NORTHWESTERN
UNIVERSITY

Computer Science Department

Technical Report
Number: NU-CS-2022-07

May, 2022

High-performance and Energy-efficient Computing Systems Using Photonics

Haiyang Han

Abstract

The benefits of photonic interconnects and memories cannot be realized through a simple replacement process. This report tries to improve the energy consumption of data center level photonics and adopt emerging photonic devices to build fast and energy-saving caches. We propose architectural designs and show their positive impacts on performance and energy efficiency. Our design and evaluation philosophies lay the groundwork for realizable designs that can be quickly adopted in real life.

Keywords

Optical Interconnect, Emerging Optical and Photonic Technologies, Optical Memory, Non-Volatile Memory, Cache Hierarchy, Data Center Network

NORTHWESTERN UNIVERSITY

High-performance and Energy-efficient Computing Systems Using Photonics

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Computer Engineering

By

Haiyang Han

EVANSTON, ILLINOIS

June 2022

ABSTRACT

High-performance and Energy-efficient Computing Systems Using Photonics

Haiyang Han

Computer systems supported by photonic interconnects and photonic memory devices can reach performance and energy efficiency levels unattainable through purely electronic means across scales, from processor chips to the data center. However, the promised benefits cannot be realized through a simple replacement process; to reach their full potential, several aspects across the entire system stack may need to be redesigned. This thesis tries to explore opportunities to improve the energy consumption of existing data center level photonics as well as adopt emerging photonic devices to build fast and energy-saving cache memories. We identify and propose a number of specific architectural designs that aim at these goals. By studying their feasibility and effectiveness we have shown the positive impact of photonic devices on performance and energy efficiency. Our work also outlines the design and evaluation philosophies of integrating photonics more efficiently into existing computing systems, laying the groundwork for realizable designs that can be quickly adopted in real life.

Acknowledgements

I would first like to express my sincerest gratitude to my advisor Professor Nikos Hardavellas. I thank you for bringing me to Northwestern University, for teaching me research, for constantly challenging me intellectually, for brainstorming and analyzing along side me, for the late nights spent together revising manuscripts. You continued to support and believe in me even in the hardest times along this journey. Without you neither this thesis nor me would have reached this stage.

I would also like to thank Professor Gokhan Memik, Professor Russ Joseph, and Professor Nikos Pleros for being on my committee. Thank you for your guidance, comments, and support throughout my years at Northwestern, especially during my Dissertation Prospectus and Final Defense.

I would like to thank Professor Nikos Pleros, Pavlos Maniotis, Chris Vagionas, and Theoni Alexoudi for laying the research groundwork for optical caches and optical phase change memories, for taking the time to explaining their research outcomes to me who is a new learner. I also thank Dimitris Syrivelis and Nikos Terzenidis for their help with the NetFPGA switch implementation, Arash Farhadi Beldachi and George Kanellos for their help with optical transceiver physical experiments, Srikanth Kandula and Professor Fabian Bustamante for network modeling, and Professor Jie Gu for his help in modeling the laser driver.

I want to give special thanks to Professor Simone Campanoni, Professor Peter Dinda, Professor Vincent St-Armour, Yigit Demir, Georgios Tziantzioulis, Ali Murat Gok, Vijay Kandiah, Enrico Armenio Deiana, and everyone else who either contributed to parts of this work or provided general guidance to me as mentors and peers.

I would also like to thank Chris and Pred, for managing our Zythos computing cluster and always getting back to me quickly. Without your support none of the works in this thesis will be possible. My experiments ran for a total of 754353.12 hours, roughly 86 years.

My friends at Northwestern also deserve my gratitude for being such good companions. To that I thank Xi Chen, Mingzheng Wu, Mingxiao Li, Lingqiao Li, Tong Wei, and Zhekai Deng.

Finally I would like to thank my parents Jun Han, Xianghong Jiao, and my girlfriend Anan Lu. You have been there for me since the beginning.

List of abbreviations

ACK: Acknowledgment

AG: Access gate

ASIC: Application-specific integrated circuit

AWG: Arrayed waveguide grating

CAM: Content addressable memory

CAS: Column address selector

CDF: Cumulative distribution function

CDR: Clock and data recovery

CICQ: Combined input-output queued

CMOS: Complementary metal-oxide-semiconductor

CMP: Chip multiprocessor

CPI: Cycles per instruction

CSW: Cluster switch

DMA: Direct memory access

DRAM: Dynamic random access memory

DVFS: Dynamic voltage and frequency scaling

DWDM: Dense wavelength division multiplexing

EDFA: Erbium-doped fiber amplifier

EDP: Energy-delay product

ELF: Executable and linkable format

EO: Electrical-optical

EPI: Energy per instruction

FC: "Fat Cat" router

FF: Flip-Flop

FIFO: First in first out

FPGA: Field programmable gate array

GST: $GeSbTe$ (germanium-antimony-tellurium)

HMC: Hybrid memory cube

HPC: High performance computing

IC: Integrated circuit

IPC: Instructions per cycle

IRDS: International Roadmap for Devices and Systems

ISA: Instruction set architecture

LC/DC: Laser control for data centers

LLC: Last level cache

MAC: Media access control

MLC: Multi-level cells

MLP: Memory-level parallelism

MR: Micro-ring

MSA: Multisource agreement

MWSR: Multiple-writer single-reader

NACK: Negative-Acknowledgment

NIC: Network interface card

NoC: Network-on-Chip

NRZ: Non-return-to-zero

NUCA: Non-uniform cache access

OCM: Optically connected memory

OE: Optical-electrical

OoO: Out-of-order

O-PCM: Optical phase change memory

QSFP: Quad small form-factor pluggable transceiver

PCI-e: Peripheral component interconnect express

PCM: Phase change memory

PhC: Photonic crystal nanocavity

PON: Passive optical network

PUE: Power usage effectiveness

RAG: Read access gate

RAM: Random access memory

RAS: Row address selector

RGMII: Reduced gigabit media-independent interface

RHEL: Red Hat Enterprise Linux

ROB: Re-order buffer

RSW: Rack switch

R-SWMR: Reservation asssited single-writer multiple-reader

RTT: Round-trip time

SERDES: Serializer/Deserializer

SFP: Small form-factor pluggable transceiver

SRAM: Static random access memory

SSD: Solid state drive

STT-RAM: Spin-transfer torque RAM

TCP/IP: Transmission Control Protocol/Internet Protocol

TDM: Time-division multiplexing

TIA: Trans-impedance amplifier

ToR: Top of the Rack

TSV: Through silicon via

TTL: Time to live

VCSEL: Vertical cavity surface emitting laser

WAG: Write access gate

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

The shift of computer architectures from single-core processors to multi-core processors [162, 64] in the last decades has increased computing power. Parallel softwares taking advantages of Amdahl's Law [80] and multi-core hardware infrastructures have both benefited from and helped push for more efficient data movement and communication. This puts on-chip and off-chip communications, data movement, and data storage in important roles in the modern computing paradigm, demanding high bandwidth, low latency, and low energy consuming approaches. The recent advancements of silicon photonics [82, 155] and optical interconnects [113, 135] provide us with a landscape that breaks current limitations of conventional electrical signaling, which is increasingly challenged by the end of Moore's Law and the breakdown of "Dennard scaling" [53]. In this thesis, we try to take advantage of this opportunity with realistic proposed designs, to provide new insights into adopting existing and future photonic technologies in modern computer architectures.

1.1. Thesis

Computer systems supported by photonic interconnect and memory devices can reach performance and energy efficiency levels unattainable through purely electronic means across scales, from processor chips to the datacenters. However, the promised benefits cannot be realized through a simple replacement process. This thesis tries to rethink several aspects of the system stack to allow photonic interconnect and memory devices to reach

their full potentials in computing systems. We design and analyze the hardware/software architectures from both macro and micro levels with high performance and energy efficiency in mind.

We first try to tackle the increasingly prominent power consumption of network components in data centers. Today, optical interconnects are already being used widely in commercial environments to support rack-to-rack and board-to-board communications [87]. However, the high signal loss of optical components demand high laser powers. The energy efficiency is further held back by the necessity to have lasers stay active, even during periods of network inactivity, which can often be as much as 70–80% of the time [14, 13]. With more power-saving optimizations applied to the server, mechanical, and cooling components of data centers, the relative power that the network consumes is more prominent. Laser power-gating is a promising method to lower laser power when network utilization is low, but hinders performance when network devices need to wait for the lasers to turn on.

We propose the LC⚡DC framework, a data center network architecture where the operating system, the switch, and the transceiver devices are co-designed. We try to turn off unnecessary high power-consuming lasers in the network when we find windows of low network utilization. At the same time full connectivity of the network is maintained. When network utilization is high LC⚡DC turns on links to dynamically adjust to the higher network traffic. We demonstrate the feasibility of LC⚡DC at different levels, including the device, switch, and node. LC⚡DC is able to save on average 60% of the optical transceivers' energy, and up to 27% of the overall data center energy consumption, while only at a cost of 6% additional packet latency.

Next we turn our attention to nanophotonic integrations on a single-node level. On-chip and inter-chip photonic technology, although extensively researched, have not been widely adopted commercially. However the continuously increasing number of cores in one computing instance as well as disintegrated architectures have created a situation where the benefits of photonic technologies can be exploited. Numerous optical on-chip network desgins have been proposed [170, 169, 127, 126, 51, 45, 47, 49, 72, 98, 116, 173] to leverage the high bandwidth and energy efficiency of silicon photonic links. All-optical static random access memory (RAM) cells have also been demonstrated to have low latency and energy consumption [4, 123].

We propose Pho$, an opto-electronic memory hierarchy for chip-multiprocessors (CMPs) that utilize emerging photonic crystal (PhC) based optical memory cells [123] for a shared, fast, and large L1 optical cache. We explore the performance and energy efficiency aspects of architectural ideas that enable this transition. Finally, we present a novel optical Network-on-Chip (NoC) topology Pho$Net and an optical network communication protocol to support the integration of on-chip optical caches. Pho$ is up to 3.89× faster (1.41× on average) over a traditional electronic cache CMP, while achieving up to 90% lower energy-delay product (31% on average). Under realistic assumptions, the Pho$Net optical NoC achieves up to 70% power savings compared to directly applying previously available optical NoC architectures.

The inherent inefficiency of Pho$ is in its electronic last level cache (LLC), which prevents an all-optical cache hierarchy. Improving upon Pho$, we propose an optical phase change memory (O-PCM) [110, 142] enabled architecture where the electronic LLC of Pho$ is replaced with an O-PCM LLC. We perform a design space exploration on the

performance impacts of write queues in mitigating O-PCM's long write latency. We also study the effects of a "no allocation" write policy to reduce the frequency of O-PCM cache writes. Our results show that by employing only an 8-entry write queue and the write policy, the combination of Pho$ and an O-PCM LLC can achieve similar performance to Pho$ and extended cache life time despite O-PCM's long write latency, while providing non-volatility at the LLC level.

As a by product of working on this thesis, we present to the architecture community our "pinballs" of the SPEC CPU2017 benchmark suite. Pinballs are shareable, OS independent files that are compatible with several architectural simulators [29, 147]. They help researchers avoid the long simulation times of modern simulators with statistically sampled program regions that can represent whole program behavior. We generate our own pinballs for SPEC CPU2017 and our validation shows that they are representative of the original benchmarks with an average absolute error rate of 12%. We also make our pinballs public available [75].

The rest of the thesis is organized as follows. Chapter 2 details the LC⚡DC architecture, a feasibility study, and an evaluation of its power and performance. In Chapter 3 we present the Pho$ architecture and evaluate its performance, power, and energy characteristics. In Chapter 4, we present the O-PCM architecture that is extended upon Pho$ and detail the results of our design space exploration in terms of performance and cache life time. Chapter 5 summarizes our motivation, collection process, validation, methodology, and results of CPU2017 pinballs. We discuss potential future work in Chapter 6, give a brief review of related work in Chapter 7, and conclude in Chapter 8.

The LC⚡DC network architecture has been submitted and is under review for NSDI 2023. It's preliminary version was presented at IEEE SUM 2017 [52]. The Pho$ architecture was presented at ISLPED 2021 [73] and was nominated for the Best Paper Award. Its extended version has been accepted and waiting for publication in ACM JETC [74]. The optical phase change memory work is under preparation for both ECOC 2022 and SPIE Photonics West 2023. The CPU2017 PinPoints work has been submitted to arXiv [76], and the artifacts publicly shared with the community [75].

CHAPTER 2

LC⚡DC : Laser Control for Data Center Networks

2.1. Introduction

Optical interconnects have emerged as a promising solution to meet the growing demand for high-bandwidth, low-latency, and energy efficient communication in data centers [3, 67, 143]. A significant fraction of the energy consumption in these networks can be attributed to the laser sources and laser drivers. As we argue, most of this energy is wasted.

For example, a typical 64-port switch with 220 W peak power draws on average 140 W [9] and employs 10G SFP+ optical transceivers at 1 W per port. With this configuration the switch consumes a third of its power on optical transceivers. QSFP (40G) transceivers consume as much as 3.5× more power, raising the power ceiling even further. However, most of the laser energy is wasted. Unlike traditional interconnects that expend most of their energy only during packet transmission, optical interconnects are always on and consume power, even during periods of inactivity. In reality, the interconnect often stays idle for long periods: compute-intensive workloads underutilize the interconnect (common in scientific and many analytic workloads), and servers in data centers often stay idle or exhibit load imbalances (data centers are typically 20–30% utilized [14, 13]).

The natural solution is to turn off the transceiver of an idle link to save energy, and turn it back on when packets arrive to facilitate communication [79]. A naive implementation

of this "*laser gating*", though, risks exposing multiple laser turn-on delays to the end application as a packet typically crosses multiple links to reach its destination, significantly increasing packet latency and lowering performance.

We propose to hide the laser turn-on delay by capitalizing on the path diversity of modern data center interconnects. To service high levels of traffic across a large number of nodes, data centers typically exploit scalable network topologies. For example, full-optical Clos networks are widely deployed in Facebook [143], Google [152] and Microsoft [67] data centers, and flattened butterfly topologies have been proposed as a cost-efficient alternative by Google [3]. All these topologies provide path diversity, i.e., there are multiple paths between any source-destination pair. Instead of turning off links arbitrarily and severing end-to-end paths, which exposes the laser turn-on delay, we propose to turn off only redundant links when utilization is low, and turn them on again when the aggregate workload needs more bandwidth. Maintaining full connectivity removes the laser turn-on latency from the critical path and results in minimal performance degradation.

We also propose to control the server-to-ToR (Top of the Rack) switch links by intercepting socket write calls at the OS level and raising a signal to the Network Interface Card (NIC) lasers to turn on. The TCP/IP processing latency is high enough that allows for ample time to notify the server NIC of the impending traffic. By the time the data are ready to be sent, the laser is already on and locked at the appropriate frequency, thereby allowing the node to save energy with zero performance penalty.

The main idea of laser gating is not new. Such techniques have been proposed before for on-chip interconnects [45, 48, 46, 50]. An earlier study capitalizing on path diversity for laser control in on-chip interconnects [50] was extended to data centers but only as

a conceptual study; no evaluation was performed on traffic similar to modern large-scale data centers (only a university data center traffic trace was used), and the feasibility of the technique was not proven or discussed. After all, the latency to perform network control plane changes has been previously shown to be in the ms scale [59], which could render such a technique useless, and the latency to perform link retraining for Clock and Data Recovery (CDR) can be prohibitively high.

In this chapter we aim to set the record straight. We evaluate laser gating on multi-stage data center networks with a Clos architecture similar to the one found in modern hyperscale data centers [67, 143, 152] using traffic patterns that closely approximate real-world traffic [67, 89, 143]. We demonstrate that control plane changes can be performed in a matter of ns by implementing LC⚡DC on a 6×6 10.8 Gbit/s switch on an FPGA. We implement a device driver on a modern Linux operating system and kernel changes that intercept socket write calls to raise a signal and alert the NIC of imminent outgoing traffic. We measure the latency of the TCP/IP processing and show that the NIC has ample time to turn on its transceiver while the outgoing packet is being prepared. Finally, through a combination of physical experiments and analog SPICE simulations of laser driver circuit models, we show that optical transceivers and their electronic drivers can be turned on at µs scales. Collectively, these results demonstrate that laser power gating at the data center scale is indeed feasible, and can be driven by the OS and network switch layers. We then estimate the energy savings that can be achieved on a variety of scenarios through simulations and technology projections.

More specifically, our contributions are:

- We propose LC⚡DC (Laser Control for Data Centers), a data center network system architecture in which the operating system, the switch, and the optical components are co-designed to achieve energy proportionality.
- We demonstrate the feasibility of employing LC⚡DC at servers and global switches through modifications in the Linux kernel and device drivers, switch design on FPGA boards, physical experiments with optical devices, and analog circuit simulations.
- We develop a data center traffic generator that models the traffic exhibited at Facebook and Microsoft. We show that our traffic generator produces CDFs of flow size and flow intervals that closely match real-world traffic.
- We evaluate LC⚡DC on models of Facebook and Microsoft data center traffic, as well as traffic traces from a university data center. LC⚡DC saves on average 60% of the optical transceiver power (68% max) at the cost of 6% higher packet delay.
- As servers and cooling, electrical and mechanical systems become increasingly more energy efficient, we project that the network will command a larger fraction of the overall data center energy consumption. We estimate the potential savings of LC⚡DC on a hypothetical future data center that applies multiple server-level energy optimizations. We find that LC⚡DC can save 12% and 21% of the data center energy on average when deactivating transceivers or transceivers and switch PHY and NIC electronics, respectively, even for cases where the server utilization approaches 70%.

2.2. Motivation

Power and energy efficiency have been at the forefront of research in circuits and computer architecture for at least 15 years [156, 157, 24]. Innovations in these fields coupled with technological advances in materials, semiconductor processes and packaging yield lower-power devices at each technology node [38, 164, 21, 95]. The end result is the rapidly increasing energy efficiency of server components, including memory, storage, processors, and ultimately the servers themselves. Simultaneously, the high power demands of modern data centers has pushed data center operators to drastically reduce power inefficiencies. As a result, modern state-of-the-art data centers reduced the power overhead for cooling, electrical and mechanical systems from more than 2× a decade ago to only 6% today [112, 66]. As servers deliver increasingly higher performance per Watt, and data center overheads are aggressively eliminated, their contribution to the overall data center power consumption drops, exposing other components that have not yet received similar attention [79]. Data center networks are one of these components, and its relative power consumption rises as innovations in other sectors reduce the power draw of other data center components.

Figure 2.1 shows the breakdown of data center power as various optimizations are applied on servers. For generality, we carry out the same study across various network designs from the literature: a Clos Facebook site [143], a Flattened Butterfly interconnect by Google [3], and three Fat-Tree networks derived from [60] that are either readily available using off-the-shelf components (Fat-Tree 1), or require board and chassis engineering for higher efficiency and lower cost (Fat-Tree 2), or require board, chassis and new ASIC design (Fat-Tree 3). All designs are modeled using the exact component counts and

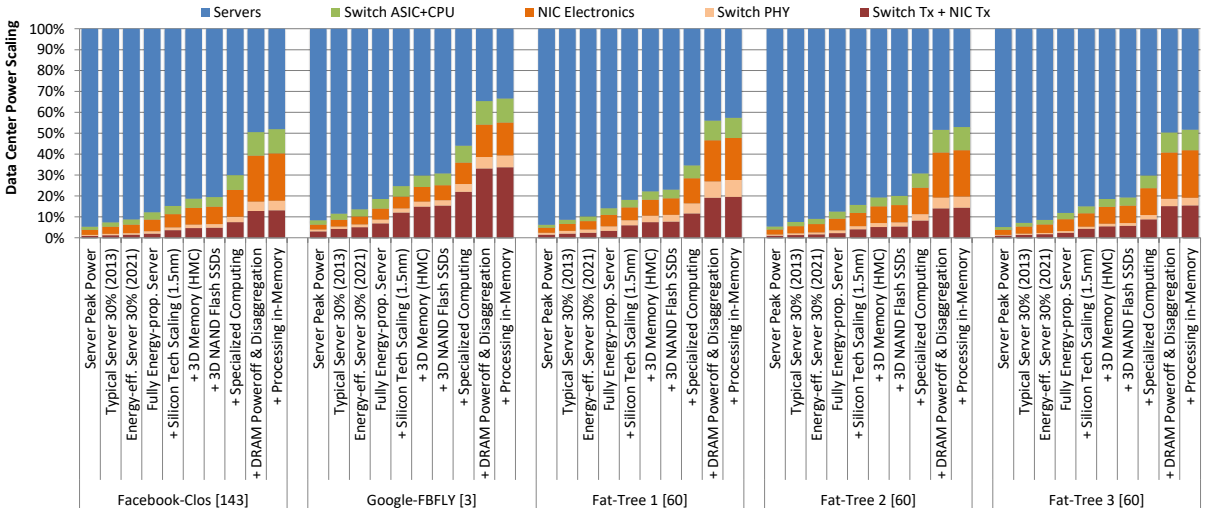


Figure 2.1. Data center power breakdown change as a function of server system optimizations, across various network designs.

connectivity described in their respective publications. Figure 2.1 breaks down the data center power into:

- Server systems
- Switch ASIC and CPU chips (28 W per switch) [60]
- Server network card (NIC) electronics (10 W) [3]
- Switch PHY chips that implement the Layer 1 protocol exposed by the switch (0.8 W, one PHY per port) [60]
- Optical transceivers on switch and NIC card ports, assuming 1 W for 10G SFP+, 2.4 W for 40G QSFP [60]

The figure excludes the overhead for cooling, electrical and mechanical systems within the data center, as the additional overhead these systems impose is proportional to the power draw of the server, storage and networking equipment. We note that modern hyperscale data centers often achieve Power Usage Effectiveness (PUE) of 1.06 [66, 112] and

report a comprehensive trailing twelve-month PUE of 1.10 [66] across all large-scale data centers, in all seasons, including all sources of overhead. Thus, the additional overhead of large-scale data centers nowadays represents a relatively small fraction of their overall energy consumption.

Most readers would be familiar with the left-most stacked bar for each network, where 92–95% of the power is consumed by the servers and only 5–8% goes to the interconnect. However, this assumes that all servers run at 100% utilization and draw peak power all the time. In reality, servers are typically only 20–30% utilized [14, 13]. Thus, their contribution to data center power is lower, and the relative importance of other components grows. The second stacked bar shows this effect assuming typical servers circa 2013 [58].

The third bar shows the effect of 30% server utilization on a best-of-class modern server that is nearly energy-proportional: the Lenovo Think System SR665, which currently has the highest performance per Watt as measured by an audited SPECpower benchmark [158]. This server expends 58% of its power at 30% utilization (compared to 70% for the 2013 server [13]). The fourth bar continues that trend and models a fully energy-proportional server at 30% utilization [14, 58, 13] at which point it consumes 40% of its peak power. On average, a data center with fully energy-proportional servers at 30% utilization seems to spend about 86% of its energy on the servers and the remaining 14% on the network.

The following bars apply successively more optimizations on the server components (and if applicable the switch and NIC electronics) to account for the transition from 7 nm to 1.5 nm CMOS technology following IRDS projections [38, 22]; the use of modern memory technology (e.g., Micron's 3D hybrid memory cube, HMC) [136, 22]; the introduction

of 16-die-stacked 3D NAND Flash for solid-state drives [164, 8]; and the employment of specialized computing which is modeled after the Catapult project [140] in Microsoft that deployed FPGAs in production data centers to off-load computations from conventional general-purpose processors.

Finally, the last two bars show the impact of applying various recent technologies such as DRAM refresh reduction [93], DRAM idle power-off [182], memory disaggregation [120], and near-memory processing [100]. We model the impact of these optimizations by employing them only on the appropriate components, following the typical power profile of data-center-class servers [58] and switches [9, 60].

As Figure 2.1 indicates, with each optimization the relative power consumption of servers drops, to the point where the network becomes a major component. Our projections indicate that, unless something is done, the thousands of transceivers employed in a data center network will account for 20% of the data center power consumption on average across network designs, after these series of optimizations are applied on the servers. Moreover, the combined switch PHY chips, server NIC electronics and transceivers will account for as much as 46% of the data center power. Thus, we argue it is time to start optimizing the network not only for latency, bandwidth and cost, but also for power and energy efficiency.

It is important to note that the optimizations we model in this study are carefully selected to be conservative and readily available, and they are solidly backed by experimental characterization of existing products. The technology projections conform to the roadmap

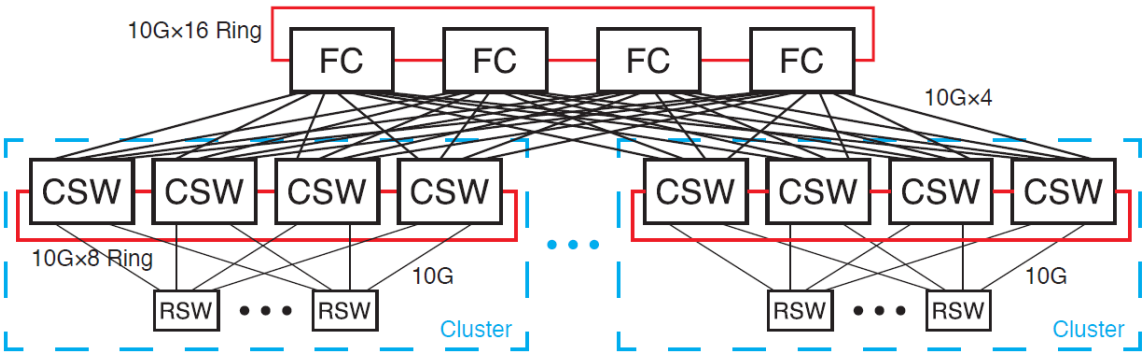


Figure 2.2. Data center network configuration, similar to design from Roy *et al.* [143].

that the semiconductor industry collectively sets every year since 1993 [38, 22], the memory and SSD devices are already commercially available [136, 125, 111, 164], nearly full-energy-proportional servers are commercially available today [158], and some data centers already deploy specialized computing (e.g., Microsoft's Catapult [140], Google's Tensor Processing Units [86]).

To capture a wider range or projections, we also include two more sophisticated energy efficiency optimizations that have been in active development in industry for more than a decade, and while they are not mainstream products yet, they are in advanced stages of development. These optimizations include near-memory processing [100] and disaggregation [120]. There is a large number of much more aggressive optimizations that we explicitly chose not to include, as their ability to scale up to production at reasonable cost is unknown, or they are not a good fit for hypercale data centers, or simply because they are not commercially available yet, despite their high potential (e.g., STT-RAM, PCM, near-threshold-voltage processors, spintronics, neuromorphic processors, and chip- and board-level photonics).

2.3. LC⚡DC Architecture

We propose LC⚡DC to minimize the amount of power spent on optical transceivers employed in switches and server NIC cards by turning them off when they are not needed. Moreover, we can extend the design of LC⚡DC to also put the switch PHY chips and the server NIC electronics into a low-power state at the same time that the laser is switched off. While we do not study this extension in this chapter, it can address as much as 46% of the projected data center power consumption (Figure 2.1), even after accounting for the CMOS scaling of the PHY and NIC electronics.

To minimize the performance impact of waiting for the lasers to turn on, LC⚡DC deactivates only redundant links, while maintaining full network connectivity. As there is always a path connecting any pair of nodes, packets can still reach their destination while links are activated. LC⚡DC monitors the interconnect traffic and turns off links when the utilization is low to save energy, and activates additional links when the utilization is high to increase performance.

2.3.1. LC⚡DC Operation at the Switch Level

LC⚡DC is applicable to any network topology with path diversity. In this chapter we evaluate it in a network similar to a site in the Facebook data center [143]. Figure 2.2 presents the network design. Each rack has 48 nodes which connect to a Rack Switch (RSW). 32 RSW's form a Cluster and connect to 4 Cluster Switches (CSW). 4 Clusters (16 CSWs) connect to 4 "Fat Cat" Routers (FC) to form a site. The RSWs have 48 10G (downlink) input-output ports and 4 10G uplinks to CSW intermediate routers (12:1

oversubscription). In turn, the CSW's provide 4 40G uplinks to FC switches (2:1 oversubscription). Each set of CSWs within a cluster and the FCs are connected on a ring formed by 8 10G links and 16 10G links respectively, for load balancing.

LC⚡DC controls each tier independently. Each RSW has 4 uplinks to connect to all the CSWs in its cluster (path divergence). Each one of these uplinks defines a *Stage*. When we say that stage k is active, we mean that links 1 through k are active. Initially only one stage per RSW is active. LC⚡DC on each switch estimates network traffic by monitoring the buffer depth (buffer utilization, aka queue backlog) of its active links, which is an accurate and lightweight method [32]. When a buffer's depth exceeds a tunable threshold (high watermark), the RSW turns on an additional stage to provide higher bandwidth and minimize queueing delay. The switch sends a control message through the already active stages to the corresponding CSW informing it of the new stage activation, and once its transceiver is active and has received an acknowledgement from the CSW that the receiving side is active, it starts using the additional link.

The newly activated stage turns off when the RSW that activated it becomes underutilized, i.e., its buffer utilization falls below a low watermark. In that case, LC⚡DC realizes that the additional bandwidth provided by the redundant link is not necessary, and should turn it off. The RSW stops receiving outgoing messages in this port, serves all the packets in its buffers, and then notifies the corresponding CSW with a stage turn-off message which deactivates the last activated stage. Once the CSW acknowledges the link deactivation, RSW turns off its transceiver too. Link activation and deactivation at CSWs and FCs is performed similarly.

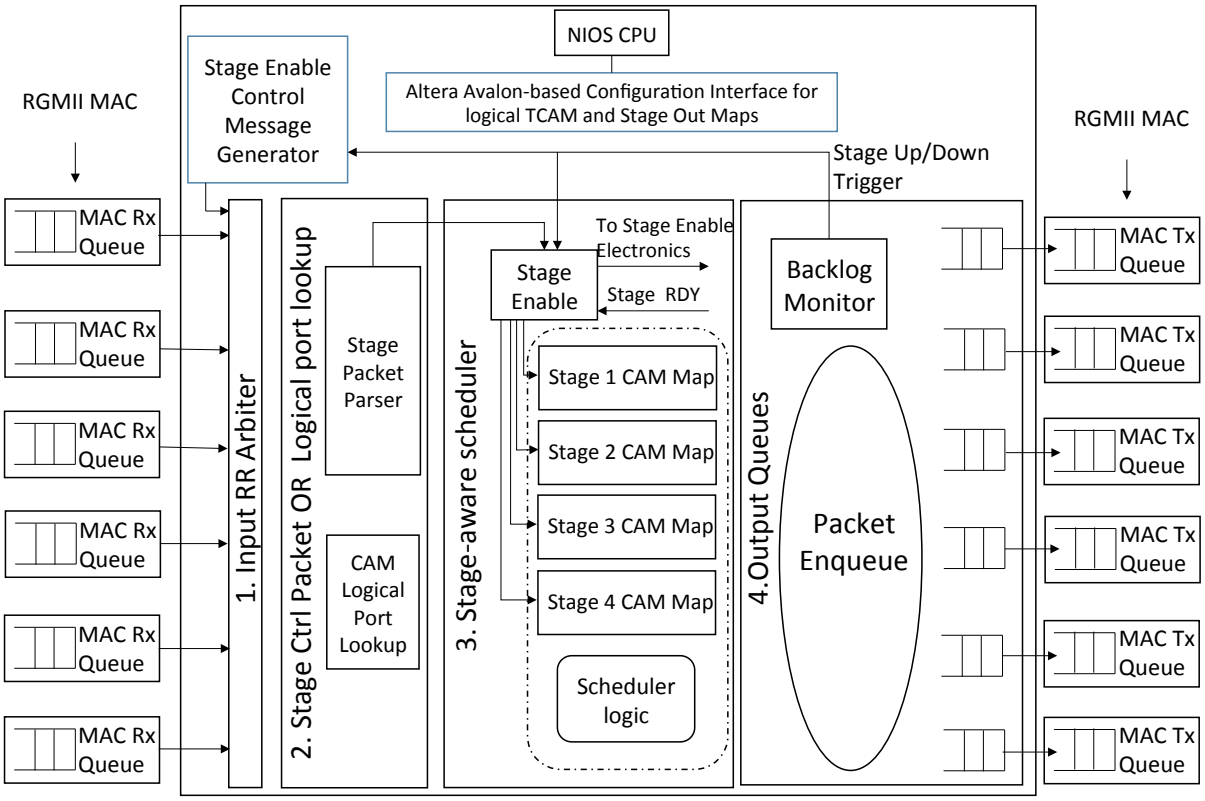


Figure 2.3. LC⚡DC switch architecture.

LC⚡DC adaptively routes traffic through only the active stages which achieves load balancing, turns on lasers on the side and hides their turn-on delay, and avoids unnecessary link activation which maximizes the energy savings.

2.3.2. LC⚡DC Switch Architecture

Figure 2.3 shows the architecture of the LC⚡DC switch. LC⚡DC is a combined input-output queued switch (CIOQ) [34]. At the input, buffering relies on the hardware queues of RGMII MAC, while at the output the design features one RAM-based queue per

physical port. The LC⚡DC datapath is 64-bit wide and adds a single annotation flit to each packet that flows through it to pass information from one stage to the next. The design implements a control message channel that uses the same physical ports as data packets (in-band).

2.3.2.1. Dataplane Pipeline Stages. The Ethernet frames are pulled from RGMII MAC queues that drive the physical interfaces using a round robin arbiter. Besides the physical input ports, the switch features a virtual port that interfaces a 2-port memory to the arbiter. The memory is pre-programmed with all flavors of control packets that are forwarded to other switches to initiate LC⚡DC stage changes. The arbiter polls the virtual port out-of-order to prioritize the forwarding of generated control packets.

In the next pipeline stage the processing differs depending on whether the packet is LC⚡DC control or not. The LC⚡DC control packet is an ethernet frame where bytes 13–14 form the LC⚡DC Ethernet type (0x9100). The next 8 bytes contain the *senderID* (4 B), the *stageID* (2 B), and the *TTL* (2 B). The packet is then padded to the minimum ethernet frame size. The *stageID* denotes the LC⚡DC stage to be enabled and the possible values are deployment-wide agreed. The *TTL* designates the number of switching layers that the control packet may flow through before it is discarded. This approach simplifies the distribution scheme by allowing control packets to get forwarded from all output ports of each switch without worrying for endless loops. Accordingly, the control packet processing checks the *stageID* and sends the appropriate notification to the LC⚡DC stage enabling component, updates the *TTL* and drops the packet (if zero) or propagates it for forwarding. Finally, the *senderID* designates the control packet sender, so it is used to determine if the packet was generated in the local switch. If it was, it is directly

forwarded to the scheduler because the stage transition process is initiated before control packet generation.

The non-local packets initiate a Content Addressable Memory (CAM) lookup to match the packet destination Ethernet MAC address with a logical port, which is a deployment-unique identifier of the destination switch where the packet recipient is attached. This switch addressing scheme is internal to LC⚡DC and has to be configured by the control plane (Section 2.3.2.2). For multicast support, special logical port identifiers are programmed by the control plane for each multicast tree. Accordingly, the logical port identifier is pushed to the packet annotation space and a multicast bit is set when needed before the packet propagates to the scheduler pipeline stage.

The scheduler load-balances the incoming traffic over the available output physical port queues based on the packet destination and the LC⚡DC stage that is enabled. As the provided packet destination is a logical port, the scheduler has access to a series of binary CAM tables, one per LC⚡DC stage, to determine the physical output port options. These CAM tables take the logical port as input and provide a binary map at the output where all the possible physical ports that may be used for this destination, based on the enabled LC⚡DC stage, are one-hot encoded. The scheduler uses each time a single CAM that corresponds to the currently enabled stage. A simple weighted scheduling algorithm chooses the output queue with the minimum backlog from the ones that belong to the given map. In case of multicast packets, the scheduler places a copy to all output queues of the encoded map, which is encoded accordingly to implement the multicast. This step concludes the LC⚡DC datapath operation.

Moreover, three peripheral components orchestrate the LC⚡DC stage transitions. First, the queue backlog monitor component inspects all the output queue backlogs. When a backlog exceeds an administrator defined high watermark, the stage up trigger is sent in parallel to the other two components: (1) the stage enable component and (2) the stage enable control message generator. The former immediately drives the electronics to enable the next stage and when it receives a stage ready signal it enables the CAM stage table that corresponds to the currently active output ports. The latter activates the virtual input port so that the arbiter pulls the proper control packet to notify the rest of the deployment for the state change. If the backlogs become low, the backlog monitor sends to the stage enable component the stage down trigger, which immediately enables the appropriate CAM table while it requests the shutdown of specific physical ports.

2.3.2.2. LC⚡DC Control Plane. The LC⚡DC switch requires control plane support that has overview of the whole network deployment and device topology in order to provide the required forwarding information on all CAM tables, the thresholds on the backlog monitors and the programming of the control packets for the underlying switch fabrics. For this particular purpose the current design features an Altera Avalon bus interface that connects the aforementioned components to the Altera NIOS processor, a softcore processor solution that is provided by the FPGA platform we use. Therefore, low-level software-based control plane support can be realized on each switch, which can then be integrated with data center network orchestration tools like OpenStack Neutron according to the SDN paradigm.

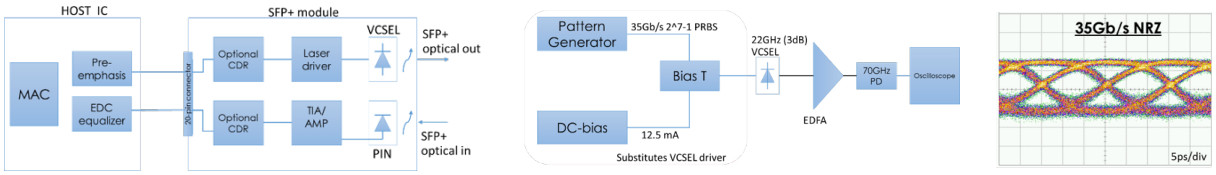
2.3.3. OS and Device Driver Design for Node-Level LC⚡DC

In addition to controlling the redundant links, LC⚡DC independently controls the transceivers on each server's NIC card, by using a modified network card device driver which is controlled by the system as a kernel module. LC⚡DC intercepts the Linux `sendmsg()` system call and replaces it with our version of `sendmsg()` by modifying the system call table at driver module initialization. Upon a user-level `socket_write()` call, the driver signals its laser to turn on and then invokes the original `sendmsg()` function. While the laser turns on, the payload goes through the TCP/IP stack processing. By the time the driver-specific transmit function is called to transmit over the fiber, sufficient time has elapsed for the laser to be fully operational and transmit the data. Changes to the Linux driver and kernel module are minimal at around 200 lines of code.

2.4. Feasibility Study

In this section we perform a comprehensive study on the feasibility of LC⚡DC . At the device level, we experimentally measured the turn-on delays of a commercially available optical transceiver and performed SPICE simulations for verification. At the switch level, we used Verilog to implement an LC⚡DC switch on FPGA and measured its delays. At the node/OS level we experimentally measured the time between an OS `sendmsg` system call and actual bits being sent to the physical link to determine if the NIC has ample time to turn on its lasers without penalty.

2.4.1. Feasibility at the Device Level



(a) SFP+ transceiver module configuration.

