Electro-Photonic NoC Designs for Kilocore Systems

JOSÉ L. ABELLÁN, CHAO CHEN, and AJAY JOSHI, Boston University

The increasing core count in manycore systems requires a corresponding large Network-on-chip (NoC) bandwidth to support the overlying applications. However, it is not possible to provide this large bandwidth in an energy-efficient manner using electrical link technology. To overcome this issue, photonic link technology has been proposed as a replacement. This work explores the limits and opportunities for using photonic links to design the NoC architecture for a future Kilocore system. Three different NoC designs are explored: ElecNoC, an electrical concentrated two-dimensional- (2D) mesh NoC; HybNoC, an electrical concentrated 2D mesh with a photonic multi-crossbar NoC; and PhotoNoC, a photonic multi-bus NoC. We consider both private and shared cache architectures and, to leverage the large bandwidth density of photonic links, we investigate the use of prefetching and aggressive non-blocking caches. Our analysis using contemporary Big Data workloads shows that the non-blocking caches with a shared LLC can best leverage the large bandwidth of the photonic links in the Kilocore system. Moreover, compared to ElecNoC-based and HybNoC-based Kilocore systems, a PhotoNoC-based Kilocore system achieves up to 2.5× and 1.5× better performance, respectively, and can support up to 2.1× and 1.1× higher bandwidth, respectively, while dissipating comparable power in the overall system.

CCS Concepts: • **Computer systems organization → Interconnection architectures**; *Multiple instruction, multiple data*; *Multicore architectures*; • **Hardware → Emerging optical and photonic technologies;**

Additional Key Words and Phrases: Networks-on-chip, manycore CMP, silicon-photonic technology, multiprogrammed workloads

ACM Reference Format:
José L. Abellán, Chao Chen, and Ajay Joshi. 2016. Electro-photonic NoC designs for Kilocore systems. J. Emerg. Technol. Comput. Syst. 13, 2, Article 24 (November 2016), 25 pages.
DOI: http://dx.doi.org/10.1145/2967614

1. INTRODUCTION

Over the past decade, the general-purpose compute capacity of the world has increased by 1.2× every year [Patterson and Hennessy 2013], and we will need to maintain this rate of growth to support the increasingly sophisticated data-driven applications of the future. The computing community has migrated towards manycore computing systems, with the goal of improving the computing capacity per chip through parallelism while staying within the chip power budget. Energy-efficient data communication has been identified as one of the key requirements for achieving this goal, and the use of silicon-photonic networks for on-chip and off-chip communication has been proposed as one

This work was supported in part by DARPA Contract No. W911NF-12-1-0211. The work in this paper was done by J. L. Abellán and C. Chen when they were a postdoctoral researcher and a Ph.D. candidate, respectively, at Boston University.
Authors' addresses: J. L. Abellán, Department of Computer Science, Universidad Católica de Murcia (UCAM), Murcia 30107, Spain; email: jlabellan@ucam.edu; C. Chen, Qualcomm Technologies Inc., 9600 N Mopac Expy Ste 900, Austin, TX 78759, USA; email: chen9810@gmail.com; A. Joshi, Department of ECE, Boston University, 8 Saint Mary's Street, PHO 334, Boston MA 02215, USA; email: joshi@bu.edu.

DOI: http://dx.doi.org/10.1145/2967614

of the technologies that can meet this requirement [D. Vantrease et al. 2008; Beamer et al. 2010; Bergman et al. 2014].

To make silicon-photonic links viable for on-chip networks, we still need to overcome the following challenges: reduce the large power consumed in the laser sources and in tuning to overcome on-chip thermal gradients, develop new architectures and applications that can leverage the large bandwidth offered by photonic links, and develop novel packaging solutions to couple a large number of off-chip lasers with a complementary metal–oxide–semiconductor (CMOS) chip or develop novel Si-based on-chip laser solutions.

To address the large power consumed in the laser sources, several runtime [Chen et al. 2015; Neel et al. 2015] and design-time solutions [Chen et al. 2014; Boos et al. 2013] have been proposed. A common theme across all the runtime solutions is the reconfiguration of the on-chip network bandwidth based on the bandwidth requirements of the overlying applications. The design-time solutions involve designing low-loss photonic devices and strategic placement and routing of these devices. For thermal management, too, both runtime [Li et al. 2015] and design-time [Zhang et al. 2015] solutions have been proposed. The runtime solutions involve workload allocation, and dynamic voltage and frequency scaling (DVFS), while the design-time solutions involve designing athermal photonic devices and strategic placement of the thermally sensitive photonic devices away from the power-hungry components. On the packaging front, there have been multiple efforts in place on designing energy-efficient coupling of off-chip lasers [Zheng et al. 2013] as well as the development of Si-based laser sources [Roelkens et al. 2014].

In this article, we focus on addressing the bandwidth challenge associated with the on-chip photonic networks. Current commercial manycore systems already have a hundred of cores on a single chip [EZchip Semiconductor Ltd. 2015]. Moving forward, this core count is expected to increase and, correspondingly, the on-chip bandwidth will also need to increase to provide rapid communication between cores and the memory hierarchy. Moreover, as we enter the age of Big Data, future computing systems will need to quickly process large quantities of data, which would make it necessary to have a high-bandwidth communication path through the memory hierarchy. Silicon-photonic links with their high bandwidth can support the large-bandwidth requirements of future manycore systems running large data-intensive workloads.

We evaluate the limits and opportunities for using the photonic link technology to design the network-on-chip (NoC) of a 1024-core system (referred to as Kilocore system in the rest of the article), which will support Big Data applications. In the literature, we can find some research works that explore Kilocore systems with photonic NoCs [Kurian et al. 2010; Sikder et al. 2015]. However, in our literature review, we have not come across any other articles that explore future Kilocore architectures with Photonic NoC for Big Data applications. We focus on the Big Data applications from the areas of cybersecurity, video surveillance, medical bioinformatics, data enrichment, social networks, new engineering processes, and large-scale data analysis. The applications from these areas are embarrassingly parallel [McAfee et al. 2012] since hundreds of thousands of interactions among components (e.g., persons, molecules, decisions, etc.) have to be computed and modeled accurately, sometimes with real-time constraints. For our analysis, we consider the Mantevo project [Heroux et al. 2009], GRAPH500 [Murphy et al. 2010], and Ubiquitous High Performance Computing (UHPC) benchmarks [Campbell et al. 2012], which are good representatives of these kinds of applications.

The major contributions of this article are as follows:

—We address the bandwidth challenge associated to silicon-photonic link technology. For that, we study a forward-looking shared-memory Kilocore system specifically designed to efficiently support contemporary large-bandwidth workloads from UHPC,

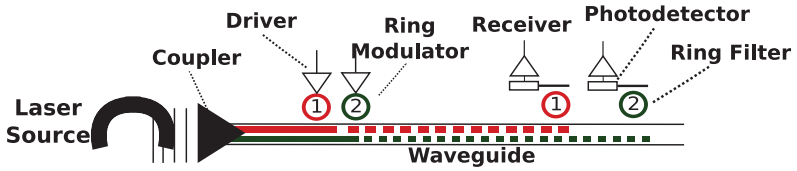


Fig. 1. Photonic link components: Two example point-to-point photonic links implemented with WDM. 1 and 2 refer to λ_1 and λ_2 wavelengths.

MANTEVO, and GRAPH500 benchmark suites. These suites are representative of the shared-memory applications of the new Big Data era. We conclude that 4TB/s on-chip bandwidth will be needed when exploring an aggressive cache hierarchy for a Kilocore system running future Big Data applications.

—We explore three different NoC architectures, an electrical concentrated 2D-mesh NoC (ElecNoC), an electrical concentrated 2D-mesh with a photonic multi-crossbar NoC (HybNoC), and a purely photonic multibus NoC (PhotoNoC), for our target Kilocore system. For all three NoCs, the parameters were chosen such that they provide the highest application performance for the benchmarks suites under consideration. Note that we do not provide a head-to-head comparison among all the possible topologies that one can have for a Kilocore system as that is beyond the scope of this article.

—We study if private and shared cache architectures with prefetching and aggressive non-blocking can leverage the large photonic NoC bandwidth of the Kilocore system.

—We determine that in our target Kilocore system the PhotoNoC provides the best application performance at comparable power consumption if the photonic links have energy cost of less than 1 pJ/bit. Moreover, we also observed that the non-blocking caches with a shared LLC can best leverage the large bandwidth of the photonic links in the Kilocore system. Compared to the ElecNoC-based and HybNoC-based Kilocore system, PhotoNoC-based Kilocore system achieves up to 2.5× and 1.5× better performance, respectively, and can support up to 2.1× and 1.1× higher bandwidth, respectively, while dissipating comparable power in the overall system.

2. SILICON-PHOTONIC LINK TECHNOLOGY

Figure 1 illustrates a generic silicon-photonic channel with two links multiplexed onto the same waveguide for communication. A laser source is used to power these two silicon-photonic links. The output of the laser is coupled into planar waveguides integrated in the chip using vertical grating couplers. At the transmitter side, an electrical modulator driver is used to imprint the electrical data onto the photonic link. Here, the electrical-to-optical conversion is performed using a ring modulator that is controlled by the electrical modulator driver. The modulated light waves propagate along the waveguide and can pass through zero or more ring filters. At the receiver side, a ring filter, whose resonant wavelength matches with the wavelength of a light wave, "drops" the light wave onto a photodetector. The resulting photodetector current is sensed by an electrical receiver. At this stage, data are converted back into the electrical domain from the photonic domain.

3. TARGET MANYCORE SYSTEM

Our target system is a Kilocore chip fabricated assuming double-gate (FinFET) 11nm CMOS technology. The operating frequency is 1GHz with 0.6V supply voltage, and the chip has an area of 400mm^2. The architecture of our Kilocore system is based on the Intel Single-Chip Cloud Computer manycore system [Gries et al. 2011]. The Kilocore system is a tiled-based architecture where the 1024 cores are divided into 256 tiles

Table I. Micro-Architecture of the Kilocore System. Two Different L2 Cache Architectures Are Considered: Private L2 Cache and Shared L2 Cache

Processor Core	
Pipeline	2-way superscalar, OoO exec.
Technology	11 nm, 0.6 Volts, 1 GHz
Instruction Queue	64 entries
Reorder Buffer	40 entries
Reservation Stations	36 entries
Branch Predictor	2 bit, 128 entries
Execution Units	1 FPU, 2 ALU, 1 MULT
Cache Hierarchy	
Private L1 I/D-Cache	4-way 32 KB @ 2 ns
a) Private Unified L2 Cache	8-way 256 KB @ 6 ns
b) Shared Distributed L2 Cache	16-way 4 MB/16 cores @ 10 ns
Cache Coherence	Directory based
Memory	16 memory controllers
	16 PIDRAM @ 50 ns

with 4 cores in each tile. We conducted an experimental evaluation of power and area overhead for the number of cores per tile and we determine that 4 cores/tile is the best granularity for our target system. Each core is a simplified version of a Pentium II that has a two-way in-order issue, out-of-order (OoO) execution superscalar pipeline, with 32KB I/D L1 cache. We consider two different L2 cache configurations: a private 256KB L2 cache for each core and a shared L2 cache that is distributed across 64 banks with 4MB per bank. Both designs provide 256MB on-chip cache capacity to the Kilocore system. This amount of on-chip cache is reasonable given that the new 100-core TILE-Mx chip [EZchip Semiconductor Ltd. 2015] already includes a 40MB on-chip cache. Cache coherency is implemented by using a directory-based protocol, and the directories are co-located with the memory controllers (MC) in the private hierarchy and with the L2 banks in the shared case. As we will detail in Section 4, we assume a baseline concentrated two-dimensional- (2D) mesh NoC for the Kilocore system. This NoC interconnects the L2 caches and MCs in the private cache hierarchy. For the shared cache hierarchy, the NoC interconnects the L1 caches and the L2 banks, and L2 banks with the MCs. A detailed description of the three different NoCs that we evaluated is provided in Section 4.

