Photonic-Based Express Coherence Notifications for Many-core CMPs

José L. Abellán[a,*], Eduardo Padierna[a], Alberto Ros[b], Manuel E. Acacio[b]

[a]*Computer Science Dept., Universidad Católica de Murcia, 30107 Murcia, Spain*
[b]*Computer Engineering Dept., University of Murcia, 30100 Murcia, Spain*

Abstract

Directory-based coherence protocols (Directory) are considered the design of choice to provide maximum performance in coherence maintenance for shared-memory many-core CMPs, despite their large memory overhead. New solutions are emerging to achieve acceptable levels of on-chip area overhead and energy consumption such as optimized encoding of block sharers in Directory (e.g., SCD) or broadcast-based coherence (e.g., Hammer). In this work, we propose a novel and efficient solution for the cache coherence problem in many-core systems based on the co-design of the coherence protocol and the interconnection network. Particularly, we propose *ECONO*, a cache coherence protocol tailored to future many-core systems that resorts on *PhotoBNoC*, a special lightweight dedicated silicon-photonic subnetwork for efficient delivery of the atomic broadcast coherence messages used by the protocol. Considering a simulated 256-core system, as compared with Hammer, we demonstrate that *ECONO+PhotoBNoC* reduces performance and energy consumption by an average of 34% and 32%, respectively. Additionally, our proposal lowers the area overhead entailed by SCD by 2×.

Keywords: Many-core CMP, Silicon-photonic Technology, Cache Coherency.

*Corresponding author. Phone: +34 968 278801. Fax: +34 868 884151
Email addresses: `jlabellan@ucam.edu` (José L. Abellán), `epadierna@alu.ucam.edu` (Eduardo Padierna), `aros@ditec.um.es` (Alberto Ros), `meacacio@ditec.um.es` (Manuel E. Acacio)

1. Introduction

During the past 15 years, the number of cores in multicore architectures has been rising steadily. Nowadays, thanks to the huge number of transistors on a chip brought by the well-known Moore's law, general-purpose manycore architectures approaching one hundred cores are becoming commercially available, such as Intel's 72-core x86 Knights Landing MIC [1]. Meanwhile, researchers are already prototyping thousand core chips such as the KiloCore chip [2].

In systems with a large number of processing cores, directory-based, write-invalidate coherence protocols appear as the only viable alternative able to bring high performance and scalability. In these protocols, a directory structure which is physically distributed among the different cores (or group of cores, depending on the particular implementation) is responsible for keeping track of the identity of the sharers of every memory block residing in one or several of the private caches. In this design, every memory block is assigned to a single directory bank and all cache misses from the private caches for that block are sent to it. On receiving a cache miss, the directory information for the particular block is retrieved, and based on that, coherence actions (if needed) are carried out so that the cache miss can be resolved.

The way the directory structure codifies the set of sharers for every memory block determines the amount of extra memory required for this structure (directory memory overhead) and ends up limiting the range of cores at which cache coherence can be provided in a practical way. Codifying the set of sharers using structures whose size per node depends linearly on the number of cores is certainly guarantee of non scalability. For example, the well-known bit-vector sharing code is absolutely incompatible with scalability.

To exemplify the difficult decision-making process to address the previous aspects at once, consider the following two protocols: *Hammer* [3] and *Directory* [4]. In ensuring coherence, *Hammer* relies on sending as many coherence messages as the number of all private caches in the system, meanwhile *Directory* keeps track of exact coherence information about sharers (through a full bit-vector) to send coherence messages just to those private caches with a valid copy of the memory block. Therefore, *Directory* is more efficient in terms of performance and energy consumption since it only injects the required coherence messages into the network-on-chip (NoC). Besides, *Hammer* minimizes on-chip area overhead, because it does not devote any extra hardware resources to encoding a sharer list for every memory block.

Efficient cache coherence is also related to efficiency of the NoC employed. For instance, the MIT RAW processor's NoC dissipates 40% of the overall chip power [5] and then, all coherence-related traffic may lead to significant energy consumption. Whereas most prior works have focused exclusively on the NoC to achieve higher efficiency through, for example new topologies or improved network routers, recent research works have opted to co-design both the NoC and the higher-level layer (i.e., the cache coherence protocol) as a means of achieving better designs [6]. In this work, we follow such trend to propose a novel and more efficient solution for the cache coherence problem in manycores. Particularly, our proposal comprises two important elements: the first is ECONO, a cache coherence protocol tailored to future many-core CMPs. ECONO is basically a protocol similar to *Hammer*, but that is able to obtain performance results similar or better than *Directory* but without any directory memory overheads (it is *directory-free*). To achieve this, ECONO resorts on the second element, a special lightweight dedicated subnetwork (called PhotoBNoC) that is implemented using silicon-photonic link technology and that guarantees efficient delivery of the atomic broadcast coherence messages used by the protocol. By combining these two elements, a higher-performance, more energy- and area-efficient solution to the cache coherence problem in future manycores can be orchestrated.

ECONO was presented in our previous work [7]. However, in that work, the broadcast network was implemented by leveraging G-Lines technology [8], which suffered important scalability hurdles. First, the transmitter and receiver circuits of a G-Line require large power dissipation to sustain signal integrity. Second, each of these links features a large area footprint. Third, G-Line technology does not scale well beyond 65-nm CMOS technology. This work extends

ECONO [7] by proposing a photonic-based broadcast network (PhotoBNoC) to achieve fast atomic broadcasts for the transmission of the ECONO's messages. We show that as silicon-photonic link technology features very large bandwidth density through Wavelength Division Multiplexing (WDM), low-latency global distance communication, and low data-dependent power dissipation, PhotoBNoC constitutes a more optimized alternative to the original ECONO's G-Lines-based network when the number of cores is large.

