 0-cycle uncertainty, T_{setup} and T_{hold} are the setup and hold times of the FF.

The worst negative slacks for the IO (WNS_{IO_setup}, WNS_{IO_hold}) for the three synchronization schemes are as follows.

Source-synchronous:

$$\begin{aligned} WNS_{IO\_setup} &= T_{per}/2 - T_{hold\_fix} - T_{skew} - T_{jit\_setup} - T_{setup} \\ WNS_{IO\_hold} &= T_{hold\_fix} - T_{skew} - T_{jit\_hold} - T_{hold} \end{aligned} \tag{7.15}$$

Synchronous:

$$\begin{aligned} WNS_{IO\_setup} &= T_{per}/2 - T_{data} - T_{hold\_fix} - \\ &\qquad T_{skew\_inter\_die} - T_{jit\_setup} - T_{setup} \\ WNS_{IO\_hold} &= T_{hold\_fix} - T_{skew\_inter\_die} - T_{jit\_hold} - T_{hold} \end{aligned} \tag{7.16}$$

Asynchronous:

$$
\begin{aligned}
WNS_{IO\_setup} &= T_{per}/8 - T_{jit\_setup} - T_{setup} \\
WNS_{IO\_hold} &= T_{hold\_fix} - T_{jit\_hold} - T_{hold} \\
T_{jit\_hold} &\simeq 0
\end{aligned}
\tag{7.17}
$$

For all three schemes, the skew and jitter are described as below, where T_{clk} is the clock insertion delay of the cluster clock. The fitted coefficients k_{15} to k_{19} are different for each clocking scheme.

$$
\begin{aligned}
T_{skew} &= k_{15} \cdot T_{clk} \\
T_{jit\_setup} &= k_{16} \cdot T_{clk} + k_{17} \cdot T_{per} \\
T_{clk} &= k_{18} \cdot A_{reg}/N_{clus} + k_{19} \cdot \frac{W_{clk} \cdot C_0 + N_{ff} \cdot C_g^{ff}}{\sqrt{(T_{tran}^{max})^2 - (T_0)^2}}
\end{aligned}
\tag{7.18}
$$

The skew for the inter-die synchronous case is much larger due to inter-die process variation (much larger k_{15}). T_{data} also follows the same form as the T_{clk} shown above. As described in Section 7.2, the source-synchronous case is not exposed to inter-die variation, and also does not have any data-delay eating into the setup slack due to the balance delay T_b shown in Figure 7.2(b).

F_{max} is calculated from the smallest T_{per} that still results in a positive WNS. F_{max} provides a better way to search for a valid design point and is also less prone to large % errors, unlike a WNS figure of merit when WNS is close to 0. For the asynchronous 3DIO, since the IO timing budget is mainly dominated by jitter and outside the scope of the timing flow described in Section 7.4, we assume 8000 Mbps as the maximum data rate. This means that the cluster frequency is limited to 1 GHz (due to 1:8 serialization) for the asynchronous case.

7.4 P&R and Timing Flow

In this section, we describe different 3D P&R flows according to the different choice of clock synchronization schemes. We use a 65nm TSMC library with our P&R flows which are realized using Synopsys IC Compiler I-2013.12-SP1. Before describing the details of our flows, we clarify the following assumptions in our design:

- There are three kinds of launch-capture paths: (i) the on-die paths from the launch to the capture FFs on bottom die (H_0); (ii) the paths going through the 3DIOs that are from the FFs on the bottom die to the FFs on the top die (H_c); and (iii) the on-die paths from launch to capture FFs on the top die (H_1). The H_0 and H_1 paths do not go back and forth between bottom and top die.

- 3DIOs are placed in the center of each cluster in a square array with 30 μm pitch.[11]

- Launch (capture) FFs of H_0 (H_1) on the bottom (top) die are uniformly distributed within a given cluster region. Pitch between any two neighboring uniformly distributed FFs is $\sqrt{\frac{A_{reg}}{N_{ff}}}$. Capture (launch) FFs of H_0 (H_1) on the bottom (top) die are placed immediately next to the 3DIOs. We assume that the rest of the logic, which is not included in the database, does not block such a placement assumption.

- H_c are unidirectional paths from bottom die to top die and only have IO circuits which are (i) double data rate (DDR) for the synchronous and source-synchronous schemes; and (ii) 1:8 serializer, 8:1 deserializer for the asynchronous scheme. These circuits are placed right next to the 3DIOs.

- In timing analysis, we use a 7.5% clock uncertainty that includes clock jitter and OCV margin.

- For timing constraints, we use 5%, 75% and 20% of the clock period as max transition time, max clock insertion delay and max clock skew, respectively.

7.4.1 Synchronous

In the synchronous scheme, the clusters in both dies share a common cluster clock tree that is balanced to endpoints in both dies. Since the commercial tool cannot synthesize the clock tree concurrently considering both dies, we use the flow shown in Figure 7.5. First, we generate a gate-level netlist and timing constraints file (i.e., SDC file) for a given BW, A_{reg}, N_c and f. We then place FFs, DDR circuits and 3DIOs based on our assumptions described above. To balance the clock tree on both dies, we synthesize the cluster clock tree on the top die, and then extract the maximum clock insertion delay T_{clk1} of the clock

[11] We have sampled the sensitivity of the P&R results to aspect ratio of the region area, non-square cluster shape, 3DIO pitch and technology. We see that the trends in our data are not significantly affected by these assumptions. This being said, we have not included these parameters in a full-factorial DOE.

tree as shown in Figure 7.2(a). Next, we annotate the delay to the clock 3DIO on the bottom die so that the commercial tool can consider the delay during the cluster clock tree synthesis of the bottom die. After CTS and routing on the bottom die, we propagate the T_{data} delays for the routing on the top die. Note that if the commercial tool performs the routing without consideration of T_{data}, the tool will insert many redundant hold buffers to compensate the clock insertion delay T_{clk1} in addition to the jitter T_{jit_hold} and FF hold time T_{hold}.

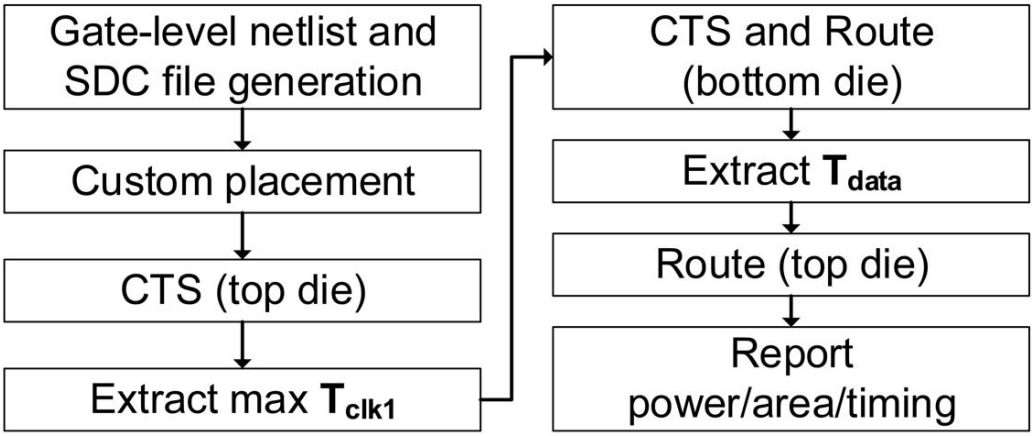


Figure 7.5: Flow for synchronous clocking scheme.

For timing analysis, we extract the netlist and parasitics (SPEF) of both the bottom and top dies, and merge the netlists and SPEFs as one netlist and SPEF file. We then use Synopsys PrimeTime to analyze setup and hold WNS. In timing analysis, we use best corner (*BC*) and worst corner (*WC*) for inter-die variation, and assign the same corner for paths that are on the same die. There are four combinations (i.e., *BC*-*BC*, *BC*-*WC*, *WC*-*BC*, *WC*-*WC*) and we report the worst setup and hold WNS out of the four combinations. We note that this is different from conventional timing analysis, which would analyze setup (hold) timing by assigning the worst (best) corner to the launch path and the best (worst) corner to the capture path, in consideration of intra-die variation. Here, we assign the same corner to the paths on the same die no matter whether the paths are launch or capture paths.

7.4.2 Source-Synchronous

The key feature of the source-synchronous scheme is to forward the clock to the clock 3DIOs which match the delays from the clock source to data 3DIOs on the bottom die. Figure 7.2(b) shows

an example launch-to-capture path from bottom die to top die. Since the launch delays (i.e., T_{data} + T_b) up to data 3DIOs including balance delays, and the delays (i.e., T_{clk0} + T_{clk1}) up to clock pins of capture FFs, are well balanced, the inter-die variation is negligible. Note that we take into account 3DIO interconnect variation and interconnect loading mismatch by adding 5% margin to the balance delay (i.e., $T_b = 1.05 \times T_{clk1}$). However, special IO circuitry is required to insert the balance delay (T_b); the additional IO circuits increase total power and area compared to the synchronous scheme.

Figure 7.6 shows our flow for the source-synchronous scheme. The first two steps (i.e., generation of the gate-level netlist, and custom placement) are the same as for the synchronous scheme. After custom placement on the bottom and top die, we synthesize the clock tree and route on the bottom die, and separately synthesize the clock tree only for the top die. We then extract T_b (i.e., T_{clk1}) for each capture FF and annotate the delays to the corresponding data 3DIOs. These annotated delays guide the commercial tool to insert proper T_{hold_fix} only considering T_{jit_hold} and T_{hold}.

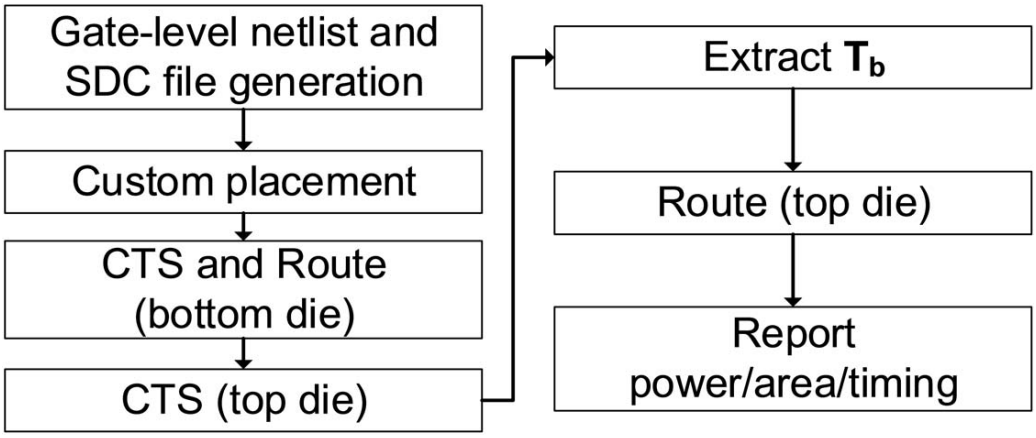


Figure 7.6: Flow for source-synchronous clocking scheme.

Timing analysis is performed in the same way as for the synchronous scheme.

7.4.3 Asynchronous

In the asynchronous scheme, the cluster clock is not propagated from bottom to top die, and separate PLLs for the IO circuits exist. There is no inter-die variation or clock skew. Therefore, we can run the traditional 2D flow on both dies separately. The only difference in the gate-level netlist is that the asynchronous scheme uses serializer and deserializer, instead of DDR module.

7.5 Results

We now describe the DOE used to generate modeling data, the model-fitting results, and the design space visualization.

7.5.1 DOE

We construct a DOE where we vary the bandwidth (10-200 GB/s), region area (25-100 mm^2), 3DIO clock frequency (ranges based on the clocking scheme: synchronous (100-2000 Mbps), source-synchronous (1500-4000 Mbps) and asynchronous (3500-8000 Mbps) and number of clusters (1-25). We have collected data for over 256 design implementations for each of the three clock synchronization schemes. We have chosen these ranges based on typical 3D bandwidth requirement ranges [189] and typical speeds based on synchronization schemes [26].

We execute our DOE for the synchronous, source-synchronous and asynchronous schemes by running the P&R and timing flow described in Section 7.4. Since our P&R flow makes the various assumptions described in Section 7.4, we limit our DOE study to these design assumptions. We then fit and validate our clock tree and data path models based on the data from the DOE by logging all intermediate nodes of the directed graph in Figure 7.3.

7.5.2 Model Fitting

We use ANN models to fit the analytical models, as described in Section 7.3. We make multiple runs with different training, validation and test data sets for improved generality and robustness of the resulting models.

The models for the internal nodes of the directed graph (clock wirelength, data wirelength, clock buffer area, data buffer area) fit within +/-20% error. Shown in Figures 7.7 and 7.8 are the distributions of % model error for total on-die area and power across the three synchronization schemes for all cases in the DOE.

The % errors of area models are 0.52/-17.8/19.76 (mean/min/max) for the synchronous scheme, -0.189/-11.28/8.56 for the source-synchronous scheme, and -0.08/-9.83/7.32 for the asynchronous scheme.

Also, the % errors of power models are 0.099/-13.4/13.39 for the synchronous scheme, 0.018/-10.4/9.75 for the source-synchronous scheme, and -0.008/ -16.65/13.22 for the asynchronous scheme. In general, the models fit within +/-15% accuracy across a large range of bandwidth, region area, 3DIO clock frequency and number of clusters for all three synchronization schemes.

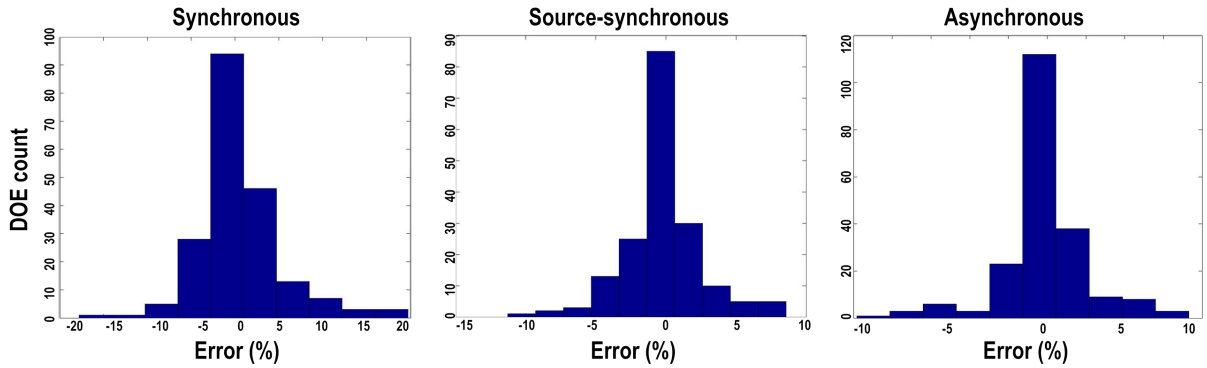


Figure 7.7: Area model percentage error.

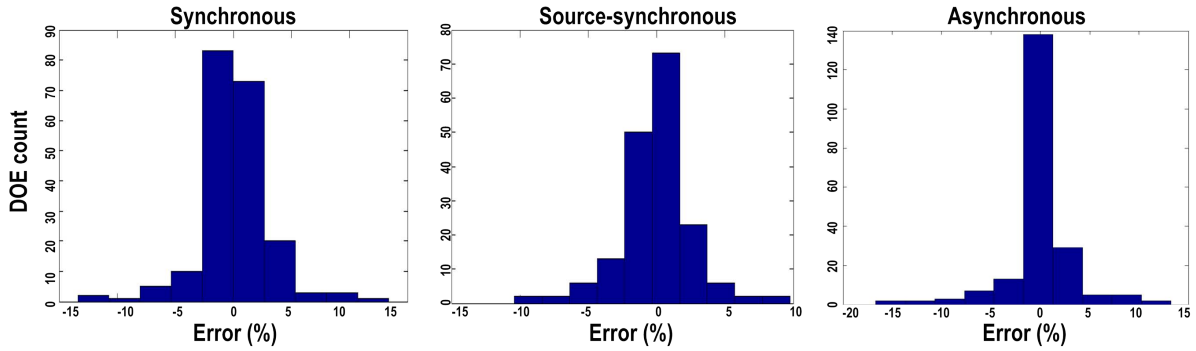


Figure 7.8: Power model percentage error.

The F_{max} error % is shown in Figure 7.9. The mean/min/max % errors of F_{max} models are -0.1/-14.78/13.91 for the synchronous scheme, 0.1/-10.02/10.15 for the source-synchronous scheme, and -0.1877/-9.47/ 12.3 for the asynchronous scheme. We use the F_{max} model to evaluate which design points meet timing, and hence are valid entries in the design space for power and area optimization. As indicated in Section 7.4, the timing for the on-die case is quite similar for the three synchronization schemes, but the 3DIO synchronization is very different: the synchronous scheme has the lowest F_{max}, followed by the source-synchronous scheme and then the asynchronous scheme, which achieves the highest 3DIO frequency.