As our focus is on determining the maximum NoC bandwidth that can be sustained on chip, we needed a memory system that avoids the main memory from becoming the performance bottleneck. Hence, we used the high-bandwidth and low-latency photonically interconnected DRAM (PIDRAM) technology [Beamer et al. 2010] with 16 MCs distributed uniformly along the four edges of the chip. This uniform distribution of MCs results in a better distribution of the on-chip network traffic and reduces memory hotspots. We assume an average latency of 50ns for the communication from the MCs to PIDRAMs and back. We ignore the variations in queuing latencies at the inputs of MCs because the high off-chip bandwidth using PIDRAM significantly reduces the number of outstanding memory requests in the queue. The main microarchitectural parameters of the components on the logic layer are shown in Table I.

4. NOC DESIGNS

In this section, we describe the three different NoC designs that we evaluate for the target Kilocore system. We start by describing the baseline concentrated 2D-mesh NoC. After that, we detail the hybrid NoC and the photonic NoC. All three NoCs are explained using the shared L2 cache hierarchy to help the reader understand how all

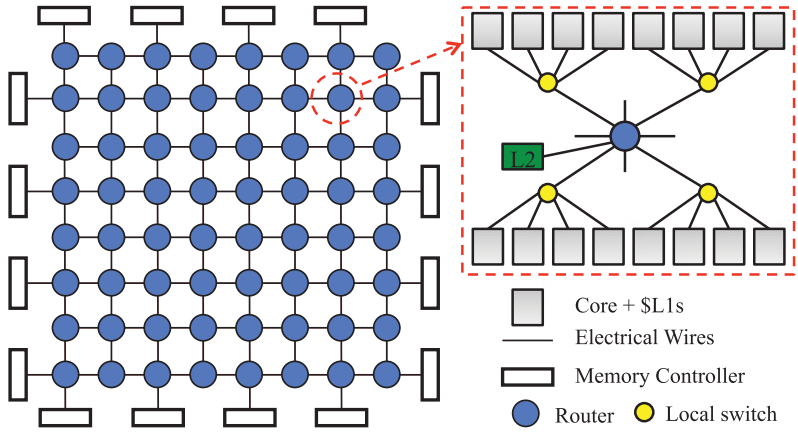


Fig. 2. Electrical NoC (ElecNoC) with shared L2 cache hierarchy. Each router is connected to 16 cores and one 4MB L2 bank, and some of the 64 routers in the periphery of the chip are also connected to one memory controller (MC).

64 shared L2 banks are distributed over the chip. The private L1/L2 cache hierarchy uses a similar NoC architecture that connects the private L2 caches to the MCs.

4.1. Electrical Concentrated 2D-mesh NoC (ElecNoC)

We considered both high-radix low-diameter topologies like crossbar, butterfly, and Clos and low-radix high-diameter topologies like mesh and torus for implementing the baseline electrical NoC. For an energy-efficient implementation of the high-radix low-diameter topologies in the electrical domain, we need to use equalized interconnects [Joshi et al. 2009b]. Design of these equalized links becomes extremely challenging, especially when we scale down to 11nm. Moreover, these networks need high-radix routers that could be very power hungry. Hence, we decided to use the low-radix high-diameter mesh topology, which is easy to design from a hardware perspective due to the use of short wires and low-radix routers. It uses distributed flow control, which contributes to efficient traffic management in highly congested traffic patterns. This 2D mesh topology has been used in the design for both commercial [Bell et al. 2008; Howard et al. 2011] and academic [Daya et al. 2014] manycore systems. However, in a 2D-mesh network, a packet has to make multiple hops to reach its destination. This can lead to high communication latency, which in turn can hurt the manycore performance. In addition, mesh networks make programming challenging as the programmer needs to carefully manage task and data placement.

To reduce the hop count for the target Kilocore target system, we use a concentrated 2D-mesh network, which is made up of 64 routers where each router has a radix of 9 (see Figure 2). The 9 Input/Output (I/O) ports of the each router are connected to the routers in the east, west, north, and south directions, one 4MB L2 bank and four tiles through a local switch that allows access to the 4 cores in each tile. Round-robin arbitration is used within each tile and in each router. These 64 routers are placed uniformly across the processor. In case of a private cache hierarchy, each router has a radix of 8. It connects to four tiles (16 cores, each core has its own private L1/L2 caches) and four inter-router links. The concentrated 2D-mesh design utilizes an X-Y routing scheme and a credit-based flow control. We assume 2-cycle pipelined routers and 1-cycle inter-router links. This makes a zero-load latency of 46 cycles for the longest path

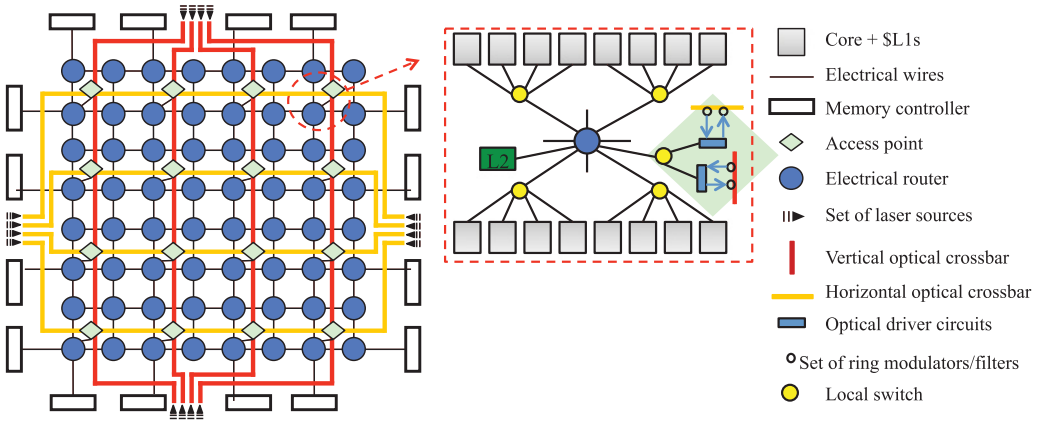


Fig. 3. Hybrid NoC (HybNoC) with shared L2 cache hierarchy. For the 400mm^2 chip size, maximum waveguide length is 40mm.

from the tile at one corner to the tile at the diagonally opposite corner (30 cycles in the routers + 1 input link + 14 inter-router links + 1 output link).[1]

4.2. Electrical Concentrated 2D-Mesh and Photonic Crossbars NoC (HybNoC)

As silicon-photonic links have lower latency and consume lower data-dependent power as compared with conventional repeater-inserted long electrical links, we study a NoC that is composed of an electrical concentrated 2D-mesh sub-network for short-distance communication (this sub-network also uses concentration as the ElecNoC), and a photonic sub-network for long-distance communication. We call this NoC design Hybrid NoC (HybNoC for short), and it is illustrated in Figure 3.

Note that we use distance as the main metric to distribute traffic in our HybNoC. The main motivation here is to minimize the communication latency for all types of messages (transmitted between any two communicating nodes cores, caches, and MCs), so we can maximize application performance. An alternate strategy for traffic distribution in the HybNoC could be using message sizes to distribute traffic. In this approach, large data messages are transferred through a high-bandwidth low-diameter high-radix photonic sub-network (like bus, clos, or butterfly), whereas short control messages are transmitted through a low-bandwidth high-diameter low-radix electrical sub-network (like mesh or concentrated mesh). However, we did not adopt this strategy. The reason is that in a large Kilocore system the zero-load latency for the longest path from the core at one corner to the core at the diagonally opposite corner would be much larger in a low-diameter high-radix electrical sub-network than in the high-diameter low-radix photonic sub-network (e.g., 46 cycles vs. 18 cycles as explained below if we take advantage of the photonic sub-network). This means that even if we provide a large amount of bandwidth in all the electrical sub-networks channels, all critical short messages (e.g., read requests and coherence related invalidation messages) will have high latency, which will lower application performance. To avoid such a scenario, we use distance as the metric to distribute traffic so we can minimize the latency for all kinds of messages and in turn maximize application performance.

For the photonic sub-network, we utilize a crossbar-based topology as photonic links provide more energy-efficient and lower latency global communication. Moreover, the

[1]For the three considered NoC designs, we also model another clock cycle from the tile's local switch to the target core and vice versa.

crossbar topology is easier to program. In particular, we utilize a multi-crossbar NoC whose layout is illustrated in Figure 3 and is composed of four vertical Multiple-Write-Single-Read (MWSR) crossbars (with channels routed in the vertical dimension) and four horizontal MWSR crossbars (with channels routed in the horizontal dimension). The interface between the electrical sub-network and the photonic sub-network is implemented through 16 access points (AP) that are uniformly distributed across the manycore system. Each AP connects a router from the electrical sub-network with a pair of vertical and horizontal crossbars in the photonic sub-network. Note that moving a message from a vertical crossbar's channel to a horizontal crossbar's channel (or vice versa) requires optical-to-electrical and electrical-to-optical conversions. We have considered the followings conversions when implementing the Hybrid NoC: OE (from photonic horizontal crossbars channel to AP), AP crossing (this is an electrical 3×3 crossbar), and and EO conversion (from AP to photonic vertical crossbars channel). The electrical concentrated 2D mesh in the HybNoC is the same as the ElecNoC topology introduced in the previous section, except for one important difference: It implements a smaller channel bandwidth as part of the network traffic is diverted to the photonic sub-network.

Each photonic crossbar in the HybNoC interconnects four APs and two MCs. Each AP includes the photonic devices to convert the signals between an electrical medium and an optical medium. Moreover, each AP includes one 3×3 local electrical crossbar. The three input ports (and output ports) of the electrical crossbar are connected to one of the vertical photonic crossbars, one of the horizontal crossbars, and one of the routers in the electrical sub-nework. In Section 5.2, we describe the experimental methodology carried out to size the HybNoC, that is, determining the channel bandwidth for the electrical inter-router links, and the bandwidth for the photonic crossbars.