Note that, our ECONO protocol uses broadcast transmissions for the maintenance of coherence in order to save on-chip area as in Hammer protocol, and our main goal is to achieve very efficient broadcast transmissions so that we can obtain similar performance and energy efficiency to Directory but without its large on-chip area overhead.

In particular, the major contributions of this manuscript are as follows:

- We advocate the usage of silicon-photonic link technology to implement the dedicated broadcast on-chip network required by ECONO.

- We propose and discuss the design of an efficient photonic-based broadcast network design called PhotoBNoC that guarantees both atomic and fast transmissions for the ECONO's atomic broadcast messages. PhotoBNoC leverages a novel network layout with distance-dependent segmented Single-Write Broadcast-Reader (SWBR) channels to save laser power. Atomic transmissions are guaranteed through SWBR channels, and both lightweight local arbiters and buffer queues at destination nodes.

- We evaluate the efficiency of our proposal by simulating a 256-core system and contemporary application workloads. We demonstrate that our proposal achieves similar performance and energy efficiency as compared to Directory but without its large memory overhead. Also, considering a range of many-core sizes (128, 256, 512 and 1024 cores), we demonstrate that our solution is more scalable than the most-optimized SCD-based directory design.

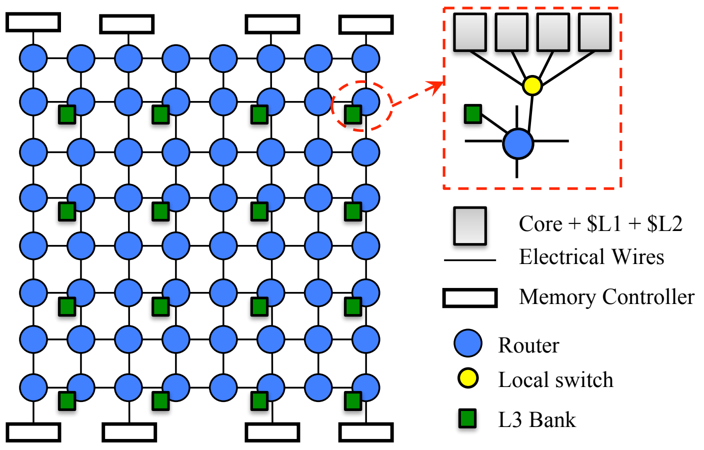


Figure 1: Baseline 256-core system with 8×8 2D-mesh NoC, 16 shared L3 banks uniformly distributed over the chip, and 8 memory controllers placed along two edges of the chip.

- We compare our proposed hybrid NoC with a complete state-of-the-art photonic NoC, namely PhotoNoC [9], to implement ECONO. Our evaluation reveals that although PhotoNoC marginally beats our hybrid NoC in terms of performance (an average of 3.3% lower execution times), PhotoNoC requires a more expensive and complex 3D stacking integration of the photonic devices due to its much larger area overhead (30×), and power dissipation (4×) when compared with PhotoBNoC.

2. Preliminaries

2.1. Target Many-core System

Our target system is a 256-core microprocessor with an electrical concentrated 2D-mesh NoC topology. Figure 1 shows a logical view of the many-core chip. Each core is a simplified version of a Pentium II that has a 2-way in-order issue, out-of-order execution superscalar pipeline. The target system uses an inclusive cache hierarchy that is based on the TILE-Gx72 many-core chip [10]: each core is equipped with a private 32-KB I/D L1 cache and a private 256-KB L2 cache. Moreover, there is also a shared 64-MB L3 cache that follows a NUCA distribution. As we can observe in Figure 1, the L3 cache is partitioned into

Table 1: Micro-architecture of the 256-core system (11 nm, 1 GHz and 0.6 Volts).

Processor Core	
Pipeline	2-way superscalar, OoO exec.
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 and Memory	
Cache Line Size	64 Bytes
Private I/D L1 Cache	4-way 32 KB @ 1+4 ns
Private unified L2 Cache	8-way 256-KB @ 3+8 ns
Shared Distributed L3 Cache	16×[16-way 4096-KB] banks @ 6+16 ns
Main memory	8 memory controllers, 8 PIDRAM @ 50 ns
On-chip Network	
Topology	Concentrated 2D-mesh
Routing	X-Y Dimension Ordered
Control Flow	Virtual-channel (3 VCs)
	Credit-based backpressure
Flit size	32 Bytes (3 Flits/VC)
Router latency	2 cycles
Packet size	72 Bytes (Data); 8 Bytes (Control)

sixteen 4MB banks. The amount of memory for the shared L3 cache is proportional to the amount of L3 cache capacity integrated in the TILE-Gx72 chip (18 MBytes for 72 cores, whereas 64 MBytes are assigned to our 256-core system). Cache coherency is implemented by using MESI directory-based protocol with cache-to-cache transfers where the directories are co-located with the L3 banks. Moreover, the target system integrates 8 memory controllers (MCs) that are uniformly distributed along two edges of the chip.