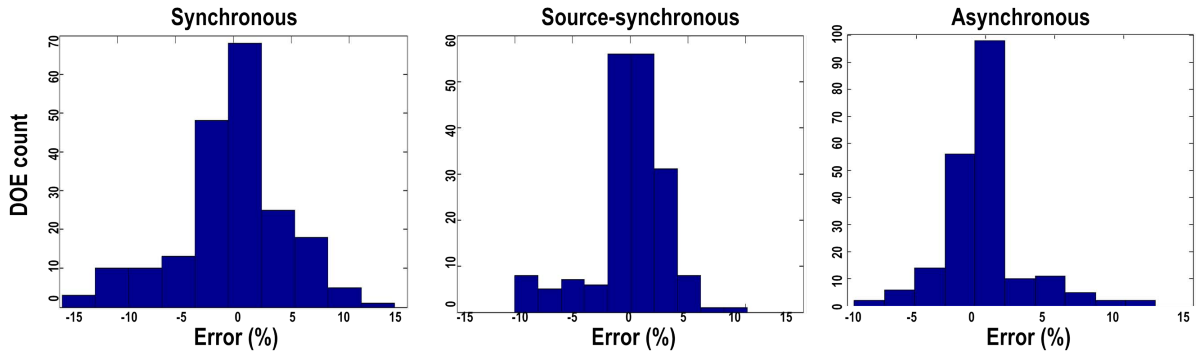


Figure 7.9: F_{max} model percentage error.

As noted above, use 3DIO power and area analytical models from CACTI-IO [58]; and CACTI-IO model fitting results are given in [58]. We combine the CACTI-IO models for the 3DIO power and area with the ANN models. Given these models for power, area and timing, we may visualize the design space, as we discuss next.

7.5.3 Design Space

For an optimal power solution given an upper bound on area, we want to pick the lowest 3DIO frequency that meets the area constraint (as frequency goes lower the 3DIO area, sink area, and wire area all go up). This is because lower-frequency and wider buses are lower power for both the 3DIO and the clock tree. Once we have the lowest 3DIO frequency that meets this area constraint, we pick the 3DIO topologies that work at this frequency and search for the one with the lowest power. As the area constraint becomes tighter, the lowest 3DIO frequency that meets this area constraint becomes higher and requires us to move toward asynchronous/highly-clustered solutions. The caveat to this is that once the 3DIO frequency becomes high enough to require termination, the power efficiency of the 3DIO begins to improve as we scale frequency higher. This makes for an interesting tradeoff as we get to the asynchronous synchronization scheme. Further, the asynchronous synchronization scheme affords the unique advantage of having a slower cluster clock frequency for an equivalent 3DIO frequency due to the 1:8 serialization, so the CTS power can be quite low at relatively high 3DIO frequencies.

The design space is depicted in Figure 7.10. We use axes of max power constraint versus max area constraint, and show iso-bandwidth lines. We observe that the iso-bandwidth contour lines hit a

vertical and horizontal *wall*. That is, for a given bandwidth, there is a minimum power and minimum area achievable across all synchronization schemes, number of clusters and cluster clock frequency.

Figures 7.11, 7.12 and 7.13 show how these three variables change as we move along the iso-bandwidth lines. Figure 7.11 shows the synchronization scheme, Figure 7.12 shows the number of clusters and Figure 7.13 shows the cluster clock frequency. The black dotted lines in Figures 7.10, 7.12 and 7.13 are the overlay of the three synchronization schemes shown in Figure 7.11.

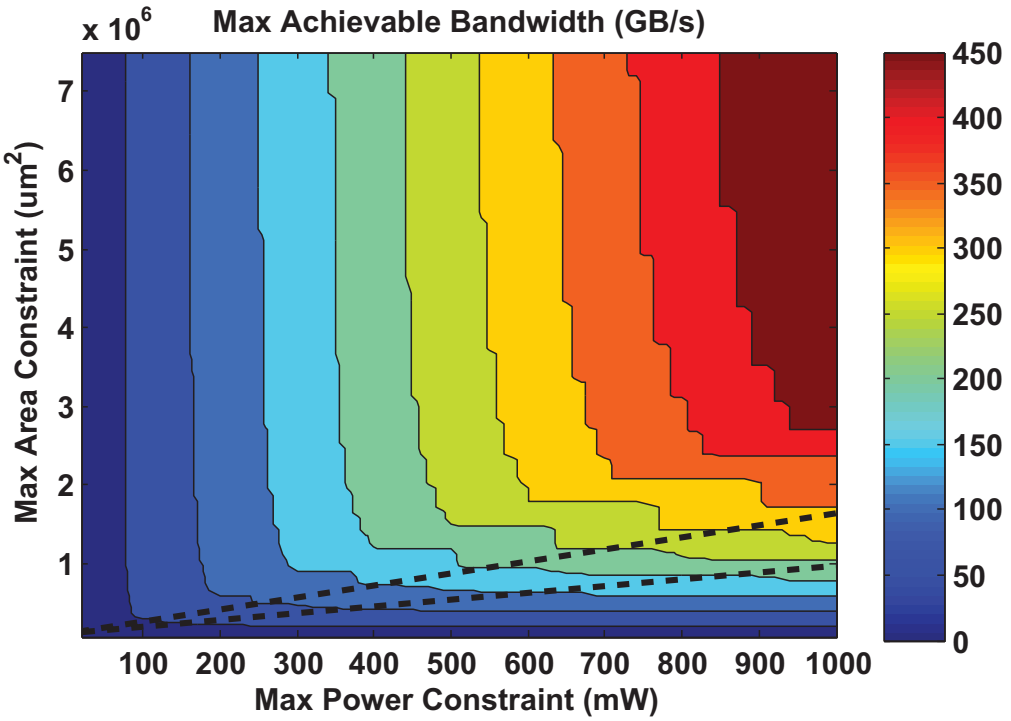


Figure 7.10: Maximum achievable bandwidth for given power and area constraints.

We observe that the asynchronous scheme is area-efficient while the synchronous scheme is power-efficient. The source-synchronous scheme provides a valuable tradeoff between power and area along the knee of the iso-bandwidth curve. The interesting tradeoffs between the schemes occur along these knee points as we change the power/area constraint tradeoffs. The synchronous scheme achieves lower power at lower frequencies since there is no overhead associated with balancing delays (of the source-synchronous scheme) or with serialization (of the asynchronous scheme). The asynchronous scheme achieves a lower area as it can operate at a higher frequency and needs fewer 3DIO.

We also observe that the optimal cluster clock frequency and number of clusters – for a given area

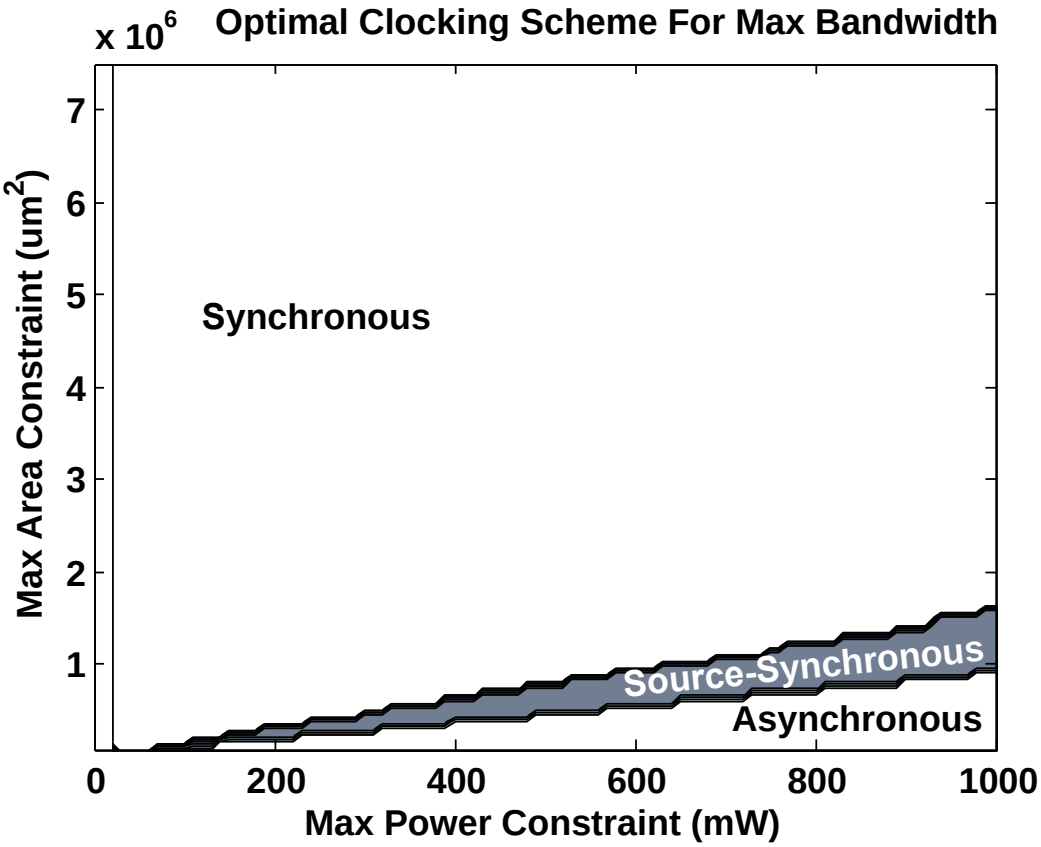


Figure 7.11: Guidance for synchronization scheme for given power and area constraints.

constraint – both increase as we increase bandwidth (or the power constraint). However, the trend for number of clusters is not monotonic when we change synchronization schemes, and is also sensitive to the edge of the search hypercube (e.g., the largest bandwidth in the search window is 450 GB/s, so when this is reached, the bandwidth lines larger than 450 GB/s are limited to 450 GB/s, and the number of clusters stops following the trend as shown in the right top of Figure 7.12).

7.6 Summary

We have presented a power, area and timing model for 3DIO and CTS that includes clustering and three different clock synchronization schemes (synchronous, source-synchronous, asynchronous). The model combines analytical closed-form expressions with metamodel-based fitting to achieve within 10%

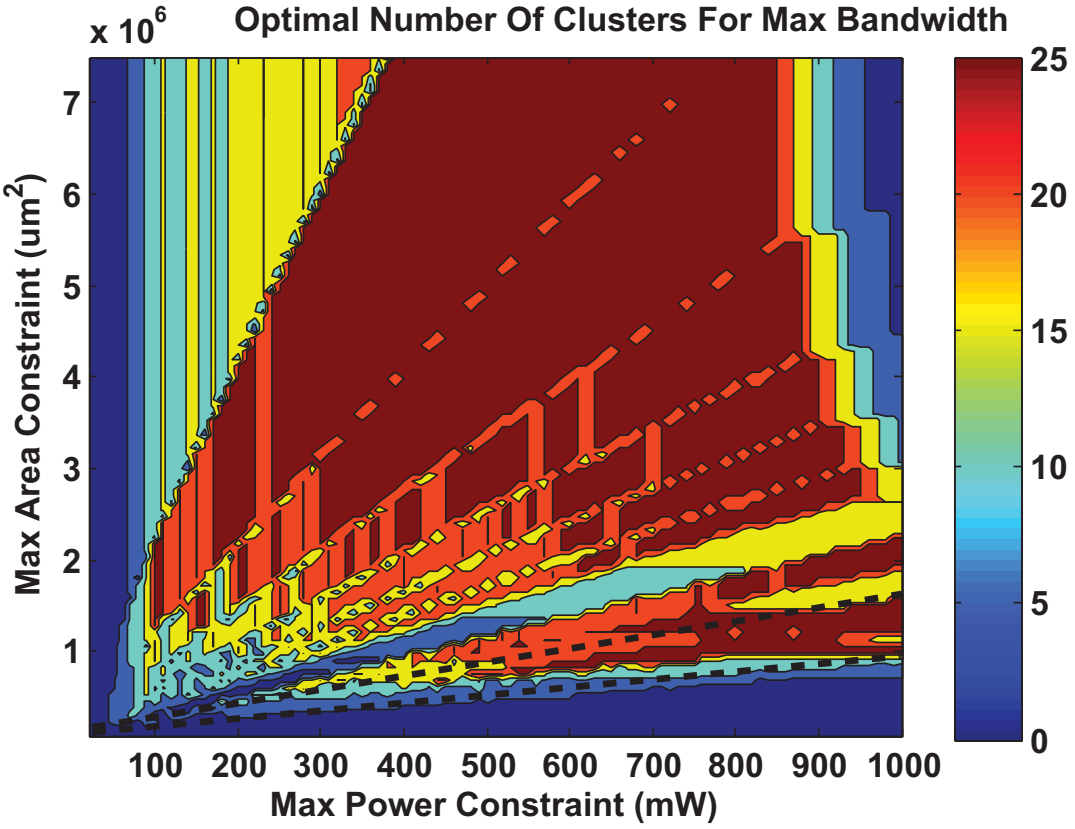


Figure 7.12: Optimal number of clusters for given power and area constraints.

error for power, area and timing across a large range of bandwidths, region areas, numbers of clusters and 3DIO frequencies. Such a model enables us to visualize the design space and support architectural optimization with respect to choice of synchronization scheme, number of clusters and 3DIO frequency for given power and area constraints, as well as bandwidth targets. We believe that our modeling methodology will enable architects to study and optimize such a design space.

Our results show an interesting structure of the design space for the synchronization scheme, number of clusters and 3DIO frequency, wherein the area and power constraints on the design clearly affect the optimal design choices. Generally, the synchronous scheme is more power-efficient but has poor area efficiency, while the asynchronous scheme is more area-efficient and reduces the number of IO but has poor power efficiency. The source-synchronous scheme provides a good design point along the knee of the power-area tradeoff for a given bandwidth. Further optimization of the asynchronous serial IO could extend the asynchronous design space by lowering the power.

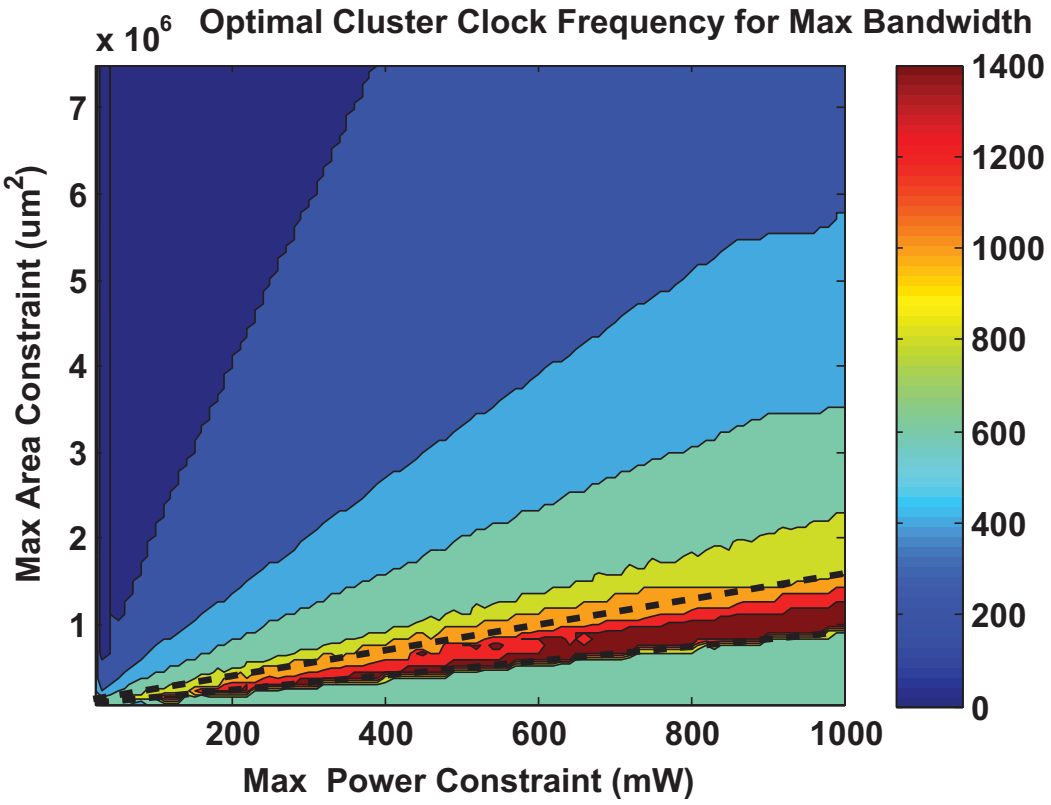


Figure 7.13: Cluster clock frequency based on power and area constraints.

7.7 Acknowledgments

Chapter 7 contains a reprint of S. Bang, K. Han, A. B. Kahng and V. Srinivas, "Clock Clustering and IO Optimization for 3D Integration", *Proc. ACM International Workshop on System-Level Interconnect Prediction*, 2015. The dissertation author is a main contributor to this paper. I would like to thank my coauthors Samyoung Bang, Kwangsoo Han and Andrew B. Kahng.