To route a message through the HybNoC, the source core's network interface controller estimates the Manhattan distance to destination node through the electrical sub-network. This distance is compared against certain threshold value that indicates whether the path can be considered as *short* path or as *long* path. If the path is *short*, then this means that the electrical sub-network is the cheapest path in terms of energy consumption and network latency. Otherwise, if the path is considered *long*, the photonic sub-network is chosen as silicon-photonic links are more energy-efficient and fast for long-distance communications. For our target system, we observed that if the distance between the source and destination is larger than 12 hops (in the electrical sub-network), then the photonic sub-network provides a lower latency and more energy-efficient transmission of packets.

As a result of this routing scheme, the HybNoC has significantly lower zero-load latency for packets that need to travel across the chip. In particular, the zero-load latency for the longest path from the tile at one corner of the chip to the tile at the diagonally opposite corner is reduced to just 18 cycles (recall that 46 cycles are required for the ElecNoC): The packet would travel through two electrical routers (2 cycles each), two eletrical links between router and AP (1 cycle each), three APs (2 cycles each), and two photonic crossbars (3 cycles each). Note that, in case of private L1/L2 cache hierarchy, the routers illustrated in Figure 3 has lower radix as the shared L2 cache bank is not used but private L2 caches per core are employed.

4.3. Photonic Multi-Bus NoC (PhotoNoC)

This NoC is completely implemented using photonic link technology. Note that among all types of topologies explored so far to implement a photonic NoC, high-radix and low-diameter topologies are commonly accepted to be the appropriate topologies when considering silicon-photonic link technology. In this work, we have chosen a multi-bus NoC (PhotoNoC for short) similar to Chen and Joshi [2013], which is depicted

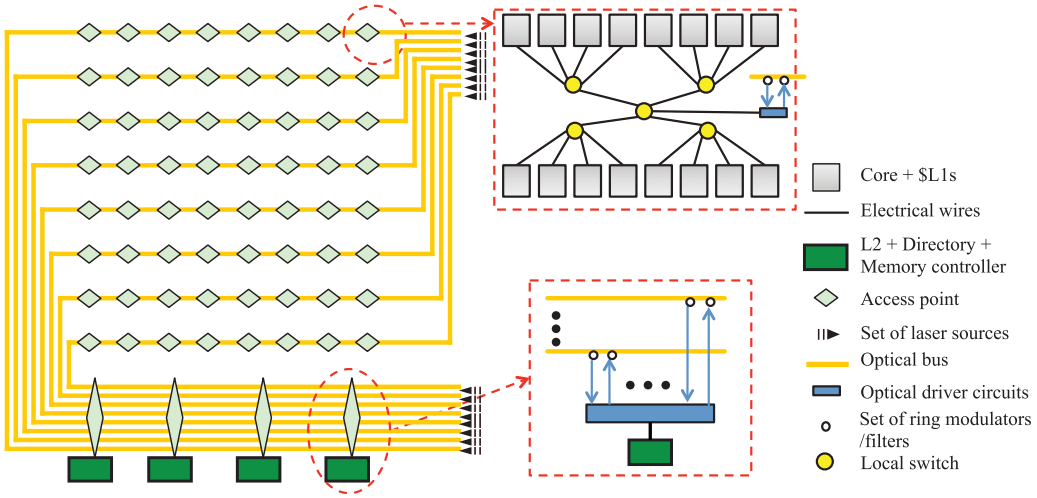


Fig. 4. Photonic Multibus NoC (PhotoNoC) with shared L2 cache hierarchy. Each of the 8 buses consists of two uni-directional buses: one from core to L2/MC direction and another one in the reverse direction. For the 400-mm^2 chip size, maximum waveguide length is 60 mm.

in Figure 4. We chose this topology as it has been demonstrated to be efficient for application performance, and it can be optimized to reduce laser power dissipation dynamically. PhotoNoC is made up of 16 silicon-photonic Multiple-Write-Multiple-Read (MWMR) buses (8 in the core-to-L2/MC direction and 8 in the L2/MC-to-core direction). Each bus is attached to 8 access points (AP) at core sides (16 cores share one AP) and 8 APs at L2/MC side (2 MC and 8 L2 shared banks share one AP). We need to use concentrations at the L2/MC side and the core side to reduce the number of APs and associated ring modulators and ring filers along each silicon-photonic bus and, in turn, reduce the laser power consumption. Note that in the case of the private L1/L2 hierarchy, the design of the PhotoNoC would be similar to this, except that the we would replace L2/MC with MC. As shown in Figure 4, for simplicity all MCs and shared L2 cache banks in the PhotoNoC-based Kilocore system are placed along one edge of the chip.

Each uni-directional bus uses one dedicated wavelength for token arbitration and two dedicated wavelengths for notification. In each uni-directional bus for core-to-L2/MC communications, the eight APs at the core side use token-based arbitration to compete for the use of the bus. For global arbitration of the MWMR buses, we leverage a token stream based arbitration similar to FeatherWeight [Pan et al. 2011]. To allow a well-organized clock distribution, we assign an entire central processing unit (CPU) clock cycle for token-stream distribution and arbitration; therefore, the ring filters at source APs only need to be synchronized at the precision of CPU clock frequency (1Gb/s), instead of the higher modulation speed of the silicon-photonic links (10Gb/s).[2] Tokens are circulated among the eight APs at the core side through a dedicated silicon-photonic token wavelength. As one AP at the core side obtains the access to the bus, it notifies the target destination AP at the L2/MC side through the two dedicated silicon-photonic notification wavelengths. The notification is composed of 8 bits (1 bit for each destination AP). After the destination AP checks its associated notification bit and knows that it is the intended destination, it tunes the ring filters on the data channels. Then the

[2]We can assume that standard serializer/deserializer circuits are used for matching the difference in frequency between CPU cores and silicon-photonic link.

communication between source AP and destination AP is established, and data are transmitted through the 103 data wavelengths[3] in the following cycles. Therefore, compared to the 103 wavelengths for data communication, the hardware overhead of token wavelength and notification wavelength is minimal. A similar mechanism is used in each silicon-photonic bus for L2/MC-to-core communications, in which the tokens are circulating among APs at the L2/MC side, and the data wavelength and notification wavelengths of each bus are transmitting the information from the APs at L2/MC side to the APs at core side. To reduce the number of silicon-photonic buses, we do not implement silicon-photonic buses for direct core-to-core communications. All core-to-core packets are transmitted over one core-to-L2/MC bus to the coherency directory located at the L2/MC and then forwarded onto one L2/MC-to-core bus towards its final destination core. The zero-load latency for transmission of a packet through a silicon-photonic bus in PhotoNoC is seven clock cycles: one cycle at injecting AP, one cycle for token based arbitration among AP at source side, one cycle for notifying and tuning the AP at receiver side, three cycles for transmission on the optical buses, and one cycle at the ejecting AP.

5. EVALUATION METHODOLOGY

5.1. Simulation Platform

We chose the parallel, x86-based full-system Sniper simulator [Carlson et al. 2011] for our evaluation. We use Sniper 5.3 with a cycle-level core model (called the reorder buffer (ROB) core model in Sniper 6.0) in order to provide higher accuracy of the simulated cores' pipeline without noticeable loss in simulation speed. In our work, Sniper 5.3 was extended to implement all the three NoC configurations explained in Section 4 and was configured to run simulations for all cache hierarchies explained in Section 5.3. Sniper 5.3 can be easily interfaced with the latest McPAT 1.0 tool [Li et al. 2009a], which can be used to obtain power dissipated by the processor cores and cache hierarchy of the manycore system when running standard application benchmarks. We use standard technology scaling rules to scale down the power values obtained by McPAT 1.0 at 22nm to the target 11nm technology.

To estimate the power of our proposed NoCs, we distinguish between the electrical components and the photonic components as described in Section 4. First, the power for the electrical network components of ElecNoC and HybNoC is estimated using a detailed transistor-level circuit model at the target 11nm technology. In particular, we use the predictive technology model PTM-MG [Yu et al. 2012] in our estimation of NoC power. We use the RC model to calculate the power dissipation of wires and crossbars in routers [Wang et al. 2003]. The routers use a standard matrix-based crossbar with an static random-access memory (SRAM) array for holding flits at every input port. We implement one-cell SRAM and two-cell SRAM with PTM model and estimate the per bit energy of larger SRAM [Liang et al. 2007]. Second, the power calculation of the silicon-photonic channels implemented in HybNoC and PhotoNoC is estimated by using a range of fJ/bit values. These fJ/bit values include all three sources of energy consumption: electrical-optical-electrical (E-O-E) conversion, thermal tuning, and laser source. The main motivation for considering a range of fJ/bit values is that a variety of photonic device designs, integration approaches, physical layouts, and laser source designs have been proposed over the years. Various combinations of device designs, integration approaches, physical layouts, and laser source designs can be used to achieve the same energy per bit value. For instance, one could use monolithic integration of photonic devices to reduce the E-O-E conversion cost, but that would lead

[3]As we will explain in Section 5.2, each unidirectional logical bus requires 128GBytes/s bandwidth. 128GB/s per bus divided into 1.25GB/s per wavelength = 103 wavelengths/bus.

Table II. Configuration Parameters for the Three Simulated NoCs. At the Listed Channel Bandwidth the Corresponding NoC Achieves Maximum Performance for the Workloads under Study

NoC	Network parameters
ElecNoC	64 GBytes/s per channel
HybNoC	Electrical sub-NoC: 16 GBytes/s;
	Photonic sub-NoC: 32 GBytes/s per crossbar's channel
PhotoNoC	128 GBytes/s for each uni-directional silicon-photonic bus

to a higher laser source cost. On the other hand, one could use 3D integration of photonic devices to reduce the laser source cost, but that would increase the E-O-E conversion cost. Both approaches could be optimized to achieve a comparable fJ/bit value. To ensure that we are using a valid range of fJ/bit values for analyzing the HybNoC and PhotoNoC (and also to ensure the feasibility of the HybNoC and PhotoNoC), we design and analyze the HybNoC and PhotoNoC using state-of-the-art silicon-photonic link technology. The details of this analysis are provided in the next section.

5.2. Networks-on-Chip Setup

We configured the NoCs proposed in Section 4 for the target Kilocore system to yield the maximum performance when running the workloads that will be introduced in Section 5.4. To accomplish that, we carried out a preliminary analysis to determine the best-performing configuration for each of the three proposed NoC designs. To do that, for each of the three NoC designs, we executed each workload using a range of channel bandwidths (from 32 bytes/cycle to 256 bytes/cycle) to determine the channel bandwidth at which the application performance saturates. As different cache hierarchy configurations can affect application performance (e.g., private L2 cache vs. shared L2 cache), and in turn affect offered bandwidth for the three NoC designs, this study also considers all cache hierarchy configurations explained in Section 5.3. Essentially, every experiment on channel bandwidth for each of the three NoC is repeated for every cache hierarchy configuration. As a result, we are obtaining the best channel bandwidth for each NoC design and all cache hierarchy configurations under study (Section 6.1 will discuss in detail the cache hierarchy configuration that obtains the best application performance). Table II summarizes the resulting network parameters that achieve the highest NoC performance for the three NoCs and cache configurations evaluated in this work.