Our system includes a packet-switched concentrated 2D-mesh topology. This design is a typical choice when dealing with large many-core systems [11, 10, 12, 13]. In our NoC, each router has an input/output port from/to a local switch that interfaces four cores. We use network concentration through local switches in order to lower the hop count between distant nodes and to simplify the routers (four cores are attached to a router through a single input/output link using the local switch), thereby we can obtain lower router power dissipation and on-chip area. We consider 2-cycle pipelined routers and 1-cycle inter-router links to model a competitive NoC design to provide high performance for the data-intensive application workloads under study (Section 5.3). In consequence, our 2D-mesh NoC features a zero-load latency of 46 cycles for the longest path from the L2 cache at one corner to the L3 bank at the diagonally opposite corner (30 cycles in the routers + 1 input link + 14 inter-router links + 1 output link). Besides, our model of the 2D-mesh NoC also takes into account another clock cycle to reach the target L2 cache from the local switch, and another clock cycle in the reverse direction.

To provide a high-bandwidth and low-latency off-chip DRAM memory, the target system's MCs are connected to memory through state-of-the-art PIDRAM technology [14]. For the PIDRAM technology, we assume an average latency of 50 ns 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.

Table 1 lists the main system parameters. For our experimental evaluation, we assume double-gate (FinFET) 11 nm CMOS technology, an operating frequency of 1 GHz with 0.6 V supply voltage, and the chip has an area of 256 mm^2. To validate that our many-core chip can be integrated into a 256-mm^2 floorplan, we estimated area of all the main architectural components by using McPAT 1.0[15] tool for the cores and cache hierarchy, and DSENT[16] for the 2D-mesh network. We obtained that the memory hierarchy (L1, L2 and L3 caches) represents 85.6 mm^2, the 256 cores needs 98.7 mm^2 and the 2D-mesh requires 42.6 mm^2. Overall, this equals 226.8 mm^2, which is within the chip area constraint of 256 mm^2.

Table 2: Silicon-photonic link technology parameters based on [21, 22, 23]. **This includes both dynamic energy (data-traffic dependent energy) and fixed energy (clock and leakage).*

Laser source efficiency	15%
Coupler loss, Splitter loss	1 dB, 0.2 dB
Modulator insertion loss	1 dB
Waveguide loss	2 dB/cm
Crossing loss	0.05 dB
Filter through loss	1e-2 dB
Filter drop loss	0.5 dB
Photodetector loss	0.1 dB
Non-linearity loss	1 dB
Modulator driver circuit energy*	0.035 pJ/b
Receiver circuit energy*	0.035 pJ/b
Thermal tuning power	16 μW/K
Receiver sensitivity	-17

2.2. Silicon-Photonic Link Technology

Silicon-photonic links have been proposed as a promising replacement to electrical links for on-chip communication in forthcoming many-core systems [17, 18, 19, 20, 9]. The reason is that they provide much higher bandwidth density ($Gbps/\mu m$), lower global communication latency, and lower data-dependent power than electrical links. Due to these appealing characteristics, we consider silicon-photonic links for the transmission of broadcast messages.

We employ monolithically integrated silicon-photonic links driven by on-chip laser sources since they simplify packaging, reduce cost, and improve laser source control when compared to off-chip laser sources [24]. This monolithic integration of photonic devices is a better choice compared to 3D integration as the link transmitter circuits experience lower parasitics, which leads to lower link energy consumption, as shown recently in working prototypes [21, 22]. We assume state-of-the-art silicon-photonic link technology parameters (Table 2), a link bandwidth of 8 Gbit/second/wavelength and 16 wavelengths/waveguide/direction, and a conventional 3-cycle link latency (1 cycle in flight and 1 cycle each for electrical-to-optical (E-O) and optical-to-electrical (O-E) conversion) plus serialization latency.

3. ECONO Cache Coherence Protocol

3.1. Baseline Operation

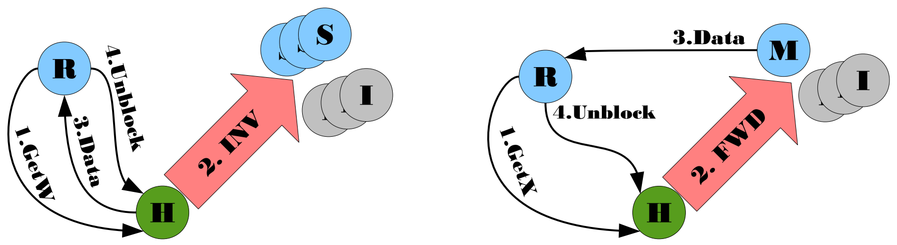
Without loss of generality, we describe the ECONO operation for the system described in Section 2.1 focusing on the coherence maintenance in the last two levels of the cache hierarchy, i.e., private L2 caches and shared L3 banks.

Figure 2 exemplifies how ECONO works under two typical coherence scenarios: invalidation of sharers and request forwarding. Situations where coherence actions are not necessary (e.g., read miss that gets the block from the shared cache level) are treated in a conventional way using the electrical network.