(b) Experimental setup for the optical modules.

(c) Output Eye diagram.

Figure 2.4. Physical on/off timing experiment with a VCSEL laser device.

2.4.1.1. Optical Transceiver Turn-on Delay. The datacom transceiver modules that are largely deployed in current data center implementations mainly rely on SFP+ modules for up to 10 Gbit/s link bandwidths and QSFP modules, which is a quadruple form of SFP+, for link bandwidths in the range of 40 Gbit/s to 100 Gbit/s. The simplest SFP+ form of transceiver is shown in Figure 2.4a and comprises a transmitter side which includes a laser component followed by the laser driver circuitry, and a receiver side that includes a photodetector followed by the pre-amplifier, also called trans-impedance amplifier (TIA), and the post-amplifier. Optional additional circuitry may include a CDR (clock and data recovery) circuit in both transceiver sides.

In such transceiver configurations, the transmitter turn on/off timings are denoted as Tx_Disable assert time / Tx_negate assert time and are defined in the SFP+ multisource agreement (MSA) [39]. The MSA specifies that the transmitter turn on/off timings are 100 µs and 1 ms respectively, while the receiver turn on/off times denoted as Rx_LOS assert delay / Rx_LOS negate delay are 100 µs each. However, the MSAs have set the timing boundaries well beyond any safety margins of stable operation in order to fully relax the design and cost requirements of the transceivers [39]. In fact, though both optical

and electrical components comprising the transceivers allow for significantly faster operations, this has never been a design objective for commercial manufacturers of Datacom transceiver modules, and hence they are not optimized for it.

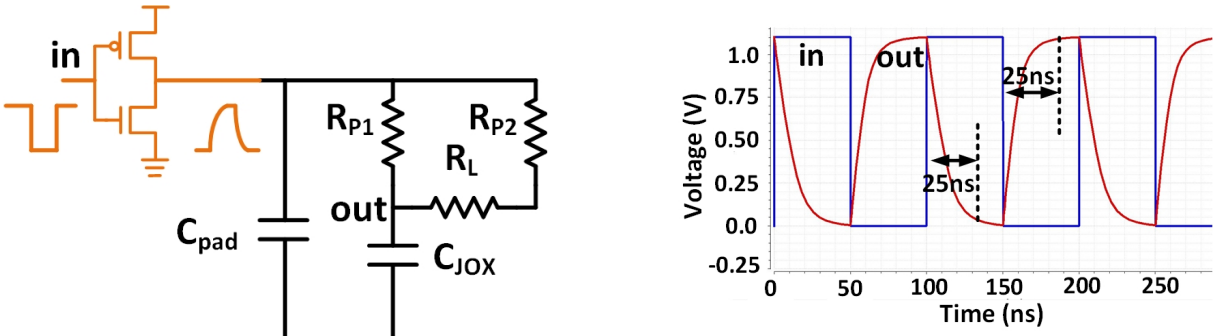
This becomes evident for transceivers in the SFP+ form-factor that are deployed in PON (Passive Optical Network) applications, such as the 10GE-PON SFP+ transceivers. For these transceivers, the necessity for burst operation has led to their commercial implementations exhibiting turn on/off times of 512 ns each [57, 81, 56]. These transceivers retain all other Datacom SFP+ specifications, such as power consumption, bit rate, etc, and thus demonstrate that transceiver for Datacom applications (e.g., data centers) can also be implemented with equally fast µs-scale turn on/off timings.

To provide further evidence that the optical components can be turned on/off at such high speed, we set an experimental setup of a manufactured VCSEL device and a simulation model of the electrical circuitry to assess the minimum timings required for turning on/off such transceivers. Figure 2.4b shows the experimental setup that reveals the turn on/off timings for the optical components. We have chosen the 22 GHz VCSEL laser of [91] as the optical source, since VCSELs are typical laser sources for Datacom transceivers. We consider the non-return-to-zero (NRZ) direct modulation bandwidth as the laser turn on/off frequency. A pseudo-random binary sequence with a word length of 2^7-1 and a bit rate of 35 Gbit/s was generated from a commercial pattern generator. The output of the pattern generator was a single-ended NRZ signal with a swing of 650 mV peak-to-peak, while a bias tee superimposed this data signal on a DC bias current of 12.4 mA. This setup substitutes the laser driving circuitry of Figure 2.4a, and defines the electrical signal specs to be applied to the VCSEL through RF probes. The optical

signal generated by the VCSEL was then pre-amplified through an erbium-doped fiber amplifier (EDFA) and launched to a Finisar XPDV3120R 70 GHz photodetector with 3 V bias. The electrical signal at the output of the PD was then captured by an Infinium sampling scope, revealing a clear eye-pattern for up to 35 Gbit/s as shown in Figure 2.4c. This implies that both transmitter and receiver optical components have the ability to be turned on/off at timing below 15 ps respectively.

The experiment above demonstrates that the turn on/off speed achievable by laser devices is well below the µs-scale that LC⚡DC requires. The electrical integrated circuit for the receiver can also be sufficiently fast for LC⚡DC application. An electrical circuit that even includes a burst mode CDR has been recently shown to exhibit optical power calibration in 12.5 ps and phase lock in 18.5 ps [144], both well below 1 µs. We emphasize that switching off links and their lasers does not modify the links; the link characteristics are the same when the link is powered up again. This makes the link amenable to clock phase caching, which was recently demonstrated to provide clock and data recovery times below 625 ps [36, 35, 12] on a real-time prototype with commercial transceivers, comfortably within the µs requirement of LC⚡DC . This result was further validated against temperature variation and clock jitter [36], demonstrating its resilience and applicability to real-world scenarios.

To fully assess the lower possible boundaries of the transceiver timing specifications, the only remaining component in our device-level feasibility study, we derived a SPICE-based analog simulation model for the laser driver depicted in Figure 2.5a. In this configuration, we considered the RLC electrical circuit equivalent for the VCSEL as provided by Finisar in [63].



(a) Lumped model for VCSEL laser [63].

(b) Spice simulation waveform in a 45 nm CMOS technology.

Figure 2.5. Spice simulation on power gating a VCSEL laser.

2.4.1.2. Electronic Circuit Simulation. We created a spice simulation in a 45 nm CMOS technology following the specification provided from VCSEL laser model [63]. As shown in Figure 2.5a, the model contains simple lumped components for bonding pad C_{PAD} (68.3 pF), distributed resistance components R_{P1}(215 Ω), R_{P2}(147 Ω), and R_L(1690 Ω) for mirror, and combined junction and oxide capacitance C_{JOX}(34 pF). This simplified lumped model has been verified from DC to high frequency (25 GHz) to match well with measured responses from the physical VCSEL laser [63]. In our simulation, a small CMOS driver with transistor sizes of 10 µm was used to turn on and off the laser block. Simulation in Figure 2.5b shows that the laser junction voltage can be driven within 25 ns with a large signal magnitude using our small CMOS driver. This simulation shows that such a laser source can be power gated well within 100 ns or 1 µs time window.

2.4.2. Feasibility at the Switch Level

Similar to transceiver modules, commercially available switches do not allow for fast nanosecond-scale updates to port maps or rules. While technologically this should be

feasible within a few processor cycles, commercial designs never had the incentive to optimize it to a few nanoseconds.

To demonstrate the feasibility of LC⚡DC at the switch layer we implemented a prototype of the described LC⚡DC pipeline architecture in Verilog as a 6×6 switch, with small-sized CAMs (100 entries each) and support for four LC⚡DC stages (Figure 2.3). The design targets an Altera Stratix V GT platform where it achieves a clock rate of 169.32 MHz, providing a 10.8 Gbit/s backplane. The overall latency from the time a packet flit enters the pipeline until it is delivered to the output queues is 7 cycles (2 cycles for the logical port lookup, 2 for the stage out port map lookup, 2 for the scheduler, and 1 for placing at the output queue). The cycle count is expected to grow with the number of output ports because the scheduler checks all backlogs before queueing a packet.

Our design also demonstrates fast stage trigger generation. When the backlog monitor observes a threshold violation (checked on every cycle) it signals the stage enable component in the same cycle (<5.8 ns delay). When a control packet that initiates the stage change appears at the input, 2 cycles elapse before the proper flit is parsed (12.8 ns). As soon as the stage enable component receives the ready signal from the output ports, it enables the appropriate stage CAM lookup table on the next cycle so that the next packet (whenever it arrives) is forwarded according to the new stage.

Our FPGA prototype demonstrates that it is feasible to implement a fast LC⚡DC switch with ns-scale latencies. ASIC implementations would be even faster than our FPGA implementation, but they are outside the scope of this chapter.

2.4.3. Feasibility at the Node Level

Using our modified Linux network device driver we measured the latency between the hypothetical "laser turn on" command that the device driver will issue to the NIC transceiver, and the subsequent call to the device-specific transmit function that starts sending the bits through the physical link. To do this we took timestamps with our version of the `sendmsg()` system call and NIC device transmit function were invoked. Our test-bed consists of an Intel 82579LM Ethernet card and an Intel Core™ i5 2520M 2.5 GHz processor running Linux kernel 4.2.0 on Ubuntu 15.10. We measured a mean elapsed time of 3.2 µs over 100k samples on a completely idle system at runlevel 1 to minimize perturbations from other kernel services, `TCP_NODELAY` to eliminate small packet aggregation, and with hyper-threading, frequency governors, and all but one cores disabled to minimize perturbations due to thread migration or core frequency changes.

Our results corroborate independent measurements in literature of 3.7 µs for a packet to traverse the TCP/IP stack [97]. These results independently measured that it takes 950 ns for a process to send a message to the socket interface on a connection that has already been established. Evoking a socket write begins the TCP layer to initiate transmission, copy the application buffer into the transmit queue in kernel space and prepare a datagram for the IP layer (260 ns). Then the IP layer does routing, segmentation, processes the IP header, and eventually calls the network device driver (550 ns). The network device driver constructs the output packet queue entry and calls the precise hardware implementation of the NIC card to transmit the frame by passing a pointer to the packet descriptor (430 ns). This causes a control register write within the NIC to set up a DMA transfer to fetch the pointer, and when it completes control is handed to the NIC card

(400 ns). Another 760 ns are consumed by the NIC to process the core register write, interpret the descriptor, and based on the descriptor initiate a DMA to fetch from main memory the data of the packet to transmit. Each 64-byte cache line access to memory takes an estimated 400 ns to propagate from the PCIe signal pins to memory and back. Thus, it takes a total of 3.75 µs for an application to launch a packet onto the fiber interface. Thus, the server's NIC card will have ample time to turn on its laser and it will impose no laser turn-on delay to the sending process.

2.5. Experimental Methodology

The goals of our evaluation are to determine LC⚡DC 's power and energy savings on the transceivers and data center levels as well as any performance overheads. We model the data center network in Figure 2.2 using the BookSim cycle-accurate network simulator [41] that we modified in-house. We faithfully model all traffic, including the additional control packets generated by LC⚡DC . The simulator is supplied packets from a traffic generator that shapes traffic to conform to the distributions prevalent in large-scale data centers. The traffic generator models the traffic characteristics shown in [67, 89, 143]. In addition to large-scale data centers with heavy traffic, we evaluate the performance of LC⚡DC on more traditional data centers that exhibit lower traffic demand. For that, we use snippets of traces collected from routers in a university data center [18].

The feasibility study (Section 2.4) demonstrated that optical devices can turn on/off at ns–µs scale. Commercial 10 Gbit/s SFP+ products [44] which have maximum turn-on/off delays of 100 ns are available, but to remain conservative in our evaluation, we model laser turn on/off times based on a commercially available SFP+ module (MRV

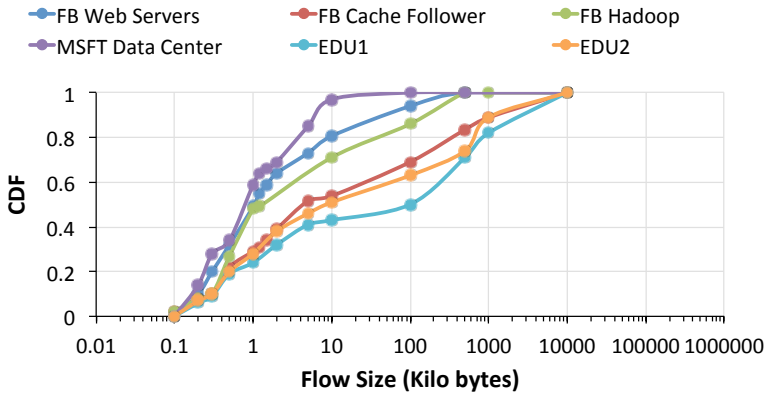
SFPFC401) which has a turn-on/off delay of 1 µs /10 µs [115] (only the delay numbers are used). We model the delay characteristics of routers as derived from our feasibility study (Section 2.4), and we calculate link latency based on the traversed optical fiber length.

We estimate network performance by measuring the average packet delivery latency, and the energy savings by measuring the fraction of the time each link is deactivated. We set the high watermark at 75% buffer utilization for stage activation, and the low watermark at 22% buffer utilization for stage deactivation (experimentally determined to balance energy savings with network performance).

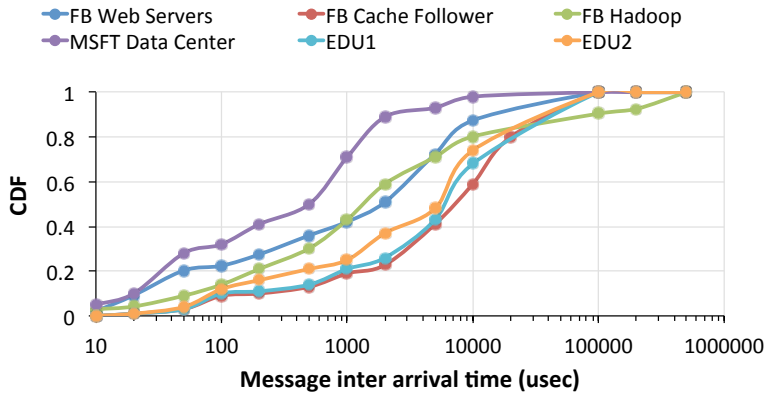
2.6. Experimental Results

2.6.1. Input Data

Figure 2.6 shows the data traffic injected into our simulated network. We created a traffic generator that produces traffic that closely approximates the network traffic originating from large-scale data centers in Facebook [143], Microsoft [67, 89], as well as in higher education settings [18]. Comparing the distribution of traffic from our generator with the large-scale data center traffic from published measurements [143, 67, 89], we confirm that they have similar CDFs, as is shown in Figure 2.7. We measure the Pearson r coefficient to be between 0.979–0.992 for the flow size CDF, and 0.894–0.998 for the flow interval CDF. Thus, our simulations are conducted under conditions that closely approximate real-world environments.



(a) Flow size CDF.

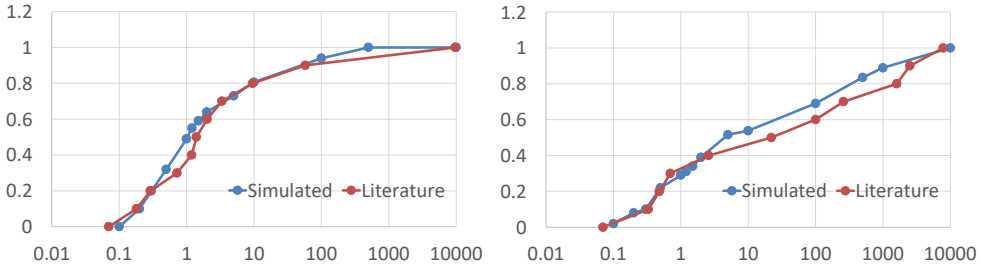


(b) Flow interval CDF.

Figure 2.6. CDF of traffic data input used in simulation.

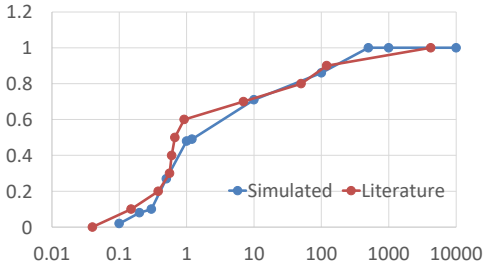
2.6.2. Results

Figure 2.8 shows the portion of the network that is activated during the execution of the traffic workload for each modelled traffic. Most traffic types exhibit sparse and bursty packet injection trends. Thus, LC⚡DC finds windows of low utilization to deactivate a link, and 87% of the time on average half of the network is deactivated, indicating the potential to achieve significant power savings. The Microsoft data center traffic presents

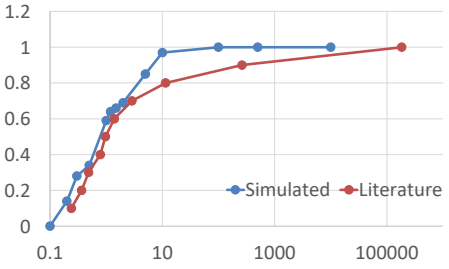


(a) Flow size: FB web server.

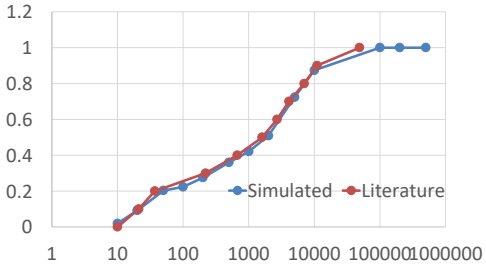
(b) Flow size: FB cache follower.



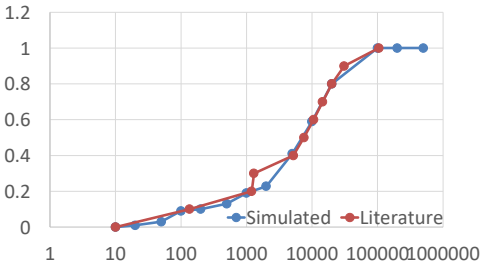
(c) Flow size: FB Hadoop.



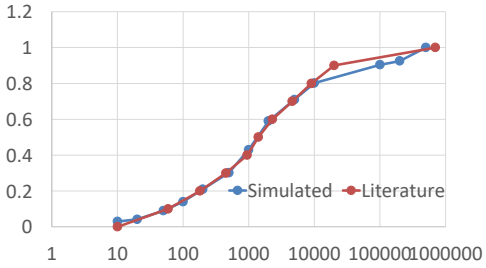
(d) Flow size: MS data center.



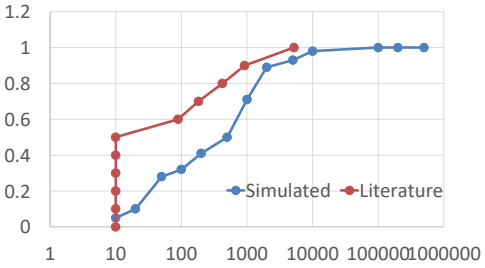
(e) Flow interval: FB web server.



(f) Flow interval: FB cache follower.



(g) Flow interval: FB Hadoop.



(h) Flow interval: MSFT data center.

Figure 2.7. Comparison of simulated traffic CDF of flow size and flow intervals vs. target large-scale data center traffic from published measurements.

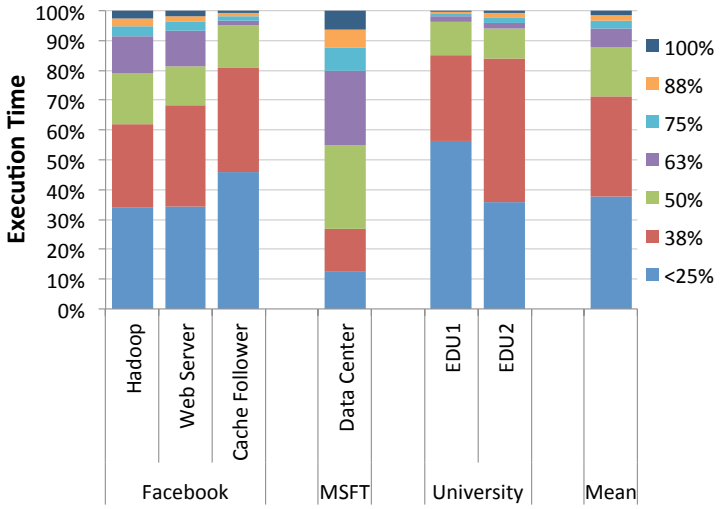


Figure 2.8. Breakdown of partial network activation. The legend indicates the fraction of the network links that are on.

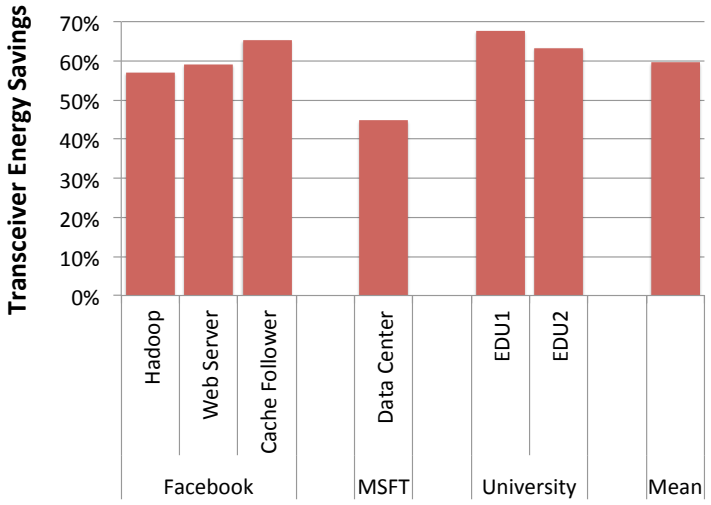


Figure 2.9. LC⚡DC transceiver energy savings.

the most challenges, but LC⚡DC still manages to turn off half of the network half the time.

As a result, LC⚡DC saves on average 60% of the optical transceivers' energy (Figure 2.9). The lower energy savings relative to the time breakdown (Figure 2.8) are due to

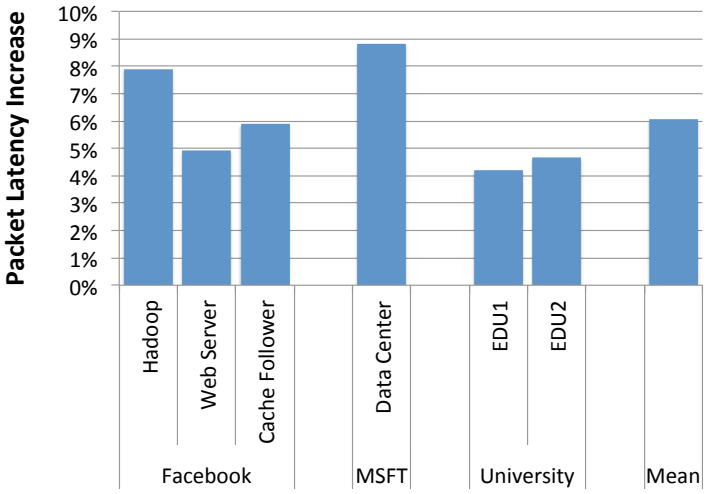
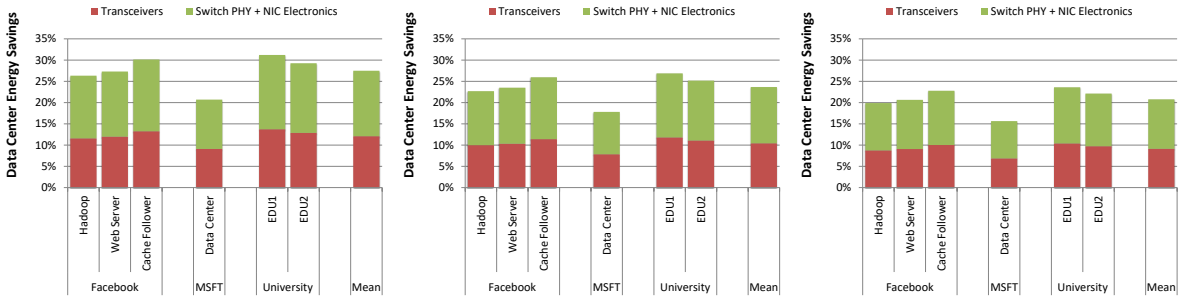


Figure 2.10. Impact of LC⚡DC on packet latency.

the fact that while a transceiver is in the process of turning on or off, the link is still considered deactivated, but we conservatively charge the full transceiver power consumption to the network. The energy savings come at the cost of 6% higher average packet latency (Figure 2.10) as queueing in network buffers may slightly increase, which we argue can be largely absorbed by the application layer.

Following the analysis in Section 2.2 we estimate the data center energy savings of LC⚡DC in Figure 2.11. We consider the data center to be at an average utilization of 30%. Additionally we calculate the energy savings of LC⚡DC in cloud servers, which are shown to typically have higher utilizations at 40%-70% [124]. Figures 2.11b and 2.11c show energy savings at 50% and 70% utilization, respectively. For data centers at 30% utilization, assuming the data center servers are optimized for energy while maintaining high performance, LC⚡DC can save 12% of the data center energy by deactivating links when they are not needed. It is also possible to extend LC⚡DC to deactivate or put



(a) Cloud servers at 30% utilization.

(b) Cloud servers at 50% utilization.

(c) Cloud servers at 70% utilization.

Figure 2.11. Impact of LC⚡DC on overall data center energy.

into a sleep mode the switch PHY chips and the server NIC electronics whenever the corresponding link is deactivated. With that extension, LC⚡DC 's data center energy savings can reach up to 27% on average, even after accounting for the CMOS scaling of the switch PHY and NIC card electronics. While we do not directly explore turning off the switch PHY and NIC card electronics, it is a promising future direction for this work. For cloud servers with utilizations of 50% and 70%, LC⚡DC can respectively save 10% and 9% of total data center energy by just turning off links. By deactivating the switch PHY and NIC electronics, LC⚡DC can respectively save 23% and 21% of total data center energy.

2.7. Conclusions

As technological innovations steadily reduce the power consumption of data center components, network power consumption becomes increasingly prominent. We argue that it is time to start optimizing the network designs not only for latency, bandwidth and cost, but also for power and energy efficiency. We present LC⚡DC , a data center network system architecture in which the operating system, the switch, and the optical components

are co-designed to achieve energy proportionality. We demonstrate LC⚡DC 's feasibility at all levels (electrical circuitry, optical devices, node-level architecture, and switch architecture). Our results show that LC⚡DC saves on average 60% optical transceiver power (68% max). We also show that it can save 9%–12% of total data center energy by turning off links and 21%–27$ energy by also turning off the switch PHY and NIC electronics.

CHAPTER 3

Pho$: A Case for Shared Optical Cache Hierarchies

3.1. Introduction

It has been nearly 25 years since the performance gap between CPUs and main memory, or the "Memory Wall", was identified as the main obstacle in increasing the performance of computer systems [179]. To mitigate the memory wall, stemming from the high latency of electronic memories and the limited bandwidth of electronic off-chip memory interconnects, modern chip multiprocessors (CMPs) have resorted to deep cache hierarchies. However, on-chip caches can occupy as much as 40% of the die area [23] and 32% of the processor's power [148]. As a result, multiple efforts on the device and architecture levels have focused on mitigating these issues, including caches based on STT-RAM [178, 69, 161, 154, 84], Phase Change Memory [107, 176, 85], and 3D-die stacking [160, 103, 106, 104, 83, 17, 180].

Alternatively, optical interconnects and nanophotonic technologies have emerged as promising yet underdeveloped solutions to tackle the disparity between processor and memory speeds. Today, we appear to have all the ingredients necessary to design novel optical cache architectures supported by optical interconnects. Optical Networks on Chip (NoCs) demonstrate higher bandwidth and energy efficiency than the traditional electronic NoCs used in CMPs [170, 169, 127, 126, 51, 45, 47, 49, 72, 98, 116, 173]. Optically connected memory (OCM) raises the possibility to switch much of the data

transports between the processor and DRAM chips to the optical domain [15, 71, 70]. Silicon photonic IC optical interfaces have been integrated with an electronic IC using a 65 nm DRAM, providing a fast 4× wavelength 10 Gbit/s optical interface for HPC applications [27]. Optical Flip-Flops (FFs) in photonic crystal nanocavities (PhC) [123, 4] can form the building blocks of all-optical memory cells [5], which have demonstrated both speed and energy benefits over their electronic counterparts by boasting read/write speeds up to 40 Gbps [123, 166]. Several optical Flip-Flops (FFs) have been developed with materials like coupled semiconductor optical amplifiers (SOAs) [165], III-V-on-SOI microdisk lasers [101], polarization bistable vertical cavity surface emitting lasers (VCSELs) [146], coupled semiconductor optical amplifier-based Mach-Zehnder Interferometers (SOA-MZIs) [102], and photonic crystal nanocavities (PhC) [123, 4]. The confluence of these technologies seems to be all we need to develop an optical cache hierarchy.

However, the application of an optical cache is not a simple plug-and-play replacement of its conventional electronic counterpart. While prior works [108, 109] have tried to explore this topic, the proposed designs are infeasible for capacities larger than a few kB due to unrealistically high power consumption, and do not consider the challenges of interconnecting the electronic and optical domains. They also lack analysis for whole system power and energy, and their performance is compared against unrealistically weak baselines. In this chapter, we address the issues that arise with the introduction of such optical cache devices, and bridge the gap between device- and architecture-level designs. More specifically, our contributions are:

- For the first time to our knowledge, we make optical caches practical. We employ a cascaded two-level row decoder to reduce laser power, active rather than passive

components to reduce off-ring optical losses, and use a new technology for the optical bit cells that dramatically lowers the static power consumption.

- We propose Pho$[1], an opto-electronic memory hierarchy for CMPs. Pho$ replaces all the core-private levels of a conventional electronic cache hierarchy with a single-level shared L1 optical cache (split I/D) that utilizes PhC-based optical memory cells [123] operating at 20 GHz. Pho$ enables for the first time L1 caches to be **high capacity** (multiple MB), **fast** (2-processor-cycle access time at 3.2 GHz), and **shared** (obviating cache coherence).
- We propose Pho$Net, a novel hybrid MWSR/R-SWMR optical NoC to connect processor cores with optical cache banks in Pho$. Pho$Net disaggregates the request/reply paths to reduce laser power, and co-arbitrates both subnets simultaneously through a novel arbitration protocol. The optical network extends to the electronic LLC and main memory.
- We perform comprehensive modeling and evaluation of Pho$'s performance, power, and energy characteristics. Pho$ is up to 3.89× faster (1.41× on average) over a traditional electronic cache hierarchy, while achieving up to 90% lower energy-delay product (31% on average). Under realistic assumptions, the Pho$Net optical NoC achieves up to 70% power savings compared to directly applying previously available optical NoC architectures.

In the following section, we provide the necessary background on optical NoCs and the optical cache technologies on which Pho$ is based.

[1]Pronounced "*phos*", a word play between the Greek "$\phi\omega\varsigma$", meaning light, the word "photonic", and "*$*", the symbol often used to denote a cache.

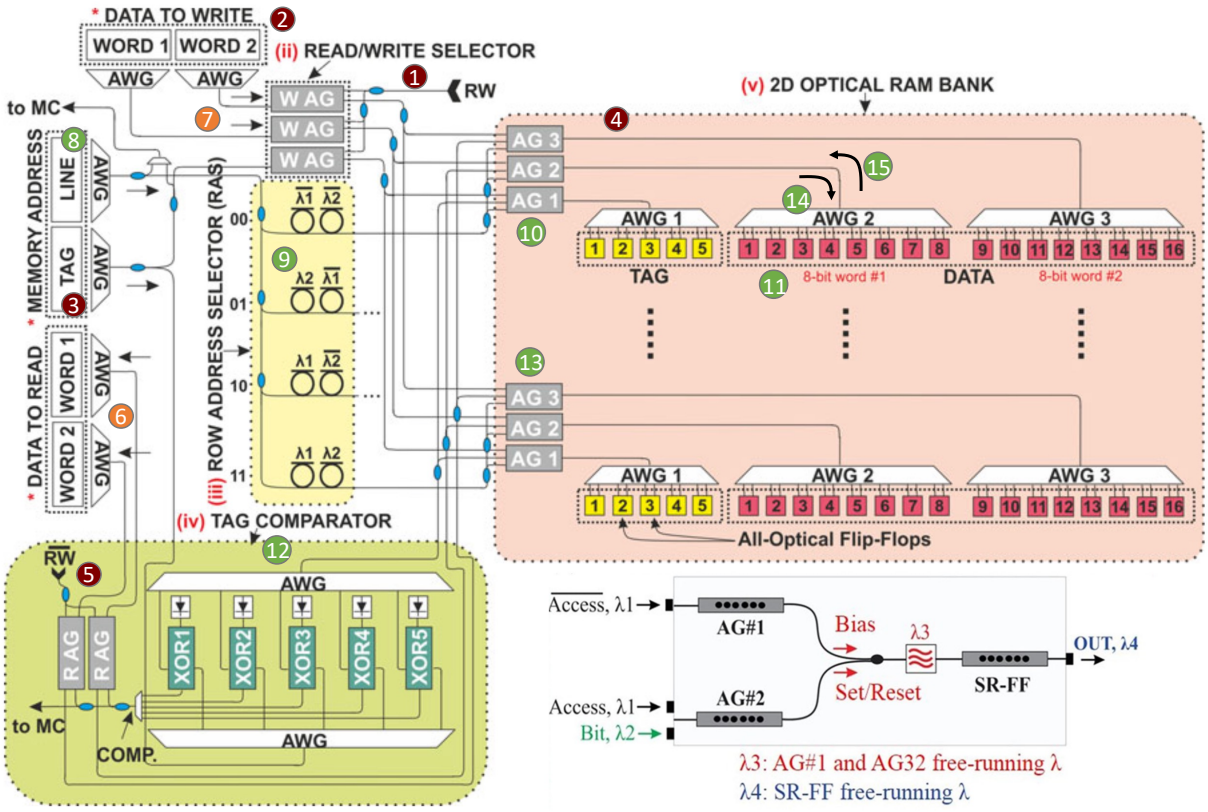


Figure 3.1. 8 B optical cache [108] and PhC nanocavity optical SRAM cell [4].

3.2. Background

3.2.1. Optical Cache Operation

Figure 3.1 shows the layout of an 8 B direct-mapped optical cache with a 2 B cache line, 2-bit index, and 5-bit tags [108]. Each bit is encoded with two wavelengths.

Read/write operations are controlled by the RW and $\overline{RW}$ signals. During a write to the cache, a RW signal representing a logical "0" activates the Write Access Gates (WAG) ❶ and allows the incoming *data* bits ❷, the *tag* bits ❸, and their complements $\overline{data}$ and $\overline{tag}$ to enter the optical RAM bank ❹. At the same time, $\overline{RW}$ represents a

Table 3.1. Index-RAS truth table.

λ_1	$\overline{\lambda_1}$	λ_2	$\overline{\lambda_2}$	Row 00	Row 01	Row 10	Row 11
0	0	1	1	**0**	$\overline{\lambda_2}$	$\overline{\lambda_1}$	$\overline{\lambda_1\lambda_2}$
0	1	1	0	λ_2	**0**	$\lambda_2\overline{\lambda_1}$	$\overline{\lambda_1}$
1	0	0	1	λ_1	$\lambda_1\overline{\lambda_2}$	**0**	$\overline{\lambda_2}$
1	1	0	0	$\lambda_1\lambda_2$	λ_1	λ_2	**0**

logical "1", blocking the Read Access Gates (RAG) (5) and preventing a read operation. In the case of a read, the RW and $\overline{RW}$ signals are set to logical "1" and "0", respectively. This allows the data from the RAM bank to propagate onto the data reply channel (6) and blocks the WAG to prevent any data from being overwritten (7).

The cache line to read or write is designated by the incoming $index$ (8) and $\overline{index}$ bits which drive the passive Row Address Selector (RAS) (9). In Figure 3.1's example, the RAS consists of 4 rows of two micro-rings (MRs) each. Each MR is tuned to a specific wavelength such that a pair of wavelengths λ_i and $\overline{\lambda_i}$ encode the logical "1" and "0" of the i-th bit of the $index$. The 2-bit $index$ is encoded with 4 wavelengths: λ_1, $\overline{\lambda_1}$, λ_2, and $\overline{\lambda_2}$. As a result, only one of the four rows will have a logical "0" after the $index$ bits pass through the RAS. For example, when the $index$ bits are "10", meaning to select the third line, the corresponding logical values for the wavelengths are: $\lambda_1 = 1$, $\overline{\lambda_1} = 0$, $\lambda_2 = 0$, and $\overline{\lambda_2} = 1$. Only the third set of MRs is capable of absorbing both λ_1 and $\overline{\lambda_2}$, creating a logical "0". For each cache line, two access gates (AGs) (10) are responsible for the data words and a third AG is responsible for the tag bits. The AGs of the selected cache line now have a control signal of "0", which allows either incoming data-to-write and tags to pass through to the optical Flip-Flops (FFs) for writing (11), or the contents of the FFs pass through to the tag comparator for reading (12). Table 3.1 shows the possible

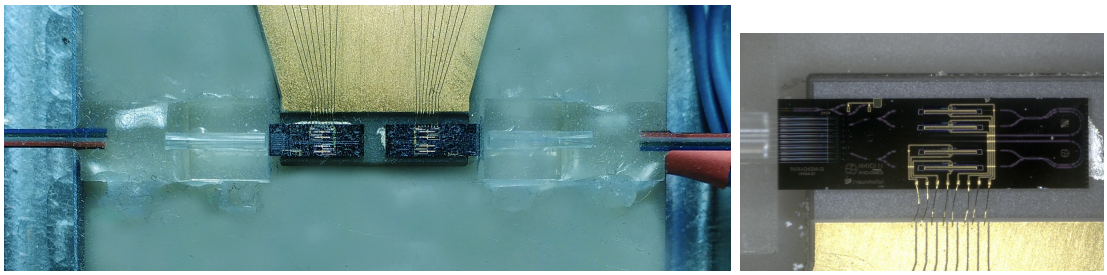


Figure 3.2. (a) Packaged optical memory. (b) Monolithic InP Flip-Flop.

combinations of the 2-bit *index* and the corresponding rows they activate. All other lines will have some wavelengths still propagating to their corresponding AGs, not activating them and blocking any data (13).

When the data and tag bits enter the optical RAM bank and propagate through the AGs in the row denoted by the index (10), the wavelengths are distributed to their corresponding optical FFs through Arrayed Waveguide Gratings (AWGs) (14). AWGs act as optical demultiplexers that retrieve individual wavelengths from Dense Wavelength Division Multiplexing (DWDM) optical channels [167]. Each pair of wavelengths λ_i and $\overline{\lambda_i}$ drive the optical FF at the i-th bit in each 8-bit optical word. For a read, the AWGs multiplex the bits from the FFs into a single waveguide in the reverse direction (15).

3.2.2. Optical SRAM Cells

We experimentally verified and characterized in our lab integrated photonic RAMs and optical FFs (Figure 3.2) which adopt the cross-coupled circuit-layout RAM cell architecture presented in Figure 3.1, and use technologies of optical gain elements integrated hybridly with InP PhC-on-SOI [4].