To ensure that the PhotoNoC and HybNoC systems that we are considering are feasible, we have conducted detailed power calculations and on-chip area overhead of both NoCs considering state-of-the-art measurements and projections of photonic link technology [Orcutt et al. 2012; Georgas et al. 2014; Liu et al. 2014; Zheng et al. 2012] (see Table III). We assume a photonic link design that enables up to 64λ for each waveguide (32λ in each direction), with a 10Gbps wavelength modulation speed that can be realizable according to recent works [Liu et al. 2012]. The latency of the photonic link is assumed to be three cycles (one cycle in flight and one cycle each for electrical-to-optical (E/O) and optical-to-electrical (O/E) conversion) + serialization latency + latency due to contention for NoC resources. All the silicon-photonic links are driven by off-chip laser sources. The waveguides are single mode and have a pitch of 4μm to minimize the crosstalk between neighboring waveguides. Modulator ring and filter ring diameters are $\sim$10μm. Table IV shows the number of different photonic components and the area occupied by those devices in both HybNoC and PhotoNoC.

Our calculations using the photonic link technology parameters shown in Table III, the number of photonic components shown in Table IV, and the workload with the highest bandwidth demand in Section 6.4 show that the PhotoNoC and HybNoC topologies have an average energy per bit cost of 923fJ/b and 429fJ/b, respectively. These values are within the 100fJ/b to 2pJ/bit range that we have considered in this article. Our

Table III. Silicon-Photonic Link Technology Assumed for HybNoC and PhotoNoC

Laser source efficiency	15%
Coupler loss, Splitter loss	1dB, 0.2dB
Modulator insertion loss	1dB
Waveguide loss	1dB/cm
Crossing loss	0.05dB
Filter through loss	1e-3dB
Filter drop loss	0.5dB
Photodetector loss	0.1dB
Non-linearity loss	1dB
Modulator driver circuit energy	0.035pJ/b
Receiver circuit energy	0.11pJ/b
Thermal tuning power	16μW/K
Receiver sensitivity	-17

Table IV. Number of the Different Photonic Components in HybNoC and PhotoNoC—We Use Channel Bandwidths Illustrated in Table II

Network	WL	MD	FL	WG	PDA (% total chip area)
HybNoC	1,248	7,488	1,248	40	22.4 (5.6)
PhotoNoC	1,648	13,184	13,184	52	43.68 (10.9)

We assume 10 Gbit/second/wavelength; 32 wavelengths/waveguide/direction. WL = Wavelengths, MD = Modulators, FL = Filters, WG = Waveguides. PDA = Photonic device area in mm^2 assuming 10μm-radius rings, 4μm waveguide pitch and a 400 mm^2 floorplan area. We assume waveguide lengths of Figure 3 (HybNoC) and Figure 4 (PhotoNoC).

area calculations show that the area of the photonic devices of the PhotoNoC would be 43.6mm^2, which represents a 10.9% of the total chip area. This overhead is acceptable for both monolithic as well as 3D integrated design. In the case of our HybNoC system, the area overhead of the photonic devices is 5.6% of the chip. Thus, from an area perspective, HybNoC can be implemented using both monolithic as well as 3D integration. Therefore, we can confirm that our proposed NoC architectures are feasible in terms of both energy consumption and on-chip area overhead.

Apart from these three NoC designs for the target system, our experimental evaluation also considers a hypothetical ideal NoC where all of its routing paths have a fixed three-cycle zero-load latency (one cycle at the source core's NI, one cycle for channel traversing, and one cycle at destination core's NI). We refer to this NoC as IdealNoC.

5.3. Cache Hierarchy Configurations

In Section 3, we explained that the target Kilocore system has an inclusive cache hierarchy composed of two levels of cache memories (L1 and L2 caches). We configure Sniper simulator to simulate the two L2 configurations under evaluation—a private L2 and a shared L2 cache. In addition, we explore the opportunities to take advantage of the larger bandwidth density of silicon-photonic links in comparison to conventional electrical RC-based links. For that, we study other cache hierarchy configurations that can potentially provide higher application performance at the cost of higher network traffic.[4] In particular, we explore cache hierarchy configurations that use an efficient prefetching technique and aggressive non-blocking caches (also named as lockup-free caches).

[4]Higher memory bandwidth can be also required but we are leveraging the PIDRAM system that provides much higher bandwidth for processor-to-memory communication.

The prefetching technique [Tullsen and Eggers 1995] was proposed to hide the large memory access latency resulting from the well-known processor-memory performance gap. By using prefetching, a cache can fetch data from lower levels of the memory hierarchy (closer to main memory) in advance (i.e., before memory blocks are actually requested by the processor cores), so it can potentially avoid cache misses. In this approach, all memory accesses are analyzed at runtime to detect memory access patterns that can take advantage of prefetching. Between L1 and L2 caches, the L2 cache is a better place to use prefetching. The reason is that the L1 cache is too small, and there is a high likelihood of evicting useful data from L1 compared to L2 whose larger size reduces the cache pollution effect.

The main drawback of using prefetching is the cost in terms of extra bandwidth that may be required due to both cache pollution and the unnecessary movement of memory blocks (that will never be referenced by the processor cores) to upper levels of cache (closer to the processor core). We propose to alleviate this likely negative effect on performance (and also on energy consumption) when using a prefetching technique by leveraging the large bandwidth density of our silicon-photonic-based NoCs (HybNoC and PhotoNoC) along with the usage of the PIDRAM interface in our target 1024-core system. Among all implementations of prefetching in the literature, we consider a global history buffer (GHB [Nesbit and Smith 2005]) that holds the most recent miss addresses in FIFO order and contains a more complete cache miss history in order to improve accuracy of prefetching. This implementation for prefetching is already integrated in Sniper simulator.

Non-blocking caches allow the processor cores to continue executing instructions while a cache miss is being handled. This way, by reducing memory stalls, the memory-level parallelism is increased that in turn improves application performance. To achieve that, a non-blocking cache integrates a hardware structure that is used by the cache controller to keep track of outstanding misses. This structure is called the Miss Status Holding Register (MSHR) [Kroft 1981]. Every entry of the MSHR stores the physical address of the requested memory block that produced the cache miss, along with other information such as the word in the block that was requested, the destination register number where the word will store the data after the cache miss is processed, and so on.

The main drawback of non-blocking caches is that it leads to extra network/memory traffic due to the larger number of in-flight cache misses that need to be solved across all levels of the memory hierarchy. Similarly to what we proposed for mitigating the negative side-effect of prefetching, we observe here an opportunity to leverage the large-bandwidth density of silicon-photonic link technology to efficiently accommodate the extra data traffic generated by non-blocking caches. Sniper simulator supports non-blocking cache that can be configured by setting the maximum number of outstanding misses (the number of entries for the MSHR). In this work, we study aggressive non-blocking caches with an MSHR configuration of 16 entries (the maximum prefetch degree evaluated in Nesbit and Smith [2005])—up to 16 outstanding misses can be handled at a particular time by a single cache controller. To take full take advantage of non-blocking caches, we configure the target Kilocore system with non-blocking caches for both levels of the cache hierarchy (L1 and L2) .

Table V summarizes all the cache configurations for the target manycore system under evaluation. The first column shows the codename for each configuration that will be used in next sections. Note that a cache hierarchy that combines both non-blocking caches and a prefetching technique has not been included in this work. The reason is that we observed that this configuration provides no performance advantage as compared to only using non-blocking caches or only using prefetching technique.

Table V. Cache Hierarchy Configurations Evaluated for the Kilocore System

Cache Hierarchy Configuration	Description
(1) PL1-PL2	Private L1 and Private L2
(2) PL1-SL2	Private L1 and Shared L2
(3) PL1-PL2pref	Private L1 and Private L2 with prefetching activated
(4) PL1-SL2pref	Private L1 and Shared L2 with prefetching activated
(5) PL1nb-PL2nb	Non-blocking Private L1 and Non-blocking Private L2
(6) PL1nb-SL2nb	Non-blocking Private L1 and Non-blocking Shared L2

Table VI. List of Benchmarks Evaluated in this Work

Suite	Applications	Input Data Sets
SPLASH2	cholesky	tk29.O matrix
	fft	1M complex data points
PARSEC	canneal, fluidanimate, swaptions	sim_medium
NAS	cg, ep, is, ua	large
GRAPH500	graph500	scale=20, edges=16
UHPC	sar, graph, md	graphoutfile_large.bin, large.ini, water_xlarge.tpr
	chess, shock	large

5.4. Application Workloads

To quantify the efficiency of the three different NoCs proposed for our target Kilocore system along with the different cache hierarchy configurations under study, we explore a broad variety of applications selected from different benchmarks suites from NAS Parallel Benchmarks (NPB) [Bailey et al. 1994], Stanford Parallel Applications for Shared-Memory (SPLASH-2) [Woo et al. 1995], Princeton Application Repository for Shared-Memory Computers (PARSEC) [Bienia et al. 2008], Mantevo project [Heroux et al. 2009], GRAPH500 [Murphy et al. 2010], and UHPC [Campbell et al. 2012]. A summary of the applications with their respective input data sets is shown in Table VI. To choose the applications, we carried out a preliminary performance evaluation study in which we obtained the applications that scale well to 256 cores (the minimum amount of threads per application simulated in this work as it is explained in Section 5.4). In addition, for an adequate evaluation of our different NoCs, another important consideration was to select the applications that are good representative of applications of the new Big Data era. These applications are expected to have very high data traffic injection rate for each core in the target manycore system. As Sniper does not run an Operating System and has limited support for a message-passing programming model (e.g., MPI), we could only use benchmark suites for simulation in Sniper that follow a shared-memory programming model (i.e., applications written in OpenMP or POSIX threads)—we could not simulate Big Data benchmarks suites such as BigDataBench [Wang et al. 2014] that supports a complex software stack based on Hadoop, Spark, and MPI.

We configure a diverse set of workloads using the above-described applications as the building blocks for studying the NoCs and cache configurations in the target Kilocore system. We consider two different types of workloads: multi-threaded workloads, in which the applications are configured with a number of threads equal to the number of cores of the target system (1024), and multi-programmed workloads in which distinct applications are running simultaneously in the target system. For the multi-programmed workloads, we partitioned the Kilocore system into four partitions where in each partition there is a 256-thread application. This way, applications that do not scale well to 1024 cores but generate large data traffic can also be studied. Moreover, this approach also helps us explore a more heterogeneous scenario.

We concentrate our analysis on the parallel phases of the applications' execution. Given that execution times of applications differ, the total execution time of our

multi-programmed workload was set to be equal to the largest value of the four execution times of the four applications. During the simulation period, the other three applications were restarted (one or more times) whenever they finished execution to ensure there is NoC traffic at all times.