The first scenario is depicted in Figure 2a, which shows the case of a memory block shared by several L2 caches (in `S` state) and a requesting core (`R`) that suffers a write miss in its L2 cache. The requester sends a write request for exclusive access for write permission (`1.GetW`) to the home L3 bank (`H`). The home L3 bank has to invalidate all L2 cached copies before sending the valid copy and granting write permission to the requester. In Hammer, unlike Directory, since home L3 banks do not store a list of sharers for every memory block, the home L3 bank sends a number of invalidation messages equal to the total number of private L2 caches in the system. Every invalidation message results in an acknowledgment response message (ACK) that is sent back to the home L3 bank. Once the home L3 bank has collected all ACK messages from the L2 caches, the home's valid copy of the block is sent to the requester (`3.Data`). After the requester gets the memory block with the write permission, it replies back with an unblock message (`4.Unblock`)[1]. Upon reception of the unblock message, the home bank can continue handling requests to that block.

ECONO does not store any information about sharers for every cached block, so it requires broadcast as in Hammer. The broadcast process is improved

[1]All studied coherence protocols use unblock messages to ease protocol races handling [25].



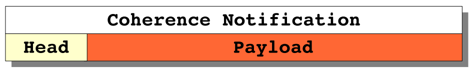
(a) Invalidation of block sharers (b) Data recovery from a block owner

Figure 2: Coherence maintenance in ECONO protocol. *Messages depicted above fine black arrows travel through the main interconnection network, whereas thick pink arrows illustrate the ACN messages transmitted over a dedicated broadcast on-chip network.*

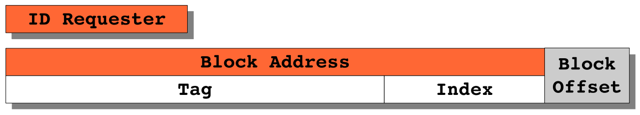
thanks to our dedicated network. As illustrated in Figure 2a, instead of sending as many invalidation messages as the total number of L2 caches in the many-core system, ECONO makes use of a single broadcast message (`2.INV`) which is sent atomically through a dedicated broadcast on-chip network (Section 4). As we will see, by using a dedicated subnetwork to transfer these atomic broadcasts we can improve overall system performance and energy consumption.

Additionally, the home L3 bank does not waste time waiting for the ACK messages because our coherence protocol can safely operate without them. The reason is that invalidations are transferred atomically with a predictable and constant propagation latency that takes just a few clock cycles to complete (Section 4). Then, the home L3 bank (i.e., the directory) can exactly know when all L2 caches have received the invalidations and then provide the block to the requesting L2 cache. The atomic broadcast invalidation simplifies protocol design and verification complexity [26]. The rest of the coherence transaction is as in the Hammer protocol (`3.Data` and `4.Unblock`).

Figure 2b shows the actions performed in the second scenario where there is a single modified copy of the block in an L2 cache, i.e., the owner L2 cache (`M`) and a requesting core that wants to write or read (see `1.GetX` for the general case) that block. In this context, the owner must invalidate (for a write miss) or downgrade (for a read miss) its copy and forward the memory block to the



(a) ACN messages in ECONO



(b) Data Payload

Figure 3: Format of the ACN messages in ECONO protocol.

requester. In ECONO, the home L3 bank requests the forwarding of the block to the owner (`M`) with the transmission of a single atomic forward message. The owner then sends its copy (`3.Data`) to the requesting core, which sends the unblock message to the home L3 bank (`4.Unblock`).

3.2. Atomic Coherence Notification Messages

ECONO keeps coherence through Atomic Coherence Notification (ACN) messages such as the `INV` and `FWD` messages in Figure 2. ACN messages are sent atomically to all private L2 caches over a dedicated broadcast network.

Figure 3a illustrates the format of an ACN message which has two different fields: the *head* field, used to identify the type of action to be applied (e.g., invalidation or forward); and the *payload* field, which stores the information required to identify the blocks that will be affected by the coherence action (e.g., the block address). The *head* field needs to cover three coherence actions (2 bits): (i) invalidation of all cached copies (Figure 2a), (ii) forward-downgrade of the owner's copy (case of a read miss in Figure 2b), and (iii) forward-invalidate of the owner copy (case of a write miss in Figure 2b).

The *payload* field contains both the block address of the requested memory block, and the requester's ID that the receiver of the ACN message will use to identify the destination of the forward message (the requester). This information is shown in Figure 3b, where the ID sub-field requires a number of bits equal to $\log_2(\#PrivateLLCs)$ (e.g., 8 bits for a 256-core system), in order to codify the ID owner. Since the block address is codified with 58 bits, the size of an ACN message for a 256-core system is 68 bits. For convenience, we will round-up this amount of bits to 72 bits so that we can work with network flits that are multiple of a byte (also making the message size valid for up to a 1024-core system). Shorter ACN messages were explored in our previous work [7] to save network bandwidth. However, the large bandwidth density of silicon-photonic technology allows us to explore precise ACN messages.

4. Photonic-based Broadcast Network-on-Chip Design

ECONO relies on atomic broadcasts that are transmitted over a dedicated on-chip network. In our previous work [7], the broadcast network was implemented by leveraging G-Lines technology [27]. The main issue with this technology is that the amount of G-Lines that can be included in a chip is very limited. First, the transmitter and receiver circuits of a G-Line require large power dissipation to sustain signal integrity. Second, each of these links features large area footprint. Third, G-Line technology does not scale well beyond 65-nm CMOS technology (to the best of our knowledge, there is no G-Line link design implemented with contemporary technology nodes). In addition, if we consider that each G-Line can only support a 1-bit signal per transmission, a network channel implemented by using G-Lines technology can support very low bandwidth density. This is why G-Lines have been proposed to implement lightweight on-chip networks to interchange extremely fast global signals (e.g., to interconnect distant routers in a NoC [27] or for global synchronization in many-core systems [28, 29]). To meet this constraint, our G-Lines implementation [7] shortens the size of the ACN messages (to save network bandwidth) by getting rid of some of the bits of the memory block addresses at the cost of increasing coherence actions (due to false positives). This reduction leads to higher energy costs in some cases, e.g., the invalidation of unnecessary cached blocks due to the inaccuracy in representing a memory block address with less bits than necessary.