Chapter 8

PDN Co-Optimization

Apart from pathfinding for signal interconnect, which we have explored in the past few chapters, another critical piece of off-chip interconnect is the power delivery network (PDN). The PDN consists of the power supply and ground connections. The off-chip portion of the PDN is a critical piece of the overall PDN as it contributes to a significant portion of the overall voltage droop. It consists of the package and printed circuit board (PCB), that includes the PDN routing, the capacitors and the voltage regulator. The off-chip PDN is constrained by the cost of the package and PCB, and also competes for routing resources with the signal interconnect and the various components on the PCB. Pathfinding for the off-chip PDN begins with floorplanning the die of the SoC along with the package and the PCB. Such a co-design requires that the pinmap of the package (the BGA assignment) and the bumpmap of the die (the bump assignment) are done with consideration to the die, package and PCB design. This also allows for the power supply and ground bumps and balls to be assigned in a manner that the PDN is robust and the signal routing is optimal. In this chapter, we explore PDN co-optimization between die, PKG and PCB from a bump and BGA assignment perspective.

8.1 Introduction

Due to lack of design automation and complex trial and error design iterations, Package (PKG) Power Delivery Network (PDN) design in a modern SoC is challenging and time-consuming. Multiple

PKG PDN modeling tools exist today and different companies have different methodologies and flows to handle the tradeoff between accuracy and runtime. Often a 3D full-wave electromagnetic (EM) solver is used as the golden signoff tool (*Golden* tool), providing accurate PDN modeling at the cost of long runtime. A quick assessment tool (*Quick* tool) that is used for quick PDN inductance modeling during the design phase offers fast runtime at the cost of accuracy.[12] The discrepancy between Golden tool and Quick tool may introduce more iterations during the overall design cycle.

PKG PDN quality is usually assessed by measuring the inductance of each *die bump*. To the best of our knowledge, there is no existing tool that can predict achievable bump inductance in the early design stage before an actual design layout implementation. As a result, an unpromising pinmap, including micro bump and *solder ball* assignment, can only be identified after trial design.

The primary input of a PKG PDN design problem is the pinmap, which includes locations and supplyrail assignments for both die bumps and solder balls. Even a small change in pinmap and layout can cause large variations in bump inductance. Figure 8.1(a) shows a partial initial layout and a map of bump inductance. Figure 8.1(b) shows a variant of initial layout where a few BGA balls, metal traces and vias are removed and the corresponding percentage change (increase) in bump inductance, relative to inductance in the original layout, is shown.

Predicting bump inductance without routing is challenging because bump inductance depends not only on the pinmap, but also on the routing resource allocation including metal utilization, via availability, reference plane completeness, etc. Figure 8.2(a) shows die bump locations of a pre-layout PKG design. Figure 8.2(b) shows the corresponding part of post-layout design. Other examples that convey the nature of PKG substrate routing, showing both bump and ball locations, are given in Figure 8.4 below.

To close the gap between Quick tool and Golden tool, a more accurate bump inductance predictor is desired. Such a predictor is difficult to build because (i) the predictor needs to abstract and understand complex interactions between various metal shapes (e.g., solid metal fill and thin metal trace), vias and metal stackup, and (ii) traditional EM simulation tools are not applicable due to their long runtimes.

[12]The two commercial PKG PDN analysis tools, "Golden" and "Quick", are among various commercial tools that are listed under Power Analysis and Optimization by the industry analyst firm Gary Smith EDA [237]. The universe for these tools includes *Cadence Sigrity XtractIM* [172], *Ansys Sentinel PSI [166]*, *Applied Simulation Technology ApsimSPE [167]* and *Mentor HyperLynx Power Integrity [218]*. We are unable to identify the tools more specifically due to license restrictions and sensitivity of EDA vendors.

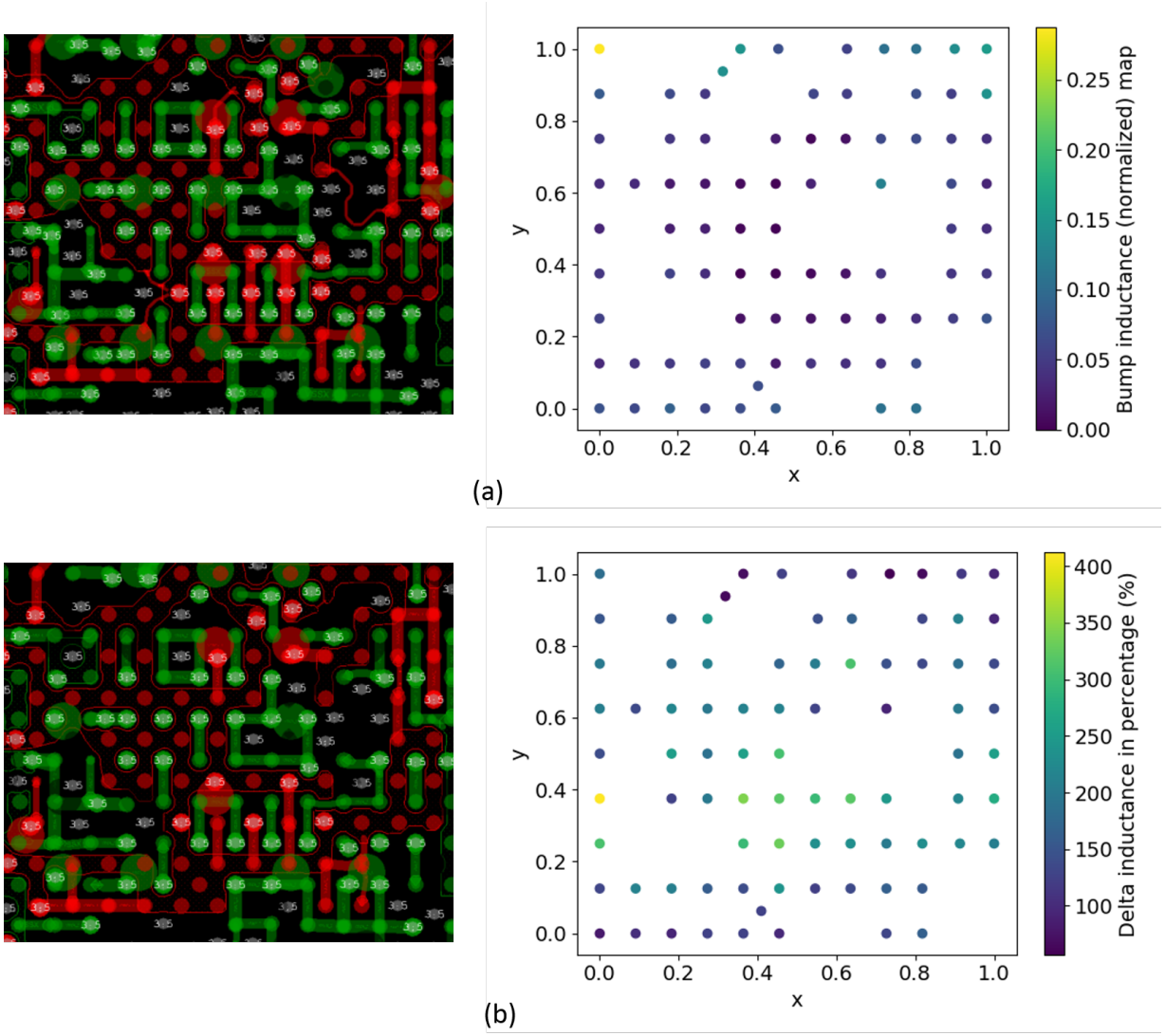


Figure 8.1: (a) Original layout fragment and map of bump inductance (a.u.). (b) Perturbed (degraded) layout with several balls, metal traces and vias removed, and map of percentage change in per-bump inductance relative to original layout.

In this work, we use machine learning techniques to achieve accurate modeling and prediction of bump inductance. Given the pinmap and PKG technology information, but *without* any PKG layout (i.e., routing) information, we make a prediction of bump inductance in a well-optimized post-layout design. In what follows, we call this estimation at pre-layout stage an estimate of *achievable* bump inductance. Moreover, given pinmap and PKG layout information, we also make a prediction of post-layout bump inductance. In what follows, we call this estimation at post-layout stage an estimate of *actual* bump

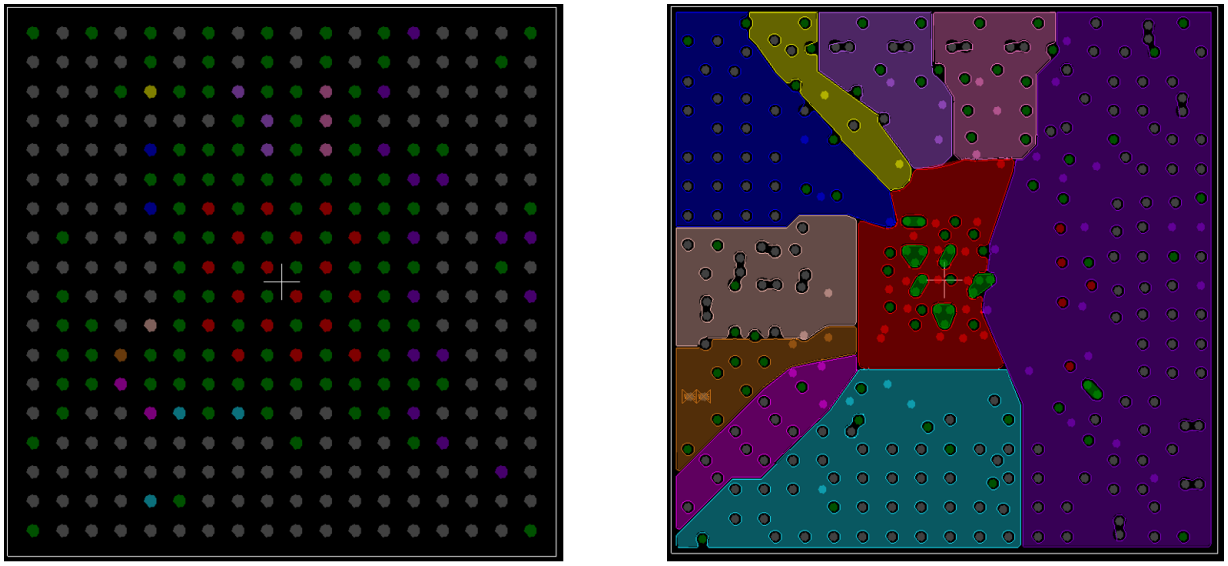


Figure 8.2: An example of package PDN design: (a) pre-layout, and (b) post-layout. Colors indicate distinct rails. Figure courtesy of ASE Group.

inductance. Our models of *achievable* and *actual* bump inductance are aimed at helping to prevent expensive iterations of PKG PDN designs. Our main contributions are summarized as follows.

- To the best of our knowledge, we are the first to propose a predictive modeling methodology which addresses the need for early-stage *achievable* bump inductance assessment.
- We identify appropriate modeling parameters, and separately apply predictive modeling methodology, to shrink the gap between Golden tool and Quick tool for design-phase bump inductance evaluation.
- We validate our bump loop inductance predictive modeling methodology and predictive model across various industry PKG designs used in modern product SoCs, and show that our model achieves an accurate prediction of bump inductance (and, further, is more accurate than the Quick tool).

The remainder of this chapter is organized as follows. Section 8.2 overviews the main approaches to bump inductance modeling. Section 8.3 describes our predictive modeling methodology. We describe our experimental setup and results in Section 8.4, and give conclusions in Section 8.5.

8.2 Approaches to Bump Inductance Modeling

An extensive literature on bump inductance modeling mostly focuses on modeling bump inductance by solving EM equations. For excellent overviews, the reader is referred to [127] and [136]. Shi et al. [127] summarizes the three categories of bump inductance models: lumped models, distributed models and S-parameter models.

A *lumped element* model lumps components in the system (e.g., bumps and BGA balls) together to reduce computational complexity [140] [93]. Therefore, PDN in a lumped model consists of a small number of RLC components. As a result, a lumped model usually has poor accuracy as it does not accurately capture the distributed nature of package routing.

A *distributed model* improves model accuracy from lumped models by introducing more RLC components. The partial element equivalent circuit (PEEC) method [9] is applied to achieve more accurate inductance values. However, with the increased number of RLC components, runtime becomes the limitation in applying distributed models.

An *S-parameter model,* which treats the PDN as a black box, is a high-bandwidth model [141]. An incident wave with varying frequencies is applied to the input port, and reflected wave and transmitted wave are measured at the input and output ports; an S-parameter matrix is then constructed. This S-parameter matrix can be post-processed to a Z-matrix, i.e., may separate out the imaginary part and then calculate the loop inductance. S-parameter models are widely used for PKG PDN signoff.

As mentioned in Section 8.1, PKG PDN modeling tools from different companies handle the tradeoff between accuracy and runtime differently. Figure 8.3(a) shows the comparison of extracted bump inductance from Golden tool and Quick tool. Figure 8.3(b) shows the percentage difference distribution between Quick tool and Golden tool. We observe that the percentage difference between Quick tool and Golden tool can exceed 500%, which may mislead the PKG PDN optimization during the design phase.

Generally speaking, lumped model-based approaches suffer from accuracy challenges, while distributed and S-parameter model-based approaches require significant bandwidth of EM and circuit simulation tools. All of these approaches cause many iterations of design cycles, as well as intensive manual interactions.

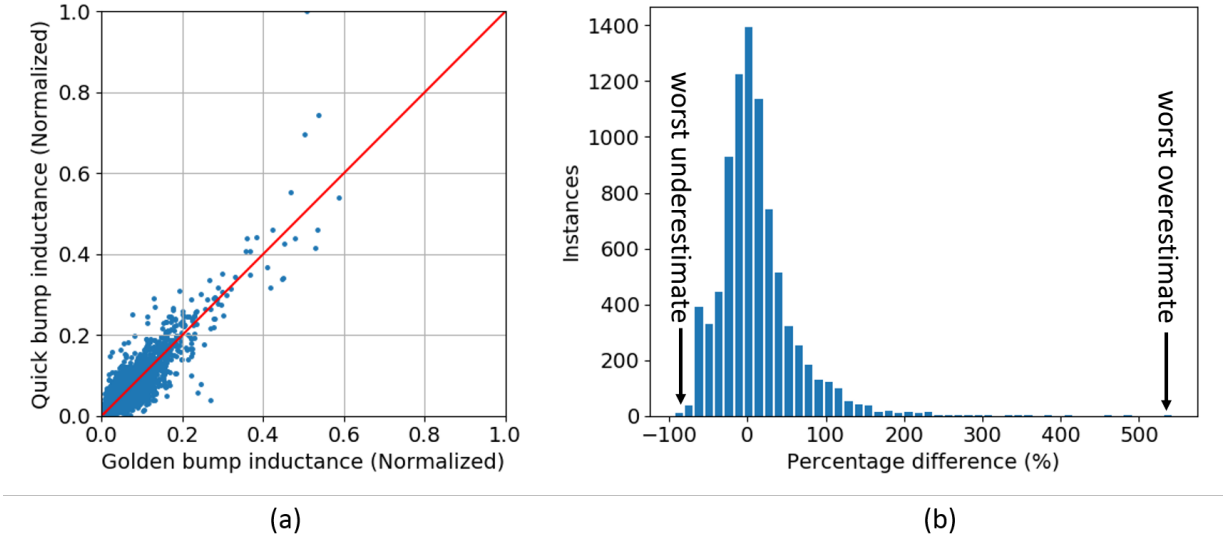


Figure 8.3: Comparison between Quick tool and Golden tool. (a) Golden versus Quick bump inductance (a.u.). (b) Histogram of percentage difference between Quick tool and Golden tool.

8.3 Our Methodology

In this section, we describe our predictive modeling methodology. Our modeling methodology includes (i) the selection of model parameters that impact bump inductance, and (ii) the application of machine learning techniques to capture the complex interactions between model parameters for accurate bump inductance prediction.

8.3.1 Selection of Model Parameters

Bump loop inductance is composed of self inductance and mutual inductance. Various components of bump inductance are determined by different factors including return path length, die bump grouping, solder ball group, etc. [229]. Therefore, in order to model *achievable* and *actual* bump inductance, it is necessary to find a set of parameters that abstracts and describes the characteristics of a die bump. We validate our selection of model parameters by studying the impact of each model parameter in Section 8.4. Figure 8.4 illustrates some notations we use in our parameters.

- ***VDD:*** Power rail that the bump of interest belongs to.
- ***GND:*** Ground rail.

- ***Other:*** Power rails other than the power rail that the bump of interest belongs to.