6. EXPERIMENTAL RESULTS

To evaluate the different NoCs (ElecNoC, HybNoC, and PhotoNoC) proposed for our target Kilocore system, we start by determining the best-performing cache hierarchy for the target system among all configurations introduced in Table V. This ensures that the comparison of the NoCs will not be performed with a sub-optimal cache hierarchy.

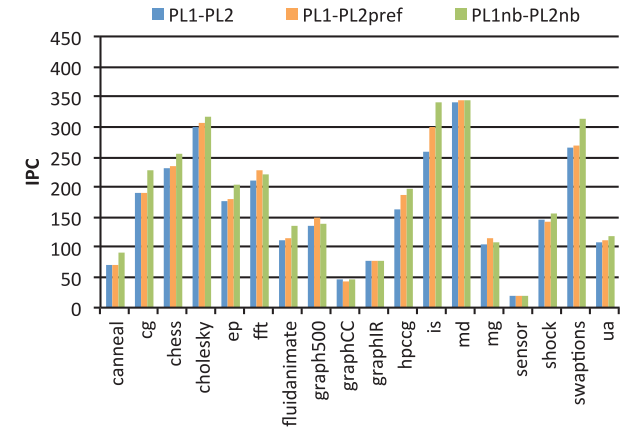
6.1. Best-Performing Cache Hierarchy

To get insight into the best-performing cache hierarchy, we perform a preliminary study utilizing the IdealNoC for the target system. Recall that IdealNoC is made up of point-to-point three-cycle fixed latency channels between any pair of communicating nodes, and it does not model congestion and contention scenarios in the network. This ensures that the cache hierarchy performance is isolated from network congestion and contention scenarios that can occur in the three proposed NoC designs. As a result, we avoid hurting maximum achievable performance of the cache hierarchies by NoC performance.

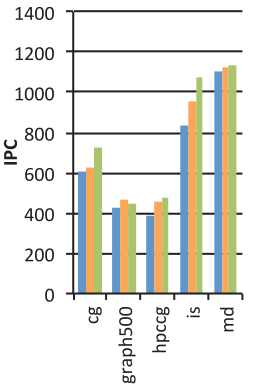
As several of the applications from the benchmark suites under study were not designed with a Kilocore processor in mind, they do not scale well to 1024 threads. Hence, apart from 1024-thread application workloads, we will study workloads composed of 256 threads each (the building block of our multi-programmed workloads). To explore the 256-thread applications, we start by analyzing these workloads for a scaled-down 256-core version of the target Kilocore system. This guarantees maximum application performance for the smallest workload utilized in this work, so we can know the performance limits of the cache hierarchy configurations under study. Note that a 256-core system has a lower network diameter than the Kilocore system when using the ElecNoC and HybNoC designs (they use a concentrated 2D-mesh topology), which means fewer numbers of hops for packet traversing, thereby shortening average network latency. We analyze the private and shared L2 cache hierarchies by running two separate sets of simulations. We also explore the use of prefetching and non-blocking caches for improving application performance.

Figure 5(a) illustrates the application performance (using Instructions-Per-Cycle (IPC) metric) when using the typical private L2 cache, the private L2 cache with prefetching, and the private non-blocking L2 caches. Note that for some benchmarks (e.g., `cholesky` or `md`) the aggregated IPC is greater than the number of cores (256 in this case) in the manycore system. The reason is that each core is a two-way superscalar OoO processor that can commit up to two instructions per clock cycle. We also present the offered bandwidth in Figure 5(c).

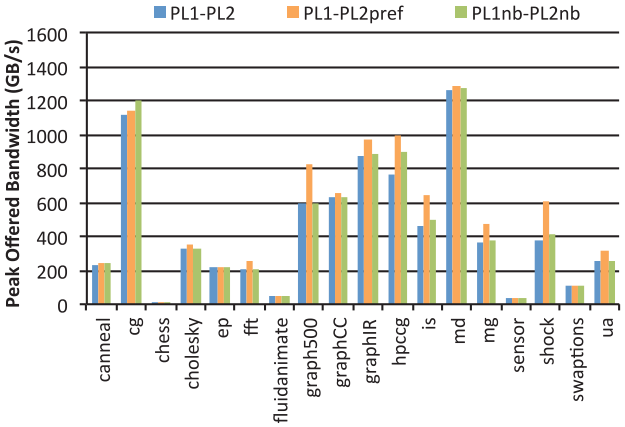
As we can see in Figure 5(a), the performance of `canneal`, `graphCC`, `graphIR`, `sensor`, and `mg` does not scale well to 256 cores. The main reason is that these benchmarks spend a significant fraction of their execution time (more than 50% on average) on synchronization operations among threads, and this percentage increases with thread count. When compared to the baseline PL1-PL2 configuration, we can see that prefetching (PL1-PL2pref) achieves up to 15% (4% on average) IPC improvement. This increased performance comes with up to 60% (16% on average) increment in NoC traffic as shown in Figure 5(c). In case of non-blocking caches (PL1nb-PL2nb), as compared to prefetching, higher improvement in IPC is obtained (up to 31%; 11% on average), with lower increment in NoC traffic (up to 17%; 3.7% on average). The reason is that, in general, prefetching incurs significant cache pollution at the L2 cache level that degrades



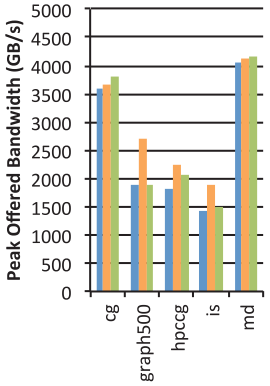
(a) 256-thread applications on 256-core system



(b) 1K-thread applications on Kilocore system



(c) 256-thread applications on 256-core system

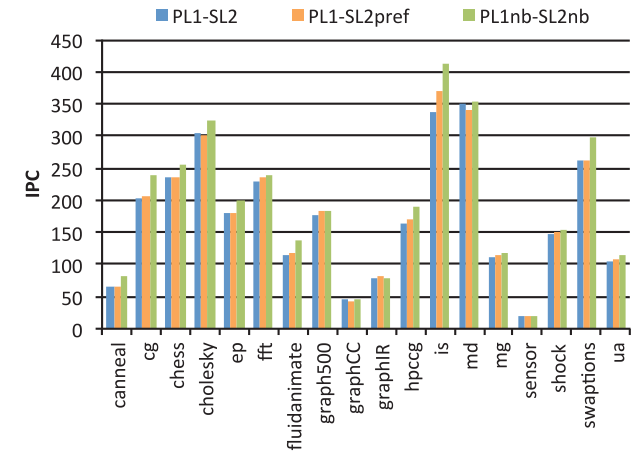


(d) 1K-thread applications on Kilocore system

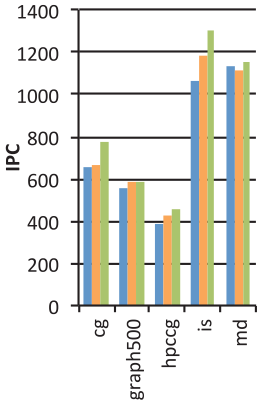
Fig. 5. Performance (in IPC) and peak offered bandwidth (GBytes per second) for a 256-core system ((a) and (c)) and a Kilocore system ((b) and (d)) with private L2 for all the applications under study. Here the NoC is an IdealNoC. We consider three different cache configurations: typical private L1 and private L2 (PL1-PL2), private L1 and private L2 with prefetching (PL1-PL2pref), and private non-blocking L1 cache and private non-blocking L2 cache (PL1-PL2nb).

application performance (the L2 cache miss rate increases by more than 40% on average) and demands more network bandwidth.

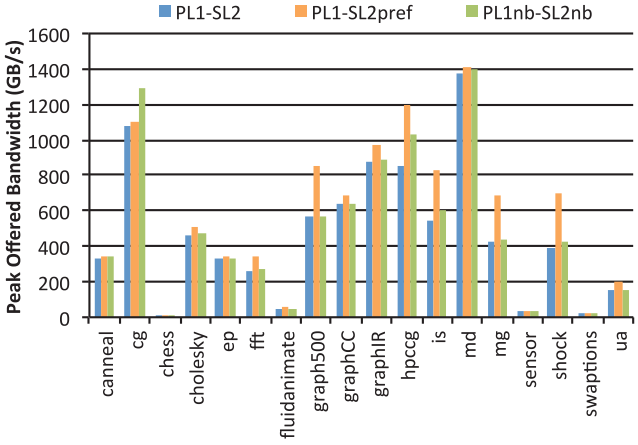
From Figure 5(c), we can see that the maximum offered traffic of 1.27TB/s is reported by `md` from UHPC suite. It is worth noting that, if we are to support such an amount of bandwidth, then the implementation of the NoC is of paramount importance. As we will explain in Section 6.2, even though ElecNoC has been sized to saturate application performance (further details in Section 5.2), this purely electrical NoC cannot provide the amount of bandwidth and performance achieved by the IdealNoC while consuming a reasonable amount of power. Nevertheless, our results reveal that silicon-photonic link technology, integrated into our HybNoC and PhotoNoC designs, can help provide



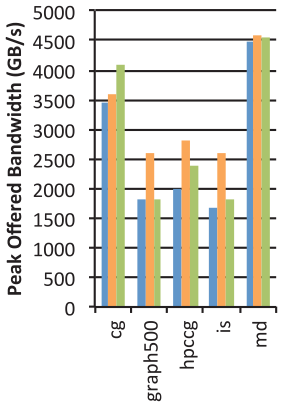
(a) 256-thread applications on 256-core system



(b) 1K-thread applications on Kilocore system



(c) 256-thread applications on 256-core system

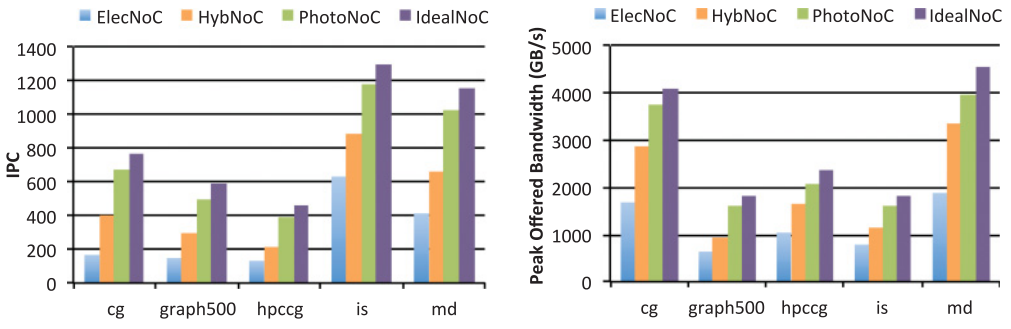


(d) 1K-thread applications on Kilocore system

Fig. 6. Performance (in IPC) and peak offered bandwidth (GBytes per second) for a 256-core system ((a) and (c)) and a Kilocore system ((b) and (d)) with shared L2 for all the applications under study. Here the NoC is an IdealNoC. We consider three different cache configurations: typical private L1 and shared L2 (PL1-SL2), private L1 and shared L2 with prefetching (PL1-SL2pref), and private non-blocking L1 cache and shared non-blocking L2 cache (PL1-SL2nb).

bandwidth close to that of an IdealNoC while consuming lower power than the ElecNoC design (further details in Section 6.4).