In this work we advocate that, as silicon-photonic link technology features very large bandwidth density through Wavelength Division Multiplexing (WDM), low-latency global distance communication, and low data-dependent power dissipation, we can design our previous non-scalable ECONO's G-Lines-based network [7] with a scalable photonic-based design (Section 2.2). Next, we describe our photonic-based broadcast network (PhotoBNoC) to achieve fast atomic broadcasts for the transmission of the ECONO's ACN messages.

As explained in Section 3, the ACN messages are transmitted from the home L3 bank to the L2 private caches. This implies that the PhotoBNoC system must implement unidirectional channels from L3 to L2. Moreover, to broadcast the ACN messages, we have to provide connectivity between every L3 bank and L2 cache controller. To support this kind of communication, we could explore a simple solution in which the PhotoBNoC is equipped with multiple independent Single-Write Broadcast-Reader (SWBR) photonic channels [30], where each SWBR channel connects an L3 bank with all 256 L2 caches. However, this design choice requires an inefficient network layout as it would require very long SWBR channels (every channel must reach all L2 caches), with a large amount of ring modulators and ring filters distributed along each silicon-photonic SWBR channel. This would aggregate considerable photonic losses per SWBR channel that would negatively impact overall laser power consumption of the PhotoBNoC. We will refer to this preliminary design as PhotoBNoC_v0.

One strategy to reduce the number of ring resonators (modulators and filters) for each SWBR channel is the use of network concentrators to lower the amount of receivers of broadcast messages. In particular, we take advantage of our electrical 2D-mesh interconnect (Figure 1), where every four L2 caches share one local switch that is connected to an input/output port of a router in the main 2D-mesh interconnect. In this way, if we consider a network router as a receiver of the broadcast message, we can reduce the amount of destination nodes by a factor of four. Then, each SWBR must interconnect a single L3 bank with 64 network routers. We will refer to this preliminary design as PhotoBNoC_v1.

To further reduce the amount of ring resonators per SWBR channel (to save laser power), we apply another optimization to this design that consists of

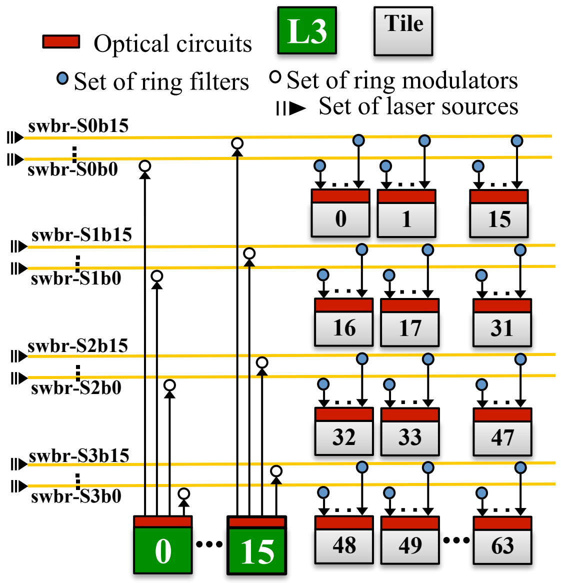


Figure 4: The 64 SWBR silicon-photonic channels required by PhotoBNoC in our target 256-core system. `swbr-SXbY`: `X` is the segment ID and `Y` is the L3 bank ID. For simplicity, instead of 256 L2 caches, we show 64 tiles where each tile encapsulates four neighboring L2 caches that utilize the same local switch to access the router of the NoC (each SWBR connects a single L3 bank and 16 routers).

splitting each SWBR channel into four separate sub-SWBR channels, namely segments. Figure 4 details the final logical design of the SWBR channels and their segments (segments are referred to as `SX`, where `X` is the segment ID ranging from 0 to 3). As we can observe, each L3 bank is connected to four SWBR channels, where each SWBR channel interconnects the L3 bank with a quarter of the routers of the NoC. We will refer to this final design as PhotoBNoC [2].

To guarantee atomic transmissions for the ACN messages with constant propagation latency (Section 3), we first utilize SWBR channels that allow an L3 bank to always send the ACN message whenever it is available for transmission (the sender does not have any other competitor in the channel). Second, to allow

[2]In Section 7, we give a thorough comparison among PhotoBNoC_v0, PhotoBNoC_v1 and PhotoBNoC in terms of number of photonic components, area overhead and power efficiency.