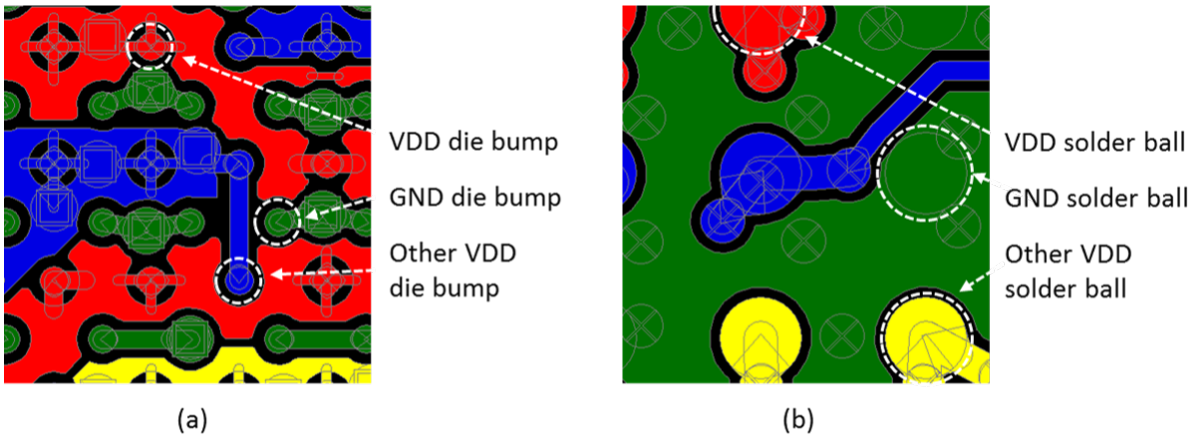


Figure 8.4: Illustration of notations in the model parameters.

For *achievable* bump inductance prediction during the early pre-layout stage of PKG PDN design, only pinmap and package technology information are available. However, *actual* bump inductance prediction can further leverage detailed layout information to make more accurate predictions. Thus, it is necessary to define different sets of parameters to distinguish between *achievable* bump inductance prediction and *actual* bump inductance prediction. We study the impact of each set of model parameters in Section 8.4.

Table 8.1 lists the parameters that we use in modeling. We use model parameters that comprehend various aspects of die bump inductance.[13] *P1, P2* and *P3* give an estimate of the primary return path length that is critical to loop inductance of a micro bump. *P4* and *P5* give an estimate of likelihood of bump grouping using metal fill. Bumps that are connected with metal fill tend to have lower inductance values. *P4, P5* and *P6* give an estimate of routing congestion. *P7* gives information on partial and mutual inductance contributed by vias, bumps and BGA balls. *P8* gives the current information on each bump.[14] *P9* indicates the availability of routing resource. *P10*, *P11* and *P12* indicate routing congestion in the surrounding area of each bump.

We divide our model parameters into three categories: pinmap-dependent (*PiM*), design-dependent

[13] A property of the set of parameters which we choose is that the number of model parameters per bump does not depend on the package size.

[14] Current information is usually only available at a per-block level rather than a per-bump level [229]. We derive reasonable per-bump current by smoothing a given per-block current map with a smoothing radius of four (4) bump pitches.

(*Des*) and layout-dependent (*Lay*). We use *PiM* and *Des* parameters to build a model for early-stage *achievable* bump inductance prediction. We use *PiM*, *Des* and *Lay* parameters to make an *actual* bump inductance prediction for design-phase evaluation.

Table 8.1: Parameters used in our modeling.

Idx	Parameter	Description	Type
$P1$	$d_{GNDBall}$	Distance to closest GND ball	*PiM*
$P2$	$d_{VDDBall}$	Distance to closest VDD ball	*PiM*
$P3$	$d_{GNDBump}$	Distance to closest GND bump	*PiM*
$P4$	$\#VDDBump[]$	Array of #bump from same supply rail within radius of {1,2,3,4} bump pitches	*PiM*
$P5$	$\#supplyrail[]$	Array of #supplyrail within radius of {1,2,3,4} bump pitches	*PiM*
$P6$	$\#OtherBump[]$	Array of #bump in different supplyrail within radius of {1,2,3,4} bump pitches	*PiM*
$P7$	*thickness*	Thickness of the PKG design	*Des*
$P8$	*current*	Derived per-bump current	*Des*
$P9$	$\#layer$	Number of metal layers used	*Des*
$P10$	$Util[]$	Array of metal utilization within radius of {1, 2, 3, 4} bump pitches	*Lay*
$P11$	$ViaCnt[]$	Array of via count within radius of {1, 2, 3, 4} bump pitches	*Lay*
$P12$	$TraceCnt[]$	Array of trace count within radius of {1, 2, 3, 4} bump pitches	*Lay*

8.3.2 Modeling Techniques

We use both linear and nonlinear learning-based algorithms such as Artificial Neural Network (ANN) [46], Support Vector Machine (SVM) [46] regression, and Multivariate Adaptive Regression Spline (MARS) [37]. For each technique, we use three-fold cross-validation to ensure the generality of the model (i.e., comparable mean-square errors (MSEs) between training and testing datasets). We use grid search to determine the length of each array parameter in the model. For each model parameter that we use, we normalize it to [0,1] before it is applied to learning-based algorithms. Nonlinear algorithms are more effective in capturing complex interactions between model parameters.

We also explore metamodeling techniques such as piecewise-linear (PWL) hybrid surrogate modeling (HSM) [61]. Data points are divided into four bins according to ANN prediction and all predictions from the three modeling techniques are combined to make the final prediction. Figure 8.5 shows the HSM modeling flow we use in this work, where $w_{i,j}$ indicates the weight of predicted response for i^{th} bin from j^{th} modeling technique. We implement our ANN in JMP Pro 13 [208]. We divide data points into four equally sized bins using predicted bump inductance threshold $t1$, $t2$, $t3$ and $t4$. We implement SVM regression and MARS in Python3 [231] using scikit-learn [240] and py-earth [230] packages, respectively.

$$y(x) \begin{cases} w_{1,1}y(x)_{ANN} + w_{1,2}y(x)_{SVMR} + w_{1,3}y(x)_{MARS}, & y(x)_{ANN} < \text{t1} \\ w_{2,1}y(x)_{ANN} + w_{2,2}y(x)_{SVMR} + w_{2,3}y(x)_{MARS}, & \text{t1} \leq y(x)_{ANN} < \text{t2} \\ w_{3,1}y(x)_{ANN} + w_{3,2}y(x)_{SVMR} + w_{3,3}y(x)_{MARS}, & \text{t2} \leq y(x)_{ANN} < \text{t3} \\ w_{4,1}y(x)_{ANN} + w_{4,2}y(x)_{SVMR} + w_{4,3}y(x)_{MARS}, & \text{t3} \leq y(x)_{ANN} \end{cases}$$

Figure 8.5: Illustration of piecewise-linear hybrid surrogate modeling based on ANN prediction.

8.3.3 Reporting Metrics

In addition to the R^2 value that is typically used to assess regression model quality, we also consider different accuracy aspects of the predictive bump inductance model when reporting results. Table 8.2 shows the various reporting metrics we use in this work. For each experiment in Section 8.4, we plot the normalized *actual* die bump inductance versus the predicted die bump inductance.[15] Also, we plot the percentage error histogram for each experiment.

8.4 Experiments

In this section, we describe our design of experiments and show our experimental results. First, we describe our experiments for model parameter validation. Second, we perform four experiments to assess

[15]We are not able to provide absolute errors due to product confidentiality constraints.

Table 8.2: Description of reporting metrics.

Notation	Meaning
R^2	Coefficient of determination
AAPE (%)	Average absolute percentage error
90^{th}-pct Worst Overestimate (%)	90% percentile value of sorted overestimating percentage errors
90^{th}-pct Worst Underestimate (%)	90% percentile value of sorted underestimating percentage errors
95^{th}-pct Worst Overestimate (%)	95% percentile value of sorted overestimating percentage errors
95^{th}-pct Worst Underestimate (%)	95% percentile value of sorted underestimating percentage errors

and measure bump inductance model quality and accuracy. Then we study the correlation between Golden tool and Quick tool. Finally, we check for the generality of our model in direct comparison with Golden tool results. Recall from Section 8.1 above that *achievable* refers to a prediction made without any routing layout information, while *actual* refers to a prediction made with routing layout information.

- **Experiment 1. Parameter set sensitivity.**
- **Experiment 2. Individual parameter sensitivity.**
- **Experiment 3. *Achievable* bump inductance modeling.**
- **Experiment 4. *Actual* bump inductance modeling.**
- **Experiment 5. Golden tool and Quick tool correlation.**
- **Experiment 6. Model generality.**

8.4.1 Design of Experiments

We use 17 industry PKG designs across various PKG technologies to build and validate our learning-based predictive model. All 8.5K data points are extracted from 2-layer, 3-layer and 4-layer designs with a variety of core thicknesses. The training time of our model is 7.5 hours on a 2.6GHz Intel

Xeon dual-CPU server. This is a one-time overhead. After training, the inference time is approximately 270 seconds for every 1K data points.

We use 67% of the data points for training and the remaining 33% of the data points for testing. We apply 3-fold cross validation in the modeling process. We perform 10 runs with different random seeds to denoise the impact of random initial weights in the modeling process.

Experiment 1. We study the sensitivity of the parameter set (i.e., PiM, Lay and Des). We remove one parameter set from the parameter list at a time and show the degradation in model accuracy after each parameter set removal.

Experiment 2. We study the sensitivity of each parameter (i.e., $P1$, $P2$, etc.). We remove one individual parameter from the parameter list at a time and show the degradation in model accuracy after each individual parameter removal.

Experiment 3. We build a model based on *PiM* and *Des* parameter sets. We validate our *achievable* bump inductance model by comparing the bump inductance prediction against the Golden tool result.

Experiment 4. We build a model based on *PiM*, *Lay* and *Des* parameter sets. We validate our *actual* bump inductance model by comparing the bump inductance prediction against Golden tool, and showing the discrepancy between Golden tool and Quick tool.

Experiment 5. We correlate Golden tool and Quick tool. We build a model based on *PiM*, *Lay*, *Des* and Quick tool results to predict the Golden tool result. We validate our correlation model by comparing the bump inductance prediction against the Golden tool result.

Experiment 6. We study the generality of our bump inductance model. We build a model based on an initial design and apply the model to a variant of the same design. We demonstrate the generality of our model by comparing the bump inductance prediction for the design variant against the Golden tool result.

8.4.2 Results for Experiments

We study model parameter sensitivity, demonstrate bump inductance quality and correlate between Golden tool and Quick tool with the six experiments described in Section 8.4-8.4.1.

Results for Experiment 1. We study the sensitivity of each parameter set on bump inductance model accuracy. Figure 8.6 shows the normalized root-mean-square error (*RMSE*) values, for both training and testing datasets, of model based on *PiM*, *Des* and *Lay* parameter sets and models based on every pair of parameter sets. We observe that RMSE degrades up to 170% and 133% for training and testing datasets respectively with the removal of PiM, which implies that PiM is the dominant parameter set among all parameter sets.

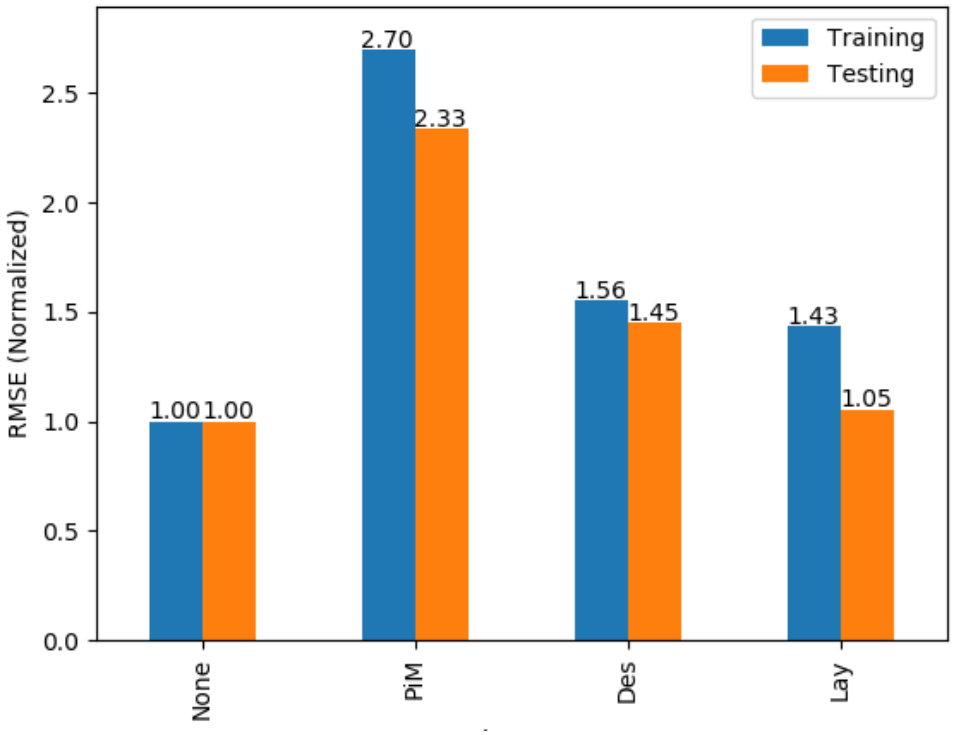


Figure 8.6: Sensitivity of RMSE (a.u.) to parameter set removal.

Results for Experiment 2. We study the sensitivity of each individual parameter on bump inductance model accuracy. Figure 8.7 shows the normalized RMSE values, for both training and testing datasets, of model based on all parameters and models with one parameter removed. We observe that RMSE degrades up to 135% and 102% for training and testing datasets, respectively, when just a single parameter is removed.[16]

[16]Readers may notice that removal of individual parameter (*P10*, *P11* or *P12*) causes more degradation in model accuracy

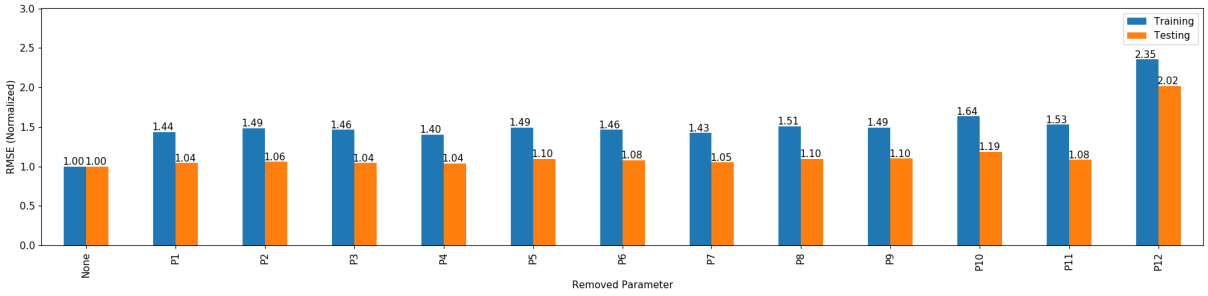


Figure 8.7: Sensitivity of RMSE to individual parameter removal.

Results for Experiment 3. We build a model based on *PiM* and *Des* parameter sets and compare the accuracy of our *achievable* bump inductance prediction against the Golden tool result. Figure 8.8(a) shows the predicted *achievable* and Golden bump inductance values and Figure 8.8(b) shows the distribution of percentage error. Our model can predict *achievable* bump inductance within an absolute percentage error of 57.0% for more than 95% of all data points.

Table 8.3 shows the accuracy metrics of the model. We observe that our model can predict *achievable* bump inductance with average absolute percentage errors of 21.2% and 18.7% for training and testing datasets respectively. We achieve R^2 values of 0.89 and 0.90 for training and testing datasets respectively.

Table 8.3: Accuracy metrics for *achievable* bump inductance model.

Metric	Training	Testing
R^2	0.89	0.90
AAPE (%)	21.2	18.7
90^{th}-pct Worst Overestimate (%)	54.7	46.7
90^{th}-pct Worst Underestimate (%)	-34.6	-32.4
95^{th}-pct Worst Overestimate (%)	70.6	59.1
95^{th}-pct Worst Underestimate (%)	-43.3	-37.6

Results for Experiment 4. We build a model based on *PiM*, *Des* and *Lay* parameter sets and compare the accuracy of our *actual* bump inductance model against the Golden tool result. We also compare the extracted bump inductance value from both Golden tool and Quick tool. Figure 8.9(a) shows the Golden versus predicted bump inductance and Figure 8.9(b) shows the percentage error distribution

compared to removal of *Lay* parameter set.