Figure 6(a) shows the same set of experiments but when considering a shared L2 cache (PL1-SL2). In this case, as compared to PL1-SL2 cache hierarchy, prefetching (PL1-SL2pref) achieves up to 10% performance improvement (1.3% on average), while increasing traffic by up to 80% (22% on average). On the other hand, through non-blocking caches (PL1nb-SL2nb) an IPC improvement of up to 25% (9% on average) is observed, whereas data traffic is increased by up to 21% (5% on average). For shared L2 cache, too, the maximum offered bandwidth is observed for `md` that requires 1.4TB/s.



(a) Performance in instructions per cycle

(b) Peak offered bandwidth

Fig. 7. Simulations for a Kilocore system with three different NoC configurations: ElecNoC, HybNoC, and PhotoNoC. We use non-blocking private L1 and non-blocking shared L2 as that was the best-performing cache hierarchy.

The conclusions of this preliminary study are as follows:

—In general, compared to private L2 (PL1-PL2), shared L2 (PL1-SL2) reports higher IPC values and has higher offered bandwidth due to more effective usage of on-chip cache capacity. On average, shared L2 reports 4.7% higher IPC and 4.8% higher offered bandwidth compared to private L2.
—Prefetching improves IPC and can increase offered bandwidth by up to 80%. However, there are some benchmarks such as `canneal`, `ep`, `sensor`, and `swaptions` where a significant amount of new cache data conflicts is observed due to prefetching (we see that L2 cache miss rate increases by more than 40% on average), thereby resulting in a minimal performance improvement as it is shown in Figure 5(a) and Figure 6(a).
—Non-blocking caches through advanced MSHR with 16 entries (due to higher MLP of up to 16 outstanding misses) achieve the maximum performance improvement (an average of 10% higher IPC compared to the baseline PL1-PL2 and PL1-SL2 cache configurations) and increases the amount of traffic injected in the NoC by an average of 20%.

Based on this preliminary study, we conclude that the best-performing cache hierarchy configuration for 256-thread applications running on a 256-core system is non-blocking private L1 and non-blocking shared L2 cache (called PL1nb-SL2nb), because it has the highest IPC for all evaluated workloads.

As shown in Figure 5(b), Figure 5(d), Figure 6(b), and Figure 6(d), we extend this experimental evaluation for the target Kilocore system with the IdealNoC using the applications that scale well to 1024 threads. As we can see, the best-performing cache hierarchy configuration is the same as in the evaluation for the 256-thread applications running on the 256-core system. Further discussion on these applications for the proposed ElecNoC, HybNoC, and PhotoNoC will be provided in Section 6.2. In the remaining sections of this work, we will utilize this configuration to compare the three NoC designs with the IdealNoC. From this preliminary study, we also select the applications that scale well with core count and have large offered network bandwidth. In particular, we choose the following applications: `cg`, `cholesky`, `graph500`, `graphIR`, `hpccq`, `is`, `md`, and `shock`.

6.2. Study of 1024-Thread Applications on Kilocore System

Figure 7 compares the three proposed NoCs (ElecNoC, HybNoC, and PhotoNoC) in terms of performance (measured as IPC) and in terms of offered bandwidth (in terms

Table VII. Multi-programmed Workloads: Four 256-Thread Applications Each

Codename	Applications Used
Workload A	md, md, md, md
Workload B	cholesky, graph500, is, shock
Workload C	cg, hpccg, graphIR, md

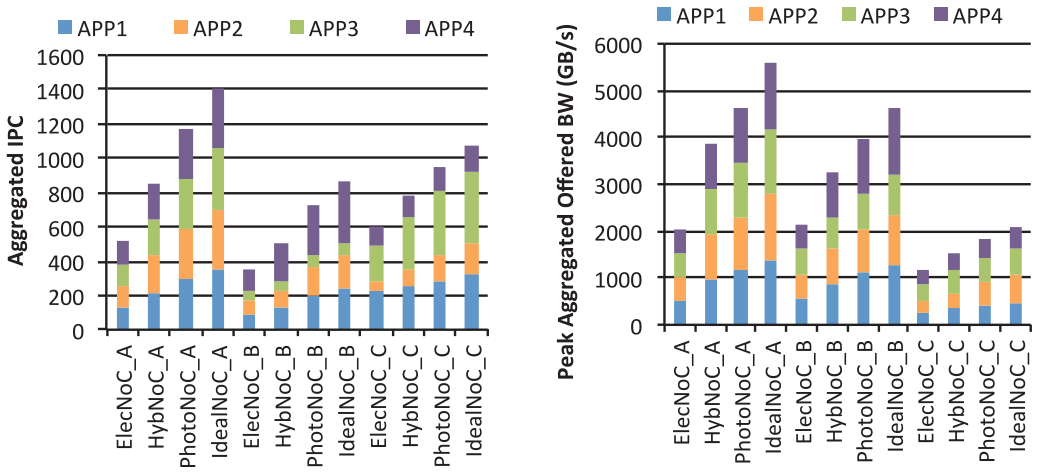
of GBytes per second), when a single 1024-thread workload is running onto the Kilocore system. We also include the IdealNoC simulation results for comparison purposes. Recall that each of the three NoCs were sized to reach saturation of application performance (further details in Section 5.2), so we will quantify the limits of each NoC, and we will show the benefits of integrating silicon-photonic link technology in the NoC architecture. We selected the applications that scale well to 1024 cores. These applications are cg, graph500, hpccg, is, and md.

Figure 7(a) shows the results of the performance comparison among the three proposed NoCs and the ideal NoC. Here, the same trend is observed when comparing ElecNoC, HybNoC, and PhotoNoC to the IdealNoC. In particular, normalizing the IPCs achieved by the IdealNoC to the proposed NoCs shows that, on average, the ElecNoC, HybNoC, and PhotoNoC report performance that is 0.32×, 0.55×, and 0.88× lower, respectively, than when using an IdealNoC. Similarly to the previous section, the reason of these results can be understood by observing the limitation in terms of peak offered network bandwidth that can be supported by each of the different NoCs. This can be observed in Figure 7(b) where, on average, ElecNoC, HybNoC, and PhotoNoC can support 59%, 34%, and 11% lower offered bandwidth, respectively. The larger offered network bandwidth that can be supported by the HybNoC and PhotoNoC designs is the result of lower network diameter compared to the ElecNoC, which uses a concentrated 2D-mesh layout. Note that lower network diameter reduces the average packet latency of the communication. Moreover, the large bandwidth achieved through silicon-photonic technology in HybNoC and PhotoNoC allows these NoCs to process more packets per cycle—and, hence, they can support a higher offered bandwidth. For instance, as shown in Section 4, in the worst-case scenario for the zero-load latency, ElecNoC reports 48 clock cycles, while HybNoC achieves 18 clock cycles, and PhotoNoC reports just only 7 clock cycles. That is why, in general, application performance is the highest when using PhotoNoC and the lowest when using ElecNoC.

6.3. Study of Multi-Programmed Workloads on Kilocore System

In the previous section, we demonstrated that PhotoNoC is the most suitable NoC configuration in terms of performance and offered network bandwidth for the target manycore system running 1024-thread application workloads. In this section, we further explore the limits of the proposed NoCs by analyzing multi-programmed workloads. Note that a workload composed of four 256-thread applications may jointly require more bandwidth than a single 1024-thread application. The reason is that, in the latter scenario, the threads require more time to finish synchronization operations (e.g., barriers and highly contended lock/unlock operations). For this study, we define three multi-programmed workloads (see Table VII). Workload A is the worst-case scenario where four instances of the application with the largest offered network bandwidth according to our previous analysis (md application in our case) are used.

Figure 8 illustrates the performance comparison of the different NoCs with the multi-programmed workloads. In Figure 8(a), the aggregated IPC is calculated for each experiment and a breakdown of each individual IPC per application is also shown. As expected, the ElecNoC reports the worst performance while the PhotoNoC reports the best performance. On average, the ElecNoC has 0.44×, HybNoC has 0.63×, and



(a) Performance in instructions per cycle

(b) Peak offered bandwidth

Fig. 8. Simulations for a Kilocore system using a multi-programmed workload. We evaluate different NoCs (ElecNoC, HybNoC, PhotNoc, and IdealNoC). We study non-blocking caches and shared L2 as the best-performing cache hierarchy.

PhotoNoC has 0.85× the performance of the IdealNoC. The performance degradation for the different NoCs when compared to the IdealNoC can be understood when analyzing Figure 8(b) that shows peak aggregated offered bandwidth results (sum of offered bandwidth reported per application in the multi-programmed workload). Compared to IdealNoC, ElecNoC has 54% lower offered bandwidth, HybNoC has 30% lower offered bandwidth, and PhotoNoC has 15% lower offered bandwidth. Note that multi-programmed workload `A` (4×`md`) has the highest offered bandwidth (5.5TB/s for the IdealNoC) and 4.6TB/s can be supported by the best-performing PhotoNoC, while ElecNoC can only support a maximum of roughly 2TB/s.

6.4. Power Dissipation

From previous sections, we conclude that silicon-photonic link technology can address the performance bottleneck issue of NoC for large manycore systems. The PhotoNoC is demonstrated to be the best-performing NoC for the target Kilocore system. In this section, we study power dissipation of the proposed NoCs for the target Kilocore system by calculating power dissipation utilizing the worst-case workload `A` in order to show the maximum power required for the workloads under study. Figure 9 shows a breakdown of the overall power dissipated by the Kilocore system. In particular, we report processor core power, cache power, and the power dissipated by the NoC. For the NoC we provide a breakdown of the power consumed in the electrical components (`NoC: Electrical`) and silicon-photonic components (`NoC: Photonics`). As silicon-photonic technology is continuously evolving, we consider a range of photonic link energy costs: 100fJ/bit, 250fJ/bit, 500fJ/bit, 1pJ/bit, 1.5pJ/bit, and 2pJ/bit. This energy/bit includes E-O-E conversion energy, laser power, and thermal tuning power.

From Figure 9, we can observe that ElecNoC dissipates almost 64W of power, which constitutes 34% of the total system power. Note that the cores and caches in the target system with ElecNoC dissipate lower power compared to the core and cache when using the other NoCs. The reason for this is the lower performance achieved by the target system with ElecNoC. For the system with the HybNoC and PhotoNoC networks, the power dissipated by the NoC is smaller than the purely electrical ElecNoC power.