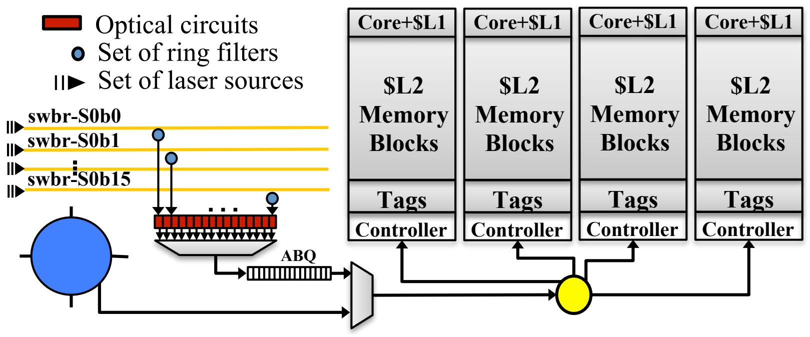


Figure 5: Main components at the receiver side of the ACN broadcast messages. ABQ refers to ACN Buffer Queue that stores the ACN messages transmitted from the L3 banks through the PhotoBNoC.

that the receivers can always accept the broadcast message, we dedicate a small buffer to store the ACN messages as it is shown in Figure 5. This buffer is called as ACN Buffer Queue (ABQ) that must be equipped with a sufficient number of entries to hold all possible ACN messages transferred from all senders in the system (i.e., the 16 L3 banks). We have experimentally determined that with just 16 entries for each ABQ we can meet this requirement. Due to the small size of the ABQ, we assume that this additional component required by PhotoBNoC represents both negligible energy consumption and on-chip area overhead. As we consider 9-Byte ACN messages (Section 3) and because every SWBR channel was designed to support a single wavelength to reduce energy consumption (Section 6.2) as we assume 8 Gbit/second/wavelength technology (Section 2.2), the serialization latency to transmit the ACN messages is 9 processor clock cycles. To this latency we have to add 3 clock cycles for the silicon-photonic link end-to-end transmission (Section 2.2), so it is guaranteed that an L3 bank can broadcast an ACN message in just 12 clock cycles. To this latency, we add one clock cycle to store the message in the ABQ.

Since the L2 cache controllers can be receivers of both ACN messages and regular data/coherence messages injected from the main electrical NoC, a multiplexer is needed to transmit any of these messages towards the local switch.

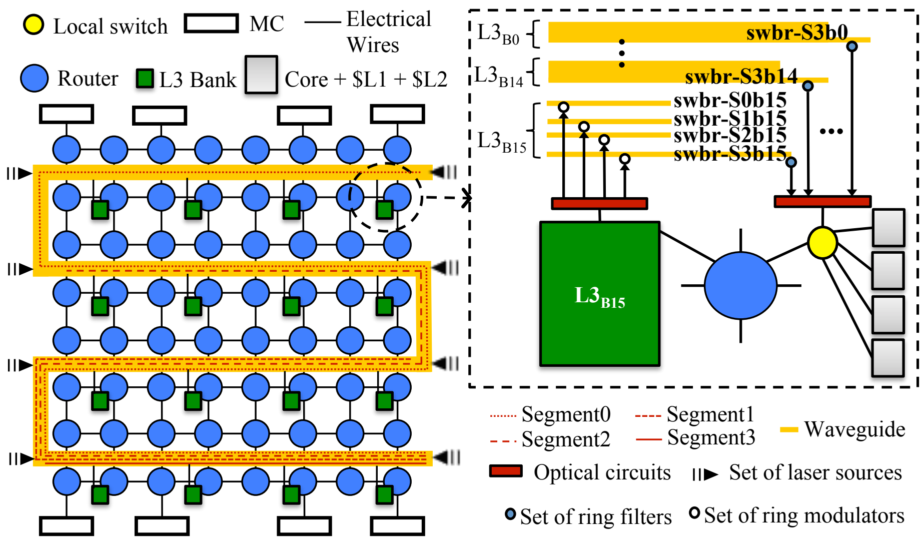


Figure 6: Target 256-core system with the baseline electrical 2D-mesh NoC and the proposed photonic-based broadcast network (PhotoBNoC). `swbr-SXbY`: a single SWBR channel, where X means the Segment ID and Y refers to the L3 bank ID (from 0 to 15). For clarity, we only draw the four segments belonging to one of the L3 banks at the bottom of the figure.

This is performed by using a local arbiter that checks both multiplexer's input ports at alternate clock cycles in a round robin fashion for the sake of fairness. For the link between a router and the multiplexer, to transmit packets without any data loss, we assume the same back-pressure mechanism implemented in the electrical NoC. This means that the multiplexer includes a 16-flit input buffer for this input port and implements credit-based flow control. It is worth noting that, once the local switch receives a message, if this is an ACN message the local switch broadcasts the ACN message towards the four private L2 cache controllers. On the other hand, if the message is a regular one that comes from the router, this is forwarded towards the target L2 cache controller.

Figure 6 shows the target 256-core system equipped with the PhotoBNoC design. As we can see, the 64 SWBR channels are routed by using a serpentine layout in order to be able to interconnect all sources (16 L3 banks) with all destinations (64 routers) in our target 256-core system. We show an example of the four segments in which one of the SWBR channels is divided into (16 routers per segment). As we can see, to provide a more energy-efficient PhotoBNoC design, to further save laser power, we propose *distance-dependent SWBR channels* in which the wavelengths are powered just to reach the needed destination ring filters. For instance, despite the serpentine layout of the photonic waveguides shown in Figure 6, wavelengths of `Segment3` are powered just to keep signal integrity (i.e., to enable reliable communication) for the ring resonators in the last two rows of 16 routers. However, the corresponding wavelengths of `Segment0` are provided with laser power to traverse the whole serpentine layout.