The optical SRAM cell can be built using emerging PhC technologies to reap the speed, energy, and footprint benefits they offer, as previously validated using a hybrid

InP/Si PhC laser [4], and InGaAsP Buried Heterostructure cavities [123, 96]. Figure 3.1 shows a possible principle of operation of an optical SRAM cell when energy efficient and compact hybrid PhC lasers are employed [4]. Two PhC-based nanocavity lasers act as AGs and another laser acts as the optical FF. The AGs are controlled by a pair of *Access* and $\overline{Access}$ signals. When *Access* is a logical 1, AG#1 outputs the Bias pulse, and AG#2's output is suppressed. Thus, only the Bias pulse enters the FF, enabling the FF to output its previous stored value, successfully reading the content of the FF. When the *Access* is a logical 0, AG#1's output is suppressed, and AG#2 outputs the Bit signal. The value of the Bit signal facilitates a Set/Reset operation, thereby performing a write.

During a read operation, the data and tag bits of the selected cache line pass through the AGs and propagate to the tag comparator. The tag bits are demultiplexed through an AWG and each bit is XOR-ed with the corresponding bit of the tag array that the processor sent. The results of all XOR gates are then multiplexed to form a COMP signal, which is 0 if the tags match and 1 otherwise. The COMP signal is then used to drive the RAGs along with the $\overline{RW}$ signal, and allow the data to be replied to the processor only if there is a read operation and the tags match, i.e., a cache hit.

Both types of optical SRAM cells demonstrate extremely fast switching speeds. PhC nanolasers [4] exhibit 50 ps switching latency for fast memory operations. Alternatively, the InGaAsP/InP buried heterostructure PhC [123] has a 44 ps switch-on latency but requires 7 ns to switch off.

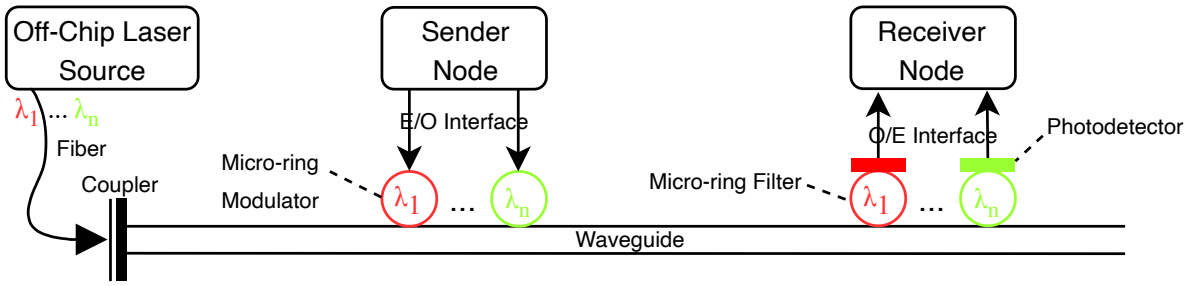


Figure 3.3. Basic nanophotonic components.

3.2.3. Optical Network-on-Chip

Recent breakthroughs in silicon photonics have propelled researchers to consider optical interconnects for on-chip communications. Optical NoCs provide low latency due to the fast propagation of light in silicon waveguides, and high bandwidth data transmission through dense wavelength-division multiplexing (DWDM), making them strong candidates to replace or partially replace traditional electronic NoCs. We first review the components for building optical NoCs, then briefly discuss two types of existing optical network interconnects.

3.2.3.1. Nanophotonic Building Blocks. Figure 3.3 shows the optical components to perform data transmission between a sender and a receiver on a chip. An off-chip laser source emits light with wavelengths $\lambda_1 \ldots \lambda_n$, which travels through an optical fiber and is brought onto the chip through a coupler. A single waveguide is capable of carrying multiple wavelengths in parallel by employing DWDM. The sender converts electrical signals into optical signals of specific wavelengths and modulates them onto the waveguide through micro-ring resonant modulators. MRs are placed next to waveguides and are tuned to modulate a specific wavelength by controlling their radius and temperature.

The modulated wavelengths travel along the waveguide until they arrive at the receiver. MRs are also used on the receiver's side as filters to extract individual wavelengths from the waveguide. Then each wavelength is directed to a photodetector to convert the signal back to electrical currents, which subsequently go through amplifiers to be strong enough to drive electrical logic circuits. A DWDM density of n wavelength requires n modulator/filter pairs.

3.2.3.2. Corona. Corona [170] implements an optical crossbar to interconnect 64 four-core clusters. The crossbar is formed by 64 Multiple-Writer Single-Reader (MWSR) buses laid out in a serpentine fashion to connect all clusters. For each MWSR bus, 63 of the total 64 nodes can transmit on the waveguide while the remaining one can receive from all others. An arbitration protocol is needed as multiple source nodes cannot transfer data to the same destination simultaneously. Token-based optical arbitration protocols [169] employ additional waveguides, in which receivers inject optical tokens for senders to acquire. A node can only transmit data to a destination when it has consumed the token on the waveguide corresponding to that destination node, meanwhile blocking other nodes from writing to the data bus. When the sender finishes transmitting data, it injects a new token onto the arbitration bus.

3.2.3.3. Firefly. Firefly [127] introduces an opto-electronic NoC using reservation-assisted Single-Writer Multiple-Reader optical crossbars (R-SWMR). A single R-SWMR bus involves one sender and multiple receivers. Arbitration, or reservation, is performed by the sender broadcasting a small optical reservation packet to all receivers. Upon receiving this packet, all nodes except the destination receiver turn off their corresponding receiving MRs, allowing only the destination node to receive the data from the sender. This saves

power compared to a broadcast-based SWMR bus because all MRs on the optical path between the sender and receiver are off and induce minimal optical losses. An R-SWMR optical crossbar with a total of N nodes has N data channels, each with a data width of w bits, and N reservation channels of $\log N$ bits each.

3.3. The Pho$ Architecture

The optical cache prototype presented in Section 3.2 achieves very low latency. The optical SRAM cells can perform reads and writes in under 50 ps, and the outside decoding processing time is 100 ps, resulting in 150 ps cache read and write latencies. As long as the core-to-cache optical bus takes no more than 50 ps, such an optical cache can perform single-cycle cache accesses for core frequencies up to 5 GHz. However, while the InP/Si PhC laser-based optical SRAM cells have fast on/off switching speeds, each cell requires a pump power of 103.5 µW for storage operations [4]. Considering the number of components needed for a reasonably sized cache, static power quickly reaches hundreds of Watts, which is unrealistic. Thus, prior designs in this space [108, 109] are not implementable above 8 kB despite only taking up a physical footprint of 7.89 mm^2.

To avoid the additional pump power needed for biasing, Pho$ instead utilizes the InGaAsP-based optical SRAM cells demonstrated by Nozaki *et al.* [123]. These cells require a static power of only 30 nW, and their switch-on latency of 44 ps is on par with the 50 ps latency of the InP/Si PhC laser, allowing cache reads to still be completed within one cycle at 5 GHz. Cache writes are slow at 7 ns, but this can be mostly mitigated by memory-level parallelism (MLP) and a modern core's store queue. MLP allows for multiple concurrent memory requests, and store queues allow arithmetic operations and

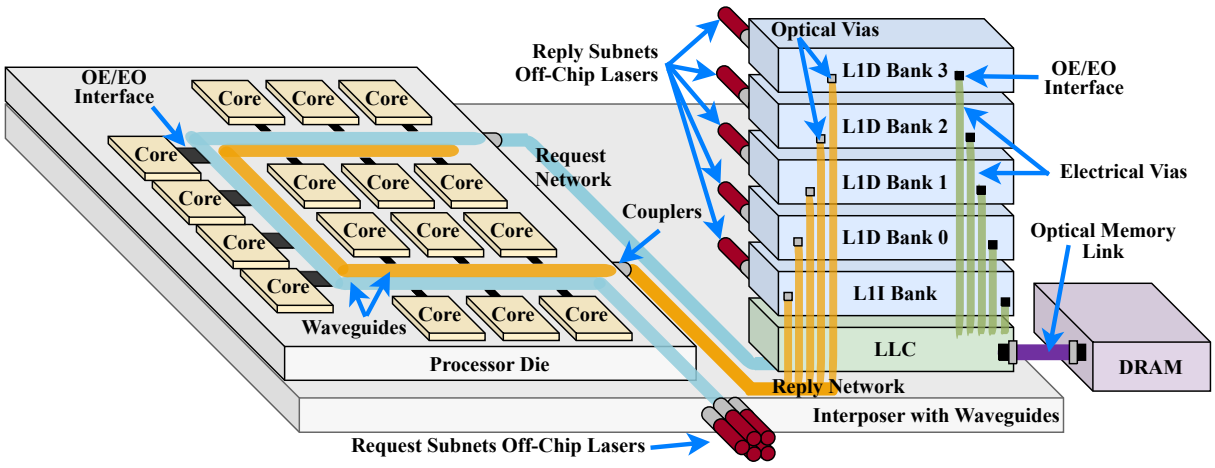


Figure 3.4. Pho$Net optical network topology.

loads to bypass pending older writes. Thus, both MLP and store queues allow a core to overlap long write latencies with other work.

We propose Pho$, an opto-electronic cache hierarchy architecture that replaces all the electronic L1D, L1I, and L2 caches in a traditional CMP with a single, shared, high-capacity all-optical cache. Due to its high capacity and disaggregation from the cores, it is natural for the optical cache to be shared among all cores. We envision a shared optical L1D that employs 4 banks to provide high capacity and parallelism, and a shared optical L1I with one bank. The optical cache banks are fabricated on separate optical dies, while the processor cores remain on their original electronic die. The cores and optical caches are 2.5D-integrated on the same package and interconnected by an optical NoC, which handles arbitration and data transmission between them.

3.3.1. Pho$Net Network Topology

Figure 3.4 shows a high-level view of Pho$'s optical network topology, Pho$Net. The electronic processor die on the left houses the cores (16 cores in a 4×4 mesh layout)

and sits atop an interposer with photonic waveguides. The dies on the right are 3D-stacked. The L1D and L1I banks are on optical dies, while the Last Level Cache (LLC) is a traditional electronic cache with its own die. Each optical cache bank has one input and one output port. We model single-ported cache banks throughout the manuscript, with the exception of the power investigation shown in Figure 3.11 where we analyze the impact of multi-ported caches on power consumption.

Communication between the cores and caches is entirely in the optical domain. Two sets of optical waveguides are laid between the processor and L1 cache dies. Each waveguide line in the figure is abstracted to represent multiple sub-networks, each comprising a bundle of waveguides with DWDM. The blue line depicts the subnets that carry requests from the cores to the cache banks (one subnet per bank). Within each request subnet, the cores are the writers and only one of the optical cache banks is the reader. Thus, each request subnet forms a Multiple-Writer Single-Reader (MWSR) crossbar [170] and uses token-based arbitration [169]. For each individual MWSR link, the cores are the writers and one of the optical cache banks is the reader. The orange line represents the reply subnets used by the cache banks to send data to the cores. For each reply subnet, one of the cache banks is the writer and the cores are the readers. Thus, the reply subnets are designed as Reservation-assisted Single-Writer Multiple-Reader (R-SWMR) crossbars [127]. Apart from waveguides for carrying data packets, additional waveguides are needed for both the request networks' token arbitration channels and the reply networks' reservation channels. In essence, Pho$Net is a hybrid MWSR/R-SWMR optical network.

For a 16-core processor with 5 optical cache banks (as in Figure 3.4), and assuming single-port cache banks, there are in total 5 hybrid subnets, each comprising an MWSR

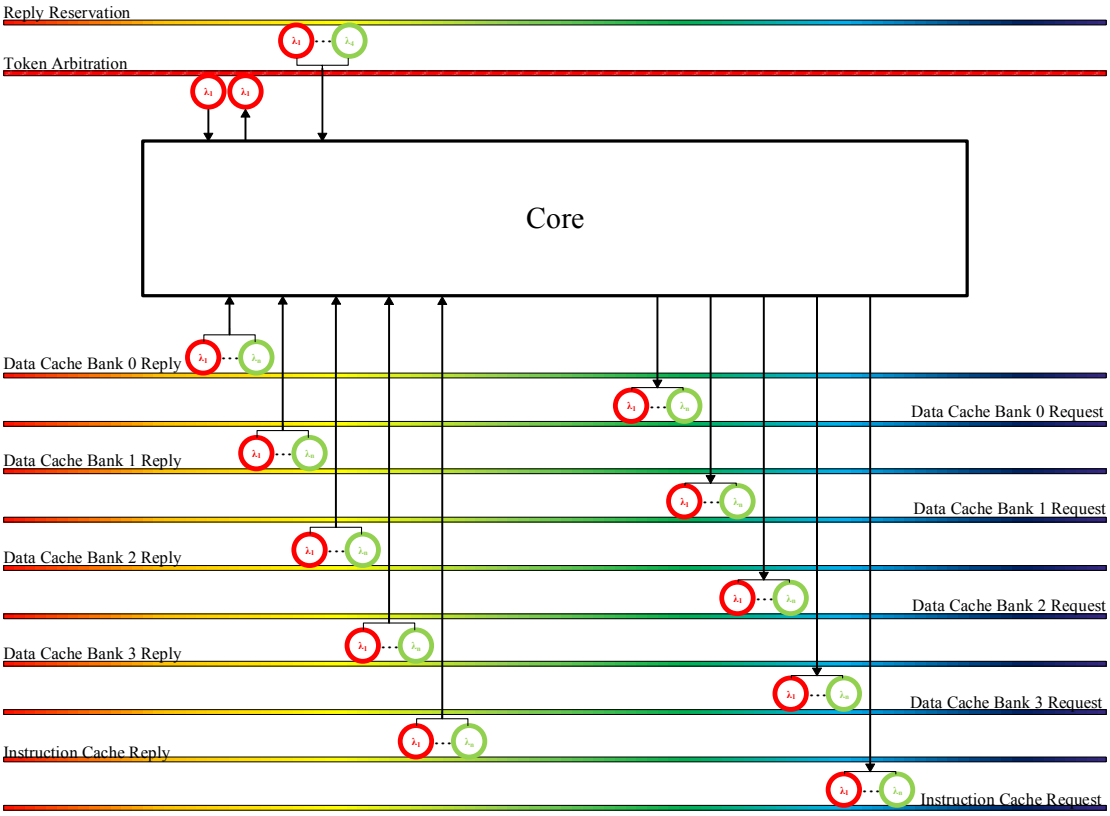


Figure 3.5. Core optical connections.

request and an R-SWMR reply crossbar with arbitration and reservation channels, respectively. If multi-port optical caches become possible in the future, we can further break the sub-networks to include only a subset of the processor cores. Note that with 16 cores, the reservation channel needs 4 wavelengths to represent the core ID.

The black squares in Figure 3.4 represent the Electrical-Optical (EO) and Optical-Electrical (OE) conversion interfaces for the cores to interact with the optical network, including modulators, filters, detectors, etc. More details are shown in Figure 3.5. All cores have a pair of send/receive interfaces to interact with each cache bank and its

corresponding set of waveguides, thus any core can send and receive on any sub-network. For diagram simplicity, not all connections between waveguides and the interfaces are shown.

Core-private caches, as employed by traditional multicores, require core-to-core communication to maintain coherence, which in turn requires full-blown MWSR or R-SWMR crossbars with all-to-all connectivity. By employing an L1 cache that is shared among all cores, Pho$ physically decouples the cores from the caches and removes the need for cache coherency and inter-core traffic. Thus, it is no longer necessary to build physical links between cores. It suffices to implement separate networks for carrying either requests or reply packets directly to and from caches, and optimize each for their purpose. The hybrid Pho$Net network capitalizes on this observation to shrink the network by avoiding full connectivity among all nodes, saving power, area, and cost.

The request and reply subnets are powered by separate off-chip lasers to minimize laser power (Section 3.3.4). Finally, the LLC can be connected to the DRAM through an optical interconnect [15] for low latency, high bandwidth DRAM accesses. The adoption of DWDM enables OCM to send out the entire cache line in one burst, which decreases the time needed to transmit and receive data on the memory interconnect. This, along with a higher propagation speed, reduces memory access latencies and increases bandwidth.

3.3.2. Pho$Net Arbitration Protocol

For each cache bank, all cores on the same request (or reply) subnet share the same channel, thus it is important to ensure that requests from (or replies to) different cores

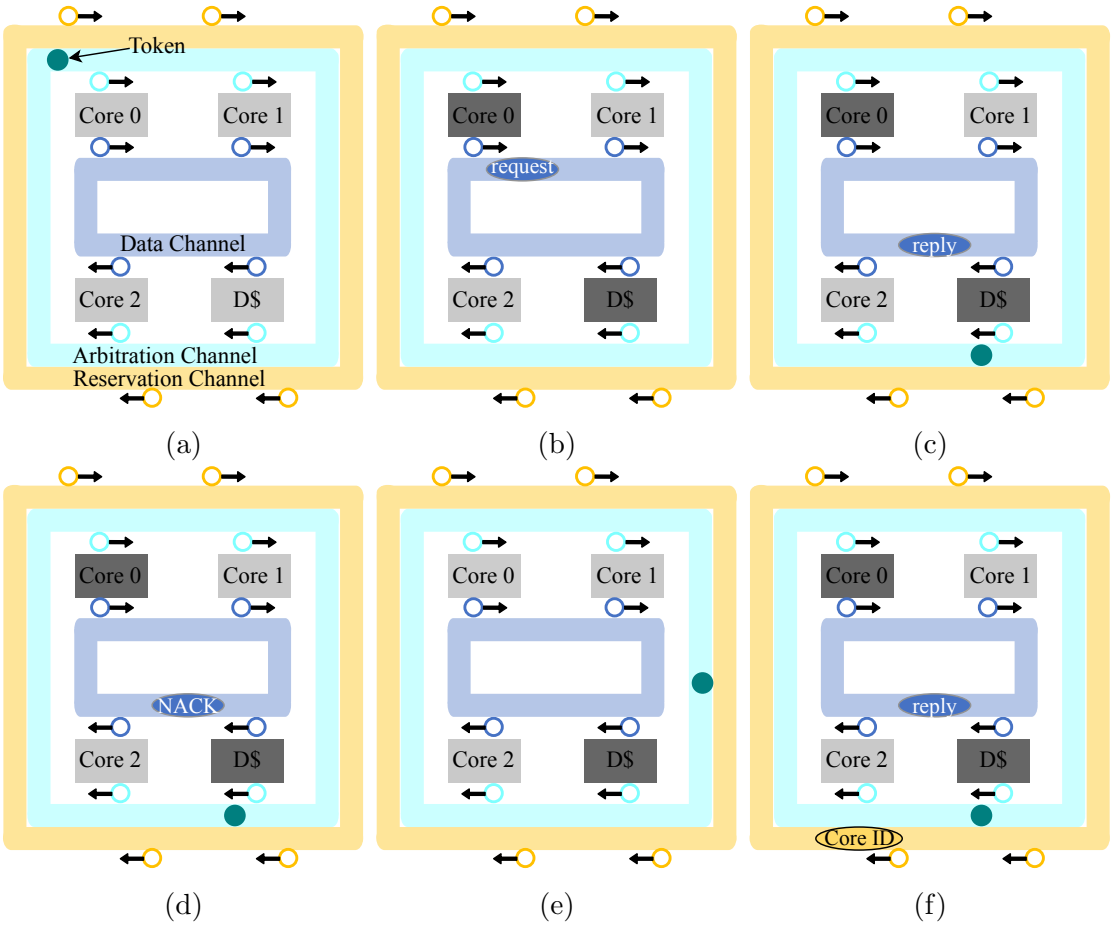


Figure 3.6. Arbitration protocol. (a) Token circles the arbitration channel waiting to be grabbed. (b) Core 0 grabs the token and sends a request packet on the data channel. (c) Cache hit: reply packet sent, followed by a new token. (d) Cache miss: NACK sent, followed by a new token. (e) Cache has data following the miss, tries to grab token first. (f) Cache notifies core 0 with reservation channel, sends reply packet followed by new token.

do not conflict. As Pho$Net is half MWSR and half R-SWMR (Section 3.3.1), it requires a new way to arbitrate packets.

Arbitration in Pho$Net is achieved through a protocol similar to optical token channel arbitration [169]. A single optical token circulates through each bank's request-reply subnets. When a core needs to send a cache request, it turns on its MRs on the arbitration

channel to try to consume the token. If the token reaches the core's receiving MRs the core absorbs the token, acquiring the exclusive access to the request channel. The core then is able to send request packets to the optical cache. After sending a request, the core turns on its receiving MRs and starts listening on the reply channel without the need for further arbitration.

The cache, upon receiving the request, processes it in the optical domain. Upon an L1 hit, the cache injects the data to the reply network followed by a new token. The reservation channel is not used at this moment because all cores except the original requester have their receiving MRs turned off (they are not expecting a data packet from the cache). If a cache miss is detected, the optical cache forwards the request to the electronic LLC after an OE conversion. It also sends out a Negative-Acknowledgement (NACK) packet on the reply waveguide. The requester core, who is still listening on the reply channel gets the NACK packet, realizes there is a cache miss, and turns off its receiving MRs. This mechanism ensures that during cache hits, the cache does not need to arbitrate for the reply channel as the requester core and cache have exclusive permissions to transmit on the request and reply channels, respectively. Regardless of a cache hit or miss, a new token is injected in the arbitration channel. This new token can be grabbed by any core who wants to send a cache request. The requester, after consuming the reply packet (be it data from the cache or a NACK) also turns on its MRs on the arbitration channel so that it can grab the new token. In both cases, the total latency for the read and reply packets should be constant as there is always exactly one round trip made from core to the cache and back to the core again: $t_{total} = t_{arbitration} + t_{request} + t_{reply} + t_{SERDES} =$

$t_{arbitration} + t_{RTT} + t_{SERDES}$ (our design does not need SERDES, but we include it in the equation for completeness).

In the case of an L1 cache miss, the electronic LLC eventually responds with the data requested by the optical L1. The L1 first tries to arbitrate for the data channel like any other core would; this ensures that no other data packets are transferring on the data channel and no additional request can be made to the bank. When it has successfully grabbed the channel, it first broadcasts on the reservation channel so that the core that sent the original request for this cache access can turn on its receiving MRs. Then the cache transmits the reply data on the data channel, followed by a new token on the arbitration channel that all cores can grab to start a new cache request. The data packet needs to trail the reservation packet by a fixed delay to allow the receiving core enough time to turn on its data channel MRs. It is also important to note that the core ID needs to be included in the original request packet so that in case of a cache miss, the L1 knows which core to send the reply to once it gets the data from the LLC.

Figure 3.6 shows an example arbitration in a simplified 3-core 1-cache-bank setup. For simplicity, we have also combined the request and reply data channels in the figure.

3.3.3. Pho$ Optical Cache Architecture

This section describes the architecture design of the 1 MB optical cache banks employed by Pho$, as well as the components' optical losses for calculating the optical power budget. In a nutshell, Pho$ employs a cascaded two-level row decoder to reduce laser power, active rather than passive components to reduce off-ring optical losses, and uses PhC for the optical SRAM bit cells to dramatically lower the static power consumption. These design

(a) Row decoder (b) Column decoder

Figure 3.7. Optical cache peripheral circuit.

innovations allow Pho$ to be implementable within a reasonable power budget even for multi-MB cache capacities, in contrast to prior designs in this space [108, 109] that are not implementable above 8 kB.

Assuming a 64 B cache line, each of Pho$'s five 1 MB direct-mapped cache banks has 16384 lines. Row decoding with an MR-based matrix, as in prior work [108], is impractical: the number of MRs needed for each line increases as the matrix scales up, consuming inordinate amounts of power. Instead, Pho$ uses a two-level cascaded row decoding process (Figure 3.7a). The first-stage demultiplexing uses an active 9-to-512 tree global row selector, implemented with PhC nanocavity-based resonant switches [122], which activates only one of the 512 5-to-32 passive MR-based row decoders in the second stage. The second-stage row decoder then selects one line to activate, allowing a read/write operation to perform on the correct cache line. In this way, we build a 16384-line row selector with only 5 MRs per line instead of 14, drastically lowering laser power.

Active optical devices such as an PhC active tree are estimated to have optical losses of 0 dB. This is because in order to function, the active components within the device also provide some small amplification, sufficient to compensate its own losses. As a result, we can lower the laser power required for light to reach the optical FFs.

For the column decoding optical circuit (Figure 3.7b), we use 8 1-to-128 AWGs to demultiplex the wavelengths in the incoming light into their respective optical FFs. Each 1-to-128 AWG serves 64 bits, with 2 complementary channels per FF, so a total of 8 AWG-based column decoders are needed for a 64 B cache line. For each AWG, an AG controls the direction of data when switching between writing and reading the FFs. The AGs are controlled by 8 WAGs acting as read/write selectors. Data are fed into the reply waveguides through 8 RAGs (Section 3.2).

For a 48-bit physical address with 14 bits used for the index, 6 bits used for the offset, and 2 bits used for bank selection, the tag field requires 26 bits (28 for L1I). Because our optical cache is direct-mapped, one 1-to-128 AWG is enough to demultiplex the incoming tag signal into separate wavelengths. As with the data cells in the above paragraph, an extra AG per line and one global WAG are needed to control the reading and writing of the tag cells. The tag comparator is built with 26 XOR gates and two 1×26 AWGs. The XOR gates and AGs are also implemented using active PhC resonant switches. The data and tag output of all 16384 cache lines need to be multiplexed onto one waveguide before the tag bits are passed into the tag comparator and the data bits into the RAGs. This can be done with active PhC trees acting as multiplexers.

Table 3.2 summarizes the component counts and the optical loss parameters for a 1 MB optical cache bank in Pho$. They will be used in calculating the laser powers required for the core and cache optical networks.

Table 3.2. Optical cache components. See Section 3.3.3 for an explanation of 0 dB losses.

Component	Count	Passive/ Active	Optical Loss
Tag Comparator AWGs	2 AWGs	Passive	3 dB
XOR gates for Tag	52 PhCs	Active	0 dB
WAG	9 PhCs	Active	0 dB
RAG	8 PhCs	Active	0 dB
AG	147456 PhCs	Active	0 dB
Global Row Selector	1024 PhCs	Active	0 dB
Row Decoder	81920 MRs	Passive	#MRs × filter drop
AWG Column Decoder	147456 AWGs	Passive	3 dB
Optical RAM Bank	8814592 PhCs	Active	0 dB
Total PhCs: 8963141	Total MRs: 81920		
Total AWGs: 147458	Min power at optical FF: −14 dBm		

3.3.4. Laser Power Sources and Optical NoC Parameters

The request subnet and the optical cells and reply subnet are powered by separate laser sources. The laser used to power the request subnet also powers the row decoders, column decoders, read/write selectors, and AGs before the optical FFs, because additional lasers along the path can overwrite any data already traveling on the waveguide. The token arbitration and reservation channels are also powered by the same laser. On the other hand, the FFs in the optical cache cells need a continuous power source to store data using photons, and the same laser can be used to power the tag comparators as well as the reply network. Table 3.3 details the specific components of the optical system that each of the two lasers is in charge of powering. We call them the "core network" and "cache network", respectively.

Pho$ uses off-chip lasers because on-chip lasers may generate a lot of power and heat. The lasers are brought onto the chip using optical fibers and couplers, and then the light is distributed onto waveguides using splitters. There is only one splitter per

Table 3.3. Optical power source responsibilities.

Laser Source	Optical NoC Components	Optical Cache Components
Core network	Request network Token arbitration channel Reservation channel	Read/write selector (WAGs) Row decoder, Column decoder AGs, RAGs
Cache network	Reply network	Optical FFs, RAGS Tag comparator

Table 3.4. Nanophotonic parameters for Pho$Net.

Component	Conservative	Aggressive	Component	Conservative	Aggressive
Waveguide	1 dB/cm	0.05 dB/cm	Waveguide bending	0.005 dB	0 dB
Coupler	2 dB	1 dB	Waveguide crossing	0.12 dB	0.05 dB
Nonlinearity	1 dB	1 dB	Photodetector	0.1 dB	0.1 dB
Ring-through	0.01 dB	0.001 dB	Modulator insertion	1 dB	0.001 dB
Filter drop	1.5 dB	0.5 dB	Detector sensitivity	−16 dBm	−28 dBm
Splitter	0.2 dB	0.1 dB	Laser Efficiency	30%	30%
Trimming	20 µW/ring	5 µW/ring	Modulation / Demod.	150 fJ/bit	20 fJ/bit

waveguide in our design. We have a total of 105 waveguides combined across all subnets for the single-port design. We consider a comprehensive range of parameters for optical components by grouping the parameters of several seminal optical NoC designs from recent years [94, 126, 51, 116, 98, 72, 50, 163, 173] into two groups, conservative and aggressive (Table 3.4), which represent the worst-case and best-case parameters among these works, respectively. Showing both conservative and aggressive parameters highlights the spread of possible values for each nanophotonic parameter. We expect each device to exhibit losses between these two values.

The waveguides of the data channel in the request network feed directly into the input ports of the optical cache and continue onto the optical FFs after passing various optical

Table 3.5. Pho$: Simulated system parameters.

	Baseline	Pho$	Pho$_OCM
Cores	16 cores, x86 ISA, 3.2 GHz, OoO, 4 wide dispatch/commit 224-entry ROB, 72-entry load queue, 56-entry store queue		
L1 ICache	electronic, private, 64 B line, 32 kB/core, 8-way, 4 cycles	optical, shared, 64 B line, 1 MB direct-mapped, 2-cycle read, 23-cycle write	
L1 DCache	electronic, private, 64 B line, 32 kB/core, 8-way, 4 cycles	optical, shared, 4 banks, 64 B line, 4 MB direct-mapped, 2-cycle read, 23-cycle write	
L2	electronic, private, 64 B line, 256 kB/core, 4-way, 14 cycles	N/A	
LLC	electronic, shared, non-inclusive, 64 B line, 22 MB, 11-way, 50 cycles		
Core-L1 Netw.	electronic, point-to-point	hybrid optical	
LLC Network	electronic, 4×4 mesh (NUCA)		
Memory	electrically connected, 49.37 ns	optically connected, 41.61 ns	

components, so we use the minimum power needed at optical FFs in Table 3.2 in the calculation of the laser power for those data waveguides. For all other waveguides, including those in the reply network data channel, token arbitration and reservation channels, the detector sensitivity is used instead.

3.4. Experimental Methodology

3.4.1. Performance Simulations

We evaluate Pho$ using the Sniper simulator [28, 29] running workloads from SPEC CPU2017 [25] (SPECspeed, ref inputs) and Parsec 3.0 [19] (simlarge inputs) benchmark suites. For CPU2017, we used Pinpoints [131] to collect representative regions. We compare our results with a baseline electronic multicore whose configuration is similar to a 16-core Intel Skylake [61, 42, 117, 139, 174, 175]. For Pho$, we perform experiments with a per-bank capacity of 1 MB. We model both a conventional DRAM for Pho$, as

well as an optically connected one (Pho$_OCM). DRAM bandwidth is modeled, but not the internals of DDR circuitry. Other non-cache and non-network related parameters are consistent across all the configurations. Table 3.5 summarizes the detailed configurations for our experiments. We run multi-threaded workloads in Parsec by pinning threads to individual cores. We also use methods introduced by Heirman *et al.* [78] to construct cycle stacks for better analysis of experiment results.

We perform physical measurements on a Dell PowerEdge R710 server [43] and estimate a 15 cm average distance between the LLC and DRAM DIMMs. We estimate the latency for DRAM accesses over that distance to be 46.7 ps/cm for light propagation in optical waveguides [31], and 50.4 ps/cm for electrical pulse propagation speed in electronic links[114].

3.4.2. Modeling Power, Energy, and Area

To get an insight into the optical NoC's power consumption, we compare our hybrid optical NoC, Pho$Net, against three network configurations. The first is a fully connected MWSR crossbar with 21 21-to-1 MWSR links (16 cores and 5 cache banks, a total of 21 nodes) with a token arbitration protocol. The second is a fully connected R-SWMR crossbar with 21 1-to-21 reservation-assisted SWMR links. Finally, we also compare against a "one channel" network where requests and replies share the waveguides as a single data channel, while all other characteristics are the same as in Pho$Net. For this comparison, we ignore the static power needed for optical FFs to operate as this depends on the number of cache components and not the network configuration.

Each data packet contains 512 bits of data, 42 bits of address, and 4 bits dedicated to the core ID (for the full R-SWMR configuration, 5 bits are used because there are 21 nodes in each R-SWMR link). Each bit is encoded with complementary wavelengths λ_i and $\overline{\lambda_i}$ to drive the optical cache circuits. We model a 64-λ DWDM. Because we stay in the optical domain for L1 cache accesses, we do not employ SERDES. All optical packets are sent in one burst.

To calculate the cores' die size, we use McPAT [99] to estimate the area of processor cores under the 14 nm technology node. Parameters are adapted from the International Technology Roadmap for Semiconductors 2015 Edition [37] and Fincacti [149]. We estimate the core die size to be 59 mm^2 and assume the distance between the core and cache dies is 1 mm. Using the scaling methodology presented by Maniotis *et al.* [108], we analyze the area footprint of a 1 MB direct-mapped optical cache by considering different component alignments and determine the optimal area to be 89 mm^2, where the distance traveled by the data and tag bits is 34.7 mm and that traveled by the index bits is 24.4 mm. We calculated the round-trip time for a cache access by considering EO/OE conversion latencies of 14.3 ps and 0.2 ps [31], the total distance traveled by the request and reply optical packets (175 ps roundtrip on the optical network), and the latency to access the optical cache bank itself (44 ps bitcell latency and 100 ps row/column decoder). This ensures 2-cycle access for up to 5 GHz. We calculated the area overhead for EO/EO interfaces of Pho$ to be 11.8% per core tile and 8% for optical dies by scaling numbers from Sun *et al.* [159]. To calculate the total area consumed by the NoC on the electronic die we use a waveguide pitch of 3 µm [183] and a micro-ring pitch of 5 µm [119]. The total

area overhead is 0.17 mm^2 if the waveguides are stacked vertically and 3.32 mm^2 if laid out on the same plane (0.3% and 5.7% of total die area, respectively).

For every network configuration, we also explore the possibility of multi-port caches. For an N-port optical cache, each port serves $\frac{16}{N}$ cores. For each sub-network, the original 16-to-1 request MWSR link becomes a $\frac{16}{N}$-to-1 link and the 1-to-16 reply R-SWMR link likewise. Individual links can become shorter and need fewer optical components, but more links are needed. In this chapter, we consider 1-port, 2-port and 4-port optical caches in our power analysis.

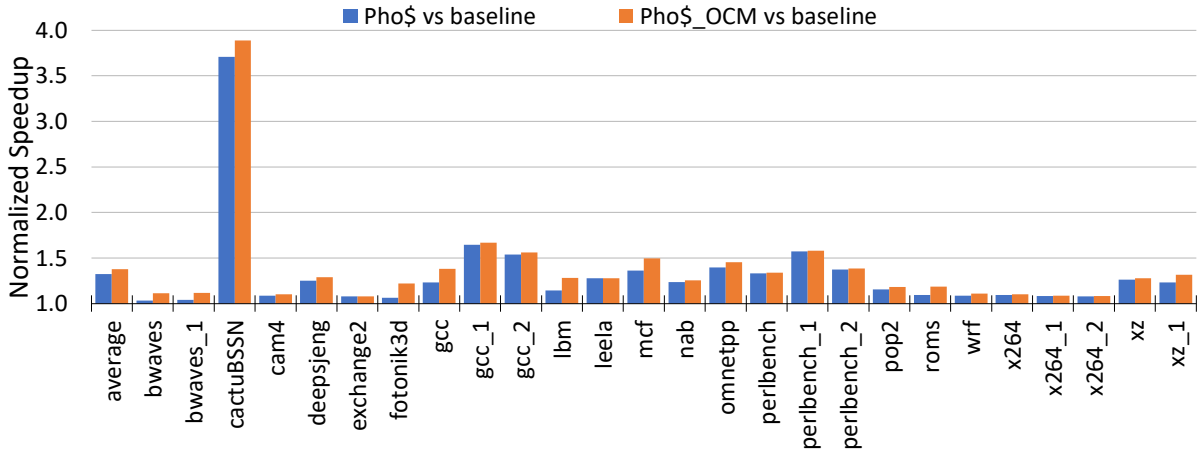
We estimate the energy consumption of cores, electronic caches, electronic on-chip interconnects, and DRAM using McPAT [99]. The energy consumption of the optical caches and Pho$Net are calculated analytically. We used detailed simulation results such as the number of cache loads, stores, misses, and evictions and calculated each operation's cache access, network arbitration, reservation, and data transfer energy. As the request and reply subnet lasers power the passive optical cache components and there is no need for additional modulation/demodulation within the optical domain, the optical cache dynamic energy is categorized as part of the NoC. Thus, to avoid double-counting, we do not include it in the energy of the optical cache, as it has already been included in the overall energy calculation as part of the NoC. The overall optical cache static power is calculated by multiplying the number of active components with the static power of each component. We use the 30 nW reported by Nozaki *et al.* [123] as the static power needed for every optical FF. For Pho$Net we model the best configuration determined by our design-space exploration (Figure 3.11). The NoC dynamic power accounts for the modulation/demodulation during the EO/OE conversions at the cores and LLC.

Finally, we compare Pho$'s performance and energy efficiency against prior works [108]. For fair comparison, we scaled important metrics like data cache capacity and processor frequency of Maniotis *et al.*'s implementation to the same level as Pho$.

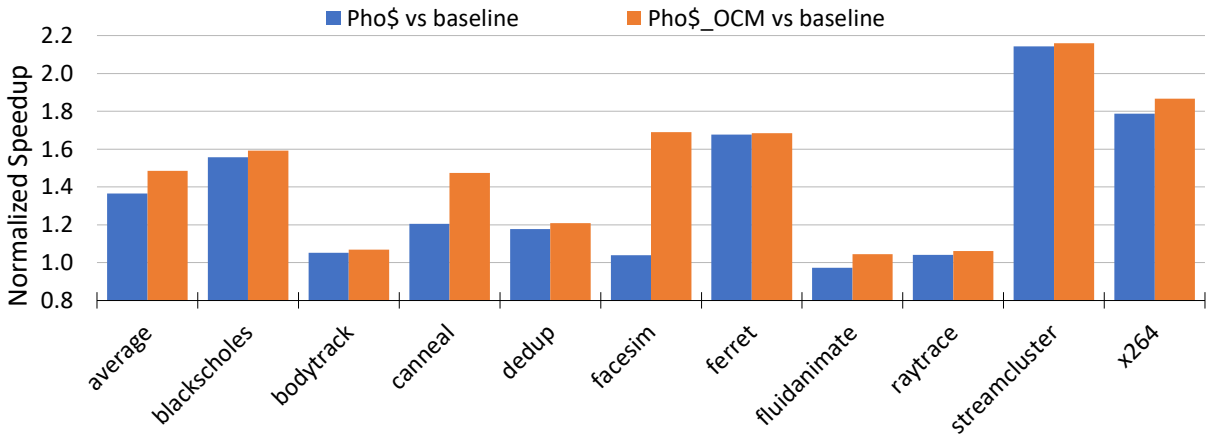
3.5. Experimental Results

3.5.1. Benchmark Performance

Figures 3.8a and 3.8b summarize the speedup of Pho$ and Pho$_OCM over the baseline running SPEC CPU2017 and Parsec 3.0. Figures 3.9a and 3.9b show the normalized CPI stacks [78], respectively. Each bar shows the relative values of cycles per instruction that are spent waiting for a particular component in the system. The "busy" sub-bar denotes the fraction of time spent within the core itself. For each application, the left, middle, and right bars represent the normalized CPI stacks of baseline, Pho$, and Pho$_OCM, respectively. Pho$ achieves an average speedup (we use arithmetic average for all averages in this chapter) of 1.34× and 1.41× without and with OCM, respectively. For CPU2017, we see an improved execution time across all applications, with *cactuBSSN* having a maximum of 3.89× speedup. Pho$ is able to significantly decrease instruction fetch delays because of its fast L1 read latency and large L1I capacity. Similarly, most applications enjoy a decrease in total L1D and L2 delay, like *leela* and *gcc_1*. The increased L1 capacity also means there are fewer misses that must visit the much slower LLC, and this is indicated by a reduced CPI for *mem-llc* in applications like *gcc*, *mcf*, and *xz*. The slow 7 ns L1 write time does not seem to have much adverse effect. OCM-enabled Pho$ makes an impact in applications like *fotonik3d* and *lbm*, providing on average an additional 5% speedup across the suite.