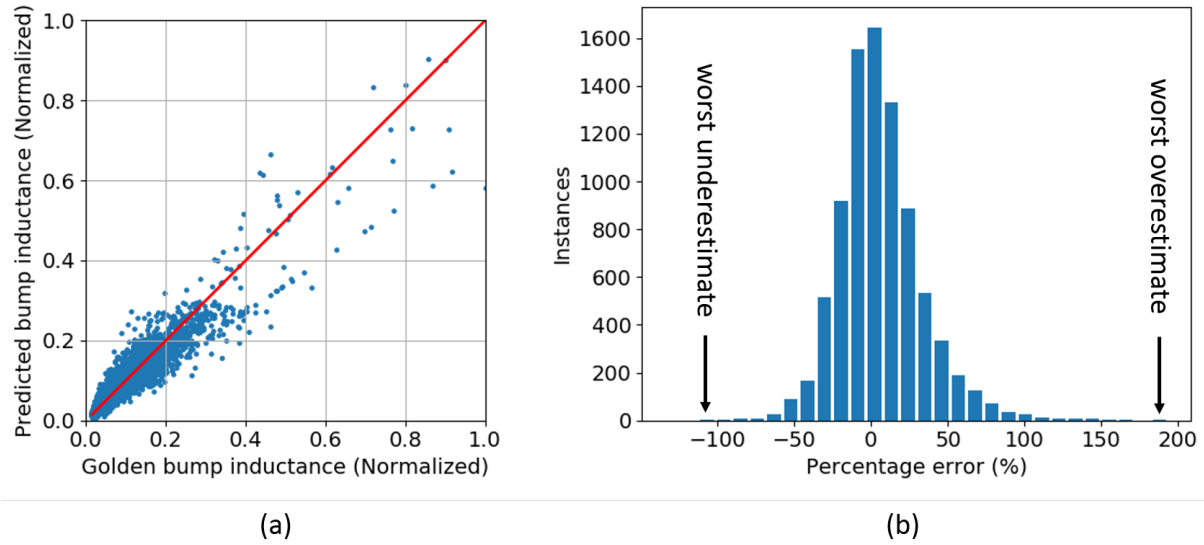


Figure 8.8: Results for *achievable* bump inductance model. (a) Golden versus predicted *achievable* bump inductance. (b) Percentage error histogram.

of our predictive model. Our model can predict *actual* bump inductance within an absolute percentage error of 44.0% for more than 95% of all data points. Table 8.4 shows the model accuracy for *actual* bump inductance prediction. We achieve average absolute percentage errors of 13.5% and 19.3% for training and testing datasets respectively.

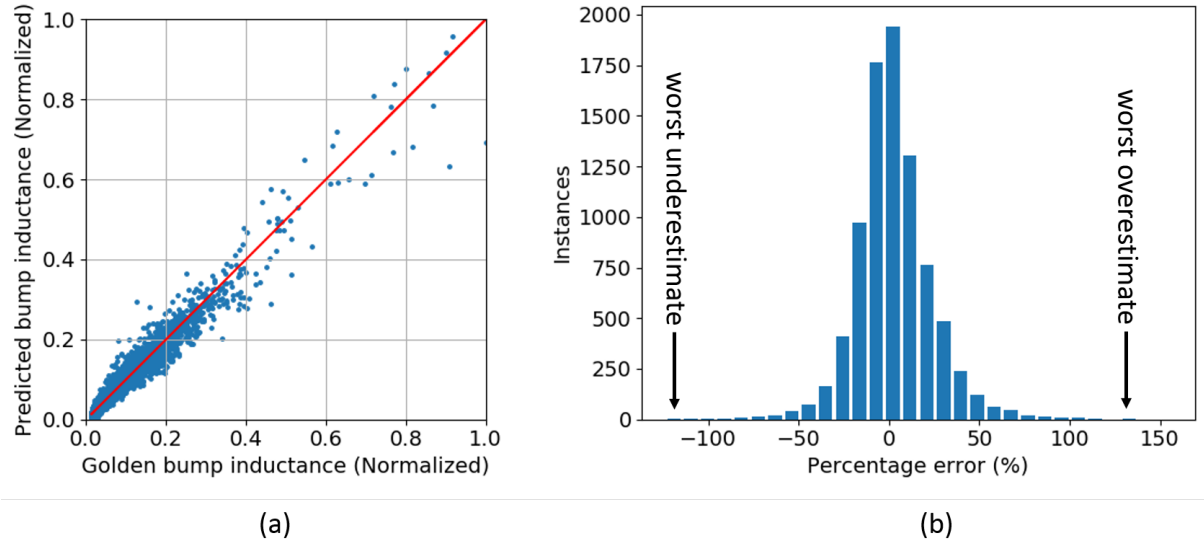


Figure 8.9: Results for *actual* bump inductance model. (a) Golden versus predicted *actual* bump inductance and (b) Percentage error histogram.

Table 8.4: Accuracy metrics for *actual* bump inductance model.

Metric	Training	Testing
R^2	0.97	0.92
AAPE (%)	13.5	19.3
90^{th}-pct Worst Overestimate (%)	32.2	48.0
90^{th}-pct Worst Underestimate (%)	-24.9	-32.5
95^{th}-pct Worst Overestimate (%)	41.2	62.9
95^{th}-pct Worst Underestimate (%)	-33.7	-40.6

Figure 8.3(a) shows the bump inductance values from Golden and Quick and Figure 8.3(b) shows the percentage difference distribution of Quick tool compared to Golden tool. Table 8.5 shows the accuracy of Quick tool. Compared to our *actual* bump inductance model, Quick tool is 82.6% less accurate for 95^{th}-pct percentage error.

Table 8.5: Accuracy metrics comparison between Quick tool and our model.

Metric	Quick	Our Model
R^2	0.77	0.95
AAPE (%)	33.2	15.4
90^{th}-pct Worst Overestimate (%)	97.6	48.6
90^{th}-pct Worst Underestimate (%)	-56.9	-37.1
95^{th}-pct Worst Overestimate (%)	125.1	37.5
95^{th}-pct Worst Underestimate (%)	-61.5	-28.3

Results for Experiment 5. Similar in spirit to [44], we build a model based on *PiM*, *Des* and *Lay* parameter sets and Quick tool results to correlate Quick tool results and Golden tool results. Figure 8.10(a) shows the Golden versus predicted bump inductance values and Figure 8.10(b) shows the percentage error distribution of our *actual* inductance model when Quick tool result is considered as an input parameter. Our model can predict *actual* bump inductance within an absolute percentage error of 32.9% for more than 95% of all data points. Table 8.6 shows the corresponding model accuracy for *actual* bump inductance prediction. Compared to the *actual* bump inductance model in **Experiment 4**, we observe that the model that considers the Quick tool result as an input achieves better accuracy metrics.

Results for Experiment 6. We study the generality of our bump inductance model by applying the model from Experiment 4 to a variant design. The variant design is created from a real design by

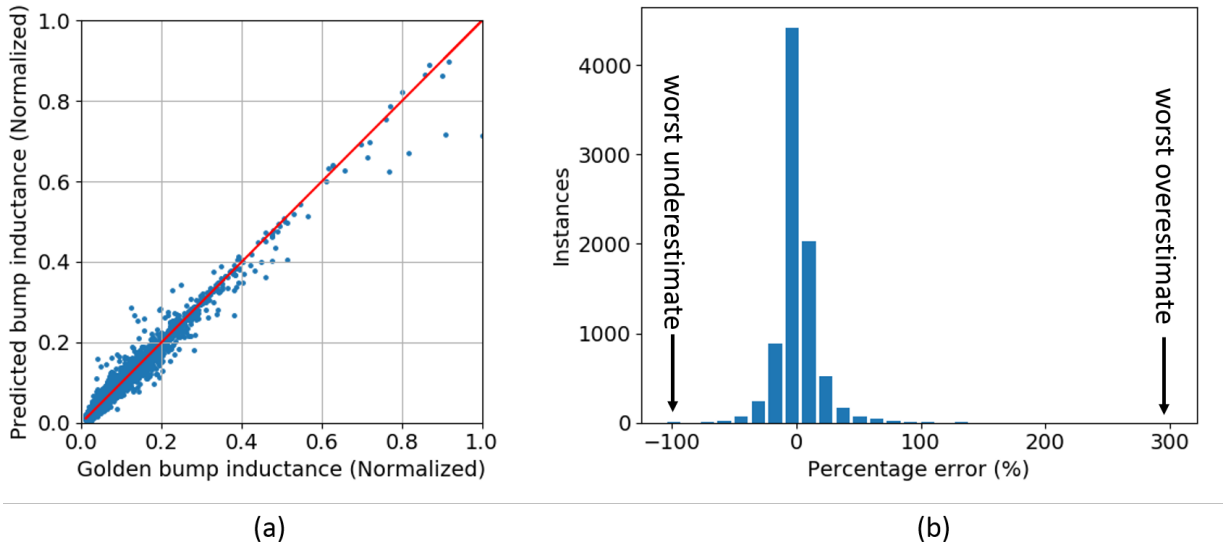


Figure 8.10: Results for Quick tool and Golden tool correlation model. (a) Golden versus predicted bump inductance. (b) Percentage error histogram.

Table 8.6: Accuracy metrics for correlation between Quick tool and Golden tool.

Metric	Training	Testing
R^2	0.99	0.94
AAPE (%)	6.2	17.5
90^{th}-pct Worst Overestimate (%)	15.7	45.5
90^{th}-pct Worst Underestimate (%)	-12.7	-31.8
95^{th}-pct Worst Overestimate (%)	20.9	60.5
95^{th}-pct Worst Underestimate (%)	-17.5	-39.9

strictly degrading PDN quality through removal of a number of balls and vias. Figure 8.11(a) shows the Golden versus predicted bump inductance values for the variant design and Figure 8.11(b) shows the percentage error distribution. We observe that more than 95 percent of the data points from the variant design have absolute percentage errors within 55.0%. As the variant design is in some sense "intentionally worse than the original PDN design", we believe that this result supports that our bump inductance model is generalized.

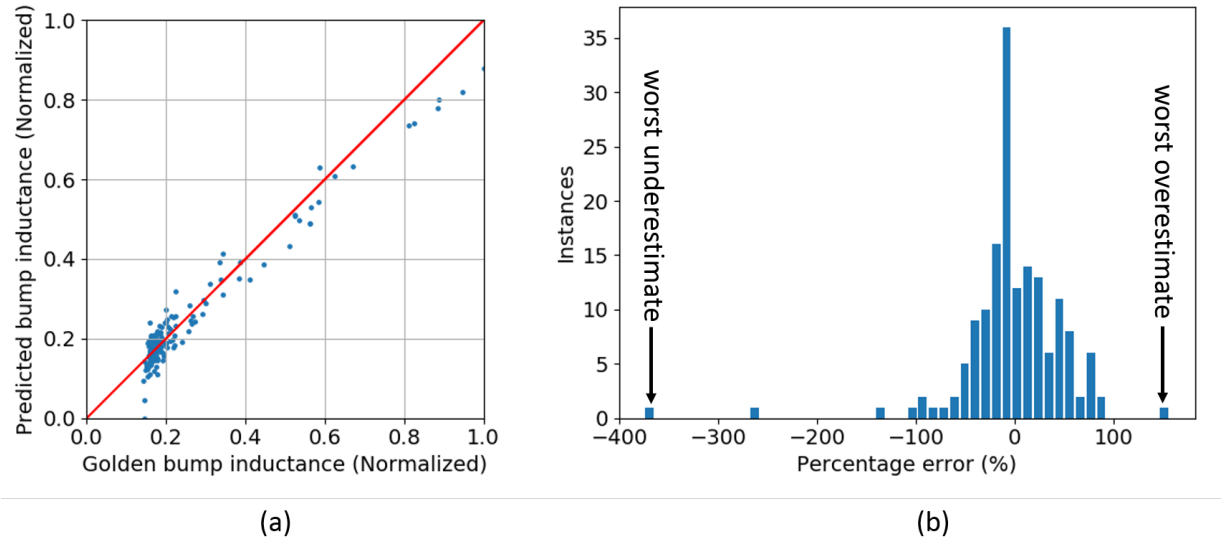


Figure 8.11: Results for model generality study. (a) Golden versus predicted bump inductance of variant design. (b) Percentage error histogram.

8.5 Summary

Prediction of PKG PDN quality, especially PKG bump inductance, is crucial to reduce design cost and turnaround time. We propose a learning-based methodology to predict *achievable* bump inductance and help make an assessment of achievable PKG PDN quality, given pinmap information and design information. We observe that the sets of pre-layout parameters designated as *PiM* and *Des* in Section 8.3-8.3.1 have the strongest impact on the PKG PDN quality. Hence, we propose a pre-layout *achievable* inductance model using the *PiM* and *Des* parameters as inputs. PKG PDN designers can use the model to avoid the need to iterate layout to evaluate multiple pinmap options during the early stages of the design. That is, a closer to optimal pinmap allows engineers to focus on a smaller subset of viable pinmaps in the layout phase. The average absolute percentage error is 21.2% or less for the *achievable* inductance model.

We then use the same learning-based methodology to build a post-layout *actual* inductance model that predicts bump inductance when layout information is available. The average absolute percentage error is 19.3% or less for *actual* bump inductance model. We extend the *actual* inductance model by feeding the results of the Quick tool as an additional input. This further reduces the average absolute percentage error to 17.5%. Our post-layout model provides more accurate layout-phase bump inductance prediction

compared to the Quick tool. PKG PDN designers can use our post-layout model to provide accurate feedback without using Golden tool. We also demonstrate that our model is generalized against a variant design. Our models enable quicker pre- and post-layout optimizations for PKG PDN, reducing weeks of pinmap and layout optimizations in a typical SoC design cycle.

8.6 Acknowledgments

Chapter 8 contains a reprint of Y. Cao, J. Li, A. B. Kahng, A. Roy, V. Srinivas and B. Xu, "Learning-Based Prediction of Package Power Delivery Network Quality", *Proc. Asia and South Pacific Design Automation Conference*, 2019. The dissertation author is a main contributor to this paper. I would like to thank my coauthors Yi Cao, Joseph Li, Andrew B. Kahng, Abinash Roy and Bangqi Xu.

Chapter 9

Conclusion

This thesis has presented tools and methods that enable cross-layer pathfinding for off-chip interconnects. The design space for pathfinding includes mobile and server memory interconnect, silicon photonic NoCs, 2.5D silicon interposers, 3D TSV, and PDN. The tools include (i) models and calculators that use PAT models to estimate system-level metrics relating to performance, power and cost, as well as (ii) optimizers and planning tools that help architects evaluate tradeoffs and are integrated into die/PKG/PCB design flows. This enables an efficient design solution for off-chip interconnects early in the design cycle.

Chapter 2 built on an early memory interface calculator and expanded it to CACTI-IO, a tool that models power-area-timing for any type of memory interface across server and mobile systems. Pathfinding with CACTI-IO included voltage and timing margin calculators that rolled up the eye metrics on the interface based on the channel, and helped assess system timing and power without the need to run detailed signal and power integrity analysis. This proved useful in evaluating a variety of memory interface options to help optimize bandwidth, capacity, power and cost. Chapter 3 presented the early work that formed the basis of the memory interface calculator, based on mobile memory interfaces. It also highlighted the cross-layer dependencies between the top-down requirements and the bottom-up capabilities of the memory subsystem. Chapter 4 extended CACTI-IO to custom memory interfaces for server designs. This was achieved through an API that helped users define a new interconnect with underlying models for the custom building blocks. It then used this API to explore two new memory architectures - cascaded channel

and narrow channel architectures. Future work for memory interconnect includes updated models for the newly evolving design space, especially for NVMs. Additionally, studying memory interconnect for a 2.5D system is of great interest, given the heterogeneous bandwidth and latency from different chiplets in the system. An important descision is the optimal placement or partitioning of the DDR controller relative to the many masters it services. An interconnect tool that can rapidly prototype the system-level impact and help plug into a pathfinding framework would be very useful here.

Chapter 5 presented a thermally-aware floorplan optimizer for PNoCs that performs simultaneous placement and routing for the PNoC. Our MILP P&R formulation takes thermal effects of cores on photonic components into consideration and integrates all sources of power consumption - including laser power, electrical-optical-electrical (EOE) power, and thermal tuning power - that are required to reliably operate photonic devices. The chapter also proposed the concept of a *power weight* that enables the study of PNoCs for heterogeneous-core designs. In the future, we plan to evaluate how such heterogeneous clusters allow for an independent cross-layer design knob that is distinct from job scheduling.

Chapter 6 presented a cross-layer co-optimization framework for 2.5D interconnect. The framework comprehended the logical, physical and circuit layers and allowed for co-optimization across them. This enabled us to look at NoC topologies, chiplet placement and link signaling and pipelining options at the same time, and to find an optimal overall solution. A key aspect that enabled the cross-layer co-optimization was a routing MILP that comprehended placement, timing based on link length and circuit layer choices, and bandwidth needs based on the NoC topology. The framework optimized system performance, power, cost and thermal management. There are huge opportunities to build on this framework in the growing area of 2.5D interconnects and systems. These opportunities include detailed P&R capability along with assignment of microbumps, as well as intra-chiplet and inter-chiplet floorplan optimization that is thermally and electrically (with respect to both signal integrity and power integrity) aware. Extending the interconnect choices to different serial and parallel bus options, including the clocking scheme, would complete the off-chip interconnect design space. A chiplet partitioning tool that integrates such a framework would be very valuable for architects evaluating a cross-layer optimization. Such a framework could also be extended to study on-chip and off-chip interconnects together, and tradeoffs across performance, power and cost. Options like wafer scale chips from Cerebras [173] provide on-chip alternatives for a high-end

network for a large design. Our framework would be able to quickly evaluate system metrics for off-chip and on-chip interconnects for various technology options at the same time.