5. Evaluation Methodology

5.1. Simulation Platform

To evaluate our proposal, we use the Sniper 6.1 simulator [31]. We extended Sniper to implement our ECONO protocol and the two other baseline cache protocols: Hammer and Directory, which will be detailed in Section 5.2. We also modified Sniper to evaluate our PhotoBNoC infrastructure, and we configured the 256-core system explained in Section 2.1. To estimate power dissipated by the processor cores and cache hierarchy of the many-core system when running the application workloads, Sniper 6.1 comes interfaced with the latest McPAT 1.0 tool [15] that utilizes a 22 nm technology node. We use standard technology scaling rules [32] to scale down the power values obtained by McPAT 1.0 at 22 nm to the target 11 nm technology node. On the other hand, to estimate the power dissipation and on-chip area overhead of both the baseline electrical 2D-mesh and the PhotoBNoC, we interfaced DSENT tool [16] with Sniper 6.1. We configured DSENT with the parameters described in Table 1, and we employed the TG11LVT model for estimations at the target 11 nm technology node. In case of the PhotoBNoC, we configured DSENT with the photonic link technology parameters listed in Table 2.

Table 3: Coherence actions carried out by the coherence protocols under study. `N` = Numer of L2 caches in the target system. `Sharers` = Numer of sharers of the memory block. **This is a single ACN message that is broadcast through the PhotoBNoC.*

Core Request (State@LLC)	**Directory**	**Hammer**	**ECONO**
Read (M/E)	Fwd_Read to Owner	N×{Fwd_Read}	Fwd_Read*
Write (M/E)	Fwd_Write to Owner	N×{Fwd_Write}	Fwd_Write*
Write (S)	Sharers×{Invalidation}	N×{Invalidation}	Invalidation*

5.2. Cache Coherence Protocols

We compare ECONO to Hammer and Directory. There are three main situations in which the three protocols maintain coherence in a different way. First, guaranteeing exclusive write access to a processor core for a memory block in `S` state through invalidation of all sharers. Second, the same situation but when the memory block is in `M/E` state, which invalidates the owner copy and sends the block to the requester leveraging cache-to-cache transfers. And third, guaranteeing read access to a memory block that is in `M/E` state by forwarding the block from owner cache and leaving both copies in `S` state. These actions are described in Table 3. In all these cases Hammer needs to broadcast the coherence messages to all the available private L2 caches in the target system. Nonetheless, in our ECONO protocol, a single broadcast message is sent through the lightweight PhotoBNoC. For convenience, we will refer to these three messages as βroadcast messages, although in Directory these coherence messages are actually a single coherence message (`Fwd_write` and `Fwd_read`), or a multicast message depending on the number of block sharers (`Invalidation`).

Table 4 lists the set of actions carried out by each private L2 cache upon reception of the ACN messages. We distinguish whether the L2 cache has a copy of the memory block (second column) or not (third column). The latter case could occur in ECONO and Hammer because they do not keep track of a list of block sharers/owner at their directories. Note that, in case of `Invalidation` messages, unlike Hammer, ECONO does not need to wait for any response ACK

Table 4: Responses given by private L2 caches to coherence messages sent from LLC banks. In Hammer and ECONO protocols, since coherence actions are broadcast to all private L2 caches in the system, it could be possible that some L2 caches that do not hold the memory block receive the coherence message.

Incoming Request	**L2 Cache with Block**	**Incorrect L2 Cache**
Fwd_Read	Directory: M$\rightarrow$S; Block to new Owner	Directory: N/A
	Hammer: M$\rightarrow$S; Block to new Owner	Hammer: Ignore
	ECONO: M$\rightarrow$S; Block to new Owner	ECONO: Ignore
Fwd_Write	Directory: M$\rightarrow$I; Block to new Owner	Directory: N/A
	Hammer: M$\rightarrow$I; Block to new Owner	Hammer: Ignore
	ECONO: M$\rightarrow$I; Block to new Owner	ECONO: Ignore
Invalidation	Directory: S$\rightarrow$I; ACK to L3 Bank	Directory: N/A
	Hammer: S$\rightarrow$I; ACK to L3 Bank	Hammer: ACK to L3 Bank
	ECONO: S$\rightarrow$I; N/A	ECONO: Ignore

messages because the LLC bank precisely knows when the invalidation has been received by all private L2 caches (as we described in Section 4). Also, as in [33], we eliminate the need for invalidation acknowledgements by exploiting the ordering properties in PhotoBNoC. As we will see, as compared to Hammer, this mechanism will save large amount of network traffic and energy consumption from the main interconnect.

5.3. Application Workloads

We explore a broad variety of application workloads selected from different benchmark suites. Particularly, we run applications that belong to NPB [34], SPLASH-2 [35], PARSEC [36], AIB [37], MANTEVO [38] and UHPC [39]. Given that our target system is equipped with a large number of cores, most of the applications belonging to the benchmark suites do not have enough data/thread-level parallelism to scale well for a 256-core system. To select the applications that report the highest scalability for our experimental evaluation, we based on the IPC metric achieved by the applications when running on the target system for the two baseline coherence protocols: Directory and Hammer.