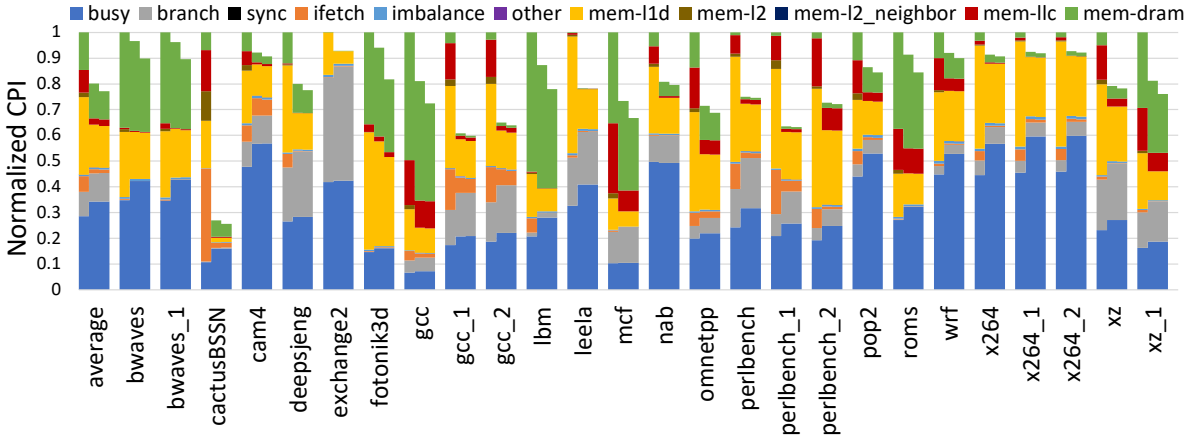
(a) CPU2017 Speedup over *baseline* (electronic multicore).



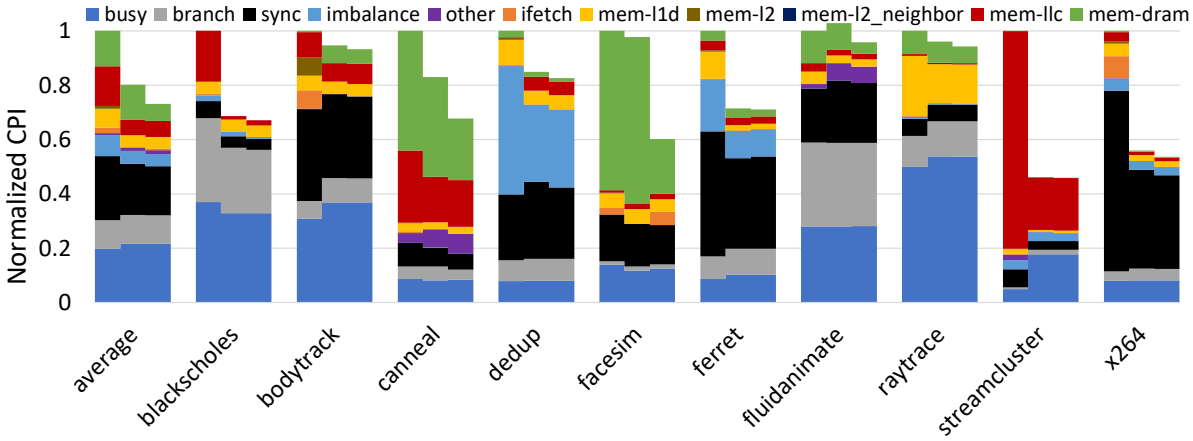
(b) Parsec Speedup over *baseline* (electronic multicore).

Figure 3.8. Normalized performance speedup of CPU2017 and Parsec.

For the multi-threaded workloads in Parsec, Pho$ is able to speed up the execution of most applications, obtaining on average 1.37× speedup. Instruction fetch delays are greatly reduced, which is most prominent in *bodytrack* and *x264*. We find that Pho$ does not suffer from high contention from a shared L1I cache. This is due to Pho$ combining the aggregate capacity of the individual L1Is in baseline into a larger shared



(a) CPU2017 CPI Stacks normalized to *baseline* (electronic multicore).



(b) Parsec CPI Stacks normalized to *baseline* (electronic multicore).

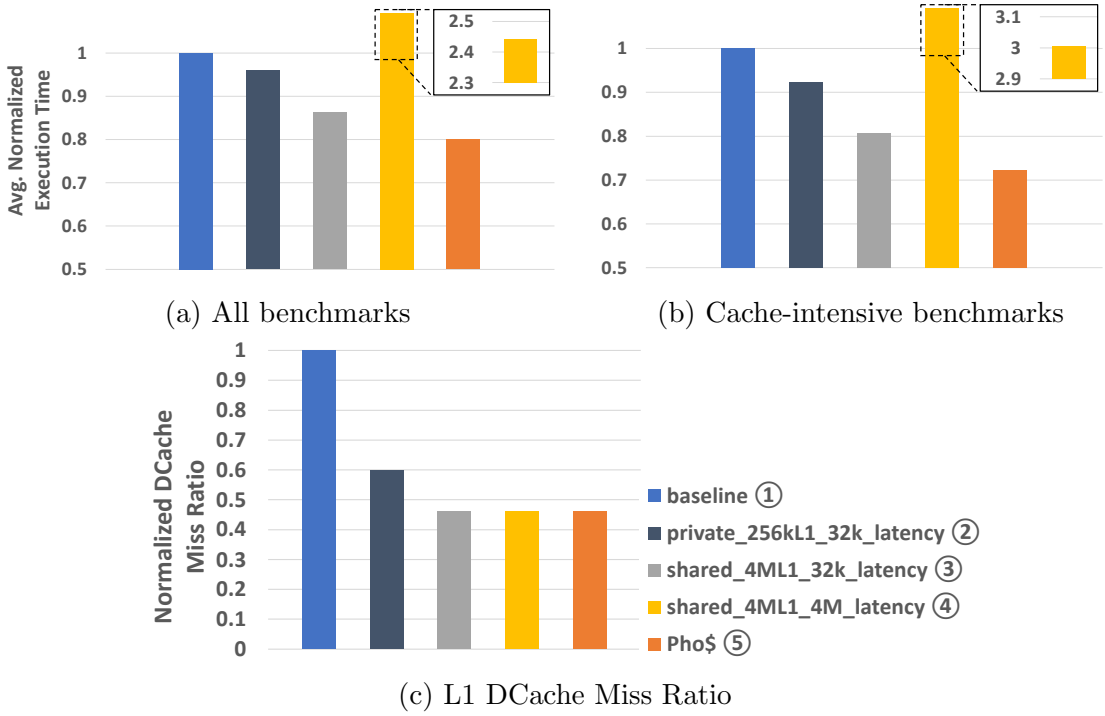
Figure 3.9. Normalized CPI Stacks of CPU2017 and Parsec. The three bars per benchmark in the CPI stack correspond to baseline, Pho$, and Pho$_OCM.

L1I, allowing more of the instruction stream to be L1-resident. Each fetched cache line also includes multiple instructions, eliminating the need for fetching on every cycle. The CPI component for L1D in Pho$ and Pho$_OCM is 44% lower on average than the CPI

contribution of L1D+L2 in the baseline. The benefits of a low read latency and large capacity outweigh the disadvantage of a high write latency. Like in CPU2017, the large capacity of Pho$'s L1 cache also results in fewer visits to the LLC and thus fewer stalls. For example, Pho$ in *blackscholes* almost eliminates the CPI contribution of LLC and in *streamcluster* reduces it by about 4×. On average, Pho$ decreases LLC delays by 2.5×. Adding OCM to Pho$ reduces the average CPI spent waiting for DRAM by 2× and increases the overall speedup to 1.48×.

Pho$ shows a slight performance slowdown in *fluidanimate*. This is caused by serialization instructions, which force the processor to flush all pending writes in its store buffer before executing the next instruction [40], and the long write latency of the optical cache stalls the processor for a prolonged period of time. This shows up as a significant increase of the "other" component in the CPI stacks for *canneal* and *fluidanimate*. However, OCM helps to outweigh this scenario and allows Pho$ to attain speedups in all of Parsec's applications, even in *fluidanimate* which experiences slowdown without OCM.

To isolate the source of the performance gains (capacity, latency, sharing), we examine three additional configurations derived from the electronic baseline (①): a hypothetical private L1 cache with increased size of 256 kB (8-way set-associative) but latency of a 32 kB cache (②), a 4 MB direct-mapped shared L1 but also with the latency of a 32 kB cache (③), and a 4 MB direct-mapped shared L1 with its respective real-world latency (④). Figure 3.10a shows the normalized average execution time of the baseline, the three hypothetical configurations, and Pho$. Pho$ gains 4% performance from increased capacity, 10% from sharing the L1, and 6% from latency. Designs ② and ③ are unrealistic but help us isolate the source of gains. Design ④ is realistic but impractical (2.4× slower than



(a) All benchmarks

(b) Cache-intensive benchmarks

(c) L1 DCache Miss Ratio

Figure 3.10. Sources of Pho$'s speedup. (a) presents the average normalized execution time of the electronic baseline (blue) and hypothetical electronic caches with zero-cost higher capacity (dark blue), zero-cost higher capacity plus sharing (grey), and a realistic high-capacity shared electronic cache (yellow) vs. Pho$ (orange). (b) presents the above results but only for cache-intensive benchmarks ($\frac{cacheCPI}{totalCPI} > 40\%$). (c) shows the average normalized L1 DCache miss ratio of all benchmarks.

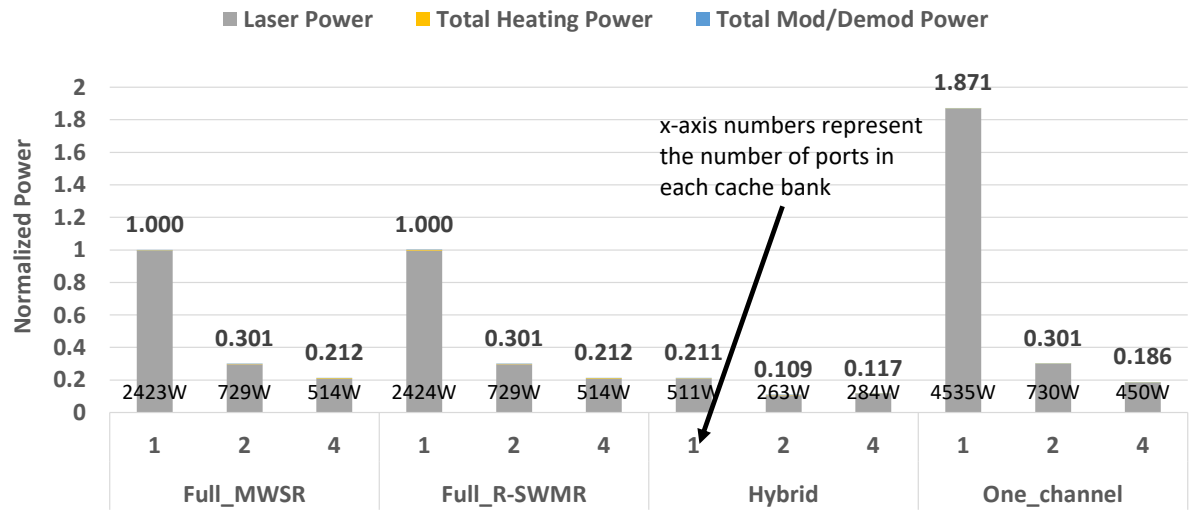
baseline; Pho$ beats it by 3.1×). Figure 3.10b shows the normalized average execution time of the same configurations, but only considering cache intensive benchmarks where the percentage of CPI spent on the cache hierarchy in total CPI is greater than 40%. The performance gain from more capacity, sharing, and latency increased to 8%, 12%, and 8%, respectively. Figure 3.10c shows the normalized L1 DCache miss ratio of all five configurations across all benchmarks. Comparing ② and ③, while an 8-way set-associative cache may lower miss rates compared to a direct-mapped one, increasing capacity by 16×

lowers misses even more. When running single-threaded workloads (SPEC) the entirety of the 4 MB cache is available to the running thread, far surpassing the performance of a 256 kB 8-way cache, even with the unreasonably fast access of a 32 kB one. In multithreaded workloads (PARSEC), when this 16× larger L1 cache is shared, the threads act as prefetchers for one another, both for data and instructions, and also avoid cache-to-cache coherence traffic. As a result, ③ has a 14% lower average L1Dcache miss ratio than ②.

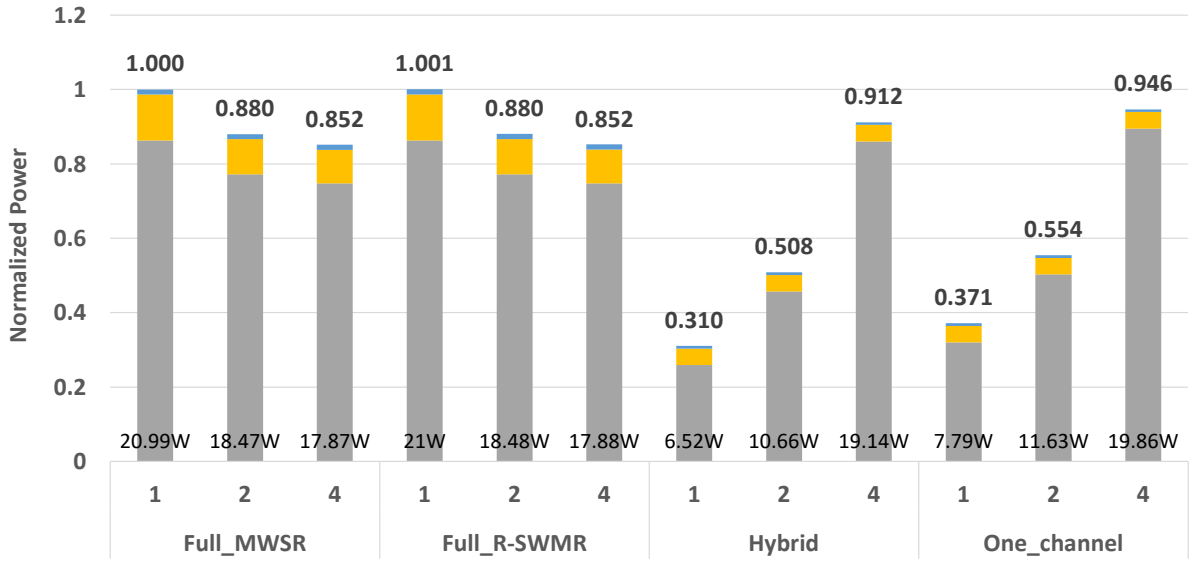
To further validate the need for a larger L1 DCache, we collected the working set sizes of SPEC CPU2017 and Parsec [153, 20] and found that most applications have working set sizes larger than 32 MB, and would not fit within Pho$'s L1D and LLC combined capacity (26 MB). Some benchmarks like *canneal* and *dedup* might even be considered to need unbounded cache and memory sizes. This implies the potential performance benefits that larger and faster cache hierarchies like Pho$ can bring.

3.5.2. Optical NoC Power Analysis

Figure 3.11 shows the normalized optical power consumption of Full MWSR, Full R-SWMR, Pho$Net, and One Channel normalized to the Full MWSR configuration (normalized separately for the conservative and aggressive nanophotonic technologies) with 1-, 2-, and 4-port optical banks (x-axis). Our estimates include the power consumption of the off-chip laser, heating for MRs, and modulation/demodulation. Table 3.6 summarizes the different subnet and MR counts for the four optical NoC configurations with 1-, 2-, and 4-port optical caches.



(a) Conservative nanophotonic parameters



(b) Aggressive nanophotonic parameters

Figure 3.11. Optical NoC power for a range of nanophotonic parameters. All results are normalized to the 1-port Full MWSR design.

Under conservative nanophotonic parameters, Pho$Net shows the lowest power consumption among alternatives and for all port numbers. Laser power constitutes over 99%

Table 3.6. Configuration comparison of different optical networks with 1-, 2-, and 4-port caches.

N_{subnet}: Number of sub-networks;
N_{ring}: Total number of MRs in the NoC;
$N_{ring.data}$: Number of MRs for each wavelength on the *data* channel;
$N_{ring.arb}$: Number of MRs for each wavelength on the *arbitration* channel;
$N_{ring.res}$: Number of MRs for each wavelength on the *reservation* channel.

	Network	Full MWSR	Full R-SWMR	One Channel	Pho$Net
1 port	N_{subnet}	1	1	5	5
	N_{ring}	515970	517293	187390	187390
	$N_{ring.data}$	21	21	32	16
	$N_{ring.arb}$	42	NA	34	34
	$N_{ring.res}$	NA	21	17	17
2 port	N_{subnet}	2	2	10	10
	N_{ring}	395460	396136	187330	187330
	$N_{ring.data}$	13	13	16	8
	$N_{ring.arb}$	26	NA	18	18
	$N_{ring.res}$	NA	13	9	9
4 port	N_{subnet}	4	4	20	20
	N_{ring}	379080	379728	187280	187280
	$N_{ring.data}$	9	9	8	4
	$N_{ring.arb}$	18	NA	10	10
	$N_{ring.res}$	NA	9	5	5

of optical power for all configurations. This is due to the high optical loss accumulated along the data path. For each waveguide with a DWDM of 64 wavelengths, 64 MRs need to be placed at each node as either modulators or demodulators for Full MWSR, Full R-SWMR, and Pho$Net topologies (128 for One Channel). This causes the optical loss incurred by all MRs along one data waveguide to be high with a conservative ring-through loss of 0.01 dB. Pho$Net gains an advantage over the other three topologies because it does not need to keep all nodes fully connected, requiring the fewest MRs along each datapath as well as the fewest data channels, thus reducing its total off-ring losses. One Channel has the worst optical loss because for each waveguide twice as many MRs are needed.

However, the high optical loss per device under conservative technology parameters still results in unrealistically high power requirements. For single port optical caches, even the most power-efficient Pho$Net configuration under the highly conservative nanophotonic parameters consumes 511 W for the network, requiring a 506 W laser power.

When we increase the number of ports of our optical cache to 2 and 4, the number of cores in each sub-network is halved and quartered, respectively, reducing the number of MRs that need to be placed along each waveguide and the total optical loss. At the same time, more sub-networks increase the number of waveguides, potentially offsetting the benefit above. All four configurations obtain lower laser power. However, Pho$Net still consumes the least power. Compared to the other topologies, Pho$Net saves 64% of total power with 2-port caches and 37–45% with 4-port caches.

We perform the same analysis using the aggressive nanophotonic parameters. The optical loss for off-resonance rings decreases from 0.01 dB to 0.001 dB. As a result, the total laser power can be lowered to a reasonable level. For single-port caches, Pho$Net achieves the lowest optical power of 6.52 W, requiring 5.43 W for the laser, 0.94 W for ring heating, and 0.15 W for modulation/demodulation. Compared to the other designs, Pho$Net still benefits from removing unnecessary links from the network and employing fewer MRs per waveguide. Having fewer MRs also reduces the MR heating and modulation/demodulation power. As a result, Pho$Net saves 70% of power compared to the two fully connected topologies and 16% compared to One Channel.

Increasing the number of cache ports under aggressive parameters increases power consumption for both Pho$Net and One Channel. As technology scales, off-ring losses have smaller weights in the overall loss. The total loss for a wavelength and even one

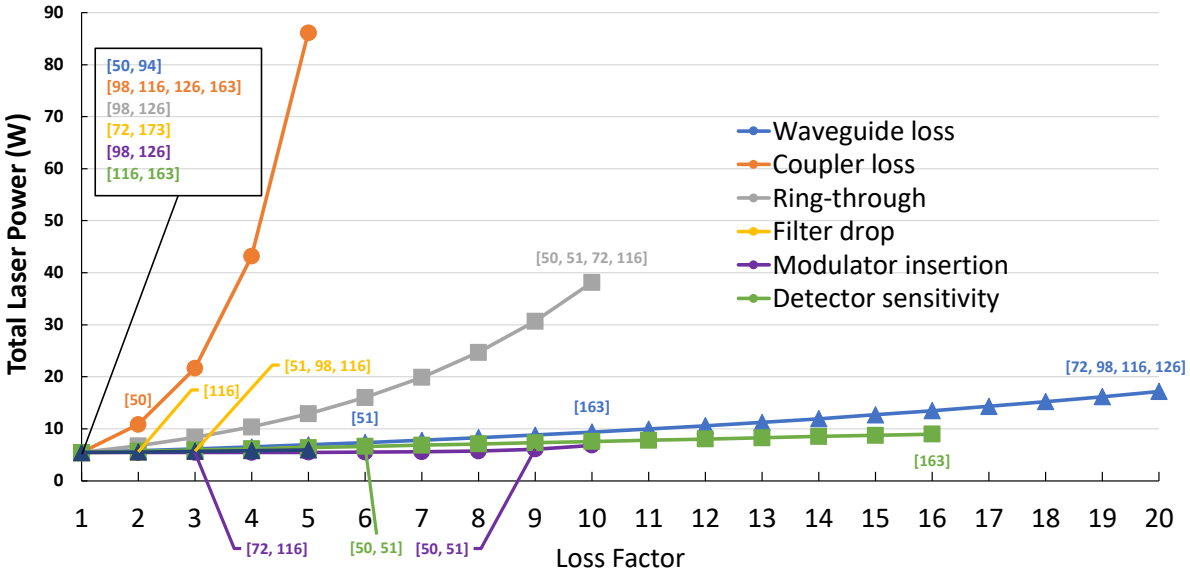


Figure 3.12. Laser power sensitivity to nanophotonic parameters. The sources for each parameter value are noted in the figure, following the same color coding scheme shown at the legend.

waveguide does not decrease by much even if we can halve the number of off-rings. For example, the per-wavelength laser power required for Pho$Net's request network under conservative parameters decreases from 16.2 mW to 4 mW when we compare 1-port to 2-port caches; however, that number only decreases from 0.27 mW to 0.24 mW under aggressive parameters. Optical loss is now more sensitive to the number of parallel waveguides. For Pho$Net and One Channel, 2-port and 4-port cache designs result in twice and four times the number of waveguides. As a result, their power consumptions increase when we have multi-port caches. On the other hand, because the MWSR and R-SWMR topologies are fully connected no matter the number of ports, they benefit from shorter individual links and fewer off-rings per link. With 4-port caches, Pho$Net has almost the same total power consumption as the other configurations. However, multi-port caches in

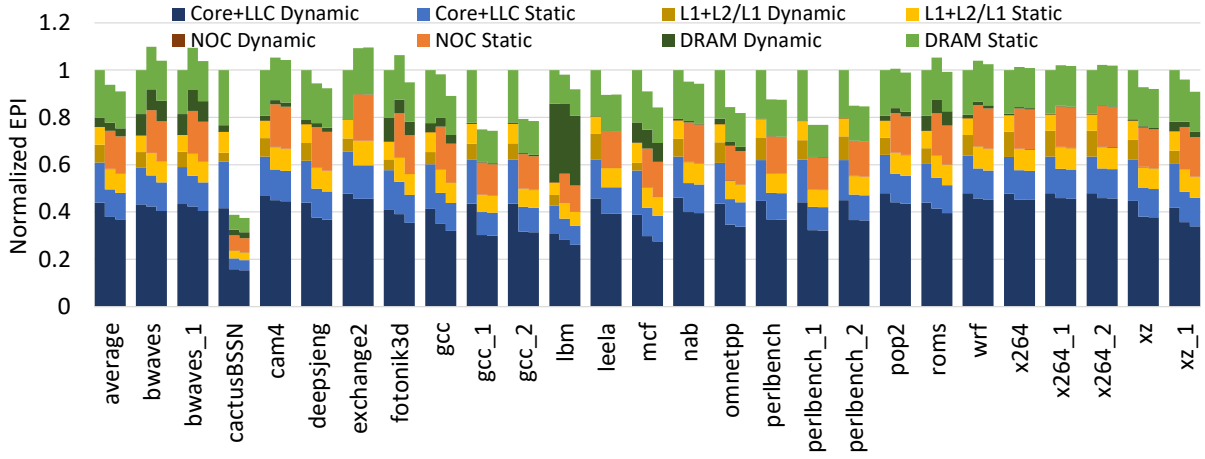
theory should provide more performance benefits by being able to serve multiple requests simultaneously. The optimal performance-power choice is beyond the scope of this thesis.

Overall, the study using aggressive nanophotonic parameters gives us a very promising power consumption outlook with the lowest power consumption being under 7 W.

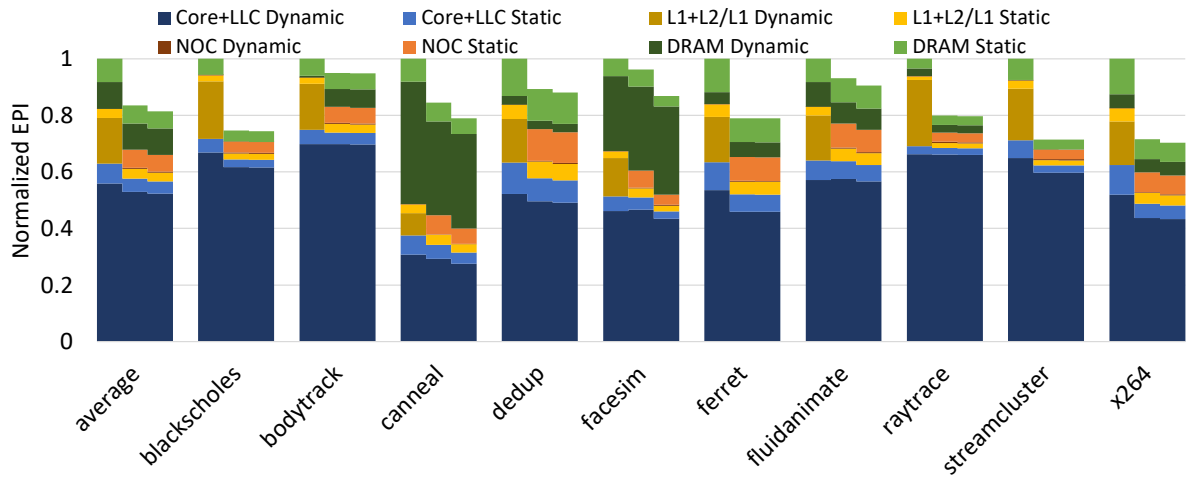
Figure 3.12 shows the sensitivity of Pho$Net's laser power to changes the scaling of optical loss for each nanophotonic parameter. Each parameter is scaled from its aggressive number up to its conservative counterpart (for modulator insertion loss, the maximum scaling factor is 1000, so we plot using the log_2 of the scaling factor on the x-axis). Pho$Net's laser power is most sensitive to coupler loss. It is also relatively sensitive to ring-through loss due to the large number of MRs required. It is relatively insensitive to all other nanophotonic parameters. This demonstrates the robustness of Pho$Net's laser power consumption under a wide range of nanophotonic technologies.

3.5.3. Energy Evaluation

Figure 3.13 shows Pho$'s normalized energy per instruction (EPI, J/$insn$) and Figure 3.14 shows shows Pho$'s normalized energy $\times$ delay product (EDP, J $\times$ s). The three bars for each workload represent baseline, Pho$, and Pho$_OCM. By replacing conventional processors' electrical L1, L2, and mesh network with Pho$'s optical architecture, the original components' energy consumptions now become the energy consumed by the optical L1 cache and optical NoC. Pho$'s L1 static energy is considered to be the total pump energy needed for optical FF operations to be stable, and it is mostly on the same level with the combined L1 and L2 static energy in the baseline. We do not consider Pho$'s L1 dynamic energy as these dynamic energy consuming operations are considered as part of



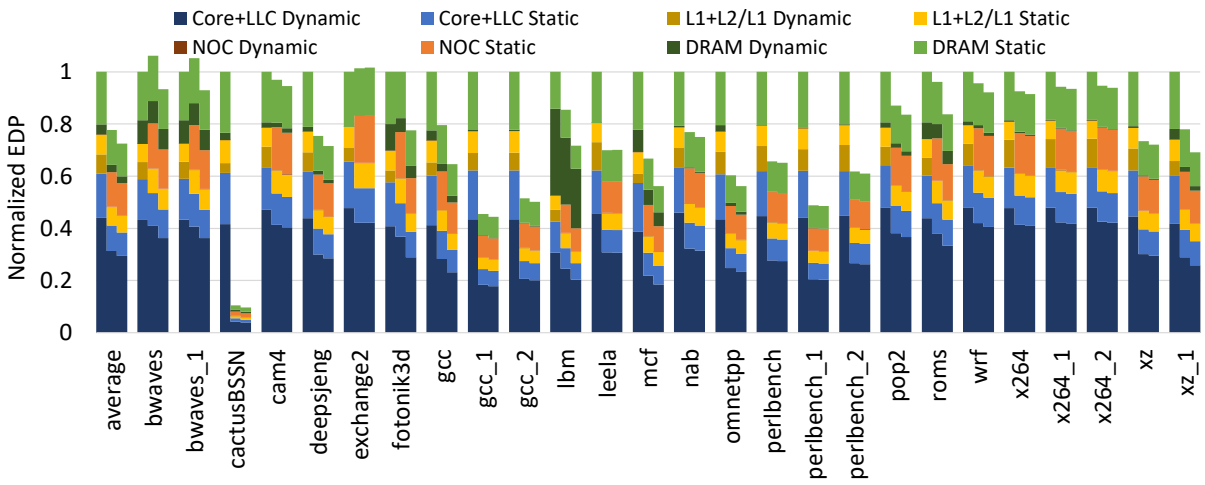
(a) CPU2017 energy per instruction normalized to *baseline* (electronic multicore).



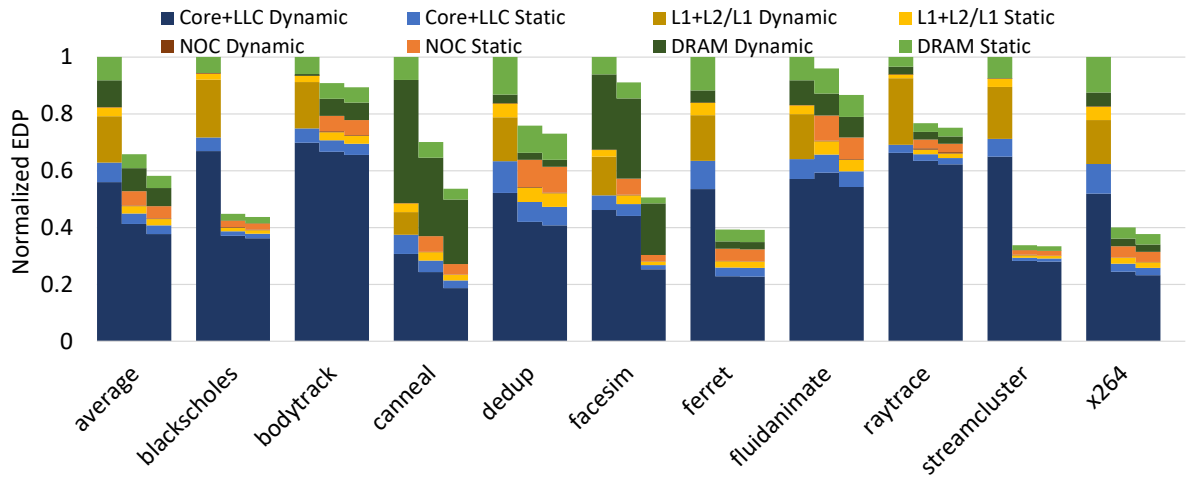
(b) Parsec energy per instruction normalized to *baseline* (electronic multicore).

Figure 3.13. Normalized energy per instruction of CPU2017 and Parsec. For each benchmark, the three bars from left to right correspond to baseline, Pho$, and Pho$_OCM, respectively.

the NoC's operations. Pho$ also has lower core and LLC energy consumption as there are less frequent core stalls and fewer LLC accesses. The EO/OE conversion energy overhead for Pho$ is minimal, which is represented by NOC Dynamic. Overall, Pho$_OCM saves



(a) CPU2017 energy×delay product normalized to *baseline* (electronic multicore).



(b) Parsec energy×delay product normalized to *baseline* (electronic multicore).

Figure 3.14. Normalized energy×delay product of CPU2017 and Parsec. For each benchmark, the three bars from left to right correspond to baseline, Pho$, and Pho$_OCM, respectively.

on average 12% EPI and 31% EDP, and is most energy efficient in applications such as *blackscholes*, *streamcluster*, and *cactuBSSN*.

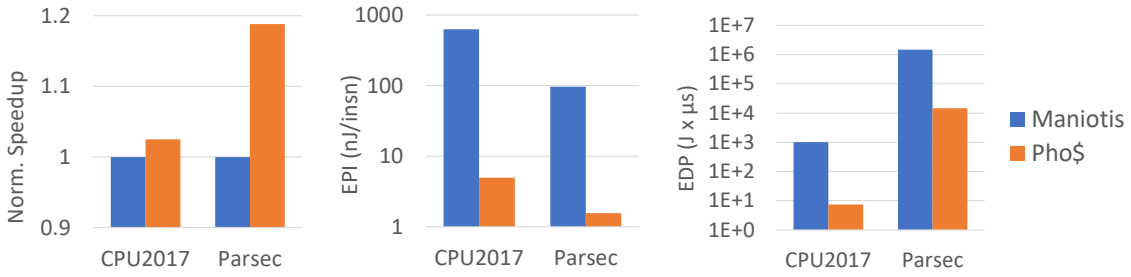


Figure 3.15. Normalized speedup, energy per instruction (nJ/insn), and energy × delay product (J×µs) of Maniotis *et al.*'s optical cache compared with Pho$.

3.5.4. Comparison with Previous Optical Cache Designs

When comparing against Pho$, the advantages of previous optical cache design from Maniotis *et al.* [108] are its 2-way associative cache design, fast write latency at 2-cycles, and a Time-Division Multiplexed (TDM) optical bus. The TDM optical bus is a less complex design compared to Pho$Net's hybrid optical network: (1) it requires just a single set of waveguides that link all nodes in order; (2) employing a TDM-based bus eliminates the need for network arbitration. However, a number of practical problems exist in this design. First, it relies on set-associative optical caches, but no optical cache designs are capable of set-associative replacement due to the lack of a replacement algorithm in the optical domain that optical set-associativity relies on. Second, its high static power due to all-passive decoder and power-inefficient PhC cells [4] makes it impractical. Finally, to avoid data collision, its TDM-based optical bus requires the entire optical system to operate at $50-80\,\text{GHz}$, as 1 CPU cycle needs to correspond to 16 optical cycles. To the best of our knowledge this is currently unattainable for optical interconnects and optical memory [5, 173]. Figure 3.15 shows the performance (speedup) and energy comparison (*log* scale) between Pho$ and Maniotis *et al.* [108], even under the assumption that the

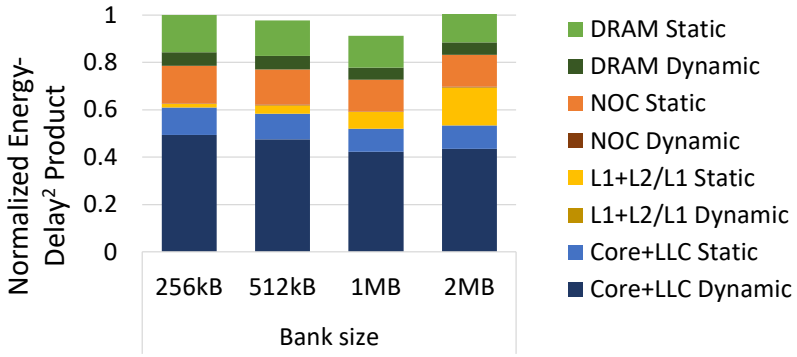


Figure 3.16. Design-space exploration of Pho$'s per-bank capacity.

associativity and TDM challenges are resolved. Pho$ is able to achieve a performance increase despite a slower writing speed, while maintaining a two orders-of-magnitude lower energy consumption.

3.5.5. Capacity-Power Inflection Point

Pho$'s static power is dominated by the number of optical bit cells. To explore the capacity-power tradeoff, we compared the average EDP and energy delay squared product (ED^2P) of Pho$ with 256 kB, 512 kB, 1 MB, and 2 MB cache banks, shown in Figure 3.16. To strike a balance between capacity and power, 1 MB is currently the best design.

3.5.6. Iso-Area Comparison

We performed iso-area comparisons by giving electronic designs the same area as Pho$. We estimate that with the additional area each core tile can employ 3 MB more cache capacity. This extra cache capacity can be used either for L2 or for the L3 slice at each tile. We explore this design space and simulate L2 + NUCA L3 slice configurations of

1+4.375 MB, 2+3.275 MB, 3+2.375 MB and 4+1.375 MB, respectively, and adjust latencies. Pho$ achieves 1.27× speedup and 37% lower EDP over the best iso-area electronic configuration.

3.6. Discussion

3.6.1. Cache Contention

While contention for the optical cache does happen, its detrimental effect (increasing the average cache access latency beyond 2 cycles) is relieved in Pho$. In our system, with 2 cycles per L1 cache access, even when 3 cores contend for the same bank at exactly the same time and serialize, one will observe a 2-cycle delay, one will observe a 4-cycle delay, and the third core will observe a 6-cycle delay, bringing the average effective cache access latency to 4 cycles per access. This exactly matches the 4–5 cycle delay of modern electronic private L1s [117, 175]. Moreover, each bank has its own private optical subnetwork in Pho$Net, so requests to different banks do not need to arbitrate with each other, effectively cutting contention by a factor of 4. This makes severe contention a much rarer event: for Pho$ to have a 4-cycle average cache delay (i.e., double the Pho$ cache latency), all 4 subnets need to have 3 requests each arriving at the same time (i.e., 12 requests contenting for the cache each time). Even the most memory-intensive application in Parsec (*streamcluster*) does not generate that amount of traffic to the cache, as less than 43% of its instructions are load/stores [20] and the baseline IPC is a mere 0.579. Thus, cache contention rarely rises to a level that presents a problem.