Chapter 7 presented a power-area-timing model for 3D interconnect, including 3DIO and clocking circuits. We studied three clocking schemes - synchronous, source-synchronous and asynchronous. We used a directed graph to break down the overall power-area-timing to intermediate metrics, and developed a combination of analytical models and neural network models to evaluate these metrics. We showed that the optimal clocking scheme depends on the area and power constraints for the link, and that source-synchronous clocking provided a good design point along the knee of the power-area tradeoff for a given bandwidth. Future work would include building a partitioning tool for 3D, along similar lines as 2.5D. Having the capability to assess PDN would be critical here given the challenges for PDN in 3D integration.

Chapter 8 presented a learning-based prediction of package PDN impedance. The predictor was capable of estimating the PDN impedance of the package based on the bump and ball assignments (pinmap) and the design of the package. Metamodeling techniques including ANN, SVM and MARS were used to build the predictor based on training data from industry designs. Such a predictor helps enabling PDN co-optimization of the SoC floorplan, bump assignment and ball assignment. Future work includes developing *Lay* parameter predictors, as well as model-based PKG pinmap optimization techniques. PKG PDN designers can use *Lay* parameter predictors for PKG PDN layout guidance; this enables more efficient exploration of the PKG PDN design space. Model-based PKG pinmap optimization can be used to identify promising PKG PDN design solutions that can be passed on to the layout phase. PKG and PCB auto-routers for PDN would also be of interest, even if they serve only to enable better insight into the impedance prediction, and are not DRC-clean or production-quality routing tools. These routers could learn from human layout techniques and improve over time using machine learning techniques.

In conclusion, the tools and methods presented in this thesis provide the foundations of a cross-layer pathfinding framework for off-chip interconnects, and enable die/PKG/PCB co-design from the architecture layer to the physical, circuit and technology layers. Future work involves (i) further extension of the design space covered; (ii) die/PKG/PCB co-design planning and routing tools that tie the three design flows together; and (iii) die or chiplet partitioning exploration for heterogeneous chiplets with inter-chiplet as well as off-chip requirements that combine thermal, signal and power integrity impact.

Bibliography

[1] M. M. Ahmed, Md. S. Shamim, N. Mansoor, S. A. Mamun and A. Ganguly, "Increasing Interposer Utilization: A Scalable, Energy Efficient and High Bandwidth Multicore-multichip Integration Solution," *Proc. IGSC*, 2017, pp. 1-6.

[2] I. Akgun, J. Zhan, Y. Wang and Y. Xie, "Scalable Memory Fabric for Silicon Interposer-based Multi-core Systems," *Proc. ICCD*, 2016, pp. 33-40.

[3] D. H. Bailey, E. Barszcz, J. T. Barton, D. S. Browning, R. L. Carter, D. Dagum, R. A. Fatoohi, P. O. Frederickson, T. A. Lasinski, R. S. Schreiber, H. D. Simon, V. Venkatakrishnan and S. K. Weeratunga, "The NAS Parallel Benchmarks," *The International Journal of Supercomputer Applications* 5(3) (1994), pp. 63-73.

[4] H. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI*, Addison-Wesley, 1990.

[5] G. Balamurugan, J. Kennedy, G. Banerjee, J. E. Jaussi, M. Mansuri, F. O'Mahony, B. Casper and R. Mooney, "A Scalable 5-15 Gbps, 14-75 mW Low Power I/O Transceiver in 65nm CMOS," *IEEE JSSC* 43 (2008), pp. 1010-1019.

[6] J. Baloria, "Micron Reinvents DRAM Memory: Hybrid Memory Cube," *Proc. IDF Workshop*, Sept. 2011.

[7] C. Batten, A. Joshi, V. Stojanovic and K. Asanovic, "Designing Chip-Level Nanophotonic Interconnection Networks," *IEEE JETCAS* 2(2) (2012), pp. 137-153.

[8] A. Boos, L. Ramini, U. Schlichtmann and D. Bertozzi, "PROTON: An Automatic Place-and-route Tool for Optical Networks-on-chip," *Proc. ICCAD*, 2013, pp. 138-145.

[9] P. A. Brennan, N. Raver and A. E. Ruehli, "Three-Dimensional Inductance Computations with Partial Element Equivalent Circuits," *IBM J. Research and Development* 23(6) 1979, pp. 661-668.

[10] J. Carballo, W. J. Chan, P. A. Gargini, A. B. Kahng and S. Nath, "ITRS 2.0: Toward a Re-framing of the Semiconductor Technology Roadmap," *Proc. ICCD*, 2014, pp. 139-146.

[11] T. E. Carlson, W. Heirmant and L. Eeckhout, "Sniper: Exploring the Level of Abstraction for Scalable and Accurate Parallel Multi-core Simulation," *Proc. SC*, 2011, pp. 1-12.

[12] B. K. Casper, M. Haycock and R. Mooney, "An Accurate and Efficient Analysis Method for Multi-Gb/s Chip-to-Chip Signaling Schemes," *Proc. VLSIC*, 2002, pp. 54-57.

[13] N. Chang, K. Kim and J. Cho, "Bus Encoding for Low-Power High-Performance Memory Systems," *Proc. DAC*, 2000, pp. 800-805.

[14] J. Charbonnier, M. Assous, J.-P. Bally, K. Miyairi, M. Sunohara, R. Cuchet, H. Feldis, N. Bouzaida, N. Bernard-Henriques, R. Hida, T. Mourier, G. Simon and M. Higashi, "High Density 3D Silicon Interposer Technology Development and Electrical Characterization for High End Applications," *Proc. ESTC*, 2012, pp. 1-7.

[15] S. Chaudhuri, J. McCall and J. Salmon, "Proposal for BER Based Specifications for DDR4," *Proc. EPEPS*, 2010, pp. 121-124.

[16] R. Chaware, K. Nagarajan and S. Ramalingam, "Assembly and Reliability Challenges in 3D Integration of 28nm FPGA Die on a Large High Density 65nm Passive Interposer," *Proc. ECTC*, 2012, pp. 279-283.

[17] C. Chen, T. Zhang, P. Contu, J. Klamkin, A. K. Coskun and A. Joshi, "Sharing and Placement of On-chip Laser Sources in Silicon-Photonic NoCs," *Proc. NOCS*, 2014, pp. 88-95.

[18] G. Chen, M. A. Anders, H. Kaul, S. K. Satpathy, S. K. Mathew, S. K. Hsu, A. Agarwal, R. K. Krishnamurthy, S. Borkar and V. De, "A 340mV-to-0.9V 20.2Tb/s Source-synchronous Hybrid Packet/circuit-switched 16×16 Network-on-chip in 22nm Tri-gate CMOS," *IEEE JSSC* 50(1) (2015), pp. 59-67.

[19] S. Cho, "Technical Challenges in TSV Integration," *Keynote, RTI 3-D Conference*, 2010.

[20] Y. Choi, H. Jeong and H. Kim, "Future Evolution of Memory Subsystem in Mobile Applications," *IEEE International Memory Workshop (IMW)*, 2010, pp. 1-2.

[21] C. Chu and D. F. Wong, "Closed Form Solution to Simultaneous Buffer Insertion/Sizing and Wire Sizing," *ACM Trans. on Design Automation of Electronic Systems* 6(3) (2001), pp. 343-371.

[22] C. Condrat, P. Kalla and S. Blair, "Channel Routing for Integrated Optics," *Proc. SLIP*, 2013, pp. 1-8.

[23] E. Consoli, M. Alioto, G. Palumbo and J. Rabaey, "Conditional Push-pull Pulsed Latches with 726fJ·ps Energy-delay Product in 65nm CMOS," *Proc. ISSCC*, 2012, pp. 482-484.

[24] A. K. Coskun, T. S. Rosing, K. Whisnant and K. Gross, "Static and Dynamic Temperature-Aware Scheduling for Multiprocessor SoCs," *IEEE Trans. on VLSI Systems* 16(9) (2008), pp. 1127-1140.

[25] A. Coskun, F. Eris, A. Joshi, A. B. Kahng, Y. Ma and V. Srinivas, "A Cross-layer Methodology for Design and Optimization of Networks in 2.5D Systems," *Proc. ICCAD*, 2018, pp. 101-108.

[26] W. Dally and J. Poulton, *Digital Systems Engineering*, Cambridge University Press, 1998.

[27] W. J. Dally and B. P. Towles, *Principles and Practices of Interconnection Networks*, Elsevier, 2004.

[28] D. Ding, Y. Zhang, H. Huang, R. T. Chen and D. Pan, "O-Router: An Optical Routing Framework for Low Power On-Chip Silicon Nano-Photonic Integration," *Proc. DAC*, 2009, pp. 264-269.

[29] D. Ding, B. Yu and D. Z. Pan, "GLOW: A Global Router for Low-power Thermal-reliable Interconnect Synthesis Using Photonic Wavelength Multiplexing," *Proc. DAC*, 2012, pp. 621-626.

[30] X. Dong and Y. Xie, "System-Level Cost Analysis and Design Exploration for Three-Dimensional Integrated Circuits (3D ICs)," *Proc. ASP-DAC*, 2009, pp. 234-241.

[31] F. Dubois, A. Sheibanyrad, F. Petrot and M. Bahmani, "Elevator-First: A Deadlock-Free Distributed Routing Algorithm for Vertically Partially Connected 3D-NoCs," *IEEE Trans. on Computers* 62(3) (2011), pp. 609-615.

[32] J. Dukovic, S. Ramaswami, S. Pamarthy, R. Yalamanchili, N. Rajagopalan, K. Sapre, Z. Cao, T. Ritzdorf, Y. Wang, B. Eaton, R. Ding, M. Hernandez, M. Naik, D. Mao, J. Tseng, D. Cui, G. Mori, P. Fulmer, K. Sirajuddin, J. Hua, S. Xia, D. Erickson, R. Beica, E. Young, P. Kusler, R. Kulzer, S. Oemardani, H. Dai, X. Xu, M. Okazaki, K. Dotan, C. Yu, C. Lazik, J. Tran and L. Luo, "Through-Silicon-Via Technology for 3D Integration," *Proc. International Memory Workshop (IMW)*, 2010, pp. 1-2.

[33] J. Ellis, "Overcoming Obstacles for Closing Timing for DDR3-1600 and Beyond," *Proc. Denali Memcon*, 2010.

[34] F. Eris, A. Joshi, A. B. Kahng, Y. Ma, S. Mojumder and T. Zhang, "Leveraging Thermally-aware Chiplet Organization in 2.5D Systems to Reclaim Dark Silicon," *Proc. DATE*, 2018, pp. 1441-1446.

[35] E. J. Fang, T. C. Shih and D. S. Huang, "IR to Routing Challenge and Solution for Interposer-based Design," *Proc. ASP-DAC*, 2015, pp. 226-230.

[36] M. Floman, "Memories Are Made for This: Mobile Device Applications," *JEDEC Flash Storage Summit*, 2010.

[37] J. H. Friedman, "Multivariate Adaptive Regression Splines," *The Annals of Statistics* 19(1) (1991), pp. 1-67.

[38] B. Ganesh, A. Jaleel, D. Wang and B. Jacob, "Fully-Buffered DIMM Memory Architectures: Understanding Mechanisms, Overheads, and Scaling," *Proc. HPCA*, 2007, pp. 109-120.

[39] M. Georgas, B. R. Moss, C. Sun, J. Shainline, J. S. Orcutt, M. Wade, Y. Chen, K. Nammari, J. C. Leu, A. Srinivasan, R. J. Ram, M. A. Popovic and V. Stojanovic, "A Monolithically-integrated Optical Transmitter and Receiver in a Zero-change 45nm SOI Process," *Proc. Symp. on VLSI Circuits Digest of Technical Papers*, 2014, pp. 1-2.

[40] P. Grani, R. Proietti, V. Akella and S. J. Yoo, "Photonic Interconnects for Interposer-based 2.5D/3D Integrated Systems on a Chip," *Proc. MEMSYS*, 2016, pp. 377-386.

[41] P. Grani, R. Proietti, V. Akella and S. J. Yoo, "Design and Evaluation of AWGR-based Photonic NoC Architectures for 2.5D Integrated High Performance Computing Systems," *Proc. HPCA*, 2017, pp. 289-300.

[42] H. Gu, J. Xu and W. Zhang, "A Low-power Fat Tree-based Optical Network-On-Chip for Multiprocessor System-on-chip," *Proc. DATE*, 2009, pp. 3-8.

[43] T. Ham, B. Chelepalli, N. Xue and B. Lee, "Disintegrated Control for Energy-Efficient and Heterogeneous Memory Systems," *Proc. HPCA*, 2013, pp. 424-435.

[44] S. S. Han, A. B. Kahng, S. Nath and A. Vydyanathan, "A Deep Learning Methodology to Proliferate Golden Signoff Timing," *Proc. DATE*, 2014, pp. 1-6.

[45] B. Hargreaves, H. Hult and S. Reda, "Within-die Process Variations: How Accurately Can They Be Statistically Modeled?" *Proc. ASP-DAC*, 2008, pp. 524-530.

[46] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer, 2009.

[47] H. Hatamkhani, F. Lambrecht, V. Stojanovic and C.-K. K. Yang, "Power-Centric Design of High-Speed I/Os," *Proc. DAC*, 2006, pp. 867-872.

[48] G. Hendry, J. Chan, L. P. Carloni and K. Bergman, "VANDAL: A Tool for the Design Specification of Nanophotonic Networks," *Proc. DATE*, 2011, pp. 1-6.

[49] M. A. Horowitz, C.-K. K. Yang and S. Sidiropoulos, "High-Speed Electrical Signaling: Overview and Limitations," *IEEE Micro* 18 (1998), pp. 12-24.

[50] M. A. Horowitz, C.-K. K. Yang and S. Sidiropoulos, "High-Speed Electrical Signaling: Overview and Limitations," *IEEE Trans. on Advanced Packaging* 31(4) (2008), pp. 722-730.

[51] J. Howard, S. Dighe, S. R. Vangal, G. Ruhl, N. Borkar, S. Jain, V. Erraguntla, M. Konow, M. Riepen, M. Gries, G. Droege, T. Lund-Larsen, S. Steibl, S. Borkar, V. K. De and R. Van Der Wijngaart, "A 48-core IA-32 Processor in 45 nm CMOS Using On-die Message-passing and DVFS for Performance and Power Scaling," *IEEE JSSC* 46(1) (2011), pp. 173-183.

[52] B. Jacob, "Trends in Memory Systems," *Sun-DARPA UNIC Architecture Workshop*, 2010.

[53] R. Jafari, F. Dabiri and M. Sarrafzadeh, "An Efficient Placement and Routing Technique for Fault-tolerant Distributed Embedded Computing," *Proc. RTCSA*, 2005, pp. 1-9.

[54] J. Jang, O. Franza and W. Burleson, "Compact Expressions for Period Jitter of Global Binary Clock Trees," *Proc. EPEP*, 2008, pp. 47-50.

[55] J. Jeddeloh and B. Keeth, "Hybrid Memory Cube – New DRAM Architecture Increases Density and Performance," *Symp. on VLSI Technology*, 2012, pp. 87-88.

[56] N. E. Jerger, A. Kannan, Z. Li and G. H. Loh, "NoC Architectures for Silicon Interposer Systems: Why Pay for More Wires when You Can Get Them (from Your Interposer) for Free?" *Proc. MICRO*, 2014, pp. 458-470.

[57] A. Joshi, C. Batten, Y. Kwon, S. Beamer, I. Shamim, K. Asanovic and V. Stojanovic, "Silicon-photonic Clos Networks for Global On-chip Communication," *Proc. NOCS*, 2009, pp. 124-133.

[58] N. P. Jouppi, A. B. Kahng, N. Muralimanohar and V. Srinivas, "CACTI-IO: CACTI With Off-Chip Power-Area-Timing Models," *Proc. ICCAD*, 2012, pp. 294-301.

[59] A. B. Kahng and V. Srinivas, "Mobile System Considerations for SDRAM Interface Trends," *Proc. SLIP*, 2011, pp. 1-8.

[60] A. B. Kahng, B. Lin and S. Nath, "Explicit Modeling of Control and Data for Improved NoC Router Estimation," *Proc. DAC*, 2012, pp. 392-397.

[61] A. B. Kahng, B. Lin and S. Nath, "Enhanced Metamodeling Techniques for High-Dimensional IC Design Estimation Problems," *Proc. DATE*, 2013, pp. 1861-1866.

[62] A. Kannan, N. E. Jerger and G. H. Loh, "Enabling Interposer-based Disintegration of Multi-core Processors," *Proc. MICRO*, 2015, pp. 546-558.

[63] M. A. Karim, P. D. Franzon and A. Kumar, "Power Comparison of 2D, 3D and 2.5D Interconnect Solutions and Power Optimization of Interposer Interconnects," *Proc. ECTC*, 2013, pp. 860-866.