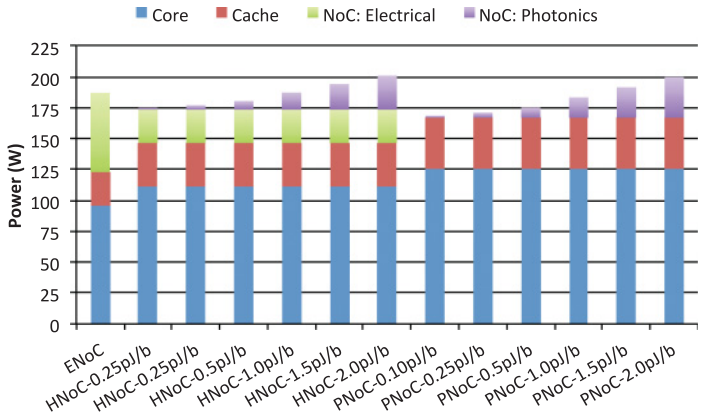


Fig. 9. Power comparison for the different NoCs for the target Kilocore system using the workload with the highest bandwidth demand: a multi-programmed workload composed of four instances of 256-thread `md`. HNoC=HybNoC; PNoC=PhotoNoC.

The 2D-mesh electrical sub-network in the HybNoC dissipates 57% less power than the 2D-mesh electrical NoC in ElecNoC. The reason is that the electrical sub-network in the HybNoC employs a 75% smaller channel size (from 64B to 16B), which in turn also reduces power dissipated by other network components such as the input buffers and crossbars in each router. The 16 electrical 3×3 APs in the HybNoC that interconnect the electrical sub-network and the photonic sub-network dissipates 33% (roughly 9W) of the electrical sub-network. The photonic sub-network dissipates 13W when using 1pJ/bit photonic link. The total power dissipation of the Kilocore system when using this 1pJ/bit optical link in the HybNoC is comparable to the power dissipation in the Kilocore system when using the ElecNoC.

In case of the system with PhotoNoC, we observe that it dissipates power comparable to the system with ElecNoC while achieving much higher performance as shown in Figure 8(a). In particular, for the 1pJ/bit configuration, the total power for the target system is very close to the 190W dissipated by the system with the ElecNoC. In this case, the photonic network accounts for only 16W, which is 9% (vs. 34% in the ElecNoC) of the total system power, as the tiles of the manycore system dissipate large power due to the higher performance of the PhotoNoC-based Kilocore system.

As we have shown, a Kilocore system with PhotoNoC reports the highest performance and has the largest offered bandwidth while consuming power comparable to the worst-performing ElecNoC. Additionally, when comparing the performance per watt metric (IPC/W) for the three NoCs in the target Kilocore system, for ElecNoC, HybNoC (1pJ/bit), and PhotoNoC (1pJ/bit), an average of 2.80IPC/W, 4.06IPC/W, and 5.51IPC/W, respectively, is obtained across all the benchmarks (multi-threaded and multi-programmed workloads) for the Kilocore system. Thus, we can conclude that the PhotoNoC is the best option for a forward-looking Kilocore system if we can develop the silicon-photonic link technology with less than 1pJ/bit.

7. RELATED WORK

Since our work explores different electro-photonic NoC architectures for Kilocore systems, this section presents an overview of the different electrical, electro-optical, and optical NoC architectures that have been proposed so far.

In terms of purely electrical NoC designs, network topologies such as shared buses have been widely adopted due to low design complexity and easy integration with simple snoop-based coherence protocols. An electrical bus, however, does not scale

well when the number of cores attached to it goes beyond 16 [Kumar et al. 2005]. To overcome this, a hierarchical segmented electrical multi-bus has been proposed [Udipi et al. 2010] that is capable of improving scalability to 64 processor cores.

Niagara [Kongetira et al. 2005] and IBM Cyclops64 [Zhang et al. 2006] utilize electrical crossbars as a network topology. However, as the radix of the crossbars increases, the power dissipation and the arbitration latency becomes impractical to be applicable for a Kilocore system. Electrical ring topology is another simple topology that has been employed for multicore systems by combining multiple rings to reduce hop count and to achieve good scalability [Ainsworth and Pinkston 2007; Seiler et al. 2008]. However, a multi-ring NoC for a Kilocore system is not feasible as it would require either long high-latency and power-hungry rings or a significant number of short rings that would considerably increase physical layout complexity.

2D-mesh topology [Ramey 2011] is popular as it is easy to integrate it on the 2D planar silicon substrate. Moreover, as compared with a ring, a 2D-mesh NoC is more scalable as network bandwidth here increases with the number of cores. However, for a Kilocore system, the network diameter of 2D-mesh NoC becomes very large. To shorten this diameter, network concentration has been proposed [Howard et al. 2011], thereby achieving a more-optimized NoC design. Our proposed ElecNoC and HybNoC systems employ concentrated 2D-mesh topology, and, in fact, the latter utilizes a multi-crossbar NoC for further reducing network diameter.

There are other topologies in the literature that try to overcome the inefficiencies of previously commented electrical NoCs. These topologies include flattened butterfly [Kim et al. 2007], clos [Scott et al. 2006], or fat trees [Ludovici et al. 2009]. However, the high radix of these topologies leads to complex switches/router nodes with a significant amount of input/output ports that would lead to high power dissipation and on-chip area overhead, thereby not being a valid solution for the Kilocore system. To overcome this, Abeyratne et al. [2013] proposed two asymmetric high-radix topologies that can be efficient for Kilocore systems. The NoC designs are based on folded-clos topology. While the authors demonstrate that proposed asymmetric NoCs report better performance than the concentrated 2D mesh, we opted to not using that topology for two reasons. First, the asymmetric NoCs are recognized to have a more complex layout while providing similar power efficiency. Second, the performance gap between the asymmetric NoC and the concentrated 2D mesh is much lower than the performance gap shown in our work when comparing ElecNoC, which is a concentrated 2D mesh, and our multi-bus based photonic NoC.

Similarly to our HybNoC, there are network implementations that integrate photonic and electrical sub-networks that leverage the large bandwidth density of silicon-photonic links to improve network performance. Pan et al. [2009] proposed a hierarchical multi-plane photonic crossbar coupled with a concentrated mesh electrical network called Firefly. Firefly implemented Reservation Assisted Single Write Multiple Read optical links to reduce power consumption. Shacham et al. [2007a] proposed a reconfigurable broadband circuit-switched on-chip nanophotonic torus network as a transmission layer with a topologically identical torus electrical network as a control layer. Li et al. [2009b] presented a planar nanophotonic broadcast bus to transmit latency-critical messages and an electrical packet switched network that handles the remaining traffic. Bahirat and Pasricha [2014] proposed a hybrid NoC fabric with concentric photonic rings coupled to a reconfigurable electrical mesh. While all these hybrid topologies are valid solutions for a Kilocore system, we opted by integrating a concentrated 2D-mesh NoC and an electronic multi-crossbar NoC. This solution is similar to that of Pan et al. [2009]; however, it targets a Kilocore system and employs concentrated routers to reduce network radix for better network efficiency. Moreover, to alleviate the expensive laser power, for instance, our multi-crossbar NoC can easily leverage the various power management techniques proposed in the literature.

Full photonic implementations of a variety of NoCs—from low-radix high-diameter mesh/torus topologies [Shacham et al. 2007b; Cianchetti et al. 2009] to medium-radix medium-diameter butterfly/clos topologies [Joshi et al. 2009a; Pan et al. 2009] to high-radix low-diameter bus/crossbar topologies [Kirman et al. 2006; D. Vantrease et al. 2008] have also been explored. These fully photonic NoCs are proposed to further leverage the large bandwidth density and lower data-dependent power advantage provided by silicon-photonic links for NoC communication. However, the large power dissipated by their laser sources makes it prohibitively expensive to be adopted to design commercial systems. Our PhotoNoC design adopts a network topology similar to the recent multi-bus NoC proposed by Chen and Joshi [2013] that is optimized to provide high application performance, while enabling a runtime management of laser power.

8. CONCLUSIONS

Application workloads from the contemporary Big Data era exhibit unprecedented network bandwidth even for a single computing node—we have observed that 4TB/s bandwidth can be needed on chip. Future computing nodes will need NoC architectures that can efficiently support such a large amount of data traffic in a single computing node. In this article, we have studied the NoC architecture of a forward-looking shared-memory Kilocore computing node to show how we can design a NoC that can accommodate such a huge data traffic. For the study, we have selected representative state-of-the-art data-intensive applications of the Big Data era from NPB, SPLASH-2, PARSEC, MANTEVO, GRAPH500, and UHPC benchmark suites to build multi-threaded and multi-programmed workloads. For the design of the Kilocore's NoC, first we utilized a competitive electrical concentrated 2D-mesh network topology (ElecNoC). We observed that as compared with an ideal fully connected NoC, where all of its routing paths have a fixed three-cycle latency (IdealNoC), ElecNoC shows an average of 68% lower application performance (56% for multi-programmed workloads). The reason is that this electrical NoC can support an average of 59% less offered network bandwidth for multi-threaded workloads (54% in case of multi-programmed workloads). We also explored two other NoC architectures (HybNoC and PhotoNoC) for the target Kilocore system that leverage silicon-photonic link technology. HybNoC is composed of an electrical concentrated 2D-mesh sub-network for short-distance communication and a photonic multi-crossbar to reduce network diameter for long-distance communication. PhotoNoC is made up of multi-bus network. In addition, to explore the possibility of leveraging large bandwidth of silicon-photonic links, we also evaluated aggressive non-blocking caches and prefetching technique for the design of the cache hierarchy of the target Kilocore system. These techniques can trade higher network bandwidth for higher application performance. Our experimental evaluation reveals that non-blocking caches with a shared L2 is the best cache hierarchy, and the PhotoNoC is the most suitable NoC design for our target Kilocore system. In particular, the Kilocore system with this configuration can support close to 0.9× the offered bandwidth and achieves close to 0.9× the application performance with respect to the IdealNoC, while consuming comparable overall power to the worst-performing system that integrates the ElecNoC. In summary, this article demonstrates that next-generation Big Data workloads can truly benefit from the large-bandwidth density of photonic links, and photonic-based NoC designs could become all the more pertinent in future manycore chips.

REFERENCES

N. Abeyratne et al. 2013. Scaling towards kilo-core processors with asymmetric high-radix topologies. In *Proceedings of the 2013 IEEE 19th International Symposium on High Performance Computer Architecture (HPCA2013)*. 496–507.

T. W. Ainsworth and T. M. Pinkston. 2007. Characterizing the cell EIB on-chip network. *IEEE Micro* 27, 5 (Sep. 2007), 6–14.

Shirish Bahirat and Sudeep Pasricha. 2014. METEOR: Hybrid photonic ring-mesh network-on-chip for multicore architectures. *ACM Trans. Embed. Comput. Syst.* 13, 3s (2014), 116:1–116:33.

D. Bailey and others. 1994. *The NAS Parallel Benchmarks*. Technical Report RNR-94-007.

Scott Beamer et al. 2010. Re-architecting DRAM memory systems with monolithically integrated silicon photonics. In *Proceedings of the 37th Annual International Symposium on Computer Architecture (ISCA 2010)*. 129–140.