Even in the case when the contention is so high as for the average access latency to exceed 4 cycles, the load and store queues on the core act as buffering for cache accesses,

which allow the out-of-order core to continue executing speculatively past these contended memory operations, and hence most of the time the contention delay will be overlapped with useful computation and will not increase execution time. This is exactly how modern cores can absorb most of the delay of L1, L2 and L3 electronic caches, and Pho$ can take advantage of the same mechanism to hide the latency of severe cache contention, when it occurs.

3.6.2. Future Technology and Scalability

Higher core counts will require Pho$ to scale capacity and avoid contention. While a proper scaling study requires physical-level details that are beyond the scope of this thesis, we can make educated guesses by drawing from prior work. Over the last 20 years optical memory cell footprints decreased by 12 orders of magnitude, compared to 3 for SRAM [5], and are fast converging to their electronic counterparts. The steep improvement slope shows little signs of a slowdown, and as this is still a nascent technology, it holds the potential to exceed them in the future as it matures. Scaling the capacity with a small area footprint can be further addressed by 3D-die stacking: the optical banks do not need to communicate with each other, only with waveguides, which can be facilitated by optical TSVs (connected to waveguides through micro-mirrors), stacked multilayer waveguides or 3D opto-electronic interconnects, which are being developed globally and have been demonstrated [30, 129, 54, 130, 121, 145, 181, 151, 184]. Optical TSVs are vital in supporting the integration of 3D stacked photonic chips, and optical TSVs that can achieve loss lower than 0.1 dB while carrying error-free operations at a bandwidth of up to 40 Gbit/s have been demonstrated and validated for fabrication [30].

If Pho$ is to be scaled to a 64-core design, there are several aspects of the design to consider. By increasing total core count from 16 to 64, there will need to be four times the number of cache banks, with each bank requiring its own optical subnet. Pho$Net's optical waveguides will also need to make four more turns on the core die to ensure all cores are connected. This ultimately will lead to longer waveguides, increasing both cache access latency and laser power. For example, with 16 cores in a 4×4 layout, the length of waveguides on the core die is approximately 10× the length of one core (Figure 3.4). With 64 cores in an 8×8 layout, the total length of waveguides on the core die is 38× the length of one core, an almost 4× increase. Scaling up the total number of cores might also require a more complicated arbitration protocol, which will need additional optical components to function. It is also worth noting that a chiplet-based design [51] may fit well with a high-performance target, while leaving individual chiplets relatively smaller in size.

Other designs are also in principle compatible with Pho$. Pho$'s interconnect can easily connect chiplets on the same network or separate sub-networks per chiplet, and use multiported optical caches (Figure 3.11) to reduce contention or optical losses. Pho$ on SMPs may also be implemented individually within each socket.

3.6.3. Cost and CMOS Compatibility

Estimating the cost of Pho$ is very challenging at this point because, while these devices have been manufactured and characterized in research lab settings, they have not been manufactured at volume, so the economy-of-scale benefits and mature yield numbers are unknown. To the best of our knowledge, PhC cells are research devices that are not

commercially available. It is important to note that in the design of Pho$ we assume separate electronic and photonic dies, which simplifies the design and reduces the associated costs. We emphasize that photonic/CMOS integration has been shown in the integration of high-speed optical modulators, optical waveguides, resonators, and sensitive avalanche photodetectors in bulk CMOS chips [10] and the manufacturing of a photonic-electronic processor [159]. The latter work adopted a "zero-change" approach to the integration of photonics. Instead of developing a custom process to enable the fabrication of photonics, which would complicate or eliminate the possibility of integration with state-of-the-art transistors at large scale and at high yield, the authors designed optical devices using a standard microelectronics foundry process that is used for modern microprocessors. Thus, there is proof-of-concept work showing that the photonic devices required on the logic die can be integrated with CMOS. The devices needed for the optical cache banks can be developed and optimized separately, as they are on a separate photonic die.

3.7. Conclusions

Recent discoveries of new materials and research on optical SRAM cells enable us to buildfast, low-power optical cache architectures. In this chapter we propose Pho$, an opto-electronic memory hierarchy architecture for multicores. Pho$ replaces private electronic L1 and L2 caches with a large shared optical cache, and on-chip electronic mesh networks with a novel optical NoC that uses a unique network arbitration protocol. We estimate that Pho$ is on average 1.41× faster and 31% more energy efficient (in terms of EDP) over purely electronic designs with similar configurations. Assuming aggressive technology projections, Pho$'s network design, Pho$Net, consumes 70% less power than previously

proposed optical NoCs. We also solve a number of problems that make previous optical cache designs impractical, achieving a performance lead and two orders-of-magnitude lower energy consumption.

CHAPTER 4

Design-Space Exploration of Optical Phase Change Cache Hierarchies

4.1. Introduction

In Chapter 3, we proposed Pho$, an opto-electronic memory architecture for multi-cores. The Pho$ design employs an optical shared L1 cache and optically connected main memory but the LLC in between is still electronic. Data between the L1 cache and LLC as well as between the LLC and DRAM will need to go through OE/EO conversions. While based on our analysis and experimental results, both the OE/EO conversion latency and energy overheads are minimal, the processes can be eliminated if the LLC is also all-optical. There are several potential benefits to building an entire cache hierarchy in the optical domain. First, the latency and energy overheads associated with the OE/EO conversions can be eliminated. Second, the components and circuits that enable the OE/EO conversion processes can also be eliminated, which saves area and cost. Finally, optical caches have proven to be more performant and energy efficient in Chapter 3, and building an all-optical LLC might lead to similar outcomes.

The simple approach will be directly using the optical SRAM bitcells [123] used in Pho$ to build an optical LLC. But such a design's feasibility is limited. Each 1 MB L1 cache bank in Pho$ has an area footprint of 89 mm^2, and Pho$ achieves a total L1D capacity of 4 MB through the 2.5D integration of four layers (with one more layer for

the L1I cache). The same technique would require 22 layers of 2.5D stacking to reach a total capacity of 22 MB, which is the capacity of the electronic LLC used in Pho$. 22 layers will greatly exacerbate existing challenges of 2.5D integrating on-chip photonics such as thermal dissippation and optical TSV lengths. As a result, it is more practical to retain 4–5 layers of stacking, which will increase the total area occupied by the LLC by approximately 5×. If this approach is used, the complexity of the optical network connecting the L1 cache and LLC will be greatly increased. In Pho$Net, each cache bank has its own sub-network. For an optical LLC, there will be 4× more waveguides as the capacity and number of banks scale up. Arbitration for the network will require more on-die optical components such as micro-rings and detectors, and longer waveguides will also be needed to account for the increased overall area, resulting in higher laser power. Simply using the same PhC SRAM cells in Pho$ for an optical LLC is not practical, with the stem of the problem being its relatively low density.

Phase Change Memory (PCM) is an emerging class of non-volatile memory (NVM) that is an attractive alternative candidate for the memory hierarchy [92, 26] due to their high bit density and low leakage. PCM stores data in either crystalline or amorphous states to distinguish between logical "1"s and "0"s. Normally, PCM cells are controlled via electrical signals where different PCM states will exhibit different resistance values. Different PCM states also have distinct optical properties, and switching between states can be performed via low-power optical excitation [142]. Reading logical values from different states is also feasible. More recently, PCM cells that are etched on photonic waveguides have been proposed [110], where memory operations at both PCM states are experimentally verified. This means that optical signals in silicon-photonic links can be

directly used to write and read PCM cells that are integrated into on-chip photonic waveguides. The time is ripe to explore the design and architectures of optical PCM (O-PCM) caches. Ultimately, this paves the way for an all-optical memory hierarchy, dramatically alleviating the memory wall problem we are faced with today. The non-volatile nature of PCM also enables non-volatility at the LLC level, and supports future endeavors such as in-memory computing, neuromorphic computing, and database recovery.

In this chapter, we perform an architectural exploration of O-PCM LLCs. We base our design on Pho$ from Chapter 3 and replace its electronic LLC with an O-PCM LLC with similar capacity to provide high-bandwidth all-optical communication between the processor and memory. Our design uses a per-bank write queue and a "no-allocate" LLC write policy to alleviate the slow write speeds and low write endurance of PCM, respectively. We perform a design-space exploration of the write queue size and show that as small as an 8-entry write queue between the L1 cache and LLC can achieve a 2% reduction in execution time over the Pho$ design (20% reduction over electronic baseline) despite a very slow write latency. We also demonstrate that with a "no allocate" write policy, Pho$ with an O-PCM LLC improves the average lifetime of the LLC by 13× when compared with just one level of O-PCM between the processor and DRAM. Ultimately, we show that an all-optical memory hierarchy design can potentially have comparable performance to Pho$ while providing non-volatility by leveraging PhC SRAM cells [123], O-PCM cells [110, 142], and on-chip optical interconnects at the same time.

4.2. Background

4.2.1. Phase Change Memory

The phase change material used in PCM is $GeSbTe$ (GST), an alloy of germanium, antimony, and tellurium. A PCM cell stores a binary value by transitioning the material between a crystalline and an amorphous state. The two states exhibit very different resistances and refractive indices, making GST suitable for both electrical and optical control operations. When the GST alloy is heated to a high temperature and quickly cooled, it switches to the amorphous state; when it is heated to a temperature that is between the crystallization and melting point and then slowly cooled, the material switches to the crystalline state.

PCM has a number of desirable characteristic as a storage device. It has a fast read access time, high density, zero leakage current, and non-volatility. It is also possible to use partially crystalline states of PCM to enable multi-level cells (MLC) storage [16, 118, 90].

The two challenges that limit using PCM to build on-chip caches are its high write latency and write endurance. The long latency for writes makes PCM unsuitable to be used as first level caches because PCM caches can have write latencies as slow as over 200 ns [92]. The high traffic in higher level caches also hinders the life time of PCM caches, and the write endurance of a PCM is only around 10^8 writes [171]. However PCM might bring performance benefits when it is used in large lower level caches or main memory, where the write latency can be hidden to some extent, and high density leveraged for larger capacities.

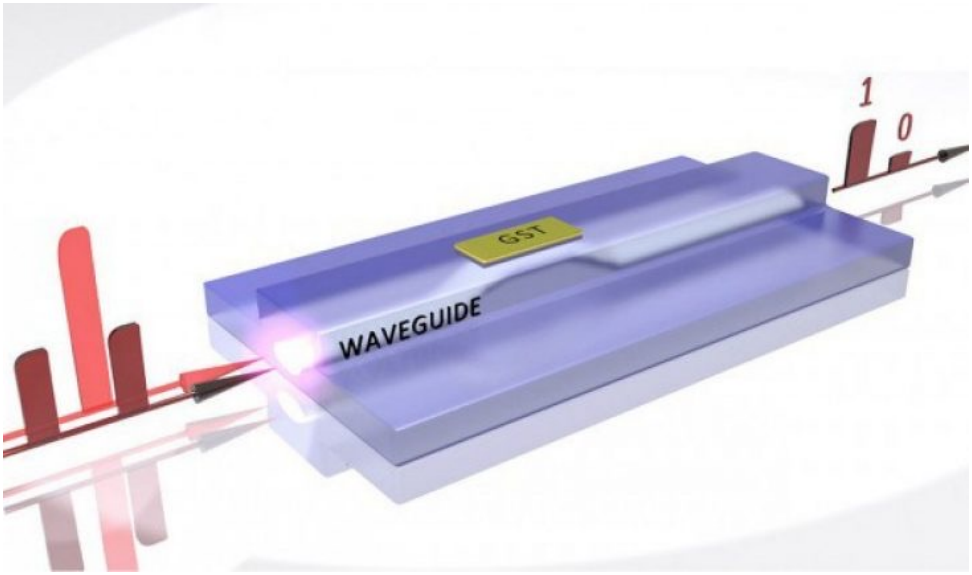


Figure 4.1. O-PCM cell structure: GST on an optical waveguide [142].

4.2.2. Optical PCM (O-PCM)

Typically, PCM cells are controlled with electrical signals. During writes, state transitions are triggered by passing electrical currents through the GST material. Reads are performed by passing a read current and measuring the voltage. Recently optical PCM (O-PCM) have been demonstrated [110, 142] where the GST material is deposited on a silicon waveguide. The proposed O-PCM has been experimentally verified to perform memory operations at both GST states and is compatible with existing CMOS and other silicon-photonics integration. Figure 4.1 shows how the GST material is coupled to an optical waveguide [142].

O-PCM write operations are performed by sending an optical signal through the waveguide on which the GST material is coupled. The optical signal has enough energy to heat the GST material and trigger a state transition. Writing a logical "0", or RESET operation requires 150 ns. This is performed when an optical pulse of 600 pJ

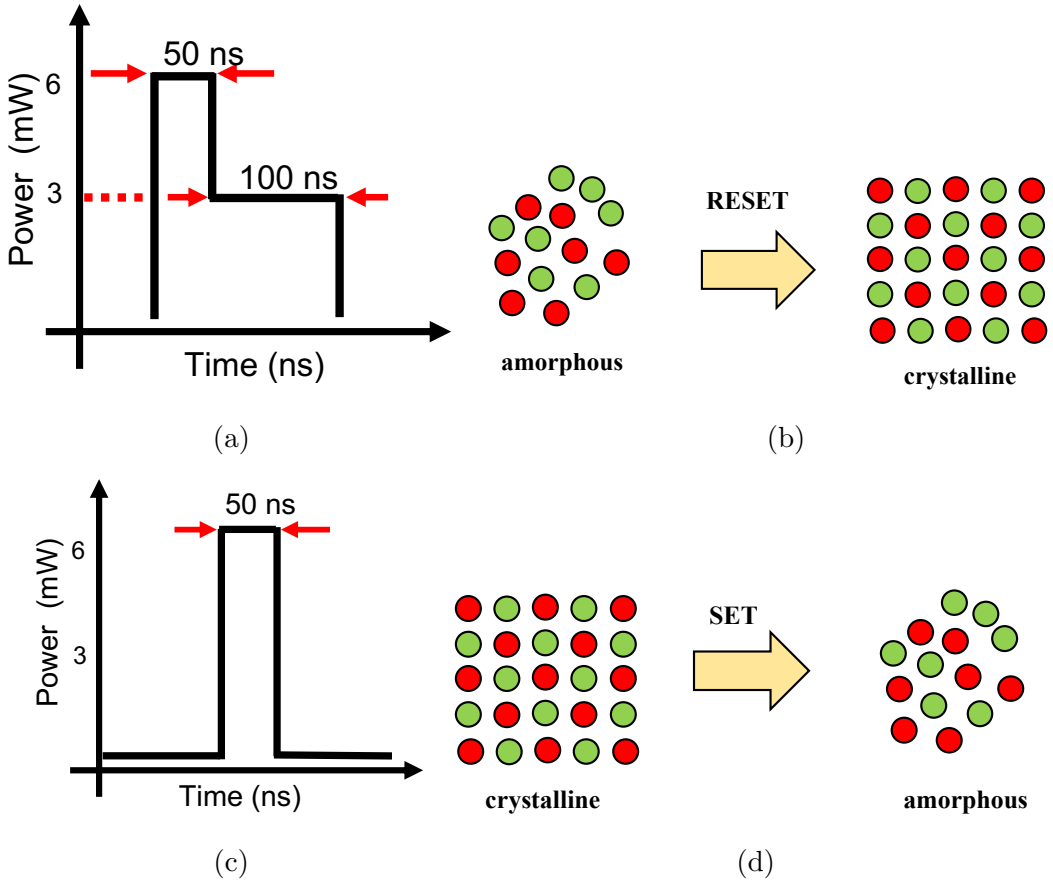


Figure 4.2. O-PCM cell write operations.
Writing a logical "0": (a) Latency and power (b) state transition;
Writing a logical "1": (c) Latency and power (d) state transition

(6 mW × 50 ns + 3 mW × 100 ns) is provided for 150 ns. The second pulse cools the GST slowly, and transitions the material to a crystalline state. Writing a logical "1", or SET operation requires 50 ns. This is performed when an optical pulse of 300 pJ (6 mW × 50 ns) is provided for 50 ns. Because only a single quick pulse is applied, the GST is able to be quickly cooled and switched to the amorphous state. Such long write latencies must be mitigated or hidden for O-PCM to be used in the LLC. Figure 4.2 shows the transition diagrams, latencies, and power requirements for O-PCM write operations.

O-PCM cell can be read by leveraging the distinct refractive indices of the crystalline and amorphous states, which absorbs different amounts of light passed through the coupled waveguide. Thus we can measure the intensity of the output light pulse and determine the read result. The latency of a read operation is only attributed to the latency of a light propagation through the waveguide, or time-of-flight. There is also no extra power required other than the Tx and Rx powers for the optical interconnects. Because the O-PCM material is directly integrated on an optical waveguide, it is possible for on-chip photonic links to directly access a cache or memory bank built using O-PCM cells.

4.3. Architecture

In this section we describe the architecture to integrate O-PCM as a last level cache to replace the electronic LLC in Pho$. We also describe the write queue for buffering writes from the upper L1 cache to the LLC as well as the modification to cache protocols to lower the number of writes to the O-PCM LLC.

4.3.1. System Architecture Overview

Figure 4.3a illustrates the architecture of the proposed design where we replace the electronic LLC in Pho$ (see Figure 3.4) with an O-PCM LLC. The L1 instruction cache has been omitted and Pho$Net, the optical NoC has been simplified. Similar to Pho$, the processor, L1 cache banks, and O-PCM LLC all sit on the interposer with photonic links. The shared L1 optical cache is built using PhC SRAM cells [123]. Instead of CMOS, the LLC is now built using O-PCM cells. This requires the off-chip laser sources to also power the O-PCM LLC die, in addition to the reply and request networks already present

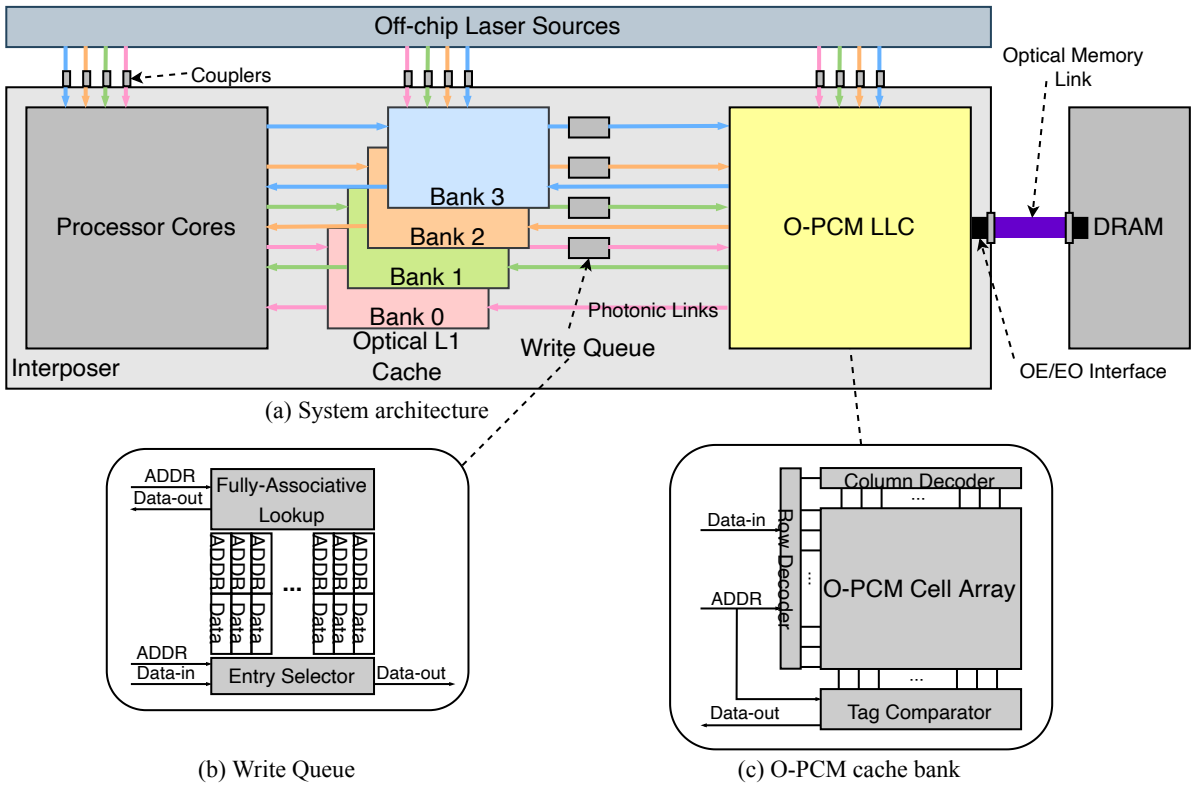


Figure 4.3. Pho$ + O-PCM architecture: (a) integration of an O-PCM LLC in Pho$ (b) structure of a write queue (c) architecture of an O-PCM cache bank.

in Pho$. This design removes the L1-LLC and LLC-DRAM OE/EO interfaces in Pho$. These interfaces were needed in Pho$ because the electronic LLC sits between the optical L1 caches and the optically connected main memory. In the new design, the entire cache hierarchy is in the optical domain.

4.3.2. The Write Queue

All write operations to the LLC that originate from the L1 caches will go through a FIFO write queue as shown in Figure 4.3b. The write queue can be designed to have different numbers of entries. Each entry contains the address and data of an in-flight write to

an LLC cache line. When the L1 cache initiates a write to the LLC, it writes to the next empty slot in the write queue. Instead of waiting for the cache line to be written, which incurs 150 ns, the L1 cache can return to its next operations immediately. In the background, the write queue will be constantly writing the in-flight cache lines, in FIFO order to the O-PCM LLC through the photonic links. Only when the write queue is full will the L1 cache be forced to stall. When the head of the queue has been written to the LLC, the L1 cache is then able to send its write to the newly available slot. Thus the maximum wait time for an LLC write initiated by the L1 cache is 150 ns, and the long write latency can be hidden from the processor to some extent. The slot selector keeps two pointers. One points to an available entry where the next incoming write will go, and the other points to the head slot of the write queue, which is the next entry to be drained onto the photonic link connecting the LLC.

When there is a cache miss in the L1 cache, it checks the write queue for a matching address first. If there is a match this means the write queue contains the most up-to-date data for that cache line and returns the data to the L1 cache. Only when no match is found does the L1 cache send the request to the LLC. The write queue is a fully-associative structure since any cache block might be written to any queue slot. As as result it is important to model the read latency of the write queue according to the number of total slots since fully-associative lookups can be expensive.

4.3.3. O-PCM Cache Bank

Figure 4.3c shows the architecture of an O-PCM cache bank. The outside decoding circuits are similar to that presented in Figure 3.1 and Figure 3.7. The optical row decoder and

column decoders can interpret the incoming cache address and select the corresponding O-PCM cells to read or write. The tag comparator will determine if there is a cache hit or miss and send back the data in the case of a read request. In principle, since each O-PCM cell can be accessed via its coupled waveguide, they are compatible with the peripheral control circuits. Due to the high density characteristic of the PCM material, we expect O-PCM caches have a larger capacity per unit area when compared with electronic caches. In our design, we envision a 32 MB O-PCM LLC with single-cycle read latency and 150 ns write latency. Even though writing a logical "1" has a latency of only 50 ns, unless the entire cache line and tag bits are composed of "1"s, a logical "0" will need to be written, which costs 150 ns. The read latency of the O-PCM cache should only be determined by the time-of-flight of the optical signal passing through the cache, and it is closely tied to the physical dimensions of the cache. The novelty of the technology, however, means that we will need to architect the exact layout of the cells to model such a cache's area and capacity, which is left for future work.

4.3.4. "No Allocation" Policy

The GST material used in O-PCM cells is limited in terms of lifetime, as each cell can only be written 10^8 times [171] before failing. To decrease the frequency of writing to the O-PCM LLC, we propose a "no allocation" policy for cache operations. For a cache hierarchy shown in Figure 4.3a, the policy can be described as follows:

- Read hit in L1: send data to processor normally
- Read miss in L1, read hit in LLC: move the line to L1; do not invalidate the line in LLC, but move the line to bottom of LRU chain

- Read miss in both L1 and LLC: fetch data from DRAM, but only allocate the cache line in L1; do not allocate in LLC
- Evictions from L1:
  - Clean eviction: send cache line to LLC and allocate in LLC
  - Dirty eviction (writeback): send cache line to LLC and DRAM (LLC lines are always clean)
- Read miss in LLC: fetch data from DRAM and forward to L1; do not allocate in LLC
- Processor write in LLC:
  - Write miss in LLC: fetch data from DRAM and forward to L1; do not allocate in LLC
  - Write hit in LLC: move the line to L1 to finish writing; do not invalidate the line in LLC, but move the line to bottom of LRU chain

This policy essentially implements a near-exclusive L1-LLC cache hierarchy. All hits in the LLC will have the cache line moved to the L1. Instead of invalidating the line in the LLC which might incur a long write latency, we move it to the bottom of the LRU replacement chain of the LLC. This ensures that the block will be used for replacement when a next write comes in, which has nearly the same effects as invaliding the line but without the extra latency. Temporarily having a not-up-to-date data in the LLC does not affect the correctness of execution because both the L1 and LLC are shared to all cores. Any access to the same address will result in a hit in the L1 (which houses the newest correct data), not the LLC. To summarize, we only allocate in the O-PCM LLC on L1 evictions (clean and dirty), which will go through the write queue first. We never allocate

in the O-PCM LLC on L1 misses only. Otherwise a write-after-write scenario can happen when the cache line is soon evicted from L1, and unnecessary writes are sent to the LLC. In this way we are able to reduce the number of writes to the O-PCM LLC, which both increases performance and expands the lifetime of the O-PCM.

The policy can also be tuned for one level of O-PCM shared cache only. In this architecture, the O-PCM is the only level of cache between the processor and DRAM.

- Read hit: send data to processor normally
- Read miss: fetch data from DRAM and send to processor; allocate in the cache
- Write hit: invalidate line and write in DRAM
- Write miss: write the line directly to DRAM; do not allocate in the cache

In this case, the O-PCM cache also only contains clean lines. Writing in the O-PCM cache is avoided during processor writes since writing to DRAM is faster. Cache writes are only performed during a read miss.

4.4. Experimental Methodology

In this section we describe the experimental methodology for a design-space exploration of the O-PCM cache architecture. Our goals are:

- determine the feasibility of just one level of O-PCM cache between the processor and DRAM
- determine the most optimal write queue size for Pho$ + O-PCM and one level of O-PCM cache in terms of performance
- compare the performance of Pho$ + O-PCM and one level of O-PCM cache

- determine which configuration benefits more from the "no allocation" cache policy in terms of performance and cell life time

4.4.1. Design Space Exploration

We wish to find the optimal write queue sizes for the configurations Pho$ + O-PCM and onelevel of O-PCM (for simplicity, we will use the names "Pho$OPCM" and "One Level" for the rest of this thesis). As the size of a write queues increase, it is able to buffer more write requests to the O-PCM LLC in the same time frame. As a result the L1 cache and processor will need to wait for an empty slot in the write queue less often, potentially increasing application performance. Increasing write queue sizes, however, comes at a cost. Because each write queue slot needs to hold the entire cache address the write queue must be a fully-associative structure to support reading in-flight writes during an L1 cache miss. Fully-associative lookup latencies and fill latencies can scale up dramatically as the number of entries increases. Thus as we increase the write queue size, eventually we will hit an inflection point and get worse performance instead. Fully-associative structures also require more circuitry, which is another reason we would like to keep the write queue size to an optimal number: to have the least performance impact and not consume too much power and area. We sweep through write queue sizes of 0, 4, 8, 16, 32, 64, 128, 256, 512, and 1024.

We conduct our design space exploration using the Sniper simulator [28, 29] running SPEC CPU2017 [25] (SPECspeed, ref inputs). For comparison, we also simulate a baseline electronic multicore similar to a 16-core Intel Skylake [61, 42, 117, 139, 174, 175] as well as Pho$ with an electronic LLC. The configurations for the eletronic baseline and Pho$ are

Table 4.1. O-PCM: Simulated system parameters.

Component	**Details**
Cores	16 cores, x86 ISA, 3.2 GHz, OoO, 4 wide dispatch/commit, 224-entry ROB, 72-entry load queue, 56-entry store queue
L1 ICache	**Baseline**: electronic, private, 64 B line,32 kB/core, 8-way, 4 cycles **One Level**: N/A **Pho$/Pho$OPCM**: optical, shared, 64 B line, 1 MB direct-mapped, 2-cycle read, 23-cycle write
L1 DCache	**Baseline**: electronic, private, 64 B line, 32 kB/core, 8-way, 4 cycles **One Level**: N/A **Pho$/Pho$OPCM**: optical, shared, 4 banks, 64 B line 4 MB direct-mapped, 2-cycle read, 23-cycle write
L2	**Baseline**: electronic, private, 64 B line, 256 kB/core, 4-way, 14 cycles **One Level/Pho$/Pho$OPCM**: N/A
LLC	**Baseline/Pho$**: electronic, shared, non-inclusive, 64 B line, 32 MB 16-way, 50 cycles **One Level/Pho$OPCM**: optical-PCM, shared, exclusive, 64 B line 32 MB, direct-mapped, 1-cycle read, 480-cycle write
Write Queue	**Baseline/Pho$**: N/A **One Level/Pho$OPCM**: 1 queue per L1D bank, see Table 4.2 for latency
Network	**Baseline**: electronic **One Level/Pho$/Pho$OPCM**: optical
Memory	electronically connected, 49.37 ns

Table 4.2. Write queue sizes and latencies.

Write queue size (# of slots)	4	8	16	32	64	128	256	512	1024
Access cycles	1	1	1	2	2	2	2	2	3

identical to that in Table 3.5 except the LLC which have a capacity of 32 MB. To isolate the experiment target to the cache hierarchy, OCM is not included in the configurations. Table 4.1 summarizes the simulation details.

To calculate the access cycles of the write queue, we used Cacti 7.0 [11] to simulate a fully-associative cache structure with the number of slots in Table 4.2 under the 14 nm

technology node. The latencies were compared against the clock cycle of a 3.2 GHz frequency and rounded up to the nearest cycle. Reading from the write queue and writing an entry to the tail of the queue will both be penalized with the corresponding access cycle.

4.4.2. Modeling O-PCM and Write Queue

To correctly account for the long write latency of O-PCM and the background FIFO operation of the write queue, we modified the Sniper simulator extensively. When the L1 cache needs to write to the O-PCM LLC and the write queue *is not* full, we should only incur the write queue's access latency as the actual writing of the O-PCM LLC is delayed until the cache line reaches the head of the queue. Between the times when the cache line initially enters the tail of the queue and the actual write at the LLC level is finished, if no other write has to wait for the queue to have an empty slot, then effectively the latency of this write is hidden from the processor and L1 cache. When the L1 cache needs to write to the O-PCM LLC and the write queue *is* full, we should incur the write queue's access latency as well as the time needed for the queue to drain its current head. Because the queue might have partially drained its head when the new write comes in, the extra latency can be between 0 ns and 150 ns.

Because Sniper uses a statistically based simulation model and is not cycle-accurate, instead of building a cache-like FIFO queue structure, we model the write queue timing using a simple queueing model . Each instance of a write queue inside the simulator is initialized with a queue size. Inside the queue, each in-flight entry records its end time (time when the cache line has been fully written into the LLC) and address. When a

Algorithm 1 Write queue modeling

```
1: function HasFreeSlot(t_start)
2:     for i ← 0, wq_size do
3:         if wq_entry[i].endtime <= t_start then
4:             return True
5:         end if
6:     end for
7:     return False
8: end function
9:
10: function AddEntry(t_start, t_process, addr)
11:     free ← 0
12:     for i ← 0, wq_size do                          ▷ Find first available free slot
13:         if wq_entry[i].endtime <= t_start then
14:             free ← i                                ▷ Slot i is free right now
15:             break
16:         else if wq_entry[i].endtime < wq_entry[free].endtime then
17:             free ← i                                ▷ Slot i is the first free slot so far
18:         end if
19:     end for
20:
21:     if t_start < wq_entry[free].endtime then ▷ Delay incoming write until the earliest
    free slot
22:         t_end = wq_entry[free].endtime + t_process
23:     else                                            ▷ No need to wait for a slot
24:         t_end = t_start + t_process
25:     end if
26:     wq_entry[free].endtime ← t_end
27:     wq_entry[free].addr ← addr
28: end function
29:
30: function GetStartTime(t_start)
31:     for i ← 0, wq_size do                          ▷ Find first available free slot
32:         if wq_entry[i].endtime <= t_start then
33:             return t_start          ▷ New entry can be written to the queue immediately
34:         else if wq_entry[i].endtime < wq_entry[free].endtime then
35:             free ← i                                ▷ Slot i is the first free slot so far
36:         end if
37:
38:         if t_start < wq_entry[free].endtime then       ▷ Delay incoming write until the
    earliest free slot
39:             return wq_entry[free].endtime
40:         else                                        ▷ No need to wait for a slot
41:             return t_start
42:         end if
43:     end for
44: end function
```

Algorithm 2 Write queue model usage

1: t_{now}
2: $t_{wq\_avail} \leftarrow t_{now}$
3: **if** not WRITE_QUEUE.HASFREESLOT(t_{now}) **then**
4: $\quad t_{wq\_avail} \leftarrow$ WRITE_QUEUE.GETSTARTTIME(t_{now})
5: $\quad wait\_time \leftarrow t_{wq\_avail} - t_{now}$
6: $\quad simulation\_time \leftarrow simulation\_time + wait\_time$
7: **end if**
8: $t_{process} \leftarrow 150\,\text{ns}$
9: WRITE_QUEUE.ADDENTRY($t_{now}, t_{process}, addr$)

write enters the write queue, the model checks the current time against other entries for any available slot. An available slot is found when an entry's end time is earlier than the current time, meaning that slot is not currently occupied by an in-flight write. If there is an available slot, the cache line is put onto the queue and we incur no extra latency except the queue access latency. If there is no available slot, the queue model calculates the earliest time a slot becomes available and total waiting time is $t_{available} - t_{now}$. Next we will use the model to calculate and record the end time of the current incoming cache line in the queue. The pseudocode of the model is described in Algorithm 1 and its usage is described in Algorithm 2. Note that $t_{wq\_available}$, t_{now}, t_{start} and t_{end} are elapsed simulation timestamps. $t_{process}$ is the time required to process a single entry in the write queue (150 ns in this experiment). To model reading from the write queue, every read to the LLC is intercepted and the address is checked in the write queue first. If there is a hit we incur the write queue's access latency instead of the read latency of the LLC (LLC hit) or DRAM (LLC miss).

To correctly attribute the added latencies to their respective hardware components, we added a *mem-write-queue* label in the CPI stack calculation process. Any read hits in the write queue and writes to the write queue will be categorized as cycles spent waiting for

the write queue itself. Any time spent waiting for a slot to become available in the queue is attributed to the O-PCM LLC or DRAM, depending on where the data is eventually supplied from.

4.4.3. Write Policy and Cache Lifetime Study

We also modified the cache protocol in Sniper to support the "no allocation" policy described in Section 4.3.4. We selected from the write queue design space exploration the optimal write queue sizes in terms of performance and recorded their memory access traces. For each benchmark, we calculated the number of writes to each physical cache block. Then we used the maximum number of writes and the lifetime of the O-PCM material (10^8) to calculate the number of times a benchmark can be repeated. We also created CDFs of the total number of writes to each LLC physical cache frame to study the policy's effect on the whole cache.

4.5. Experimental Results

4.5.1. Design Space Exploration Results

Figure 4.4 shows the average CPI stacks [78] of the baseline electronic multicore, One Level with different write queue sizes, Pho$OPCM with different write queue sizes, as well as the Pho$ architecture proposed in Chapter 3 running workloads from SPEC CPU2017. For One Level, because there is only one level of O-PCM cache between the processor and DRAM, the cache is under heavy traffic. Read misses show up as "mem-dram" components in the CPI stack because the data is fetched from DRAM, then written to the cache. Without a write queue, One Level's CPI is 143× that of baseline, and all

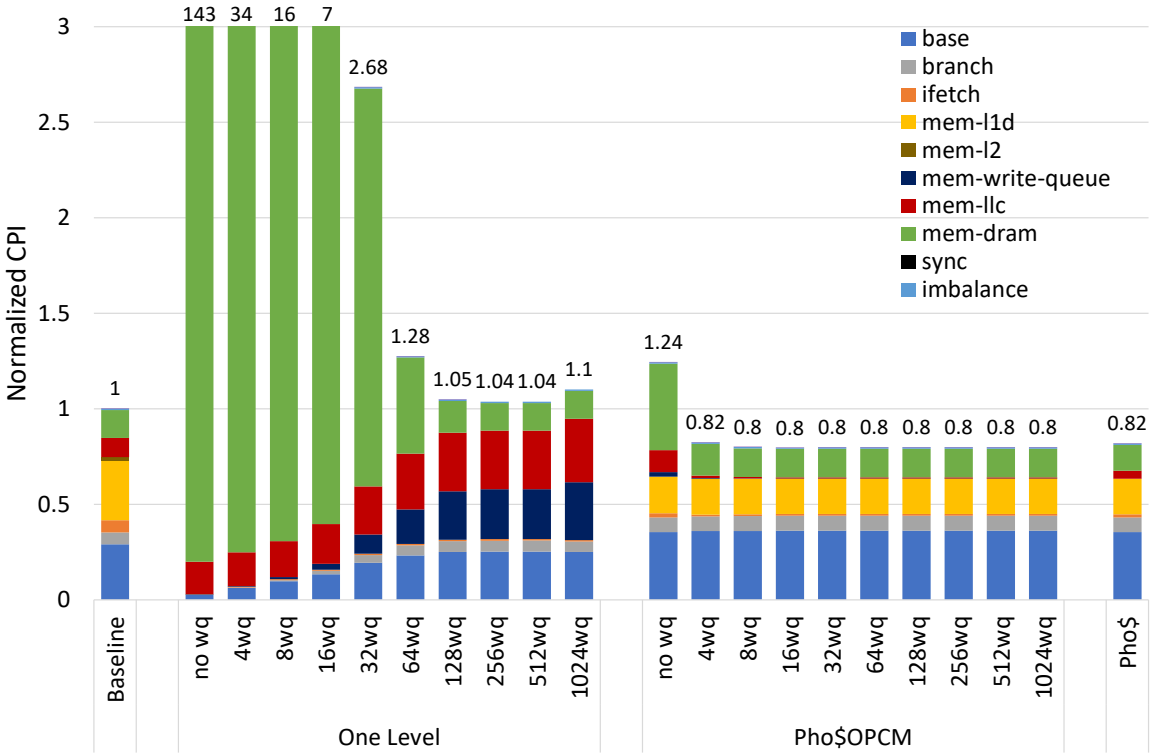


Figure 4.4. Average CPU2017 CPI Stacks normalized to *baseline* (electronic multicore).

hardware components are severely stalled. As we increase the size of the write queue, One Level's CPI decreases quickly. For write queues of 256 and 512 entries, we hit an optimal performance at 1.04× slower than baseline. We also see the CPI of the write queue (dark blue) increase as we use larger queues because (1) larger queues can store more in-flight writes, which increases the possibility that a cache read hits in the write queue and (2) larger write queues have longer access latencies. The benefits of a larger queue size diminish at a 1024-entry write queue. Compared to 256- and 512-entry write queues with 2-cycle access latencies, a 1024-entry write queue suffers by having a 3 cycle access latency. Nonetheless, despite big write queues, One Level is incapable of performing

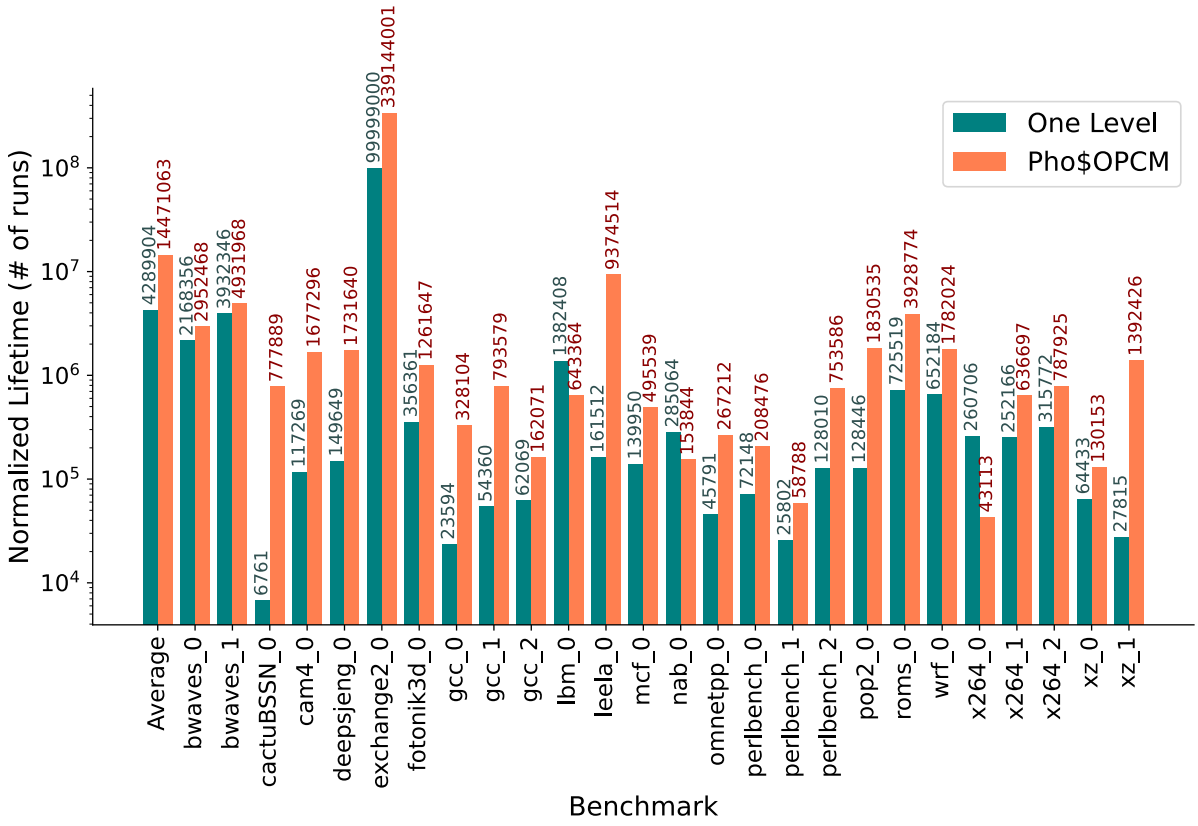


Figure 4.5. O-PCM cache lifetime.

better than the electronic baseline. It seems imperative that we add an L1 cache before the O-PCM cache.