[64] C. V. Kashyap, C. J. Alpert, F. Liu and A. Devgan, "Closed-Form Expressions for Extending Step Delay and Slew Metrics to Ramp Inputs for RC Trees," *IEEE Trans. on CAD* 23(4) (2004), pp. 509-516.

[65] P. Keller, "Understanding The New Bit Error Rate Based DRAM Timing Specifications," *Server Memory Forum*, 2012.

[66] J.-S. Kim, C. S. Oh, H. Lee, D. Lee, H. R. Hwang, S. Hwang, B. Na, J. Moon, J.-G. Kim, H. Park, J.-W. Ryu, K. Park, S. K. Kang, S.-Y. Kim, H. Kim, J.-M. Bang, H. Cho, M. Jang, C. Han, J.-B. Lee, J. S. Choi and Y.-H. Jun, "A 1.2V 12.8GB/s 2Gb Mobile Wide-I/O DRAM with 4x128 I/Os Using TSV-Based Stacking," *Proc. ISSCC*, 2011, pp. 496-498.

[67] T.-Y. Kim and T. Kim, "Bounded Skew Clock Routing for 3D Stacked IC Designs: Enabling Trade-offs Between Power and Clock Skew," *Proc. International Green Computing Conference*, 2010, pp. 525-532.

[68] D. Kim, J. Kim, J. Choi, J. S. Park, J. Kim, H. Lee, J. Lee and K. Park, "Distributed Multi TSV 3D Clock Distribution Network in TSV-Based 3D IC," *Proc. EPEPS*, 2011, pp. 87-90.

[69] D. Kim, K. Y. Au, H. Y. Li, X. Luo, Y. L. Ye, S. Bhattacharya and G. Q. Lo, "2.5D Silicon Optical Interposer for 400 Gbps Electronic-photonic Integrated Circuit Platform Packaging," *Proc. EPTC*, 2017, pp. 1-4.

[70] G. Kim, J. Kim, J. Ahn and J. Kim, "Memory-centric System Interconnect Design with Hybrid Memory Cubes," *Proc. PACT*, 2013, pp. 145-155.

[71] N. Kirman, M. Kirman, R. K. Dokania, J. F. Martinez, A. B. Apsel, M. A. Watkins and D. H. Albonesi, "Leveraging Optical Technology in Future Bus-based Chip Multiprocessors," *Proc. MICRO*, 2006, pp. 492-503.

[72] J. U. Knickerbocker, P. S. Andry, E. Colgan, B. Dang, T. Dickson, X. Gu, C. Haymes, C. Jahnes, Y. Liu, J. Maria, R. J. Polastre, C. K. Tsang, L. Turlapati, B. C. Webb, L. Wiggins and S. L. Wright, "2.5D and 3D Technology Challenges and Test Vehicle Demonstrations," *Proc. ECTC*, 2012, pp. 1068-1076.

[73] H. Lee, K. K. Chang, J. Chun, T. Wu, Y. Frans, B. Leibowitz, N. Nguyen, T. J. Chin, K. Kaviani, J. Shen, X. Shi, W. T. Beyene, S. Li, R. Navid, M. Aleksic, F. S. Lee, F. Quan, J. Zerbe, R. Perego and F. Assaderaghi, "A 16 Gb/s/Link, 64 GB/s Bidirectional Asymmetric Memory Interface," *IEEE JSSC* 44(4) (2009), pp. 1235-1247.

[74] B. Lee, E. Ipek, O. Mutlu and D. Burger, "Architecting Phase Change Memory as a Scalable DRAM Alternative," *Proc. ISCA*, 2009, pp. 2-13.

[75] M. Lee, W. Dally and P. Chiang, "Low-Power Area-Efficient High-Speed I/O Circuit Techniques," *IEEE JSSC* 35(11) (2000), pp. 1591-1599.

[76] B. Leibowitz, R. Palmer, J. Poulton, Y. Frans, S. Li, J. Wilson, M. Bucher, A. M. Fuller, J. Eyles, M. Aleksic, T. Greer and N. M. Nguyen, "A 4.3 GB/s Mobile Memory Interface with Power-Efficient Bandwidth Scaling," *IEEE JSSC* 45 (2010), pp. 136-137.

[77] R. Li, D. Zhou, J. Liu and X. Zeng, "Power-Optimal Simultaneous Buffer Insertion/Sizing and Wire Sizing," *Proc. ICCAD*, 2003, pp. 581-586.

[78] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen and N. P. Jouppi, "McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures," *Proc. MICRO*, 2009, pp. 469-480.

[79] X. Li, W. Liu, H. Du, Y. Wang, Y. Ma and H. Yang, "Whitespace-Aware TSV Arrangement in 3D Clock Tree Synthesis," *Proc. ISVLSI*, 2013, pp. 115-120.

[80] Z. Li, A. Qouneh, M. Joshi, W. Zhang, X. Fu and T. Li, "Aurora: A Cross-Layer Solution for Thermally Resilient Photonic Network-on-Chip," *IEEE Trans. on VLSI Systems* 23(1) (2015), pp. 170-183.

[81] H. Li, A. Fourmigue, S. Le Beux, X. Letartre, I. O'Connor, and G. Nicolescu, "Thermal Aware Design Method for VCSEL-based On-chip Optical Interconnect," *Proc. DATE*, 2015, pp. 1120-1125.

[82] K. Lim, J. Chang, T. Mudge, P. Ranganathan, S. K. Reinhardt and T. F. Wenisch, "Disaggregated Memory for Expansion and Sharing in Blade Servers," *Proc. ISCA*, 2009, pp. 267-278.

[83] X. Liu, Y. Peng and M. C. Papaefthymiou, "Practical Repeater Insertion for Low Power: What Repeater Library Do We Need?" *Proc. DAC*, 2004, pp. 30-35.

[84] W. Liu, H. Du, Y. Wang, Y. Ma, Y. Xie, J. Quan and H. Yang, "TSV-Aware Topology Generation for 3D Clock Tree Synthesis," *Proc. ISQED*, 2013, pp. 300-307.

[85] W. Liu, T. Chien and T. Wang, "Metal Layer Planning for Silicon Interposers with Consideration of Routability and Manufacturing Cost," *Proc. DATE*, 2014, pp. 359-364.

[86] G. H. Loh, Y. Xie and B. Black, "Processor Design in 3D Die-stacking Technologies," *IEEE Micro* 27(3) (2007), pp. 31-48.

[87] S. Lu, R. Tessier and W. Burleson, "Reinforcement Learning for Thermal-aware Many-core Task Allocation," *Proc. GLSVLSI*, 2015, pp. 379-384.

[88] C.-K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, G. Lowney, S. Wallace, V. J. Reddi and K. Hazelwood, "PIN: Building Customized Program Analysis Tools with Dynamic Instrumentation," *Proc. PLDI*, 2005, pp. 190-200.

[89] J. Macri, "AMD's Next generation GPU and High Bandwidth Memory Architecture: FURY," *Proc. HCS*, 2015, pp. 1-26.

[90] R. Mahajan, R. Sankman, N. Patel, D. Kim, K. Aygun, Z. Qian, Y. Mekonnen, I. Salama, S. Sharan, D. Iyengar and D. Mallik, "Embedded Multi-die Interconnect Bridge (EMIB) – a High Density, High Bandwidth Packaging Interconnect," *Proc. ECTC*, 2016, pp. 557-565.

[91] F. O'Mahony, J. Kennedy, J. E. Jaussi, G. Balamurugan, M. Mansuri, C. Roberts, S. Shekhar, R. Mooney and B. Casper, "A 47x10Gb/s 1.4mW/(Gb/s) Parallel Interface in 45nm CMOS," *Proc. IEEE ISSCC*, 2010, pp. 156-158.

[92] K. T. Malladi, F. A. Nothaft, K. Periyathambi, B. C. Lee, C. Kozyrakis and M. Horowitz, "Towards Energy-Proportional Datacenter Memory with Mobile DRAM," *Proc. ISCA*, 2012, pp. 37-48.

[93] T. Mandic, B. K. J. C. Nauwelaers and A. Baric, "Simple and Scalable Methodology for Equivalent Circuit Modeling of IC Packages," *IEEE Trans. on CPMT* 4(2) (2014), pp. 303-315.

[94] J. Meng, K. Kawakami and A. K. Coskun, "Optimizing Energy Efficiency of 3-D Multicore Systems with Stacked DRAM Under Power and Thermal Constraints," *Proc. DAC*, 2012, pp. 648-655.

[95] M. Mondal, A. J. Ricketts, S. Kirolos, T. Ragheb, G. Link, N. Vijaykrishnan and Y. Massoud, "Thermally Robust Clocking Schemes for 3D Integrated Circuits," *Proc. DATE*, 2007, pp. 1-6.

[96] P. Mosalikanti, C. Mozak and N. Kurd, "High Performance DDR Architecture in Intel Core Processors Using 32nm CMOS High-K Metal-gate Process," *Proc. VLSI-DAT*, 2011, pp. 1-4.

[97] N. Muralimanohar, R. Balasubramonian and N. Jouppi, "Optimizing NUCA Organizations and Wiring Alternatives for Large Caches with CACTI 6.0," *Proc. MICRO*, 2007, pp. 3-14.

[98] K. Murayama, M. Aizawa, K. Hara, M. Sunohara, K. Miyairi, K. Mori, J. Charbonnier, M. Assous, J. Bally, G. Simon and M. Higashi, "Warpage Control of Silicon Interposer for 2.5D Package Application," *Proc. ECTC*, 2013, pp. 879-884.

[99] A. Narayan, Y. Thonnart, P. Vivet, C. F. Tortolero and A. K. Coskun, "WAVES: Wavelength Selection for Power-efficient 2.5D-integrated Photonic NoCs," *Proc. DATE*, 2019, pp. 516-521.

[100] M. M. Navidi and G.-S. Byun, "Comparative Analysis of Clock Distribution Networks for TSV-based 3D IC Designs," *Proc. ISQED*, 2014, pp. 184-188.

[101] D. Oh, F. Lambrecht, J. H. Ren, S. Chang, B. Chia, C. Madden and C. Yuan, "Prediction of System Performance Based on Component Jitter and Noise Budgets," *Proc. EPEPS*, 2007, pp. 33-36.

[102] D. Oh, S. Chang, C. Madden, J. Kim, R. Schmitt, M. Li, C. Y. F. Ware, B. Leibowitz, Y. Frans and N. Nguyen, "Design and Characterization of a 12.8 GB/s Low Power Differential Memory System for Mobile Applications," *Proc. EPEP*, 2009, pp. 33-36.

[103] D. Oh and C. Yuan, *High-Speed Signaling: Jitter Modeling, Analysis, and Budgeting*, Prentice Hall, 2011.

[104] J. S. Orcutt, B. Moss, C. Sun, J. Leu, M. Georgas, J. Shainline, E. Zgraggen, H. Li, J. Sun, M. Weaver, S. Urosevic, M. Popovic, R. J. Ram and V. Stojanovic, "Open Foundry Platform for High-performance Electronic-photonic Integration," *Optics Express*, 2012, pp. 12222-12232.

[105] S. Osmolovskyi, J. Knechtel, I. L. Markov and J. Lienig, "Optimal Die Placement for Interposer-based 3D ICs," *Proc. ASP-DAC*, 2018, pp. 513-520.

[106] H. Ou, H. Chien and Y. Chang, "Simultaneous Analog Placement and Routing with Current Flow and Current Density Considerations," *Proc. DAC*, 2013, pp. 1-6.

[107] J. Ousterhout, P. Agrawal, D. Erickson, C. Kozyrakis, J. Leverich, D. Mazieres, S. Mitra, A. Narayanan, G. Parulkar, M. Rosenblum, S. Rumble, E. Stratmann and R. Stutsman, "The Case for RAMClouds: Scalable High-Performance Storage Entirely in DRAM," *SIGOPS Operating Systems Review* 43(4) (2009), pp. 92-105.

[108] R. Palmer, J. Poulton, A. Fuller, J. Chen and J. Zerbe, "Design Considerations for Low-Power High-Performance Mobile Logic and Memory Interfaces," *Proc. ASSCC*, 2008, pp. 205-208.

[109] Y. Pan, P. Kumar, J. Kim, G. Memik, Y. Zhang and A. Choudhary, "Firefly: Illuminating Future Network-on-chip with Nanophotonics," *Proc. ISCA*, 2009, pp. 429-440.

[110] G Parès, "3D Interposer for Silicon Photonics," *LETI Innovations Days*, 2013.

[111] J. T. Pawlowski, "Hybrid Memory Cube (HMC)," *Proc. HotChips*, 2011, pp. 1-24.

[112] J. Poulton, R. Palmer, A. M. Fuller, T. Greer, J. Eyles, W. J. Dally and M. Horowitz, "A 14-mW 6.25-Gb/s Transceiver in 90-nm CMOS," *IEEE JSSC* 42(12) (2007), pp. 2745-2757.

[113] M. Qureshi, V. Srinivasan and J. Rivers, "Scalable High-Performance Main Memory System Using Phase-Change Memory Technology," *Proc. ISCA*, 2009, pp. 24-33.

[114] R. Radojcic, *More-than-Moore 2.5D and 3D SiP Integration*, Springer, 2017.

[115] S. Ramalingam, "HBM Package Integration: Technology Trends, Challenges and Applications," *Proc. HCS*, 2016, pp. 1-17.

[116] L. Ramini, P. Grani, S. Bartolini and D. Bertozzi, "Contrasting Wavelength-Routed Optical NoC Topologies for Power-Efficient 3D-stacked Multicore Processors using Physical-Layer Analysis," *Proc. DATE*, 2013, pp. 1589-1594.

[117] L. Ramos, E. Gorbatov and R. Bianchini, "Page Placement in Hybrid Memory Systems," *Proc. ICS*, 2011, pp. 85-95.

[118] C. Ravishankar, D. Gaitonde and T. Bauer, "Placement Strategies for 2.5D FPGA Fabric Architectures," *Proc. FPL*, 2018, pp. 16-164.

[119] A. Ruberg, "Serial Port Memory: Bandwidth is Not Just for PCs Anymore," *Proc. Denali Memcon*, 2010.

[120] K. Ruhmer, P. Cochet, R. McCleary, R. Rogoff and R. Roy, "Lithography Challenges for 2.5D Interposer Manufacturing," *Proc. ECTC*, 2014, pp. 523-527.

[121] M. Saint-Laurent and M. Swaminathan, "Impact of Power-Supply Noise on Timing in High-Frequency Microprocessors," *IEEE Trans. on Advanced Packaging* 27(1) (2004), pp. 135-144.

[122] S. Sarkar, A. Brahme and S. Chandar, "Design Margin Methodology for DDR Interface," *Proc. IEEE EPEPS*, 2007, pp. 167-170.

[123] B. Schroeder, E. Pinheiro and W. Weber, "DRAM Errors in the Wild: A Large-Scale Field Study," *Proc. ACM SIGMETRICS*, 2009, pp. 193-204.

[124] D. P. Seemuth, A. Davoodi and K. Morrow, "Automatic Die Placement and Flexible I/O Assignment in 2.5D IC Design," *Proc. ISQED*, 2015, pp. 524-527.

[125] A. Shacham, K. Bergman and L. P. Carloni, "On the Design of a Photonic Network-on-Chip," *Proc. NOCS*, 2007, pp. 53-64.

[126] M. S. Shamim, N. Mansoor, R. S. Narde, V. Kothandapani, A. Ganguly and J. Venkataraman, "A Wireless Interconnection Framework for Seamless Inter and Intra-chip Communication in Multichip Systems," *IEEE Trans. on Computers* 66(3) (2017), pp. 389-402.

[127] Y. Shi and L. He, "Modeling and Design for Beyond-the-Die Power Integrity," *Proc. ICCAD*, 2010, pp. 411-416.

[128] T. L. Snyder and J. M. Steele, "A Priori Bounds on the Euclidean Traveling Salesman," *SIAM J. Computing* 24(3) (1995), pp. 665-671.

[129] R. Sredojevic and V. Stojanovic, "Optimization-Based Framework for Simultaneous Circuit-and-System Design-Space Exploration: A High-Speed Link Example," *Proc. ICCAD*, 2008, pp. 314-321.

[130] J. M. Steele and T. L. Snyder, "Worst-case Growth Rates of Some Classical Problems of Combinatorial Optimization," *SIAM J. Computing* 18(2) (1989), pp. 278-287.

[131] V. Stojanovic and M. Horowitz, "Modeling and Analysis of High-Speed Links," *Proc. CICC*, 2003, pp. 589-594.

[132] D. Stow, I. Akgun, R. Barnes, P. Gu and Y. Xie, "Cost Analysis and Cost-driven IP Reuse Methodology for SoC Design Based on 2.5D/3D Integration," *Proc. ICCAD*, 2016, pp. 56-61.