S. Bell et al. 2008. TILE64 - processor: A 64-core SoC with mesh interconnect. In *Proceedings of the 2008 IEEE International Solid-State Circuits Conference (ISSCC'08), Digest of Technical Papers*. 88–598.

Keren Bergman et al. 2014. Silicon photonics. In *Photonic Network-on-Chip Design*. Integrated Circuits and Systems, Vol. 68. Springer, New York, 27–78.

C. Bienia et al. 2008. The PARSEC benchmark suite: Characterization and architectural implications. In *Proceeings of the PACT*. 72–81.

A. Boos et al. 2013. PROTON: An automatic place-and-route tool for optical networks-on-chip. In *Proceedings of the 2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 138–145.

D. Campbell et al. 2012. *Ubiquitous High Performance Computing: Challenge Problems Specification*. Technical Report HR0011-10-C-0145. Georgia Institute of Technology.

Trevor E. Carlson, Wim Heirman, and Lieven Eeckhout. 2011. Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. 52:1–52:12.

Chao Chen et al. 2014. Sharing and placement of on-chip laser sources in silicon-photonic NoCs. In *Proceedings of the 2014 8th IEEE/ACM International Symposium on Networks-on-Chip (NoCS)*. 88–95.

Chao Chen, J. L. Abellan, and A. Joshi. 2015. Managing laser power in silicon-photonic NoC through cache and NoC reconfiguration. *IEEE Trans. Comput.-Aid. Des. Integr. Circ. Syst.* 34 (2015), 972–985.

Chao Chen and A. Joshi. 2013. Runtime management of laser power in silicon-photonic multibus NoC architecture. *IEEE J. Select. Top. Quant. Electron.* 19, 2 (2013).

Mark J. Cianchetti, Joseph C. Kerekes, and David H. Albonesi. 2009. Phastlane: A rapid transit optical routing network. In *Proceedings of the 36th Annual International Symposium on Computer Architecture (ISCA 2009)*. 441–450.

D. Vantrease et al. 2008. Corona: System implications of emerging nanophotonic technology. In *Proceedings of the 35th International Symposium on Computer Architecture, 2008 (ISCA'08)*. 153–164.

B. K. Daya et al. 2014. SCORPIO: A 36-core research chip demonstrating snoopy coherence on a scalable mesh NoC with in-network ordering. In *Proceedings of the 2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*. 25–36.

EZchip Semiconductor Ltd. 2015. EZchip Introduces TILE-Mx100 Worlds Highest Core-Count ARM Processor Optimized for High-Performance Networking Applications. Retrieved from http://www.tilera.com/News/PressRelease/?ezchip=97.

M. Georgas et al. 2014. A monolithically-integrated optical transmitter and receiver in a zero-change 45nm SOI process. In *Proceedings of the 2014 Symposium on VLSI Circuits Digest of Technical Papers*. 1–2.

Matthias Gries et al. 2011. SCC: A flexible architecture for many-core platform research. *Comput. Sci. Eng.* 13, 6 (2011), 79–83.

Michael A. Heroux et al. 2009. Improving performance via mini-applications. Technical Report. Sandia National Laboratories.

J. Howard et al. 2011. A 48-core IA-32 processor in 45 nm CMOS using on-die message-passing and DVFS for performance and power scaling. *IEEE J. Solid-State Circ.* 46, 1 (2011), 173–183.

A. Joshi et al. 2009a. Silicon-photonic clos networks for global on-chip communication. In *Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip, 2009 (NoCS'09)*. 124–133.

A. Joshi, Byungsub Kim, and V. Stojanovic. 2009b. Designing energy-efficient low-diameter on-chip networks with equalized interconnects. In *Proceedings of the 17th IEEE Symposium on High Performance Interconnects (HOTI'09)*. 3–12.

John Kim, James Balfour, and William Dally. 2007. Flattened butterfly topology for on-chip networks. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*. 172–182.

N. Kirman et al. 2006. Leveraging optical technology in future bus-based chip multiprocessors. In *39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-39)*. 492–503.

Poonacha Kongetira, Kathirgamar Aingaran, and Kunle Olukotun. 2005. Niagara: A 32-way multithreaded sparc processor. *IEEE Micro* 25, 2 (2005), 21–29.

David Kroft. 1981. Lockup-free instruction fetch/prefetch cache organization. In *Proceedings of the 8th Annual Symposium on Computer Architecture (ISCA'81)*. IEEE Computer Society Press, Los Alamitos, CA, 81–87.

Rakesh Kumar, Victor Zyuban, and Dean M. Tullsen. 2005. Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling. In *Proceedings of the 32nd Annual International Symposium on Computer Architecture*. 408–419.

George Kurian et al. 2010. ATAC: A 1000-core cache-coherent processor with on-chip optical network. In *Proceedings of the 19th International Conference on Parallel Architectures and Compilation Techniques (PACT'10)*.

S. Li et al. 2009a. McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Proc. MICRO-42*. 469–480.

Zheng Li et al. 2009b. Spectrum: A hybrid nanophotonic-electric on-chip network. In *Proceedings of the 46th Annual Design Automation Conference (DAC'09)*. 575–580.

Zhongqi Li et al. 2015. Aurora: A cross-layer solution for thermally resilient photonic network-on-chip. *IEEE Trans. VLSI Syst.* 23, 1 (2015), 170–183.

Xiaoyao Liang, K. Turgay, and D. Brooks. 2007. Architectural power models for sram and cam structures based on hybrid analytical/empirical techniques. In *International Conference on Computer Aided Design (ICCAD)*. 824–830.

F. Y. Liu et al. 2012. 10-Gbps, 5.3-mW optical transmitter and receiver circuits in 40-nm CMOS. *IEEE J. Solid-State Circ.* 47, 9 (Sept 2012), 2049–2067.

Yangyang Liu et al. 2014. Ultra-low-loss CMOS-compatible waveguide crossing arrays based on multimode bloch waves and imaginary coupling. *Opt. Lett.* 39, 2 (Jan 2014), 335–338.

D. Ludovici et al. 2009. Assessing fat-tree topologies for regular network-on-chip design under nanoscale technology constraints. In *Proceedings of the Conference on Design, Automation and Test in Europe*. 562–565.

Andrew McAfee et al. 2012. Big data: The management revolution. *Harv. Bus. Rev*. 90, 10 (2012), 61–67.

Richard C. Murphy, Kyle B. Wheeler, Brian W. Barrett, and James A. Ang. 2010. Introducing the graph 500. *Cray Users Group (CUG)* (2010).

Brian Neel, Matthew Kennedy, and Avinash Kodi. 2015. Dynamic power reduction techniques in on-chip photonic interconnects. In *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI*. 249–252.

Kyle J. Nesbit and James E. Smith. 2005. Data cache prefetching using a global history buffer. *IEEE Micro* 25, 1 (Jan. 2005), 90–97.

Jason S. Orcutt et al. 2012. Open foundry platform for high-performance electronic-photonic integration. *Opt. Expr.* 20, 11 (May 2012), 12222–12232.

Yan Pan et al. 2009. Firefly: Illuminating future network-on-chip with nanophotonics. *SIGARCH Comput. Arch. News* 37, 3 (June 2009).

Yan Pan, John Kim, and Gokhan Memik. 2011. FeatherWeight: Low-cost optical arbitration with QoS support. In *Proceedings of the 44th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-44)*. ACM, 105–116.

David A. Patterson and John L. Hennessy. 2013. *Computer Organization and Design, Fifth Edition: The Hardware/Software Interface* (5th ed.). Morgan Kaufmann, San Francisco, CA.

Carl Ramey. 2011. Tile-gx100 manycore processor: Acceleration interfaces and architecture. In *Proceedings of the 23th Hot Chips Symposium*.

Gunther Roelkens et al. 2014. Silicon-based photonic integration beyond the telecommunication wavelength range. *IEEE J. Select. Top. Quant. Electron.* 20, 4 (2014), 394–404.

Steve Scott et al. 2006. The BlackWidow high-radix clos network. In *Proceedings of the 33rd Annual International Symposium on Computer Architecture*. 16–28.

Larry Seiler et al. 2008. Larrabee: A many-core x86 architecture for visual computing. In *ACM SIGGRAPH 2008 Papers*. 18:1–18:15.

A. Shacham et al. 2007b. On the design of a photonic network-on-chip. In *NOCS*. 53–64.

Assaf Shacham, Keren Bergman, and Luca P. Carloni. 2007a. The case for low-power photonic networks on chip. In *Proceedings of the 44th Annual Design Automation Conference*. 132–135.

M. A. I. Sikder et al. 2015. OWN: Optical and wireless network-on-chip for kilo-core architectures. In *Proceedings of the 2015 IEEE 23rd Annual Symposium on High-Performance Interconnects (HOTI)*. 44–51.

Dean M. Tullsen and Susan J. Eggers. 1995. Effective cache prefetching on bus-based multiprocessors. *ACM Trans. Comput. Syst.* 13, 1 (1995), 57–88.

A. N. Udipi, N. Muralimanohar, and R. Balasubramonian. 2010. Towards scalable, energy-efficient, bus-based on-chip networks. In *Proceedings of the 2010 IEEE 16th International Symposium on High Performance Computer Architecture (HPCA)*. 1–12.

Hangsheng Wang, Li-Shiuan Peh, and S. Malik. 2003. Power-driven design of router microarchitectures in on-chip networks. In *Proceedings of the IEEE/ACM International Symposium on Microarchitecture (MICR0-36)*. 105–116.

Lei Wang et al. 2014. Bigdatabench: A big data benchmark suite from internet services. In *Proceedings of the 2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 488–499.

S. C. Woo et al. 1995. The SPLASH-2 programs: Characterization and methodological considerations. In *Proc. ISCA*. 24–36.

Cao Yu et al. 2012. The Predictive Technology Model (PTM) website. Retrieved from http://ptm.asu.edu/.

C. Zhang, D. Liang, G. Kurczveil, J. E. Bowers, and R. G. Beausoleil. 2015. Thermal management of hybrid silicon ring lasers for high temperature operation. *IEEE Journal of Selected Topics in Quantum Electronics* 21, 6 (2015), 385–391.

Ying Ping Zhang et al. 2006. A study of the on-chip interconnection network for the IBM Cyclops64 multi-core architecture. In *Proceedings 20th International Symposium on Parallel and Distributed Processing (IPDPS'06)*.

Xuezhe Zheng et al. 2012. 2-pJ/bit (on-chip) 10-Gb/s digital CMOS silicon photonic link. *IEEE Photon. Technol. Lett.* 24, 14 (July 2012), 1260–1262.

Xuezhe Zheng et al. 2013. A 33mW 100Gbps CMOS silicon photonic WDM transmitter using off-chip laser sources. In *Optical Fiber Communication Conference*. Optical Society of America, PDP5C–9.

Received October 2015; revised April 2016; accepted June 2016