For the Pho$OPCM configurations, the optical shared L1 cache absorbs the majority of the traffic to the O-PCM LLC. With the LLC acting as a "victim cache" for the L1, we get comparable performance to Pho$ with only a 4-entry write queue, with an 18% reduction in CPI over baseline. Pho$OPCM does hit its optimal write queue size early as increasing beyond 8 entries does not yield any significant performance benefits. With an 8-entry write queue, the majority of writes to the O-PCM LLC is hidden, and we get a 20% CPI reduction compared to baseline and 2% CPI reduction compared to Pho$.

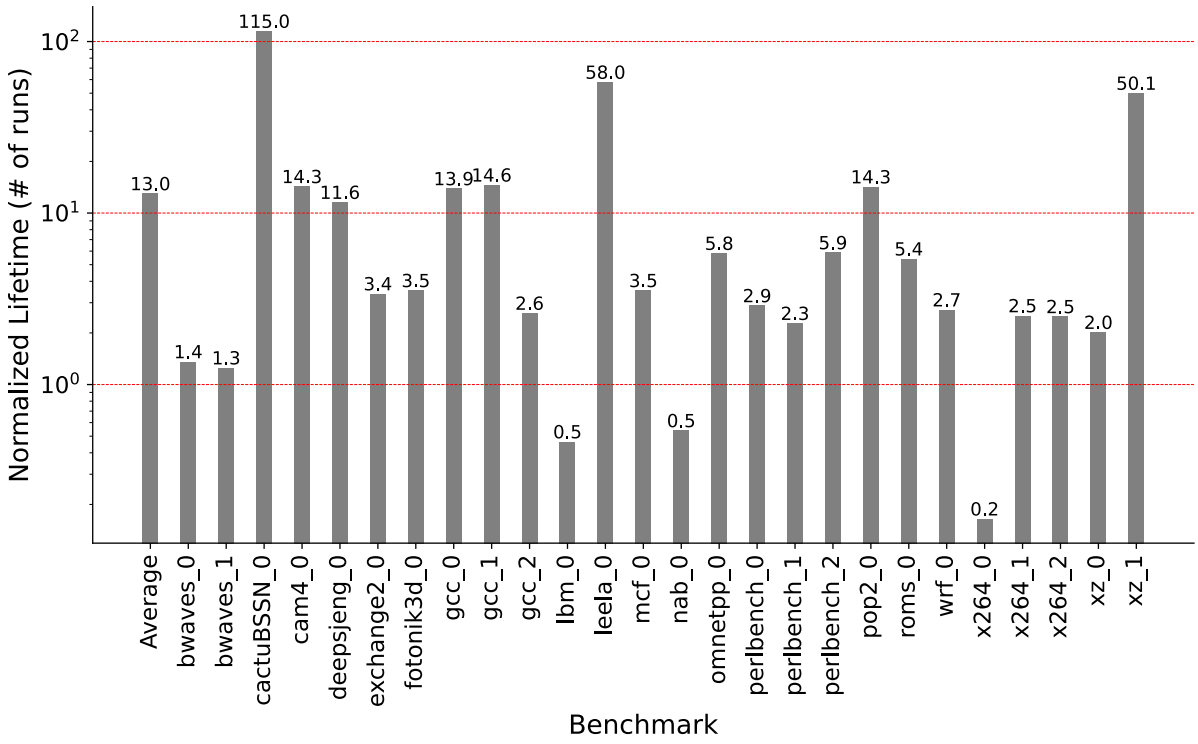


Figure 4.6. Normalized O-PCM cache lifetime (Pho$OPCM vs One Level).

By conducting the design space exploration, we can determine that from a performance aspect, the optimal write queue sizes for One Level and Pho$OPCM are 256 and 8, respectively. Detailed CPI stacks for every benchmark can be found in the Appendix (Figures A.1, A.2, and A.3).

4.5.2. "No Allocation" Policy Study

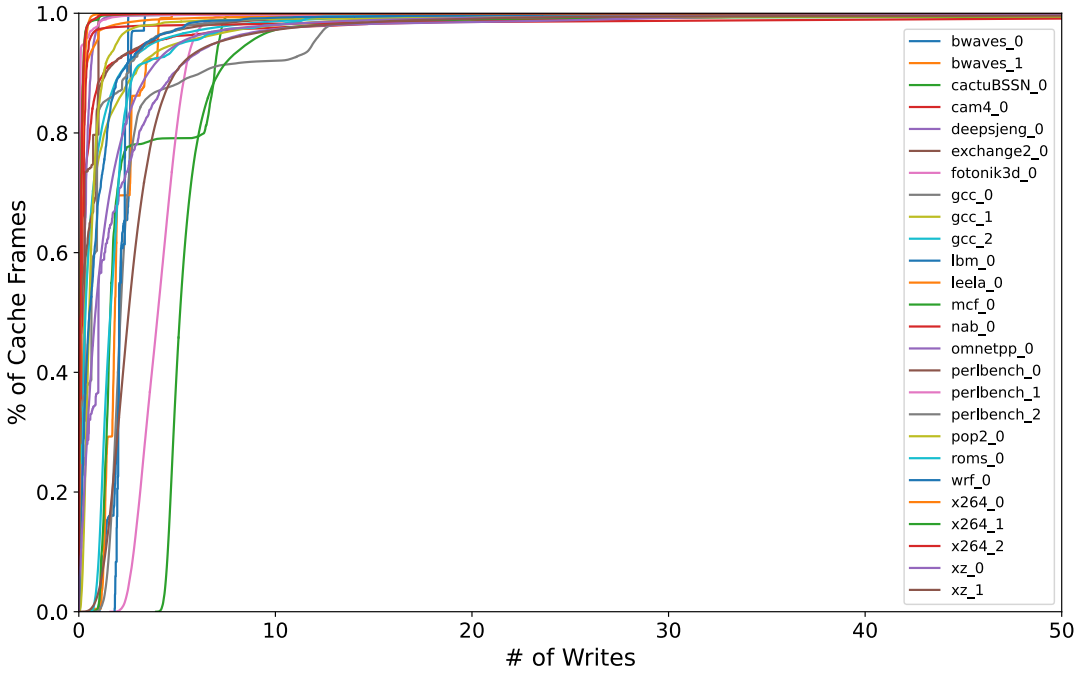
Figure 4.5 shows the lifetime of One Level and Pho$OPCM with the "no allocation" policy running workloads from SPEC CPU2017. Figure 4.6 shows the normalized lifetime of Pho$OPCM vs One Level. When employing the "no allocation" policy, Pho$OPCM has a significant longer average lifetime than One Level (13×). For *lbm*, *nab*, and *x264_0*,

Pho$OPCM shows shorter lifetimes than One Level. This can be because the write-heavy characteristic of these workloads. For One Level, when writes miss in the O-PCM cache, the "no allocation" policy governs that no block allocation happens in the cache but writes are directed to the DRAM instead. For Pho$OPCM in the case of a stream of writes to the same cache line, if the writes miss both in the L1 and the O-PCM LLC, when the cache line is fetched from DRAM it can cause an eviction from the L1, which will be a cache allocation in the O-PCM LLC. This experiment shows that the "no allocation" policy works well to improve O-PCM lifetime for general workloads but not for workloads with long streams of writes.

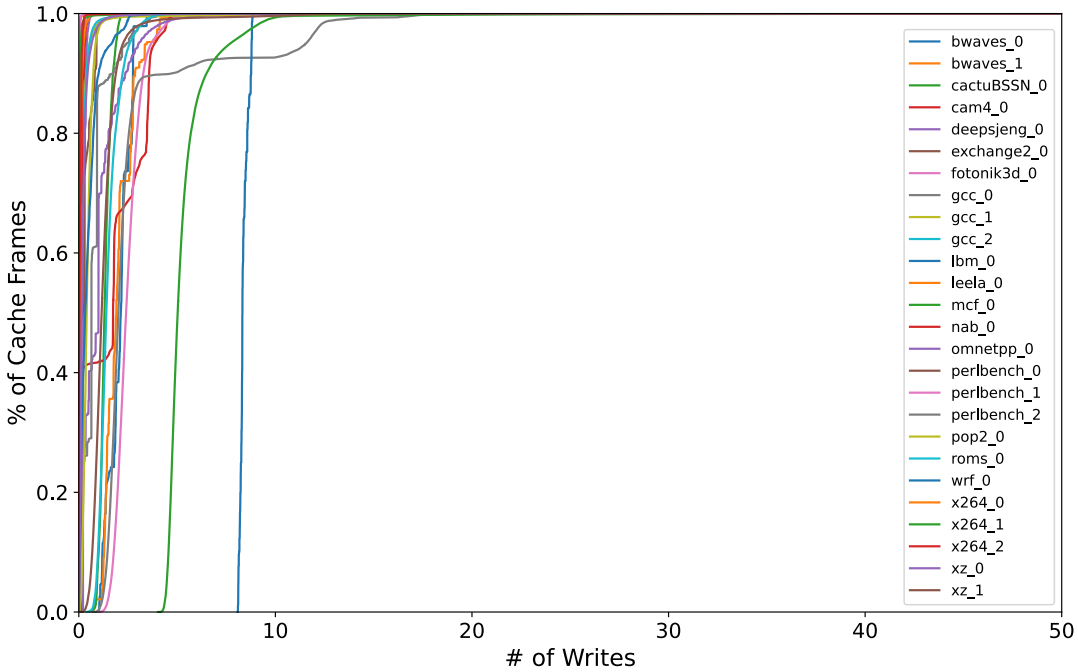
Figures 4.5 and 4.6 consider lifetime by calculating the earliest physical cache frame to fail after repeated writes. In Figure 4.7 we consider all physical cache frames by showing the CDF of the percentage of physical cache frames and their total number of writes for One Level and Pho$OPCM, respectively. For most benchmarks, Pho$OPCM shows a steeper plot and the CDF reaches 100% more quickly. This means that physical cache frames experience fewer number of writes in a benchmark run, indicating an overall longer lifetime for the O-PCM cache. The only exception is *lbm* where One Level experiences fewer writes for almost all physical cache frames, for the reason explained in the paragraph above.

4.6. Conclusions

Emerging phase change memory is a promising technology for building future memories and caches with advantages such as high density, low leakage power, and data non-volatility. However, both electrical and optical PCM suffer from high write latency, high



(a) One Level



(b) Pho$OPCM.

Figure 4.7. Cache Frames Writes CDF.

write energy, and low write endurance. In this chapter, we augmented the opto-electrical cache hierarchy architecture of Pho$ with an O-PCM last level cache for an all-optical cache hierarchy. We performed a design space exploration on the performance impacts of write queues in mitigating O-PCM's long write latency. We also studied the effects of a "no allocation" write policy to reduce the frequency of O-PCM cache writes for both One Level and Pho$OPCM configurations. Our results show that by employing only a write queue of 8 entries and the write policy, the combination of Pho$ and an O-PCM LLC can achieve (1) similar performance to Pho$ despite O-PCM's long write latency, (2) 13× the cache lifetime vs One Level, while providing non-volatility at the cache level.

CHAPTER 5

Public Release and Validation of SPEC CPU2017 PinPoints

5.1. Introduction

In architecture research, evaluating novel ideas before they are physically implemented requires modeling them on simulators that execute a wide range of representative benchmarks. Released in 2017, the SPEC (Standard Performance Evaluation Corporation) CPU 2017 benchmark suite [1] has been a popular tool in computer architecture for such tasks. However, the increased dynamic instruction counts and large memory footprints of CPU2017 over its predecessors have led to unrealistically long simulation times. The increasingly complex modeling of novel architectures, possibly with high core counts, sophisticated cache and Network-on-Chip protocols, and emerging materials further exacerbate the problem—some benchmarks take months to simulate to completion. To combat this problem, phase-based statistical sampling methods like SimPoint [150] were developed, in which simulating multiple short representative regions can predict the behavior of the whole application, significantly reducing the total simulation time. The Intel PinPoints [131] tool set automates the region-finding process by using the program instrumentation tools Pin [105] and PinPlay [133]. Each generated representative region, also called a pinball, can be replayed on simulators such as Sniper [28] and ZSim [147] and shared among researchers, liberating them from the need to collect sampled regions.

Pinballs of CPU2017 are already available online [177, 2] from other researchers. However, our own attempts at using these pinballs with the latest version of the Sniper simulator were unsuccessful as various errors would end simulations before performance statistics could be collected. Thus we decided to collect our own pinballs of the CPU2017 SPECspeed benchmarks, not only to use in our own research, but also to provide an alternate set of CPU2017 pinballs that researchers from around the world might find useful. The link to our pinballs repository can be found on our lab's website, PARAG@N, under "Pinballs" in the Artifacts section at http://paragon.cs.northwestern.edu/#Artifacts. This chapter provides the details of our pinball collecting process, pinball statistics, and validation of representability against whole programs in terms of cycles per instruction (CPI).

Our generated pinballs achieve an average absolute error rate of 12% when comparing their predicted CPIs with those of dynamically linked native applications in the SPEC CPU2017 SPECspeed suite. We also find that for applications with high CPI prediction error rates, comparing against statically linked applications can reduce their error rate by an average of 29.7%, bringing the average absolute error rate across the entire SPEC CPU2017 SPECspeed suite down to 8%.

5.2. Background

5.2.1. SPEC CPU2017

SPEC CPU2017 is a collection of 43 benchmarks categorized into four suites: SPECspeed Integer, SPECspeed Floating Point, SPECrate Integer, and SPECrate Floating Point. The SPECspeed benchmarks are mainly used for measuring execution time by always

running one copy of a benchmark while SPECrate measures throughput by running multiple concurrent copies of each benchmark. CPU2017 also provides three input sizes: *test*, *train*, and *ref*. Only the *ref* input size can be used when reporting time metrics.

5.2.2. Pinballs

Pin [105] from Intel is a dynamic binary instrumentation framework. Tools created using Pin can be used to analyze user space applications at the instruction set architecture (ISA) level. It does not require recompiling the source code of a target application because the instrumentation is performed at run time. The Program Record/Replay Toolkit [133], or PinPlay is a set of tools built using Pin that support the logging and replaying of an entire program or a part of it. Running PinPlay's logger on a program produces a pinball, which can either be replayed using PinPlay itself or fed into a simulator. A pinball is a collection of files that contain information about a program like initial memory and register states, register states before and after system calls, etc. When replayed, the pinball guarantees repeatable and deterministic behavior. A large pinball of a whole program can be sliced into regional pinballs.

SimPoint [150] is a tool that uses statistical sampling to capture multiple representative simulation regions of a large program. The simulation points produced by SimPoint contain the bulk of information about a program's execution and can be used to accurately model the run time behavior of the whole program. Each simulation point is associated with a weight that roughly represents the relative frequency by which the corresponding phase executes in the program. Because only small phases of the program are captured, simulation time with SimPoint is greatly reduced.

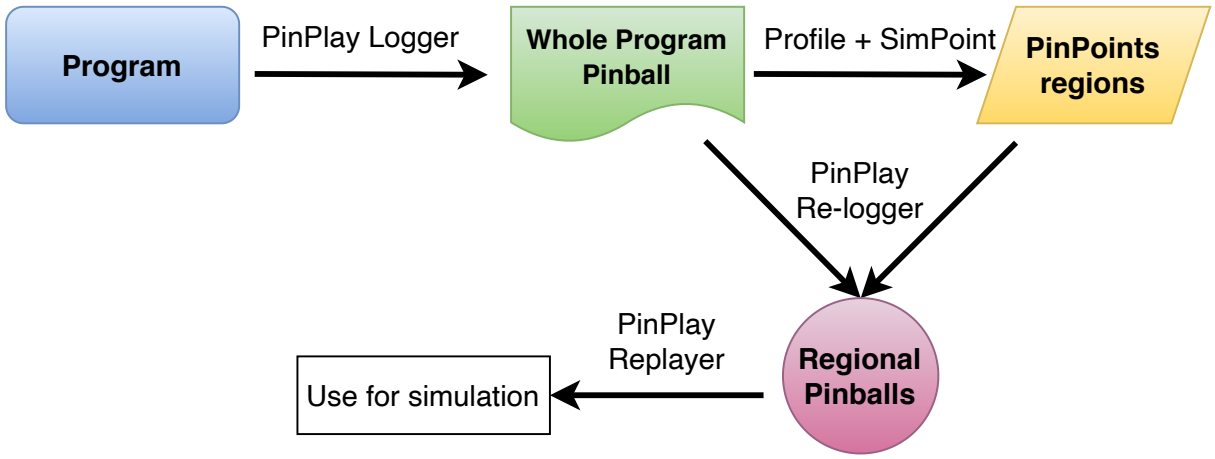


Figure 5.1. PinPoints workflow.

PinPoints [131] combines Pin with SimPoint. It uses the whole program pinball recorded with PinPlay as input to SimPoint and produces simulation points in the form of regional pinballs. Figure 5.1 shows the workflow for capturing regional pinballs. There are a number of advantages with using pinballs produced by PinPoints. They are OS independent, provide reproducible and deterministic simulation results, and can be shared among researchers.

5.2.3. ELFies

ELFies [132] or *pinballs2elf* is a tool-chain that converts a pinball into an ELF executable, which can be run natively on Linux and without extra overhead. This is useful when validating the regions produced by SimPoint. To validate SimPoint, one needs to compute the error rate by calculating (1) the weighted average of CPIs of the regional pinballs, and (2) the CPI of the whole program pinball. Typically, the CPIs are collected by running the pinballs through a simulator. While step 1 can be quite fast as the slices are relatively short, step 2 requires the simulation of the entire program, which can be unrealistically

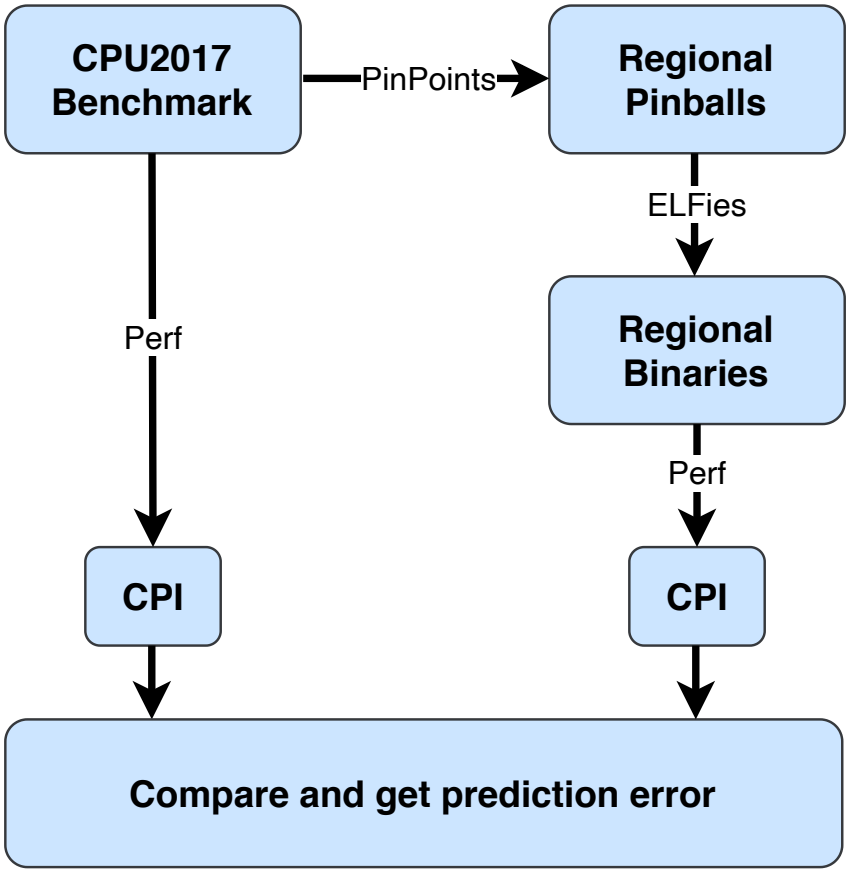


Figure 5.2. Validation workflow.

time consuming. In our experience, the fastest whole-program simulation in the CPU2017 suite using the Sniper architectural simulator finished in 28 days when running on a modern Intel-based server. The ELF executables generated by ELFies significantly reduce the validation time by allowing one to execute regional pinballs natively and use hardware counters to calculate the CPI. In this way we are able to quickly validate our pinballs by comparing the CPI obtained by hardware performance counters of the original application and the ELFies generated from the regional pinballs.

5.3. Methodology

Figure 5.2 shows the entire workflow for our experiments. We use PinPoints (with Pin version 3.7) to generate regional pinballs of the SPEC CPU2017 SPECspeed benchmarks (both INT and FP) with the *ref* input size. All benchmarks are compiled on an Intel Xeon™ E5-2695 V3 processor (Haswell, 14 cores, 2.3 GHz, 32 kB L1 cache, 256 kB L2 cache, 35 MB last-level cache) running Linux kernel 4.18 using GCC 8. We disable the OpenMP compilation flags but keep all other optimization flags consistent with the recommended example provided by the CPU2017 distribution and target the 64-bit ISA. As for tuning the parameters of SimPoint, we use a *maxk* value of 32 (32 maximum regions), a *slice_size* of 100 million instructions, and a *warmup_length* of 300 million instructions. We leave out some FP benchmarks because they could take months, as the logging and replaying process can incur as much as 200× slowdown compared to native execution [134]. Table 5.1 summarizes the SimPoints and global dynamic instruction counts of the benchmarks we run. Note that the executions of *600.perlbench_s*, *602.gcc_s*, *625.x264_s*, *657.xz_s*, and *603.bwaves_s* have multiple steps, with each step taking in a different input. We consider each step as a separate benchmark, denoted by the number index suffixes after the dot in the **Benchmark** column. The **90 Percentile SimPoints** column shows the least number of SimPoints needed to reach a cumulative weight of 0.9 or more.

To evaluate the pinballs, we use ELFies to convert the region pinballs into native executables and use Linux `perf` to read hardware performance counters and calculate the CPIs of the individual regions. We then take measurements on a system with an Intel Core™ i7-8700 processor (Skylake, 6 cores, 3.2 GHz, 32 kB L1 cache, 256 kB L2 cache, 8 MB last-level cache) and 16 GB of DRAM, running Ubuntu 18.04 with a 4.18 Linux kernel.

Table 5.1. CPU2017 SimPoints.

Benchmark	# of SimPoints	90 Percentile SimPoints	Instructions (billion)
SPEC Int			
600.perlbench_s.0	16	9	1961.85
600.perlbench_s.1	12	4	1150.03
600.perlbench_s.2	18	9	1109.25
602.gcc_s.0	19	6	4721.74
602.gcc_s.1	22	12	1412.38
602.gcc_s.2	23	11	1350.69
605.mcf_s	29	17	4066.58
620.omnetpp_s	3	2	5951.38
623.xalancbmk_s	23	18	9226.90
625.x264_s.0	25	16	1522.16
625.x264_s.1	20	14	5515.79
625.x264_s.2	15	9	5560.37
631.deepsjeng_s	6	5	4848.04
641.leela_s	21	13	13728.15
648.exchange2_s	19	15	10596.43
657.xz_s.0	18	10	12910.01
657.xz_s.1	17	11	8018.37
Average	18	10.65	5508.83
SPEC FP			
603.bwaves_s.0	27	5	59025.56
603.bwaves_s.1	31	6	55580.46
607.cactuBSSN_s	31	6	32636.33
619.lbm_s	16	8	18501.50
621.wrf_s	27	20	114200.20
627.cam4_s	21	13	38725.31
628.pop2_s	20	12	95140.64
644.nab_s	16	7	29067.15
649.fotonik3d_s	23	10	123075.22
654.roms_s	30	23	123075.22
Average	24.2	11	61211.27

Using a system for measurement that is different than the system used for collecting the pinballs reflects the actual way that the pinballs we release will be used in the wild.

We measure each region or whole program instance ten times and calculate the average CPI. The ELFies are generated with the hardware performance counters `PERF_COUNT_HW_CPU_CYCLES` and `PERF_COUNT_HW_CPU_INSTRUCTIONS` found in `/usr/include/linux/perf_event.h`. To measure whole program performance, we use `perf record` and sample the *cpu_clk_unhalted.thread* and *inst_retired.any* hardware counters. To minimize OS noise and DVFS effects, we turn off Intel Hyper-Threading, Turbo Boost, and SpeedStep, and run all experiments at the `init 3` runlevel. We use Sniper 7.4 with two versions of Pin (3.7 and 3.11) to test the generated pinballs and verify that they can run without errors. We calculate the region predicted CPIs and prediction errors according to the following equations:

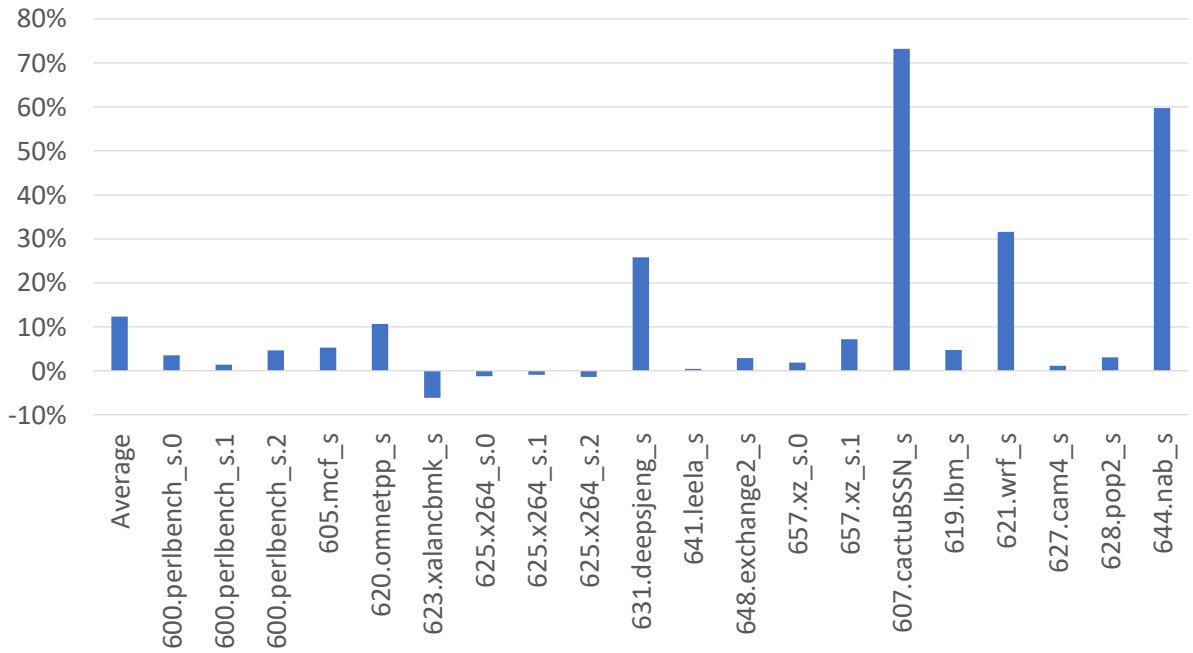
$$predicted\_CPI = \sum_{i=1}^{num\_simpoints} CPI_i \times weight_i$$

$$pred\_error = \frac{|whole\_program\_CPI - predicted\_CPI|}{whole\_program\_CPI}$$

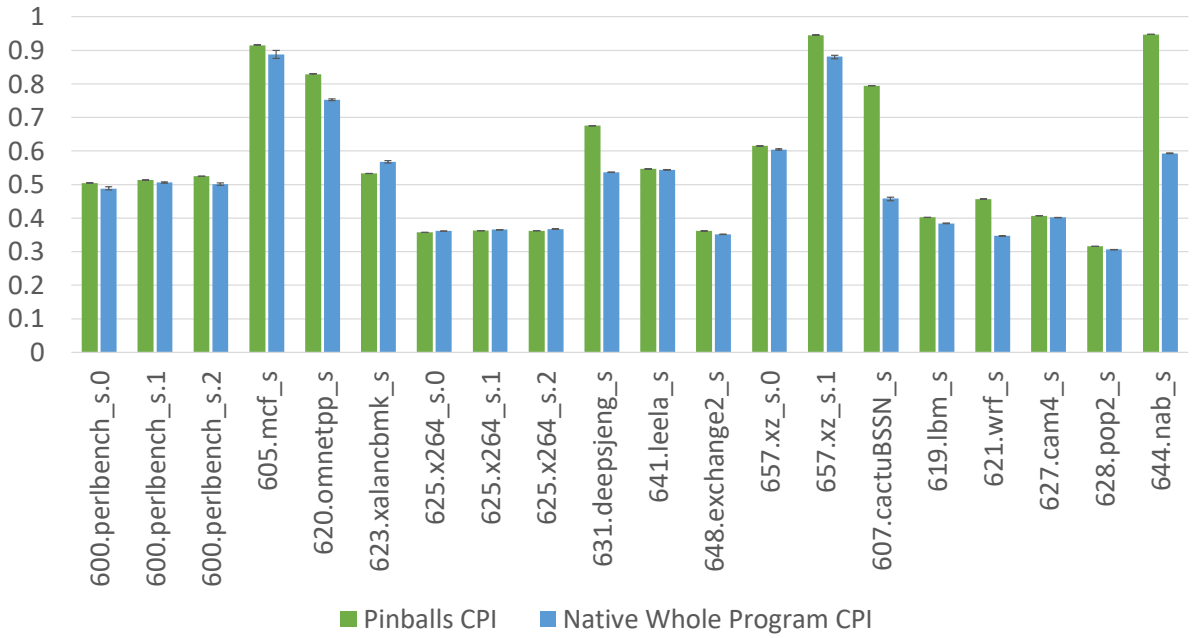
5.4. Results

Figure 5.3 shows the prediction errors and CPI comparisons of our pinballs against native runs of the benchmarks. The average absolute prediction error rate across all benchmarks is 12%. For most benchmarks, our pinballs represent native execution well, and we can expect the pinballs to reasonably represent the performance characteristics of their native counterparts, while significantly reducing simulation time.

Benchmarks *631.deepsjeng_s*, *607.cactuBSSN_s*, *621.wrf_s*, and *644.nab_s* have error rates above 25%. We posit that this is caused by the difference between dynamically linked libraries on the machine used to collect the pinballs and the machine used to collect runtime execution CPIs. To investigate this, we statically link *631.deepsjeng_s*, *607.cactuBSSN_s*, and *644.nab_s*, and compare the CPIs calculated by the pinball regions

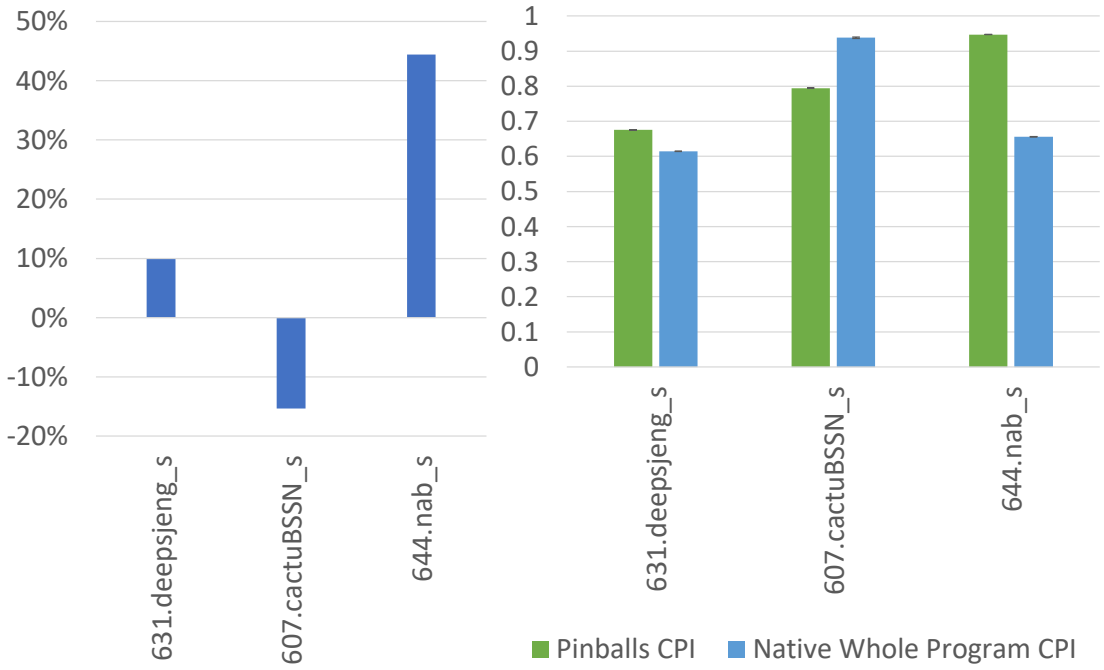


(a) Prediction errors (%).



(b) CPI comparison.

Figure 5.3. Pinball validation results.



(a) Prediction errors (%). (b) CPI comparison.

Figure 5.4. Validation results for statically linked binaries.

with the whole-program CPIs of the statically compiled benchmarks. We show the prediction errors and CPI comparisons in Figure 5.4. Statically linking *621.wrf_s* requires `libgfortran.a`, which is not available in our version of RHEL, thus we exclude this application from further investigation. Using statically linked binaries reduces the absolute CPI error of these three benchmarks by an average of 29.7%. This also brings the average error across the entire benchmark suite down to less than 8%.

Unfortunately, licensing restrictions prevent us from releasing statically linked versions of *631.deepsjeng_s*, *607.cactuBSSN_s*, and *644.nab_s*. Users of our pinballs (or any other pinball releases, for that matter) should be aware of the relatively high errors when using dynamically linked versions of these three benchmarks. Alternatively, users can statically

compile their own versions of these three benchmarks on their own platforms and recollect the corresponding pinballs, or at least verify the pinballs' accuracy on their machines by comparing the calculated CPI with our published data.

Although further in-depth tuning of PinPoint parameters to generate more accurate pinballs for these four benchmarks is possible, that is a topic for future work.

5.5. Conclusions

In this chapter we announce to the computer architecture community the public release of validated pinballs for SPEC CPU2017 SPECspeed benchmarks, and share the details of our pinball-collecting process, statistics, and validation results. Our validation shows that the average absolute CPI error rate of our pinballs is 12% for dynamically linked benchmarks. We also discover that differences exist between dynamically linked and statically linked benchmarks when their CPIs are compared against the CPI results predicted from pinballs. For benchmarks that exhibit high errors when compiled dynamically, we find that compiling them statically can reduce the CPI error rate by 29.7%. In particular, when statically linking *631.deepsjeng_s*, *607.cactuBSSN_s*, and *644.nab_s*, and dynamically linking all other benchmarks, we estimate an average absolute CPI error of less than 8% across the entire SPEC SPU2017 suite. The link to our pinballs repository can be found on our lab's website, PARAG@N, under "Pinballs" in the Artifacts section at http://paragon.cs.northwestern.edu/#Artifacts.

CHAPTER 6

Looking Ahead: Future Work

In this chapter we discuss some of the limitations of the work presented in this thesis and propose future work to conceptualize the directions to continue pursuing.

6.0.1. LC⚡DC : Impact on the Application Level and Comparison with Existing Energy Proportional Policies

In Section 2.6 our results show that LC⚡DC on average increases packet latency by 6%. The question remains, however, on how the power gating latency can be translated to performance impact at the datacenter level and to appliications. A sensitivity study on the power gating delay and the impact to network performance would be beneficial. Networking impact metrics can include packet latency and application metrics can include round-trip latency, query per second, etc. This study can build a stronger case on why LC⚡DC needs to hide the power gating latency. A quantitative analysis and evaluation of LC⚡DC compared to existing energy proportional policies (ElasticTree [79], DREAM [185], CARPO [172], etc) can help argue why these works cannot be directly translated to optical interconnects and why LC⚡DC is necessary.

Future work on LC⚡DC evaluation can use a more sophisticated network simulator that is widely accepted by the networking community. One such simulator is the NS-3 [141], a

popular discrete-event network simulator. It supports custom topologies, custom protocols, and user-defined traffic and meets the needs for evaluating data center applications on LC⚡DC .