[133] D. Stow, Y. Xie, T. Siddiqua and G. H. Loh, "Cost-effective Design of Scalable High-performance Systems Using Active and Passive Interposers," *Proc. ICCAD*, 2017, pp. 728-735.

[134] D. B. Strukov, G. S. Snider, D. R. Stewart and R. Williams, "The Missing Memristor Found," *Nature* 453 (2008), pp. 80-83.

[135] C. Sun, C. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L. Peh and V. Stojanovic, "DSENT - A Tool Connecting Emerging Photonics with Electronics for Opto-Electronic Networks-on-Chip Modeling," *Proc. NOCS*, 2012, pp. 201-210.

[136] M. Swaminathan and E. Engin, *Power Integrity Modeling and Design for Semiconductors and Systems*, Pearson Education, 2007.

[137] G. Taguchi, *Introduction to Quality Engineering*, 2nd ed., McGraw-Hill, 1996.

[138] G. E. Tellez and M. Sarrafzadeh, "Minimal Buffer Insertion in Clock Trees with Skew and Slew Rate Constraints," *IEEE Trans. on CAD* 16(4) (1997), pp. 333-342.

[139] S. Thoziyoor, J. Ahn, M. Monchiero, J. B. Brockman and N. P. Jouppi, "A Comprehensive Memory Modeling Tool and its Application to the Design and Analysis of Future Memory Hierarchies," *Proc. ISCA*, 2008, pp. 51-62.

[140] C.-T. Tsai, "Package Inductance Characterization at High Frequencies," *IEEE Trans. on CPMT* 17(2) (1994), pp. 225-229.

[141] C.-T. Tsai and W.-Y. Yip, "An Experimental Technique for Full Package Inductance Matrix Characterization," *IEEE Trans. on CPMT* 19(2) (1996), pp. 338-343.

[142] Y. Tsai, Y. Xie, N. Vijaykrishnan and M. J. Irwin, "Three-Dimensional Cache Design Exploration Using 3DCacti," *Proc. ICCD*, 2005, pp. 519-524.

[143] Y. Urino, T. Usuki, J. Fujikata, M. Ishizaka, K. Yamada, T. Horikawa, T. Nakamura and Y. Arakawa, "High-density and Wide-bandwidth Optical Interconnects with Silicon Optical Interposers," *Photonics Research* 2(3) (2014), pp. A1-A7.

[144] A. Vaidyanath, "Challenges and Solutions for GHz DDR3 Memory Interface Design," *Proc. Denali MemCon*, 2010.

[145] G. A. Van Huben, K. D. Lamb, R. B. Tremaine, B. S. Aleman, S. M. Rubow, S. H. Rider, W. E. Maule and M. E. Wazlowski, "Server-class DDR3 SDRAM Memory Buffer Chip," *IBM Journal of Research and Development* 56(1) (2012), pp. 3:1-3:11.

[146] D. Vantrease, R. Schreiber, M. Monchiero, M. McLaren, N. P. Jouppi, M. Fiorentino, A. Davis, N. Binkert, R. G. Beausoleil and J. H. Ahn, "Corona: System Implications of Emerging Nanophotonic Technology," *Proc. ISCA*, 2008. pp. 153-164.

[147] T. Vogelsang, "Understanding the Energy Consumption of Dynamic Random Access Memories," *Proc. MICRO*, 2010, pp. 363-374.

[148] P. Vogt, "Fully Buffered DIMM (FB-DIMM) Server Memory Architecture: Capacity, Performance, Reliability, and Longevity," *Intel Developer Forum*, 2004.

[149] D. Wang, B. Ganesh, N. Tuaycharoen, K. Baynes, A. Jaleel and B. Jacob, "DRAMsim: A Memory System Simulator," *ACM SIGARCH Computer Architecture News - Special Issue* 33(4) (2005), pp. 100-107.

[150] S. Wong, G. Lee and D. Ma, "Modeling of Interconnect Capacitance, Delay, and Crosstalk in VLSI," *IEEE Trans. on Semiconductor Manufacturing* 13(1) (2000), pp. 108-111.

[151] S. Woo, "DRAM and Memory System Trends," Rambus, Inc.

[152] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh and A. Gupta, "The SPLASH-2 Programs: Characterization and Methodological Considerations," *Proc. ISCA*, 1995, pp. 24-36.

[153] X. Wu, W. Zhao, M. Nakamoto, C. Nimmagadda, D. Lisk, S. Gu, R. Radojcic, M. Nowak and Y. Xie, "Electrical Characterization for Intertier Connections and Timing Analysis for 3-D ICs," *IEEE Trans. on VLSI Systems* 20(1) (2012), pp. 186-191.

[154] Y. Xie, G. H. Loh, B. Black and K. Bernstein, "Design Space Exploration for 3D Architectures," *ACM Journal on Emerging Technologies in Computing Systems* 2(2) (2006), pp. 65-103.

[155] H. Yan, Q. Zhou, X. Hong and Z. Li, "Efficient Hierarchical Algorithm for Mixed Mode Placement in Three Dimensional Integrated Circuit Chip Designs," *Tsinghua Science and Technology* 14(2) (2009), pp. 161-169.

[156] H. Yoon, J. Meza, R. Ausavarungnirun, R. A. Harding and O. Mutlu, "Row Buffer Locality Aware Caching Policies for Hybrid Memories," *Proc. ICCD*, 2012, pp. 337-344.

[157] D. H. Yoon, J. Chang, N. Muralimanohar and P. Ranganathan, "BOOM: Enabling Mobile Memory Based Low-Power Server DIMMs," *Proc. ISCA*, 2012, pp 25-36.

[158] M. Zaharia, M. Chowdhury, M. Franklin, S. Shenker and I. Stoica, "Spark: Cluster Computing with Working Sets," *Proc. HotCloud*, 2010.

[159] T. Zhang, J. Abellánn, A. Joshi and A. Coskun, "Thermal Management of Manycore Systems with Silicon-photonic Networks," *Proc. DATE*, 2014, pp. 307-312.

[160] Z. Zhang and C. P. Wong, "Recent Advances in Flip-chip Underfill: Materials, Process, and Reliability," *IEEE Trans. on Advanced Packaging* 27(3) (2004), pp. 515-524.

[161] Y. Zhang, T. E. Sarvey and M. S. Bakir, "Thermal Evaluation of 2.5D Integration Using Bridge-chip Technology: Challenges and Opportunities," *Proc. TCPMT*, 2017, pp. 1101-1110.

[162] X. Zhao, J. Minz and S. K. Lim, "Low-Power and Reliable Clock Network Design for Through-Silicon Via (TSV) Based 3D ICs," *IEEE Trans. on CPMT* 1(2) (2011), pp. 247-259.

[163] H. Zheng and Z. Zhu, "Power and Performance Trade-Offs in Contemporary DRAM System Designs for Multicore Processors," *IEEE Trans. on Computers* 59(8) (2010), pp. 1033-1046.

[164] H. Zheng, J. Lin, Z. Zhang and Z. Zhu, "Decoupled DIMM: Building High-Bandwidth Memory System from Low-Speed DRAM Devices," *Proc. ISCA*, 2009, pp. 255-266.

[165] "System-Level Impact of 3D", *GSRC Workshop* 2010.

[166] *ANSYS,* `https://www.ansys.com`

[167] *Applied Simulation Technology,* `http://www.apsimtech.com`

[168] R. Barth, *personal communication*, February 2011.

[169] Intel's Scalable Memory Buffer. `http://tinyurl.com/7xbt27o`

[170] G. W. Burr, M. J. Breitwisch, M. Franceschini, D. Garetto, K. Gopalakrishnan, B. Jackson, B. Kurdi, C. Lam, L. A. Lastras, A. Padilla, B. Rajendran, S. Raoux and R. S. Shenoy, "Phase Change Memory Technology," `http://arxiv.org/abs/1001.1164v1`, 2010.

[171] CACTI, `http://www.hpl.hp.com/research/cacti/`

[172] *Cadence Design Systems,* `https://www.cadence.com`

[173] *Cerebras Systems,* `https://www.cerebras.net`

[174] N. Chatterjee, R. Balasubramonian, M. Shevgoor, S. Pugsley, A. Udipi, A. Shafiee, K. Sudan, M. Awasthi and Z. Chishti, "USIMM: the Utah SImulated Memory Module," *technical report* UUCS-12-002, University of Utah CS Dept., 2012.

[175] IBM ILOG CPLEX, `www.ilog.com/products/cplex/`

[176] DARPA CHIPS, `http://www.darpa.mil/news-events/2016-07-19`

[177] JEDEC DDR3 Specification JESD79-3E.

[178] JEDEC DDR4 Specification JESD79-4A.

[179] DDR4 SDRAM. http://en.wikipedia.org/wiki/DDR4_SDRAM, 2014.

[180] ESG Memory Engineering, Memory For Dell PowerEdge 12th Generation Servers, 2012.

[181] Dell PowerEdge 11th Generation Servers: R810, R910, and M910. http://goo.gl/30QkU, 2010.

[182] Dell, Dell PowerEdge R910 Technical Guide, http://www.avsys.mx/es/hosting/docs/PowerEdge-R910-Technical-Guide.pdf, 2014.

[183] AMP, TE DDR2 Connector Model, http://www.te.com/documentation/electrical-models/files/slm/DDR2_DIMM_240-Solder_tail.pdf, 2014.

[184] X. Dong, C. Xu, Y. Xie and N. P. Jouppi, "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory," *technical report*, UC Santa Barbara ECE Dept., 2012.

[185] K. Chandrasekar, C. Weis, Y. Li, S. Goossens, M. Jung, O. Naji, B. Akesson, N. Wehn and K. Goossens, "DRAMPower: Open-source DRAM Power and Energy Estimation Tool," *technical report*, Delft University of Technology CE Dept., 2012.

[186] Micron Technical Note, http://www.micron.com/~/media/Documents/Products/TechnicalNote/DRAM/TN4104.pdf

[187] Hot Chips 2017: Intel Deep Dives Into EMIB, https://www.tomshard-ware.com/news/intel-emib-interconnect-fpga-chiplet,35316.html.

[188] Intel EMIB White Paper, https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/wp/wp-01251-enabling-nextgen-with-3d-system-in-package.pdf.

[189] Future-Mobile JEDEC Wide IO Specification JESD229.

[190] Future-Mobile JEDEC Draft Serial Memory Specification.

[191] Intel introduces foveros: 3D die stacking for more than just memory, https://tinyurl.com/y87d6ycc, 2018.

[192] Netlist, HyperCloud HCDIMM Outperforms LRDIMM in Big Data and Big Mem Applications, http://www.netlist.com/files/5413/4193/4890/20120301_NL_White_Paper_HCDIMM_Outperforms_LRDIMM.pdf, 2012.

[193] HP, HP ProLiant DL580 G8 Server technology, http://h20195.www2.hp.com/V2/GetPDF.aspx%2F4AA5-1116ENW.pdf, 2014.

[194] HP ProLiant DL580 G7 Server Technology, http://goo.gl/aOQ3L, 2015.

[195] HP Memory Technology Evolution: An Overview of System Memory Technologies, http://tinyurl.com/7mvktcn.

[196] HP Power Advisor, http://h18000.www1.hp.com/products/solutions/power/index.html.

[197] Hybrid Memory Cube Specification 1.0, http://hybridmemorycube.org/files/SiteDownloads/HMC\%20Specification\%201_0.pdf, 2013.

[198] Heterogeneous integration roadmap 2019 edition, https://eps.ieee.org/technology/heterogeneous-integration-roadmap/2019-edition.html.

[199] IBIS, http://www.eda.org/ibis/, 2014.

[200] Intel, Intel C102/C104 Scalable Memory Buffer, http://www.intel.com/content/dam/www/public/us/en/documents/design-guides/c102-c104-scalable-memory-buffer-tmsdg.pdf, 2014.

[201] Intel Product Specifications, http://ark.intel.com/, 2016.

[202] *International Technology Roadmap for Semiconductors*, 2011 edition, http://www.itrs.net/

[203] Heterogeneous Integration Chapter in ITRS 2.0, http://www.itrs2.net/itrs-reports.html, 2015.

[204] System Integration Chapter in ITRS 2.0, http://www.itrs2.net/itrs-reports.html, 2015.

[205] ITRS Process Integration, Devices and Structures Chapter: http://www.itrs.net/Links/2009ITRS/2009Chapters_2009Tables/2009_PIDS.pdf

[206] ITRS Process Integration, Devices and Structures Tables: http://www.itrs.net/Links/2010ITRS/2010Update/ToPost/2010Tables_PIDS_FOCUS_C_ITRS.xls

[207] JEDEC: http://www.jedec.org

[208] JMP User Guide, http://www.jmp.com

[209] N. P. Jouppi, A. B. Kahng, N. Muralimanohar and V. Srinivas, "HPL-2013-79: CACTI-IO Technical Report," HP Labs, September 2013.

[210] Y. Kim, W. Yang and O. Mutlu, "Ramulator: A Fast and Extensible DRAM Simulator," *technical report*, CMU ECE Dept., 2015.

[211] J. Knudsen, "NanGate 45nm Open Cell Library," *CDNLive, EMEA*, 2008.

[212] A. Lesea, Xilinx, Inc., *personal communication*, 2008.

[213] JEDEC LPDDR2 Specification JESD209-2C.

[214] JEDEC LPDDR3 Specification JESD209-3.

[215] Load-Reduced DIMMs, http://www.micron.com/products/dram-modules/lrdimm, 2015.

[216] K. T. Malladi and M. A. Horowitz, *personal communication*, 2011.

[217] McSim, http://cal.snu.ac.kr/mediawiki/index.php/McSim

[218] *Mentor, A Siemens Business,* https://www.mentor.com

[219] Meta-software Inc., HSPICE Users Manual, Campbell, CA, 1996.

[220] Micron DRAM System Power Calculators, http://www.micron.com/support/dram/power_calc.html

[221] Micron Technology Inc., "Calculating Memory System Power for DDR2 - Technical Note TN-47-07," 2005.

[222] Micron Data Sheets for DDR3 and LPDDR2.

[223] Micron DDR3 SDRAM Part MT41J256M8, 2006.

[224] DDR4 Networking Design Guide, https://www.micron.com/~/media/documents/products/technical-note/dram/tn_4003_ddr4_network_design_guide.pdf, 2014.

[225] Mobile-XDR technical brief http://www.rambus.com/assets/documents/products/mobile_xdr_brief.pdf

[226] R. Myslewski, "HP Busts Out New ProLiant Rack Mount Based on Intel's New Top O'line Server Chippery," http://www.theregister.co.uk/2014/02/19/hp_busts_out_new_proliant_rack_mount_based_on_intels_new_top_o_line_server_chippery/, 2014.

[227] Nvidia: NVIDIA Tesla P100, https://images.nvidia.com/content/pdf/tesla/whitepaper/pascal-architecture-whitepaper.pdf

[228] J. T. Pawlowski, "The Future of Memory Technology," Keynote at The Memory Forum, 2014.

[229] [(Package team engineer), (foundry)], *personal communication*, March 2018.

[230] Py-earth, https://github.com/scikit-learn-contri/py-earth

[231] Python3, https://www.python.org

[232] QCT Memory Analysis, Qualcomm, Inc.

[233] "Challenges and Solutions for Future Main Memory," *Rambus White Paper*, http://tinyurl.com/cetetsz, 2009.

[234] SAP, In-Memory Computing: SAP HANA, http://scn.sap.com/community/hana-in-memory, 2013.

[235] SAS, SAS In-Memory Analytics, http://www.sas.com/en_us/software/in-memory-analytics.html, 2013.

[236] Wind River Simics Full System Simulator, http://www.windriver.com/products/simics/, 2007.

[237] Gary Smith EDA, https://www.garysmitheda.com

[238] J. Stuecheli, "Power Technology for a Smarter Future," IBM, 2014.

[239] Supermicro Solutions, http://www.supermicro.com/products/nfo/Xeon_X10_E5.cfm, 2015.

[240] SVM, scikit-learn.org/stable/modules/svm.html

[241] S. Thoziyoor, N. Muralimanohar, J. Ahn and N. P. Jouppi, "CACTI 5.1. Technical Report," HP Laboratories, 2008.

[242] K. Tran, "High-bandwidth Memory White Paper: Start Your HBM/2.5D Design Today, " *technical report*, Amkor Technology Inc., 2016.

[243] UltraDIMM - Flash Based Ultra-Low Latency Sotrage Device, https://www.sandisk.com/about/media-center/press-releases/2014/sandisk-announces-ulltradimm-design-win-with-huawei, 2014.

[244] Virtex-7 FPGA VC707 evaluation kit, Xilinx.

[245] H. Wong, "A Comparison of Intels 32nm and 22nm Core i5 CPUs: Power, Voltage, Temperature, and Frequency," http://blog.stuffedcow.net/2012/10/intel32nm-22nm-core-i5-comparison/, 2012.

[246] R. Zhang, M. R. Stan and K. Skadron, "Hotspot 6.0: Validation, Acceleration and Extension," *technical report*, University of Virginia CS Dept., 2015.