6.0.2. O-PCM Cache

In Chapter 4 we performed a design space exploration of O-PCM caches. However the architectural details of an O-PCM cache is not clear yet. Below summarizes some key points that need to be considered when exploring a complete design:

- Cell size and peripheral circuits. As O-PCM cells are still a very new technology, their cell sizes are subject to change with rapid developments of related research. This directly affects the capacity of an O-PCM cache. Depending on the density, 3D stacking might also be required. The cell size also impacts latency as we will need to consider the dimensions when calculating optical round-trip time. The peripheral circuits used in Pho$ (Section 3.3) are in theory compatible with O-PCM but newer technologies can present more optimized architectures. It is important to work with our collaborators on these aspects.
- Similar to Pho$, an optical network should be designed and evaluation be performed. For an O-PCM LLC to be practical, the original Pho$Net can be extended to also connect the LLC. As can be seen from Section 3.5.2, the design of an optical NoC can greatly affect its power and feasibility. Correspondingly, an analysis on why or why not the network protocol should be expanded can also be conducted.

- Energy efficiency evaluation. Energy remains a focal point in all architectural research, and it is necessary for an energy evaluation of the O-PCM LLC for it to be desirable to future architectures.

6.0.3. PinPoints Energy

While the pinballs presented in Chapter 5 represent whole program performance well, our verification shows that the energy of pinballs do not actually reflect that of the whole program. SimPoint [150] was developed with representing performance characteristics in mind, and naturally is not applicable to energy evaluation. Research that tries to extend this work can exploit this deficiency in SimPoint and develop a new methodology that preferably can use small program regions to represent both performance and energy of the whole program. Energy is of course highly hardware dependent. We do envision that such a new methodology can be very impactful to not only the architecture community, but also research in all of computing.

CHAPTER 7

Related Work

7.1. Data Center Networks

Heller *et al.* propose ElasticTree [79], a power manager that dynamically adjusts the set of active links and switches to satisfy data center traffic while saving energy. Ananthanarayanan and Katz [7] present a switch design that estimates traffic to power down ports when possible. Zhou *et al.* propose DREAM [185], which tries to slice TCP flows across multiple paths and dynamically adjusting path selection. Wang *et al.* present CARPO [172], an adaptive link rate solution that consolidates traffic flows according to their correlations. Kandula *et al.* [88] add on-demand "flyways" (wireless and wired) links to tackle network congestions and to provide extra network capacity over a base average network. Ghobadi *et al.* [65] propose ProjectToR which uses free-space optics for inter-rack communications.

Laser gating has been proposed before for on-chip interconnects [45, 48, 46, 50], which however do not present the complexities that laser gating on data center networks has to address (e.g., fast CDR, fast power-up of commercial transceiver electronics, control plane updates at the switch, and kernel modification to set up an early warning system). A previous proposal for laser control in on-chip interconnects [50] was extended to data centers but only as a conceptual study; no evaluation was performed on traffic similar to

modern large-scale data centers (only a university data center traffic trace was used), and the feasibility of the technique was not demonstrated.

Helios [59] identifies the subset of traffic best suited to circuit switching and dynamically reconfigures the network topology at runtime based on shifting communication patterns.

7.2. Optical RAM

In the 1990s, Guilfoyle *et al.* [68] first introduced photonic random optical memory for faster and less power consuming accesses of the main memory and Chiarulli *et al.* [33] proposed an opto-electronic memory hierarchy for similar purposes. Pleros *et al.* [137], Alexoudi *et al.* [4], and Nozaki *et al.* [123] propose using different materials to build optical SRAM cells with memory operations. Alexoudi *et al.* [6], Vagionas *et al.* [168], Maniotis *et al.* [108, 109], and Pleros *et al.* [138] present a series of work that propose physical-level optical cache architectures and integrating them into processors with simple interconnects. Fotouhi *et al.* [62] exploit silicon-photonic interconnects in chiplet-based systems to build uniform memory architectures.

7.3. Optical Networks-on-Chip

Several optical NoC works try to integrate photonics into on-chip communication. Corona [170] implements an MWSR optical crossbar where nodes contend for an optical token before they are allowed to transmit data, allowing the arbitration of a shared channel. Vantrease *et al.* [169] propose a token slot arbitration mechanism to overcome the shortcomings of Corona by dividing the data channel into different slots and assigning a token to each slot. FireFly [127] partitions optical R-SWMR crossbars to connect clusters

of electrical mesh networks. FeatherWeight [126] improves network fairness by assigning a quota (maximum number of tokens) to each node with an epoch. ProLaser [49] segregates the data channel and the control channel and manages them separately in order to save laser power. LumiNOC [98] aims to reduce waveguide loss and laser power by dividing a large NoC into small subnets.

7.4. Optical PCM and NVM Caches

Phase change memory as a potential building block for on-chip caches has been previously researched. Joo *et al.* [85] introduce a number of proposals to build PCM caches, including write distribution among cells and data inversion tactics to improve write endurance. Dong *et al.* [55] develop NVSim, which is a non-volatile memory simulator that estimates access time, energy, and area for different NVM technologies, albeit not for O-PCM. Hankin *et al.* [77] present a set of heuristics for modeling NVM-based LLCs and evaluated several designs using different NVM technologies. Their results show that almost all NVM-based LLCs perform worse than electronic baselines with similar cache capacity and area.

7.5. Statically Based Workload Characterizations

A number of methods and tools have been proposed and implemented to help reduce the simulation time of benchmarks in architectural research. SimPoint [150] is an algorithm that automatically finds small regions of a program that can represent the architectural characteristics of the whole program, thereby reducing simulation time. Patil *et al.* and Intel develop PinPoints [131] which uses SimPoint and Pin [105] to automatically produces shareable and OS-independent regional pinballs that can be simulated by

simulators like Sniper [28, 29] and ZSim [147]. Patil *et al.* also develops ELFies [132] that can convert pinballs to individual ELF binaries to support faster pinball validation.

Several works have also analyzed the SPEC CPU2017 suite using different methods. Panda *et al.* [128] conduct a detailed architectural analysis of the CPU2017 suite and compare against the CPU2006 suite. Singh *et al.* [153] use methods like dynamic binary instrumentation, native hardware performance counters, and OS based tools to conduct a comprehensive study of the CPU2017 suite. Wu *et al.* [177, 2] are the first to provide publicly available pinballs for SPEC CPU2017 but the pinballs were unusable in many scenarios.

CHAPTER 8

Conclusions

Breakthroughs in optical interconnects and silicon photonics have shifted computer architecture into new terratories. As Moore's Law and Dennard scaling continue to reach the end of their life cycles, opportunities have arisen to take advantage of the low latency, high bandwidth, and high energy efficiency of photonics in modern computer architecture.

Existing optical interconnects have been a relatively overlooked aspect in optimizing for energy-proportional data centers. Laser gating has been proposed as a way to reduce laser power when the network is under low utilization, but the turn on delay remains an issue, and the feasibility of the method must be studied on all levels. We proposed LC⚡DC , an energy-proportional data center network architecture that is co-designed through the laser, switch, and OS levels. We performed a comprehensive study on the feasibility of LC⚡DC on these levels. We designed a traffic generator that closely resembles real world data center traffic. We demonstrated that LC⚡DC saves on average 60% optical transceiver energy, and up to 27% overall data center energy, at the expense of only 6% additional packet latency.

We then turn our focus to on-chip photonics and try to break through the memory wall with emerging PhC memory cells and O-PCM cells. We emphasized that replacing traditional electronic components in a processor with emerging optical components is not a simple plug-and-play process, and the entire system stack needs to be rethinked and redesigned. We introduced Pho$, which replaces traditional electronic L1 caches with a

shared, 2-cycle read latency, multi MB capacity optical L1 cache. Our results indicate that Pho$ is up to 3.89× faster (1.41× on average) than a traditional electronic multicore processor while saving up to 90% energy-delay product (31% on average). We also improved upon optical NoC designs by designing a hybrid MWSR/R-SWMR network architecture, which consumes up to 70% less power than directly applying existing topologies under realistic assumptions. Lastly we extended the electronic LLC in Pho$ with a cache built using optical phase change memory. Our design space exploration showed the potentials of such an architecture. Despite the long write latency of O-PCM, our design achieves similar performance to Pho$ but also enables non-volatility at the LLC level.

While working on this thesis, we also took the time to generate our own pinballs for simulation use. We verified that our regional pinballs of SPEC CPU2017 represent the whole program benchmarks of the suite with only 12% error rate, which can be improved to 8% by dynamically linking some benchamarks. We also released the pinballs for other researchers to use.

References

[1] 2021. SPEC CPU 2017. https://www.spec.org/cpu2017/

[2] 2021. SPEC CPU2017 and Simulation. https://www.spec.org/cpu2017/research/simpoint.html

[3] Dennis Abts, Michael R. Marty, Philip M. Wells, Peter Klausler, and Hong Liu. 2010. Energy proportional datacenter networks. In *37th International Symposium on Computer Architecture (ISCA 2010), June 19-23, 2010, Saint-Malo, France*. ACM, 338–347. https://doi.org/10.1145/1815961.1816004

[4] Theonitsa Alexoudi, Dimitrios Fitsios, Alexandre Bazin, Paul Monnier, Rama Raj, Amalia Miliou, George T. Kanellos, Nikos Pleros, and Fabrice Raineri. 2016. III-V-on-Si Photonic Crystal Nanocavity Laser Technology for Optical Static Random Access Memories. *IEEE J. Sel. Topics Quantum Electron.* 22, 6 (2016), 295–304. https://doi.org/10.1109/JSTQE.2016.2593636

[5] Theonitsa Alexoudi, George T. Kanellos, and Nikos Pleros. 2020. Optical RAM and integrated optical memories: a survey. *Light: Science & Applications* 9, 1 (2020), 1–16. https://doi.org/10.1038/s41377-020-0325-9

[6] Theonitsa Alexoudi, Sotiris Papaioannou, George T. Kanellos, Amalia Miliou, and Nikos Pleros. 2013. Optical Cache Memory Peripheral Circuitry: Row and Column Address Selectors for Optical Static RAM Banks. *J. Lightw. Tech.* 31, 24 (2013), 4098–4110. https://doi.org/10.1109/JLT.2013.2286529

[7] Ganesh Ananthanarayanan and Randy H. Katz. 2008. Greening the Switch. In *Workshop on Power Aware Computing and Systems, HotPower 2008, December 7, 2008, San Diego, CA, USA, Proceedings*. USENIX Association. https://www.usenix.org/conference/hotpower-08/greening-switch

[8] David G. Andersen and Steven Swanson. 2010. Rethinking Flash in the Data Center. *IEEE Micro* 30, 4 (2010), 52–54. https://doi.org/10.1109/MM.2010.71

[9] Arista. 2020. *7050SX Series 10/40G Data Center Switches Data Sheet*. Arista. https://www.arista.com/assets/data/pdf/Datasheets/7050SX-128_64_Datasheet_S.pdf

[10] Amir H. Atabaki, Sajjad Moazeni, Fabio Pavanello, Hayk Gevorgyan, Jelena Notaros, Luca Alloatti, Mark T. Wade, Chen Sun, Seth A. Kruger, Huaiyu Meng, Kenaish Al Qubaisi, Imbert Wang, Bohan Zhang, Anatol Khilo, Christopher V. Baiocco, Miloš A. Popović, Vladimir M. Stojanović, and Rajeev J. Ram. 2018. Integrating photonics with silicon nanoelectronics for the next generation of systems on a chip. *Nature* 556, 7701 (2018), 349–354. https://doi.org/10.1038/s41586-018-0028-z

[11] Rajeev Balasubramonian, Andrew B. Kahng, Naveen Muralimanohar, Ali Shafiee, and Vaishnav Srinivas. 2017. CACTI 7: New Tools for Interconnect Exploration in Innovative Off-Chip Memories. *ACM Trans. Arch. Code Opt.* 14, 2 (2017), 14:1–14:25. https://doi.org/10.1145/3085572

[12] Hitesh Ballani, Paolo Costa, Raphael Behrendt, Daniel Cletheroe, István Haller, Krzysztof Jozwik, Fotini Karinou, Sophie Lange, Kai Shi, Benn Thomsen, and Hugh Williams. 2020. Sirius: A Flat Datacenter Network with Nanosecond Optical Switching. In *SIGCOMM '20: Proceedings of the 2020 Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication, Virtual Event, USA, August 10-14, 2020*. ACM, 782–797. https://doi.org/10.1145/3387514.3406221

[13] Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle. 2013. The datacenter as a computer: An introduction to the design of warehouse-scale machines. In *Synthesis lectures on computer architecture*. Vol. 8. Morgan & Claypool Publishers, 1–154. https://doi.org/10.2200/S00516ED2V01Y201306CAC024

[14] Luiz André Barroso and Urs Hölzle. 2007. The Case for Energy-Proportional Computing. *Computer* 40, 12 (2007), 33–37. https://doi.org/10.1109/MC.2007.443

[15] Christopher Batten, Ajay Joshi, Jason Orcutt, Anatol Khilo, Benjamin Moss, Charles W. Holzwarth, Miloš A. Popovic, Hanqing Li, Henry I. Smith, Judy L. Hoyt, et al. 2009. Building Many-Core Processor-to-DRAM Networks with Monolithic CMOS Silicon Photonics. *IEEE Micro* 29, 4 (2009), 8–21. https://doi.org/10.1109/MM.2009.60

[16] Ferdinando Bedeschi, Rich Fackenthal, Claudio Resta, Enzo Michele Donze, Meenatchi Jagasivamani, Egidio Cassiodoro Buda, Fabio Pellizzer, David W Chow, Alessandro Cabrini, Giacomo Matteo Angelo Calvi, et al. 2009. A bipolar-selected

phase change memory featuring multi-level cell storage. *IEEE J. Solid-State Circuits* 44, 1 (2009), 217–227. https://doi.org/10.1109/JSSC.2008.2006439

[17] Majed Valad Beigi and Gokhan Memik. 2016. TAPAS: Temperature-aware Adaptive Placement for 3D Stacked Hybrid Caches. In *2nd International Symposium on Memory Systems, MEMSYS 2016, Alexandria, VA, USA, October 3-6, 2016*. ACM, 415–426. https://doi.org/10.1145/2989081.2989085

[18] Theophilus Benson, Aditya Akella, and David A. Maltz. 2010. Network traffic characteristics of data centers in the wild. In *Proceedings of the 10th ACM SIGCOMM Internet Measurement Conference, IMC 2010, Melbourne, Australia - November 1-3, 2010*. ACM, 267–280. https://doi.org/10.1145/1879141.1879175

[19] Christian Bienia. 2011. *Benchmarking Modern Multiprocessors*. Ph. D. Dissertation. Princeton University. https://parsec.cs.princeton.edu/publications/bienia11benchmarking.pdf

[20] Christian Bienia, Sanjeev Kumar, Jaswinder Pal Singh, and Kai Li. 2008. The PARSEC benchmark suite: characterization and architectural implications. In *17th International Conference on Parallel Architectures and Compilation Techniques, PACT 2008, Toronto, Ontario, Canada, October 25-29, 2008*. ACM, 72–81. https://doi.org/10.1145/1454115.1454128

[21] Shekhar Borkar. 2013. Exascale Computing - A Fact or a Fiction?. In *27th IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2013, Cambridge, MA, USA, May 20-24, 2013*. IEEE Computer Society, 3. https://doi.org/10.1109/IPDPS.2013.121

[22] Shekhar Borkar. 2015. Node Architecture: From Present Technology to Future Exascale Nodes. In *Proceedings of the 2015 ISC High Performance Conference* (Frankfurt, Germany). http://www.nextplatform.com/2015/08/12/future-systems-intel-fellow-conjures-the-perfect-exascale-machine/

[23] Shekhar Borkar and Andrew A. Chien. 2011. The future of microprocessors. *Comm. ACM* 54, 5 (2011), 67–77. https://doi.org/10.1145/1941487.1941507

[24] David M. Brooks, Vivek Tiwari, and Margaret Martonosi. 2000. Wattch: a framework for architectural-level power analysis and optimizations. In *27th International Symposium on Computer Architecture (ISCA 2000), June 10-14, 2000, Vancouver, BC, Canada*. IEEE Computer Society, 83–94. https://doi.org/10.1109/ISCA.2000.854380

[25] James Bucek, Klaus-Dieter Lange, and Jóakim von Kistowski. 2018. SPEC CPU2017: Next-Generation Compute Benchmark. In *Companion of the 2018 ACM/SPEC International Conference on Performance Engineering, ICPE 2018, Berlin, Germany, April 09-13, 2018*. ACM, 41–42. https://doi.org/10.1145/3185768.3185771

[26] Geoffrey W Burr, Matthew J Breitwisch, Michele Franceschini, Davide Garetto, Kailash Gopalakrishnan, Bryan Jackson, Bülent Kurdi, Chung Lam, Luis A Lastras, Alvaro Padilla, et al. 2010. Phase change memory technology. *Journal of Vacuum Science & Technology B, Nanotechnology and Microelectronics: Materials, Processing, Measurement, and Phenomena* 28, 2 (2010), 223–262. https://doi.org/10.1116/1.3301579

[27] Hyunil Byun, Jinkwon Bok, Kwansik Cho, Keunyeong Cho, Hanmei Choi, Jinyong Choi, Sanghun Choi, Sangdeuk Han, Seokyong Hong, Seokhun Hyun, et al. 2014. Bulk-Si photonics technology for DRAM interface. *Photonics Research* 2, 3 (2014), A25–A33. https://doi.org/10.1364/PRJ.2.000A25

[28] Trevor E. Carlson, Wim Heirman, and Lieven Eeckhout. 2011. Sniper: exploring the level of abstraction for scalable and accurate parallel multi-core simulation. In *Conference on High Performance Computing Networking, Storage and Analysis, SC 2011, Seattle, WA, USA, November 12-18, 2011*. ACM, 52:1–52:12. https://doi.org/10.1145/2063384.2063454

[29] Trevor E. Carlson, Wim Heirman, Stijn Eyerman, Ibrahim Hur, and Lieven Eeckhout. 2014. An Evaluation of High-Level Mechanistic Core Models. *ACM Trans. Arch. Code Opt.* 11, 3 (2014), 28:1–28:25. https://doi.org/10.1145/2629677

[30] Sujay Charania, Niels Neumann, Sebastian Killge, Felix Winkler, Zaid Al-Husseini, Laszlo Szilagyi, Ronny Henker, Frank Ellinger, Dirk Plettemeier, and Johann W Bartha. 2020. Design, Fabrication, and Comparison of 3D Multimode Optical Interconnects on Silicon Interposer. *Journal of Lightwave Technology* 38, 13 (2020), 3454–3460. https://doi.org/10.1109/JLT.2020.2971394

[31] Guoqing Chen, Hui Chen, Mikhail Haurylau, Nicholas A. Nelson, David H. Albonesi, Philippe M. Fauchet, and Eby G. Friedman. 2007. Predictions of CMOS compatible on-chip optical interconnect. *Integration* 40, 4 (2007), 434–446. https://doi.org/10.1016/j.vlsi.2006.10.001

[32] Lizhong Chen, Lihang Zhao, Ruisheng Wang, and Timothy Mark Pinkston. 2014. MP3: Minimizing performance penalty for power-gating of Clos network-on-chip. In *20th IEEE International Symposium on High Performance Computer Architecture,*

HPCA 2014, Orlando, FL, USA, February 15-19, 2014. IEEE Computer Society, 296–307. https://doi.org/10.1109/HPCA.2014.6835940

[33] Donald M. Chiarulli and Steven P. Levitan. 1996. Optoelectronic-cache memory system architecture. *Applied Optics* 35, 14 (1996), 2449–2456. https://doi.org/10.1364/AO.35.002449

[34] Shang-Tse Chuang, Ashish Goel, Nick McKeown, and Balaji Prabhakar. 1999. Matching output queueing with a combined input/output-queued switch. *IEEE J. Sel. Areas Commun.* 17, 6 (1999), 1030–1039. https://doi.org/10.1109/49.772430

[35] Kari Clark, Hitesh Ballani, Polina Bayvel, Daniel Cletheroe, Thomas Gerard, István Haller, Krzysztof Jozwik, Kai Shi, Benn Thomsen, Philip M. Watts, Hugh Williams, Georgios Zervas, Paolo Costa, and Zhixin Liu. 2018. Sub-Nanosecond Clock and Data Recovery in an Optically-Switched Data Centre Network. In *European Conference on Optical Communication, ECOC 2018, Rome, Italy, September 23-27, 2018*. IEEE, 1–3. https://doi.org/10.1109/ECOC.2018.8535333

[36] Kari A Clark, Daniel Cletheroe, Thomas Gerard, Istvan Haller, Krzysztof Jozwik, Kai Shi, Benn Thomsen, Hugh Williams, Georgios Zervas, Hitesh Ballani, et al. 2020. Synchronous subnanosecond clock and data recovery for optically switched data centres using clock phase caching. *Nature Electronics* 3, 7 (July 2020), 426–433. https://doi.org/10.1038/s41928-020-0423-y

[37] International Roadmap Committee. 2015. International Technology Roadmap For Semiconductors 2.0 2015 Edition. https://www.semiconductors.org/wp-content/uploads/2018/06/0_2015-ITRS-2.0-Executive-Report-1.pdf

[38] International Roadmap Committee. 2020. International Roadmap for Devices and Systems (IRDS) 2020 Edition. https://irds.ieee.org/editions/2020

[39] SFF Committee. 2013. SFF-8431 Specifications for Enhanced Small Form Factor Pluggable Module SFP+ Revision 4.1 + Addendum. http://www.fluxlight.com/content/Tech-Docs/SFPPlus%20MSA.PDF

[40] Intel Corporation. 2021. Intel 64 and IA-32 Architectures Software Developer's Manual Volume 3(3A, 3B, 3C & 3D): System Programming Guide. https://software.intel.com/content/dam/develop/public/us/en/documents/325384-sdm-vol-3abcd.pdf

[41] William James Dally and Brian Patrick Towles. 2004. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishing Inc. https://doi.org/10.5555/2821589

[42] Johan De Gelas and Ian Cutress. 2017. Sizing Up Servers: Intel's Skylake-SP Xeon versus AMD's EPYC 7000 - The Server CPU Battle of the Decade? https://www.anandtech.com/show/11544/intel-skylake-ep-vs-amd-epyc-7000-cpu-battle-of-the-decade/13

[43] Dell Inc. 2012. *PowerEdge R710 Technical Guide*. Dell Inc. https://i.dell.com/sites/doccontent/business/solutions/engineering-docs/en/Documents/server-poweredge-r710-tech-guidebook.pdf Ver. 4.0.

[44] Delta Electronics 2012. *SFP+ 10GEPON Symmetrical ONU Transceiver*. Delta Electronics. https://datasheet.datasheetarchive.com/originals/library/Datasheets-ZIHA5/DSAZIHA100086647.pdf Rev. S1.

[45] Yigit Demir and Nikos Hardavellas. 2014. EcoLaser: An Adaptive Laser Control for Energy Efficient On-Chip Photonic Interconnects. In *IEEE/ACM International Symposium on Low-Power Electronics and Design, ISLPED 2014, La Jolla, CA, USA, August 11-13, 2014*. ACM, 3–8. https://doi.org/10.1145/2627369.2627620

[46] Yigit Demir and Nikos Hardavellas. 2014. LaC: Integrating laser control in a photonic interconnect. In *Proceedings of the IEEE Photonics Conference (IPC)*. IEEE, 28–29. https://doi.org/10.1109/IPCon.2014.6995193

[47] Yigit Demir and Nikos Hardavellas. 2015. Parka: Thermally Insulated Nanophotonic Interconnects. In *9th International Symposium on Networks-on-Chip, NOCS 2015, Vancouver, BC, Canada, September 28-30, 2015*. ACM, 1–8. https://doi.org/10.1145/2786572.2786597

[48] Yigit Demir and Nikos Hardavellas. 2015. Towards Energy-Efficient Photonic Interconnects. In *Proceedings of Optical Interconnects XV, SPIE Photonics West* (San Francisco, CA), Vol. 9368. SPIE, 185–196. https://doi.org/10.1117/12.2080496

[49] Yigit Demir and Nikos Hardavellas. 2016. Energy-proportional photonic interconnects. *ACM Trans. Arch. Code Opt.* 13, 4 (2016), 54:1–54:26. https://doi.org/10.1145/3018110

[50] Yigit Demir and Nikos Hardavellas. 2016. SLaC: Stage laser control for a flattened butterfly network. In *2016 IEEE International Symposium on High Performance Computer Architecture, HPCA 2016, Barcelona, Spain, March 12-16, 2016*. IEEE Computer Society, 321–332. https://doi.org/10.1109/HPCA.2016.7446075

[51] Yigit Demir, Yan Pan, Seukwoo Song, Nikos Hardavellas, John Kim, and Gokhan Memik. 2014. Galaxy: a high-performance energy-efficient multi-chip architecture using photonic interconnects. In *2014 International Conference on Supercomputing, ICS'14, Muenchen, Germany, June 10-13, 2014*. ACM, 303–312. https://doi.org/10.1145/2597652.2597664

[52] Yigit Demir, Nikos Terzenidis, Haiyang Han, Dimitris Syrivelis, George T. Kanellos, Nikos Hardavellas, Nikos Pleros, Srikanth Kandula, and Fabian Bustamante. 2017. Harnessing path diversity for laser control in data center optical networks. In *2017 IEEE Photonics Society Summer Topical Meeting Series (SUM)*. 113–114. https://doi.org/10.1109/PHOSST.2017.8012676

[53] Robert H. Dennard, Fritz H. Gaensslen, V. Leo Rideout, Ernest Bassous, and Andre R. LeBlanc. 1974. Design of ion-implanted MOSFET's with very small physical dimensions. *IEEE J. Solid-State Circuits* 9, 5 (1974), 256–268. https://doi.org/10.1109/JSSC.1974.1050511

[54] Fuad E. Doany, Benjamin G. Lee, Daniel M. Kuchta, Alexander V. Rylyakov, Christian Baks, Christopher Jahnes, Frank Libsch, and Clint L. Schow. 2012. Terabit/Sec VCSEL-Based 48-Channel Optical Module Based on Holey CMOS Transceiver IC. *J. Lightw. Tech.* 31, 4 (2012), 672–680. https://doi.org/10.1109/JLT.2012.2217938

[55] Xiangyu Dong, Cong Xu, Yuan Xie, and Norman P. Jouppi. 2012. NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* 31, 7 (2012), 994–1007. https://doi.org/10.1109/TCAD.2012.2185930

[56] Delta Electronics. 2012. *SFP+ 10GEPON Symmetrical ONU Transceiver*. Delta Electronics. https://datasheet.datasheetarchive.com/originals/library/Datasheets-ZIHA5/DSAZIHA100086648.pdf Rev. S3.

[57] Eoptolink. 2015. *EOLX-PR-GET-30 Series Datasheet*. Eoptolink. http://www.eoptolink.com/pdf/EOLX-PR-GET-30.pdf Rev. V1.b.

[58] Xiaobo Fan, Wolf-Dietrich Weber, and Luiz André Barroso. 2007. Power provisioning for a warehouse-sized computer. In *34th International Symposium on Computer*

Architecture (ISCA 2007), June 9-13, 2007, San Diego, California, USA. ACM, 13–23. https://doi.org/10.1145/1250662.1250665

[59] Nathan Farrington, George Porter, Sivasankar Radhakrishnan, Hamid Hajabdolali Bazzaz, Vikram Subramanya, Yeshaiahu Fainman, George Papen, and Amin Vahdat. 2010. Helios: a hybrid electrical/optical switch architecture for modular data centers. In *Proceedings of the ACM SIGCOMM 2010 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, New Delhi, India, August 30 -September 3, 2010*. ACM, 339–350. https://doi.org/10.1145/1851182.1851223

[60] Nathan Farrington, Erik Rubow, and Amin Vahdat. 2009. Data Center Switch Architecture in the Age of Merchant Silicon. In *17th IEEE Symposium on High Performance Interconnects, HOTI 2009, New York, New York, USA, August 25-27, 2009*. IEEE Computer Society, 93–102. https://doi.org/10.1109/HOTI.2009.11

[61] Agner Fog. 2021. The microarchitecture of Intel, AMD and VIA CPUs: An optimization guide for assembly programmers and compiler makers, 2016. https://www.agner.org/optimize/microarchitecture.pdf

[62] Pouya Fotouhi, Sebastian Werner, Jason Lowe-Power, and SJ Ben Yoo. 2019. Enabling scalable chiplet-based uniform memory architectures with silicon photonics. In *International Symposium on Memory Systems, MEMSYS 2019, Washington, DC, USA, September 30 - October 03, 2019*. ACM, 222–334. https://doi.org/10.1145/3357526.3357564

[63] D Gazula, JK Guenter, RH Johnson, GD Landry, AN MacInnes, G Park, JK Wade, JR Biard, and JA Tatum. 2010. Emerging VCSEL technologies at Finisar. In *Vertical-Cavity Surface-Emitting Lasers XIV*, Vol. 7615. International Society for Optics and Photonics, SPIE, 50–57. https://doi.org/10.1117/12.841012

[64] Pawel Gepner and Michal Filip Kowalik. 2006. Multi-Core Processors: New Way to Achieve High System Performance. In *Fifth International Conference on Parallel Computing in Electrical Engineering (PARELEC 2006), 13-17 September 2006, Bialystok, Poland*. IEEE Computer Society, 9–13. https://doi.org/10.1109/PARELEC.2006.54

[65] Monia Ghobadi, Ratul Mahajan, Amar Phanishayee, Nikhil R. Devanur, Janardhan Kulkarni, Gireeja Ranade, Pierre-Alexandre Blanche, Houman Rastegarfar, Madeleine Glick, and Daniel C. Kilper. 2016. ProjecToR: Agile Reconfigurable Data Center Interconnect. In *Proceedings of the ACM SIGCOMM 2016 Conference, Florianopolis, Brazil, August 22-26, 2016*. ACM, 216–229. https://doi.org/10.

1145/2934872.2934911

[66] Google. 2021. Efficiency - Data Centers - Google. https://www.google.com/about/datacenters/efficiency/

[67] Albert G. Greenberg, James R. Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A. Maltz, Parveen Patel, and Sudipta Sengupta. 2009. VL2: a scalable and flexible data center network. In *Proceedings of the ACM SIGCOMM 2009 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Barcelona, Spain, August 16-21, 2009*. ACM, 51–62. https://doi.org/10.1145/1592568.1592576

[68] Peter S. Guilfoyle and Richard V. Stone. 1992. Photonic random optical memory access cache. In *Image Storage and Retrieval Systems*, Vol. 1662. International Society for Optics and Photonics, SPIE, 218–223. https://doi.org/10.1117/12.58506

[69] Xiaochen Guo, Engin Ipek, and Tolga Soyata. 2010. Resistive computation: avoiding the power wall with low-leakage, STT-MRAM based computing. In *37th International Symposium on Computer Architecture (ISCA 2010), June 19-23, 2010, Saint-Malo, France*. ACM, 371–382. https://doi.org/10.1145/1815961.1816012

[70] Amit Hadke, Tony Benavides, Rajeevan Amirtharajah, Matthew Farrens, and Venkatesh Akella. 2008. Design and evaluation of an optical CPU-DRAM interconnect. In *26th International Conference on Computer Design, ICCD 2008, 12-15 October 2008, Lake Tahoe, CA, USA*. IEEE Computer Society, 492–497. https://doi.org/10.1109/ICCD.2008.4751906

[71] Amit Hadke, Tony Benavides, S.J. Ben Yoo, Rajeevan Amirtharajah, and Venkatesh Akella. 2008. OCDIMM: Scaling the DRAM Memory Wall Using WDM Based Optical Interconnects. In *16th Annual IEEE Symposium on High Performance Interconnects (HOTI 2008), 26-28 August 2008, Stanford, CA, USA*. IEEE Computer Society, 57–63. https://doi.org/10.1109/HOTI.2008.25

[72] Parisa Khadem Hamedani, Natalie Enright Jerger, and Shaahin Hessabi. 2014. QuT: A low-power optical Network-on-Chip. In *Eighth IEEE/ACM International Symposium on Networks-on-Chip, NOCS 2014, Ferrara, Italy, September 17-19, 2014*. IEEE, 80–87. https://doi.org/10.1109/NOCS.2014.7008765

[73] Haiyang Han, Theoni Alexoudi, Chris Vagionas, Nikos Pleros, and Nikos Hardavellas. 2021. Pho$: A Case for Shared Optical Cache Hierarchies. In *IEEE/ACM International Symposium on Low Power Electronics and Design, ISLPED 2021, Boston, MA, USA, July 26-28, 2021*. IEEE, 1–6. https://doi.org/10.1109/

ISLPED52811.2021.9502487

[74] Haiyang Han, Theoni Alexoudi, Chris Vagionas, Nikos Pleros, and Nikos Hardavellas. 2022. A Practical Shared Optical Cache with Hybrid MWSR/R-SWMR NoC for Multicore Processors. *ACM J. Emerg. Technol. Comput. Syst.* (2022). https://doi.org/10.1145/3531012

[75] Haiyang Han and Nikos Hardavellas. 2021. CPU2017 Pinpoints. https://nuwildcat.sharepoint.com/:f:/s/MCC-Paragon-Lab/Emb34V55TcFGhS4JiUJFsUYBu4QLozpRBUANywlvqEEy8A

[76] Haiyang Han and Nikos Hardavellas. 2021. Public Release and Validation of SPEC CPU2017 PinPoints. https://doi.org/10.48550/ARXIV.2112.06981

[77] Alexander Hankin, Tomer Shapira, Karthik Sangaiah, Michael Lui, and Mark Hempstead. 2019. Evaluation of Non-Volatile Memory Based Last Level Cache Given Modern Use Case Behavior. In *IEEE International Symposium on Workload Characterization, IISWC 2019, Orlando, FL, USA, November 3-5, 2019*. IEEE, 143–154. https://doi.org/10.1109/IISWC47752.2019.9042051

[78] Wim Heirman, Trevor E. Carlson, Shuai Che, Kevin Skadron, and Lieven Eeckhout. 2011. Using cycle stacks to understand scaling bottlenecks in multi-threaded workloads. In *2011 IEEE International Symposium on Workload Characterization, IISWC 2011, Austin, TX, USA, November 6-8, 2011*. IEEE Computer Society, 38–49. https://doi.org/10.1109/IISWC.2011.6114195

[79] Brandon Heller, Srinivasan Seetharaman, Priya Mahadevan, Yiannis Yiakoumis, Puneet Sharma, Sujata Banerjee, and Nick McKeown. 2010. ElasticTree: Saving Energy in Data Center Networks. In *Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2010, April 28-30, 2010, San Jose, CA, USA*. USENIX Association, 249–264. http://dl.acm.org/citation.cfm?id=1855711.1855728

[80] Mark D. Hill and Michael R. Marty. 2008. Amdahl's Law in the Multicore Era. *Computer* 41, 7 (2008), 33–38. https://doi.org/10.1109/MC.2008.209

[81] Eitetsu Igawa and Satoshi Yoshima. 2012. *Optical Transceiver for 10G-EPON*. Technical Report. Mitsubishi Electric ADVANCE. 10–12 pages. https://www.advance.mitsubishielectric.com/advance/pdf/2012/139_complete.pdf

[82] Bahram Jalali and Sasan Fathpour. 2006. Silicon Photonics. *J. Lightw. Tech.* 24, 12 (2006), 4600–4615. https://doi.org/10.1109/JLT.2006.885782

[83] Djordje Jevdjic, Gabriel H. Loh, Cansu Kaynak, and Babak Falsafi. 2014. Unison Cache: A Scalable and Effective Die-Stacked DRAM Cache. In *47th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2014, Cambridge, United Kingdom, December 13-17, 2014*. IEEE Computer Society, 25–37. https://doi.org/10.1109/MICRO.2014.51

[84] Adwait Jog, Asit K. Mishra, Cong Xu, Yuan Xie, Vijaykrishnan Narayanan, Ravishankar Iyer, and Chita R. Das. 2012. Cache revive: Architecting volatile STT-RAM caches for enhanced performance in CMPs. In *The 49th Annual Design Automation Conference 2012, DAC '12, San Francisco, CA, USA, June 3-7, 2012*. ACM, 243–252. https://doi.org/10.1145/2228360.2228406

[85] Yongsoo Joo, Dimin Niu, Xiangyu Dong, Guangyu Sun, Naehyuck Chang, and Yuan Xie. 2010. Energy- and endurance-aware design of phase change memory caches. In *Design, Automation and Test in Europe, DATE 2010, Dresden, Germany, March 8-12, 2010*. IEEE Computer Society, 136–141. https://doi.org/10.1109/DATE.2010.5457221

[86] Norm Jouppi. 2016. Google supercharges machine learning tasks with TPU custom chip. https://cloudplatform.googleblog.com/2016/05/Google-supercharges-machine-learning-tasks-with-custom-chip.html

[87] Christoforos Kachris and Ioannis Tomkos. 2012. A Survey on Optical Interconnects for Data Centers. *IEEE Commun. Surveys Tuts.* 14, 4 (2012), 1021–1036. https://doi.org/10.1109/SURV.2011.122111.00069

[88] Srikanth Kandula, Jitendra Padhye, and Paramvir Bahl. 2009. Flyways To De-Congest Data Center Networks. In *Eight ACM Workshop on Hot Topics in Networks (HotNets-VIII), HOTNETS '09, New York City, NY, USA, October 22-23, 2009*. ACM SIGCOMM. http://conferences.sigcomm.org/hotnets/2009/papers/hotnets2009-final112.pdf

[89] Srikanth Kandula, Sudipta Sengupta, Albert G. Greenberg, Parveen Patel, and Ronnie Chaiken. 2009. The nature of data center traffic: measurements & analysis. In *Proceedings of the 9th ACM SIGCOMM Internet Measurement Conference, IMC 2009, Chicago, Illinois, USA, November 4-6, 2009*. ACM, 202–208. https://doi.org/10.1145/1644893.1644918

[90] D-H Kang, J-H Lee, JH Kong, D Ha, J Yu, CY Um, JH Park, F Yeung, JH Kim, WI Park, et al. 2008. Two-bit cell operation in diode-switch phase change memory cells with 90nm technology. In *2008 Symposium on VLSI Technology*. IEEE, 98–99. https://doi.org/10.1109/VLSIT.2008.4588577

[91] KS Kaur, AZ Subramanian, Paolo Cardile, Rik Verplancke, Joris Van Kerrebrouck, Silvia Spiga, Ralf Meyer, Johan Bauwelinck, Roel Baets, and Geert Van Steenberge. 2015. Flip-chip assembly of VCSELs to silicon grating couplers via laser fabricated SU8 prisms. *Optics Express* 23, 22 (2015), 28264–28270. https://doi.org/10.1364/OE.23.028264

[92] Nam Sung Kim, Choungki Song, Woo Young Cho, Jian Huang, and Myoungsoo Jung. 2019. LL-PCM: Low-Latency Phase Change Memory Architecture. In *Proceedings of the 56th Annual Design Automation Conference 2019, DAC 2019, Las Vegas, NV, USA, June 02-06, 2019*. ACM, 14. https://doi.org/10.1145/3316781.3317853

[93] Seikwon Kim, Wonsang Kwak, Changdae Kim, Daehyeon Baek, and Jaehyuk Huh. 2020. Charge-Aware DRAM Refresh Reduction with Value Transformation. In *IEEE International Symposium on High Performance Computer Architecture, HPCA 2020, San Diego, CA, USA, February 22-26, 2020*. IEEE, 663–676. https://doi.org/10.1109/HPCA47549.2020.00060

[94] Pranay Koka, Michael O McCracken, Herb Schwetman, Xuezhe Zheng, Ron Ho, and Ashok V Krishnamoorthy. 2010. Silicon-photonic network architectures for scalable, power-efficient multi-chip systems. In *37th International Symposium on Computer Architecture (ISCA 2010), June 19-23, 2010, Saint-Malo, France*. ACM, 117–128. https://doi.org/10.1145/1815961.1815977

[95] Emre Kültürsay, Mahmut Kandemir, Anand Sivasubramaniam, and Onur Mutlu. 2013. Evaluating STT-RAM as an energy-efficient main memory alternative. In *2012 IEEE International Symposium on Performance Analysis of Systems & Software, Austin, TX, USA, 21-23 April, 2013*. IEEE Computer Society, 256–267. https://doi.org/10.1109/ISPASS.2013.6557176

[96] Eiichi Kuramochi, Kengo Nozaki, Akihiko Shinya, Koji Takeda, Tomonari Sato, Shinji Matsuo, Hideaki Taniyama, Hisashi Sumikura, and Masaya Notomi. 2014. Large-scale integration of wavelength-addressable all-optical memories on a photonic crystal chip. *Nature Photonics* 8, 6 (2014), 474–481. https://doi.org/10.1038/nphoton.2014.93

[97] Steen Larsen, Parthasarathy Sarangam, and Ram Huggahalli. 2007. Architectural Breakdown of End-to-End Latency in a TCP/IP Network. In *19th Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2007), 24-27 October 2007, Gramado, RS, Brazil*. IEEE Computer Society, 195–202. https://doi.org/10.1109/SBAC-PAD.2007.12

[98] Cheng Li, Mark Browning, Paul V. Gratz, and Samuel Palermo. 2014. LumiNOC: A Power-Efficient, High-Performance, Photonic Network-on-Chip. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* 33, 6 (2014), 826–838. https://doi.org/10.1109/TCAD.2014.2320510

[99] Sheng Li, Jung Ho Ahn, Richard D. Strong, Jay B. Brockman, Dean M. Tullsen, and Norman P. Jouppi. 2009. McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In *42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-42 2009), December 12-16, 2009, New York, New York, USA*. ACM, 469–480. https://doi.org/10.1145/1669112.1669172

[100] Ke Liu, Xuan Zhang, Jinin So, Jong-Geon Lee, Shinhaeng Kang, Sukhan Lee, Songyi Han, YeonGon Cho, Jin Hyun Kim, Yongsuk Kwon, KyungSoo Kim, Jin Jung, IlKwon Yun, Sung Joo Park, Hyunsun Park, Joon-Ho Song, Jeonghyeon Cho, Kyomin Sohn, Nam Sung Kim, and Hsien-Hsin S. Lee. 2022. Near-Memory Processing in Action: Accelerating Personalized Recommendation With AxDIMM. *IEEE Micro* 42, 1 (2022), 116–127. https://doi.org/10.1109/MM.2021.3097700

[101] Liu Liu, Rajesh Kumar, Koen Huybrechts, Thijs Spuesens, Günther Roelkens, Erik-Jan Geluk, Tjibbe De Vries, Philippe Regreny, Dries Van Thourhout, Roel Baets, and Geert Morthier. 2010. An ultra-small, low-power, all-optical flip-flop memory on a silicon chip. *Nature Photonics* 4, 3 (2010), 182–187. https://doi.org/10.1038/nphoton.2009.268

[102] Y. Liu, R. McDougall, J. Seoane, E. Kehayas, M. T. Hill, G. Maxwell, S. Zhang, R. Harmon, F.M. Huijskens, L. Rivers, et al. 2006. Characterization of Hybrid Integrated All-Optical Flip-Flop. In *LEOS 2006 - 19th Annual Meeting of the IEEE Lasers and Electro-Optics Society*. IEEE, 943–944. https://doi.org/10.1109/LEOS.2006.279158

[103] Gabriel H. Loh. 2009. Extending the effectiveness of 3D-stacked DRAM caches with an adaptive multi-queue policy. In *42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-42 2009), December 12-16, 2009, New York, New York, USA*. ACM, 201–212. https://doi.org/10.1145/1669112.1669139

[104] Gabriel H. Loh and Mark D. Hill. 2011. Efficiently enabling conventional block sizes for very large die-stacked DRAM caches. In *44th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2011, Porto Alegre, Brazil, December 3-7, 2011*. ACM, 454–464. https://doi.org/10.1145/2155620.2155673

[105] Chi-Keung Luk, Robert Cohn, Robert Muth, Harish Patil, Artur Klauser, Geoff Lowney, Steven Wallace, Vijay Janapa Reddi, and Kim Hazelwood. 2005. Pin: building customized program analysis tools with dynamic instrumentation. In *Proceedings of the ACM SIGPLAN 2005 Conference on Programming Language Design and Implementation, Chicago, IL, USA, June 12-15, 2005*. ACM, 190–200. https://doi.org/10.1145/1065010.1065034

[106] Niti Madan, Li Zhao, Naveen Muralimanohar, Aniruddha Udipi, Rajeev Balasubramonian, Ravishankar Iyer, Srihari Makineni, and Donald Newell. 2009. Optimizing communication and capacity in a 3D stacked reconfigurable cache hierarchy. In *15th International Conference on High-Performance Computer Architecture (HPCA-15 2009), 14-18 February 2009, Raleigh, North Carolina, USA*. IEEE Computer Society, 262–274. https://doi.org/10.1109/HPCA.2009.4798261

[107] Prasanth Mangalagiri, Karthik Sarpatwari, Aditya Yanamandra, VijayKrishnan Narayanan, Yuan Xie, Mary Jane Irwin, and Osama Awadel Karim. 2008. A low-power phase change memory based hybrid cache architecture. In *18th ACM Great Lakes Symposium on VLSI 2008, Orlando, Florida, USA, May 4-6, 2008*. ACM, 395–398. https://doi.org/10.1145/1366110.1366204

[108] Pavlos Maniotis, Dimitrios Fitsios, George T. Kanellos, and Nikos Pleros. 2013. Optical Buffering for Chip Multiprocessors: A 16GHz Optical Cache Memory Architecture. *J. Lightw. Tech.* 31, 24 (2013), 4175–4191. https://doi.org/10.1109/JLT.2013.2290741

[109] Pavlos Maniotis, Savvas Gitzenis, Leandros Tassiulas, and Nikos Pleros. 2016. An optically-enabled chip-multiprocessor architecture using a single-level shared optical cache memory. *Optical Switching and Networking* 22 (2016), 54–68. https://doi.org/10.1016/j.osn.2016.05.001

[110] A Manolis, J Faneca, T Domínguez Bucio, A Baldycheva, A Miliou, FY Gardes, N Pleros, and C Vagionas. 2020. Non-volatile integrated photonic memory using GST phase change material on a fully etched Si3N4/SiO2 waveguide. In *Conference on Lasers and Electro-Optics: Science and Innovations*. Optical Society of America, STh3R–4. https://doi.org/10.1364/CLEO_SI.2020.STh3R.4

[111] Lucas Mearian. 2013. Micron ships Hybrid Memory Cube that boosts DRAM 15X. https://www.computerworld.com/article/2485092/data-center/micron-ships-hybrid-memory-cube-that-boosts-dram-15x.html

[112] Meta. 2021. 2020 Sustainability Report. https://sustainability.fb.com/report/2020-sustainability-report/

[113] David A.B. Miller. 2000. Rationale and challenges for optical interconnects to electronic chips. *Proc. IEEE* 88, 6 (2000), 728–749. https://doi.org/10.1109/5.867687

[114] Atar Mittal. 2020. What is a PCB Transmission Line? https://www.protoexpress.com/blog/pcb-transmission-line/

[115] MRV. 2008. *4.25 Gbps Fibre Channel SFP Transceiver*. MRV. https://pdf1.alldatasheet.com/datasheet-pdf/view/275370/MRV/SFPFC401.html Rev. A2.

[116] Vikram K. Narayana, Shuai Sun, Abdel-Hameed A. Badawy, Volker J. Sorger, and Tarek A. El-Ghazawi. 2017. MorphoNoC: Exploring the design space of a configurable hybrid NoC using nanophotonics. *Microprocess. Microsystems* 50 (2017), 113–126. https://doi.org/10.1016/j.micpro.2017.03.006

[117] NASA. 2021. Skylake Processors. https://www.nas.nasa.gov/hecc/support/kb/skylake-processors_550.html

[118] T Nirschl, JB Philipp, TD Happ, Geoffrey W Burr, B Rajendran, M-H Lee, A Schrott, M Yang, M Breitwisch, C-F Chen, et al. 2007. Write strategies for 2 and 4-bit multi-level phase-change memory. In *2007 IEEE International Electron Devices Meeting*. IEEE, 461–464. https://doi.org/10.1109/IEDM.2007.4418973

[119] Christopher J. Nitta, Matthew Farrens, and Venkatesh Akella. 2013. On-Chip Photonic Interconnects: A Computer Architect's Perspective. In *Synthesis Lectures on Computer Architecture*. Morgan & Claypool Publishers, 1–111. https://doi.org/10.2200/S00537ED1V01Y201309CAC027

[120] Vlad Nitu, Boris Teabe, Alain Tchana, Canturk Isci, and Daniel Hagimont. 2018. Welcome to zombieland: practical and energy-efficient memory disaggregation in a datacenter. In *Proceedings of the Thirteenth EuroSys Conference, EuroSys 2018, Porto, Portugal, April 23-26, 2018*. ACM, 16:1–16:12. https://doi.org/10.1145/3190508.3190537

[121] Akihiro Noriki, Isao Tamai, Yasuhiro Ibusuki, Akio Ukita, Satoshi Suda, Daisuke Shimura, Yosuke Onawa, Hiroki Yaegashi, and Takeru Amano. 2019. Optical TSV Using Si-Photonics Integrated Curved Micro-Mirror. In *2019 International 3D Systems Integration Conference (3DIC), Sendai, Japan, October 8-10, 2019*. IEEE, 1–4. https://doi.org/10.1109/3DIC48104.2019.9058779

[122] Kengo Nozaki, Akihiko Shinya, Shinji Matsuo, Tomonari Sato, Eiichi Kuramochi, and Masaya Notomi. 2013. Ultralow-energy and high-contrast all-optical switch

involving Fano resonance based on coupled photonic crystal nanocavities. *Optics Express* 21, 10 (2013), 11877–11888. https://doi.org/10.1364/OE.21.011877

[123] Kengo Nozaki, Akihiko Shinya, Shinji Matsuo, Yasumasa Suzaki, Toru Segawa, Tomonari Sato, Yoshihiro Kawaguchi, Ryo Takahashi, and Masaya Notomi. 2012. Ultralow-power all-optical RAM based on nanocavities. *Nature Photonics* 6, 4 (2012), 248–252. https://doi.org/10.1038/nphoton.2012.2

[124] The Natural Resources Defense Council (NRDC). 2014. Data Center Efficiency Assessment, Scaling Up Energy Efficiency Across the Data Center Industry: Evaluating Key Drivers and Barriers. https://www.nrdc.org/sites/default/files/data-center-efficiency-assessment-IP.pdf

[125] NVIDIA. 2016. NVIDIA Tesla P100, White Paper WP-08019-001_v01.1. https://images.nvidia.com/content/pdf/tesla/whitepaper/pascal-architecture-whitepaper.pdf

[126] Yan Pan, John Kim, and Gokhan Memik. 2011. FeatherWeight: low-cost optical arbitration with QoS support. In *44th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2011, Porto Alegre, Brazil, December 3-7, 2011*. ACM, 105–116. https://doi.org/10.1145/2155620.2155633

[127] Yan Pan, Prabhat Kumar, John Kim, Gokhan Memik, Yu Zhang, and Alok N. Choudhary. 2009. Firefly: illuminating future network-on-chip with nanophotonics. In *36th International Symposium on Computer Architecture (ISCA 2009), June 20-24, 2009, Austin, TX, USA*. ACM, 429–440. https://doi.org/10.1145/1555754.1555808

[128] Reena Panda, Shuang Song, Joseph Dean, and Lizy K. John. 2018. Wait of a Decade: Did SPEC CPU 2017 Broaden the Performance Horizon?. In *IEEE International Symposium on High Performance Computer Architecture, HPCA 2018, Vienna, Austria, February 24-28, 2018*. IEEE Computer Society, 271–282. https://doi.org/10.1109/HPCA.2018.00032

[129] Mahavir S. Parekh, Paragkumar A. Thadesar, and Muhannad S. Bakir. 2011. Electrical, optical and fluidic through-silicon vias for silicon interposer applications. In *2011 IEEE 61st Electronic Components and Technology Conference (ECTC)*. IEEE, 1992–1998. https://doi.org/10.1109/ECTC.2011.5898790

[130] Chirag S. Patel, Cornelia K. Tsang, Christian Schuster, Fuad E. Doany, Harold Nyikal, Christian W. Baks, Russell Budd, L. Paivikki Buchwalter, Paul S. Andry, Donald F. Canaperi, et al. 2005. Silicon Carrier with Deep Through-Vias, Fine

Pitch Wiring and Through Cavity for Parallel Optical Transceiver. In *Electronic Components and Technology, 2005. ECTC'05.* IEEE, 1318–1324. https://doi.org/10.1109/ECTC.2005.1441439

[131] Harish Patil, Robert Cohn, Mark Charney, Rajiv Kapoor, Andrew Sun, and Anand Karunanidhi. 2004. Pinpointing Representative Portions of Large Intel® Itanium® Programs with Dynamic Instrumentation. In *37th Annual International Symposium on Microarchitecture (MICRO-37 2004), 4-8 December 2004, Portland, OR, USA.* IEEE Computer Society, 81–92. https://doi.org/10.1109/MICRO.2004.28

[132] Harish Patil, Alexander Isaev, Wim Heirman, Alen Sabu, Ali Hajiabadi, and Trevor E Carlson. 2021. ELFies: Executable Region Checkpoints for Performance Analysis and Simulation. In *IEEE/ACM International Symposium on Code Generation and Optimization, CGO 2021, Seoul, South Korea, February 27 - March 3, 2021.* IEEE, 126–136. https://doi.org/10.1109/CGO51591.2021.9370340

[133] Harish Patil, Cristiano Pereira, Mack Stallcup, Gregory Lueck, and James Cownie. 2010. Pinplay: a framework for deterministic replay and reproducible analysis of parallel programs. In *CGO 2010, The 8th International Symposium on Code Generation and Optimization, Toronto, Ontario, Canada, April 24-28, 2010.* ACM, 2–11. https://doi.org/10.1145/1772954.1772958

[134] Harish Patil, Cristiano Pereira, Chariles(Chuck) Yount, and Milind Chabbi. 2015. Tutorial: Using PinPlay for Reproducible Analysis and Replay Debugging. In *36th ACM SIGPLAN Conference on Programming Language Design and Implementation, Portland, OR, USA, June 15-17, 2015.* ACM. https://sites.google.com/site/pinplaypldi2015tutotial/

[135] Lorenzo Pavesi and Gérard Guillot. 2006. Optical Interconnects: The Silicon Aproach. In *Springer Series in Optical Sciences.* Vol. 119. Springer. https://doi.org/10.1007/978-3-540-28912-8

[136] J. Thomas Pawlowski. 2011. Hybrid memory cube (HMC). In *2011 IEEE Hot Chips 23 Symposium (HCS).* 1–24. https://doi.org/10.1109/HOTCHIPS.2011.7477494

[137] Nikos Pleros, Dimitrios Apostolopoulos, Dimitrios Petrantonakis, Christos Stamatiadis, and Hercules Avramopoulos. 2009. Optical Static RAM Cell. *IEEE Photon. Tech. Lett.* 21, 2 (2009), 73–75. https://doi.org/10.1109/LPT.2008.2008444

[138] Nikos Pleros, Pavlos Maniotis, Theonitsa Alexoudi, Dimitris Fitsios, Christos Vagionas, Sotiris Papaioannou, Konstantinos Vyrsokinos, and George T. Kanellos.

2014. Optical RAM-enabled cache memory and optical routing for chip multiprocessors: technologies and architectures. In *Optical Interconnects XIV*, Vol. 8991. International Society for Optics and Photonics, SPIE, 206–213. https://doi.org/10.1117/12.2042732

[139] Timothy Prickett Morgan. 2017. Drilling Down Into The Xeon Skylake Architecture. https://www.nextplatform.com/2017/08/04/drilling-xeon-skylake-architecture/

[140] Andrew Putnam, Adrian M. Caulfield, Eric S. Chung, Derek Chiou, Kypros Constantinides, John Demme, Hadi Esmaeilzadeh, Jeremy Fowers, Gopi Prashanth Gopal, Jan Gray, Michael Haselman, Scott Hauck, Stephen Heil, Amir Hormati, Joo-Young Kim, Sitaram Lanka, James R. Larus, Eric Peterson, Simon Pope, Aaron Smith, Jason Thong, Phillip Yi Xiao, and Doug Burger. 2014. A reconfigurable fabric for accelerating large-scale datacenter services. In *ACM/IEEE 41st International Symposium on Computer Architecture, ISCA 2014, Minneapolis, MN, USA, June 14-18, 2014*. IEEE Computer Society, 13–24. https://doi.org/10.1109/ISCA.2014.6853195

[141] George F. Riley and Thomas R. Henderson. 2010. The *ns-3* Network Simulator. In *Modeling and Tools for Network Simulation*. Springer, 15–34. https://doi.org/10.1007/978-3-642-12331-3_2

[142] Carlos Ríos, Matthias Stegmaier, Peiman Hosseini, Di Wang, Torsten Scherer, C David Wright, Harish Bhaskaran, and Wolfram HP Pernice. 2015. Integrated all-photonic non-volatile multi-level memory. *Nature photonics* 9, 11 (2015), 725–732. https://doi.org/10.1038/nphoton.2015.182

[143] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C. Snoeren. 2015. Inside the Social Network's (Datacenter) Network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2015, London, United Kingdom, August 17-21, 2015*. ACM, 123–137. https://doi.org/10.1145/2785956.2787472

[144] Alexander V. Rylyakov, Jonathan E. Proesel, Sergey V. Rylov, Benjamin G. Lee, John F. Bulzacchelli, Abhijeet Ardey, Benjamin D. Parker, Michael P. Beakes, Christian W. Baks, Clint Schow, and Mounir Meghelli. 2015. A 25 Gb/s Burst-Mode Receiver for Low Latency Photonic Switch Networks. *IEEE J. Solid-State Circuits* 50, 12 (2015), 3120–3132. https://doi.org/10.1109/JSSC.2015.2478837

[145] Wesley D. Sacher, Jared C. Mikkelsen, Ying Huang, Jason C.C. Mak, Zheng Yong, Xianshu Luo, Yu Li, Patrick Dumais, Jia Jiang, Dominic J. Goodwill, Eric Bernier,

Patrick Guo-Qiang Lo, and Joyce K. S. Poon. 2018. Monolithically Integrated Multilayer Silicon Nitride-on-Silicon Waveguide Platforms for 3-D Photonic Circuits and Devices. *Proc. IEEE* 106, 12 (2018), 2232–2245. https://doi.org/10.1109/JPROC.2018.2860994

[146] Jun Sakaguchi, Takeo Katayama, and Hitoshi Kawaguchi. 2010. High Switching-Speed Operation of Optical Memory Based on Polarization Bistable Vertical-Cavity Surface-Emitting Laser. *IEEE J. Quantum Electron.* 46, 11 (2010), 1526–1534. https://doi.org/10.1109/JQE.2010.2052590

[147] Daniel Sánchez and Christos Kozyrakis. 2013. ZSim: Fast and accurate microarchitectural simulation of thousand-core systems. In *The 40th Annual International Symposium on Computer Architecture, ISCA'13, Tel-Aviv, Israel, June 23-27, 2013*. ACM, 475–486. https://doi.org/10.1145/2485922.2485963

[148] Rathijit Sen and David A. Wood. 2013. *Cache power budgeting for performance*. Technical Report TR1791. Univ. of Wisconsin-Madison, Computer Science Dept. http://digital.library.wisc.edu/1793/65385

[149] Alireza Shafaei, Yanzhi Wang, Xue Lin, and Massoud Pedram. 2014. FinCACTI: Architectural Analysis and Modeling of Caches with Deeply-Scaled FinFET Devices. In *IEEE Computer Society Annual Symposium on VLSI, ISVLSI 2014, Tampa, FL, USA, July 9-11, 2014*. IEEE Computer Society, 290–295. https://doi.org/10.1109/ISVLSI.2014.94

[150] Timothy Sherwood, Erez Perelman, Greg Hamerly, and Brad Calder. 2002. Automatically characterizing large scale program behavior. In *10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-X), San Jose, California, USA, October 5-9, 2002*. ACM Press, 45–57. https://doi.org/10.1145/605397.605403

[151] Nicolás Sherwood-Droz and Michal Lipson. 2011. Scalable 3D dense integration of photonics on bulk silicon. *Optics Express* 19, 18 (2011), 17758–17765. https://doi.org/10.1364/OE.19.017758

[152] Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, Anand Kanagala, Jeff Provost, Jason Simmons, Eiichi Tanda, Jim Wanderer, Urs Hölzle, Stephen Stuart, and Amin Vahdat. 2015. Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2015, London, United Kingdom, August 17-21, 2015*. ACM, 183–197.

https://doi.org/10.1145/2785956.2787508

[153] Sarabjeet Singh and Manu Awasthi. 2019. Memory Centric Characterization and Analysis of SPEC CPU2017 Suite. In *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering, ICPE 2019, Mumbai, India, April 7-11, 2019*. ACM, 285–292. https://doi.org/10.1145/3297663.3310311

[154] Clinton W. Smullen, Vidyabhushan Mohan, Anurag Nigam, Sudhanva Gurumurthi, and Mircea R. Stan. 2011. Relaxing non-volatility for fast and energy-efficient STT-RAM caches. In *17th International Conference on High-Performance Computer Architecture, HPCA-17 2011, February 12-16, 2011, San Antonio, Texas, USA*. IEEE Computer Society, 50–61. https://doi.org/10.1109/HPCA.2011.5749716

[155] Richard Soref. 2006. The Past, Present, and Future of Silicon Photonics. *IEEE J. Sel. Topics Quantum Electron.* 12, 6 (2006), 1678–1687. https://doi.org/10.1109/JSTQE.2006.883151

[156] Springer 2001. *Power-Aware Computer Systems, First International Workshop, PACS 2000, Cambridge, MA, USA, November 12, 2000, Revised Papers* (Cambridge, MA). Lecture Notes in Computer Science, Vol. 2008. Springer, Springer-Verlag. https://doi.org/10.1007/3-540-44572-2

[157] Mircea R. Stan and Kevin Skadron. 2003. Guest Editors' Introduction: Power-Aware Computing. *Computer* 36, 12 (2003), 35–38. https://doi.org/10.1109/MC.2003.1250876

[158] Standard Performance Evaluation Corporation. 2020. SPECpower_ssj2008 report, Lenovo Global Technology Think System SR665. https://www.spec.org/power_ssj2008/results/res2020q2/power_ssj2008-20200519-01031.html

[159] Chen Sun, Mark T. Wade, Yunsup Lee, Jason S. Orcutt, Luca Alloatti, Michael S. Georgas, Andrew S. Waterman, Jeffrey M. Shainline, Rimas R. Avizienis, Sen Lin, et al. 2015. Single-chip microprocessor that communicates directly using light. *Nature* 528, 7581 (2015), 534–538. https://doi.org/10.1038/nature16454

[160] Guangyu Sun, Xiangyu Dong, Yuan Xie, Jian Li, and Yiran Chen. 2009. A novel architecture of the 3D stacked MRAM L2 cache for CMPs. In *15th International Conference on High-Performance Computer Architecture (HPCA-15 2009), 14-18 February 2009, Raleigh, North Carolina, USA*. IEEE Computer Society, 239–249. https://doi.org/10.1109/HPCA.2009.4798259

[161] Zhenyu Sun, Xiuyuan Bi, Hai Li, Weng-Fai Wong, Zhong-Liang Ong, Xiaochun Zhu, and Wenqing Wu. 2011. Multi retention level STT-RAM cache designs with a dynamic refresh scheme. In *44th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 2011, Porto Alegre, Brazil, December 3-7, 2011*. ACM, 329–338. https://doi.org/10.1145/2155620.2155659

[162] Joel M. Tendler, J. Steve Dodson, J.S. Fields, Hung Le, and Balaram Sinharoy. 2002. POWER4 system microarchitecture. *IBM Journal of Research and Development* 46, 1 (2002), 5–26. https://doi.org/10.1147/rd.461.0005

[163] Christos A. Thraskias, Eythimios N. Lallas, Niels Neumann, Laurent Schares, Bert J. Offrein, Ronny Henker, Dirk Plettemeier, Frank Ellinger, Juerg Leuthold, and Ioannis Tomkos. 2018. Survey of Photonic and Plasmonic Interconnect Technologies for Intra-Datacenter and High-Performance Computing Communications. *IEEE Commun. Surveys Tuts.* 20, 4 (2018), 2758–2783. https://doi.org/10.1109/COMST.2018.2839672

[164] Toshiba. 2017. Toshiba Develops World's First 16-die Stacked NAND Flash Memory with TSV Technology. https://toshiba.semicon-storage.com/eu/company/news/2017/07/memory-20170711-1.html

[165] Christos Vagionas, Dimitris Fitsios, George T. Kanellos, Nikos Pleros, and Amalia Miliou. 2012. All Optical Flip Flop with two Coupled Travelling Waveguide SOA-XGM Switches. In *Conference on Lasers and Electro-Optics 2012*. Optical Society of America, JW4A.2. https://doi.org/10.1364/CLEO_AT.2012.JW4A.2

[166] Christos Vagionas, Dimitrios Fitsios, Konstantinos Vyrsokinos, George T. Kanellos, Amalia Miliou, and Nikos Pleros. 2014. XPM- and XGM-Based Optical RAM Memories: Frequency and Time Domain Theoretical Analysis. *IEEE J. Quantum Electron.* 50, 8 (2014), 1–15. https://doi.org/10.1109/JQE.2014.2330068

[167] Christos Vagionas, Stella Markou, George Dabos, Theonitsa Alexoudi, Dimitris Tsiokos, Amalia Miliou, Nikos Pleros, and George T. Kanellos. 2013. Optical RAM Row Access and Column Decoding for WDM-formatted optical words. In *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference 2013*. Optical Society of America, JW2A.56. https://doi.org/10.1364/NFOEC.2013.JW2A.56

[168] Christos Vagionas, Stelios Pitris, Charoula Mitsolidou, Jan Bos, Pavlos Maniotis, Dimitris Tsiokos, and Nikos Pleros. 2015. All-Optical Tag Comparison for Hit/Miss Decision in Optical Cache Memories. *IEEE Photon. Tech. Lett.* 28, 7 (2015), 713–716. https://doi.org/10.1109/LPT.2015.2505500

[169] Dana Vantrease, Nathan Binkert, Robert Schreiber, and Mikko H. Lipasti. 2009. Light speed arbitration and flow control for nanophotonic interconnects. In *42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-42 2009), December 12-16, 2009, New York, New York, USA*. ACM, 304–315. https://doi.org/10.1145/1669112.1669152

[170] Dana Vantrease, Robert Schreiber, Matteo Monchiero, Moray McLaren, Norman P. Jouppi, Marco Fiorentino, Al Davis, Nathan Binkert, Raymond G. Beausoleil, and Jung Ho Ahn. 2008. Corona: System Implications of Emerging Nanophotonic Technology. In *35th International Symposium on Computer Architecture (ISCA 2008), June 21-25, 2008, Beijing, China*. IEEE Computer Society, 153–164. https://doi.org/10.1109/ISCA.2008.35

[171] Jue Wang, Xiangyu Dong, Yuan Xie, and Norman P. Jouppi. 2013. i^{2}WAP: Improving non-volatile cache lifetime by reducing inter- and intra-set write variations. In *19th IEEE International Symposium on High Performance Computer Architecture, HPCA 2013, Shenzhen, China, February 23-27, 2013*. IEEE Computer Society, 234–245. https://doi.org/10.1109/HPCA.2013.6522322

[172] Xiaodong Wang, Yanjun Yao, Xiaorui Wang, Kefa Lu, and Qing Cao. 2012. CARPO: Correlation-aware power optimization in data center networks. In *Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012*. IEEE, 1125–1133. https://doi.org/10.1109/INFCOM.2012.6195471

[173] Sebastian Werner, Javier Navaridas, and Mikel Luján. 2018. A Survey on Optical Network-on-Chip Architectures. *Comput. Surveys* 50, 6 (2018), 89:1–89:37. https://doi.org/10.1145/3131346

[174] WikiChip. 2020. Skylake (client) - Microarchitectures - Intel. https://en.wikichip.org/wiki/intel/microarchitectures/skylake_(client)

[175] WikiChip. 2020. Skylake (server) - Microarchitectures - Intel. https://en.wikichip.org/wiki/intel/microarchitectures/skylake_(server)

[176] H.-S. Philip Wong, Simone Raoux, SangBum Kim, Jiale Liang, John P. Reifenberg, Bipin Rajendran, Mehdi Asheghi, and Kenneth E. Goodson. 2010. Phase Change Memory. *Proc. IEEE* 98, 12 (2010), 2201–2227. https://doi.org/10.1109/JPROC.2010.2070050

[177] Qinzhe Wu, Steven Flolid, Shuang Song, Junyong Deng, and Lizy K John. 2018. Invited Paper for the Hot Workloads Special Session Hot Regions in SPEC CPU2017. In *2018 IEEE International Symposium on Workload Characterization, IISWC*

2018, Raleigh, NC, USA, September 30 - October 2, 2018. IEEE Computer Society, 71–77. https://doi.org/10.1109/IISWC.2018.8573479

[178] Xiaoxia Wu, Jian Li, Lixin Zhang, Evan Speight, Ram Rajamony, and Yuan Xie. 2009. Hybrid cache architecture with disparate memory technologies. In *36th International Symposium on Computer Architecture (ISCA 2009), June 20-24, 2009, Austin, TX, USA*. ACM, 34–45. https://doi.org/10.1145/1555754.1555761

[179] William A. Wulf and Sally A. McKee. 1995. Hitting the memory wall: implications of the obvious. *ACM SIGARCH Comp. Arch. News* 23, 1 (1995), 20–24. https://doi.org/10.1145/216585.216588

[180] Lei Yi, Guangbao Shan, Song Liu, and Chengmin Xie. 2016. High-performance processor design based on 3D on-chip cache. *Microprocess. Microsystems* 47 (2016), 486–490. https://doi.org/10.1016/j.micpro.2016.07.009

[181] S.J. Ben Yoo, Binbin Guan, and Ryan P. Scott. 2016. Heterogeneous 2D/3D photonic integrated microsystems. *Microsystems & Nanoengineering* 2, 1 (2016), 1–9. https://doi.org/10.1038/micronano.2016.30

[182] Dongli Zhang, Moussa Ehsan, Michael Ferdman, and Radu Sion. 2014. DIMMer: A case for turning off DIMMs in clouds. In *Proceedings of the ACM Symposium on Cloud Computing, Seattle, WA, USA, November 3-5, 2014*. ACM, 11:1–11:8. https://doi.org/10.1145/2670979.2670990

[183] Yang Zhang, Amir Hosseini, Xiaochuan Xu, David Kwong, and Ray T. Chen. 2013. Ultralow-loss silicon waveguide crossing using Bloch modes in index-engineered cascaded multimode-interference couplers. *Optics Letters* 38, 18 (2013), 3608–3611. https://doi.org/10.1364/OL.38.003608

[184] Yu Zhang, Anirban Samanta, Kuanping Shang, and S.J. Ben Yoo. 2020. Scalable 3D Silicon Photonic Electronic Integrated Circuits and Their Applications. *IEEE J. Sel. Topics Quantum Electron.* 26, 2 (2020), 1–10. https://doi.org/10.1109/JSTQE.2020.2975656

[185] Liang Zhou, Laxmi N. Bhuyan, and K. K. Ramakrishnan. 2019. DREAM: DistRibuted Energy-Aware traffic Management for Data Center Networks. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems, e-Energy 2019, Phoenix, AZ, USA, June 25-28, 2019*. ACM, 273–284. https://doi.org/10.1145/3307772.3328291

APPENDIX A

Additional Figures

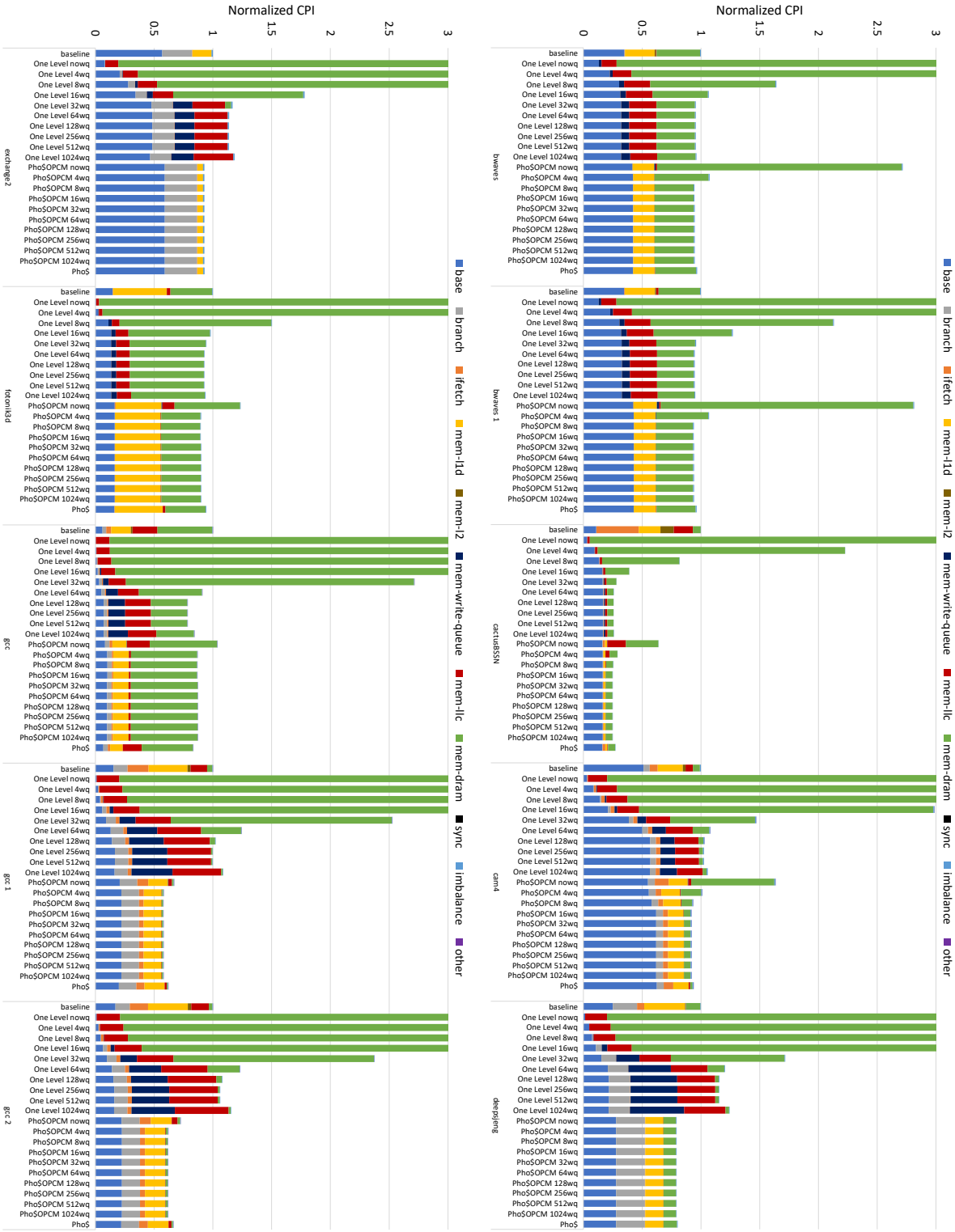


Figure A.1. Average CPU2017 CPI Stacks normalized to *baseline* Part 1 & 2.

Figure A.2. Average CPU2017 CPI Stacks normalized to *baseline* Part 3 & 4.

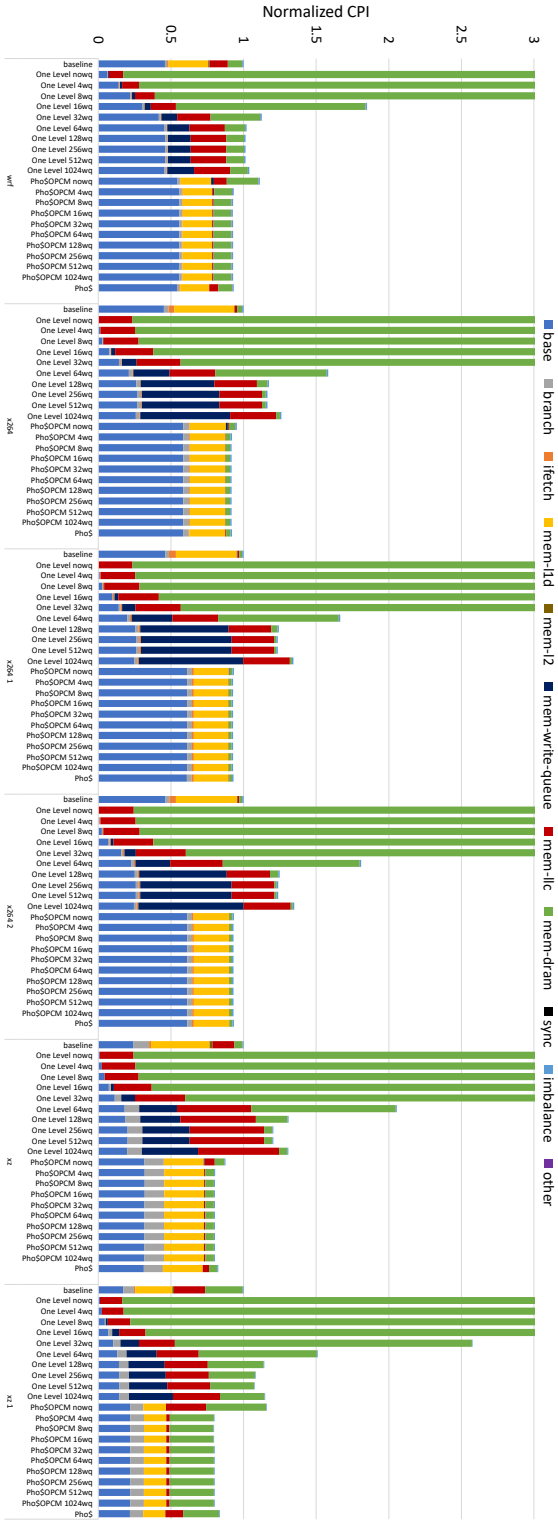


Figure A.3. Average CPU2017 CPI Stacks normalized to *baseline* Part 5.