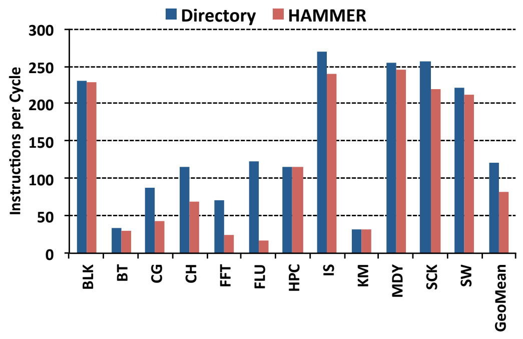
Table 5: List of benchmarks evaluated in this work.

Suite	**Applications (Abbreviation)**	**Input Data Sets**
SPLASH2	cholesky (CH)	tk29.O matrix
	fft	4M complex data points
PARSEC	blackscholes (BLK), fluidanimate (FLU)	sim_large
	swaptions (SW)	sim_large
NPB	cg, bt, is	large
AIB	Kmeans (KM)	100 clusters, 0.0001 error
MANTEVO	hpccg (HPC)	3D-matrix (100,100,100)
UHPC	mdynamics (MDY), schock (SCK)	water_xlarge.tpr, large

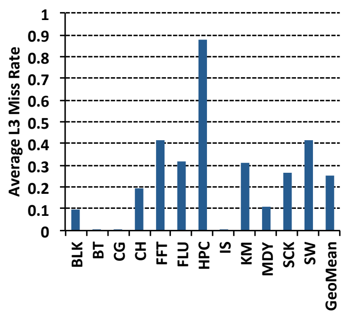
In addition, for each application we tuned its input data set to get the highest data-level parallelism and the fairest load balancing for our large target system. By doing that, the most scalable applications among all benchmark suites are the 12 applications and the input data sets listed in Table 5.

The experimental evaluation just assumes the parallel phase of the applications once all caches have been warmed up. This will be referred as the region of interest (ROI). For each benchmark execution, we run the pre-ROI region (i.e., the initialization phase) in Sniper's cache-only mode for cache warming. Given that execution time of applications is different, our evaluation employs Sniper's detailed execution for the first 150 million instructions at the ROI.

Figure 7 shows a preliminary characterization of the different application workloads in terms of reported IPC (Figure 7a), and average LLC (L3) miss rate (Figure 7b). Directory obtains an average of 46% higher IPC than Hammer protocol. This is expected as Hammer relies on broadcast messages to maintain coherence, even if there is just a single cache block owner from which to invalidate/recover the block (a more in-depth analysis will be performed in Section 6). Moreover, the average IPC across all benchmarks is larger than 100 (for 256 cores). However, even after fine-tuning the applications' input data set to report the highest IPC, we observe that not all applications scale well such as BT and KM that report roughly 30 IPC. However, we keep these two



(a) Instructions per cycle.



(b) Average LLC miss rate.

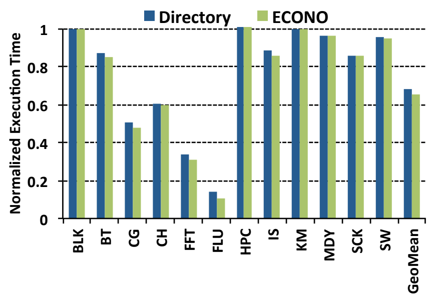
Figure 7: Characterization of application workloads in terms of scalability and working set locality for Directory and Hammer protocol.

applications in our experimental evaluation to have a more complete range of application behaviors when evaluating our ECONO protocol. As to the average LLC miss rate shown in Figure 7b, we observe that it is equal to 25%. We experimentally found that this value does not significantly improve even if we increase the size of the LLC. In this way, we assume that our total on-chip cache capacity fits the working set locality required for the benchmarks.

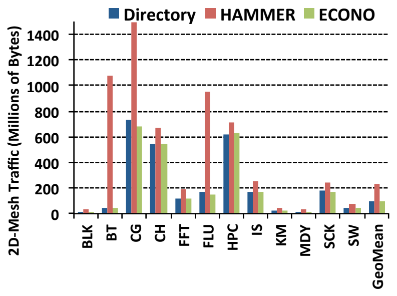
6. Experimental Results

6.1. Performance Evaluation

Figure 8a compares the execution times obtained by Directory, Hammer and ECONO protocols for the benchmarks under study when running on the target many-core system (normalized with respect to Hammer protocol). Note that, our evaluation of ECONO protocol considers the hybrid NoC explained in Section 4. In general terms, we can observe that not only ECONO outperforms Hammer but also it can obtain slightly lower execution times than Directory for some benchmarks (BT, CG, FFT, FLU and IS). In particular, ECONO shows an average of 34% lower execution time than Hammer, whereas Directory reduces it to 31%.



(a) Execution times normalized to Hammer.



(b) Total traffic in the 2D-mesh NoC.

Figure 8: Preliminary analysis among ECONO, Hammer and Directory protocols.

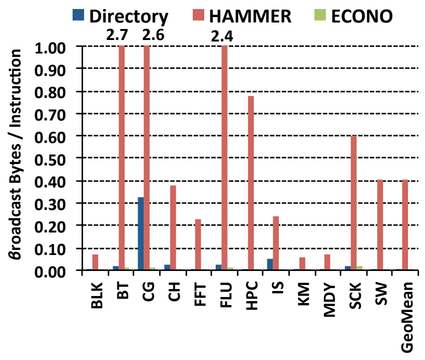
Figure 8b shows the impact on network traffic generated by each protocol through the electrical NoC. Hammer generates much more network traffic than Directory and ECONO (2.3×, on average). This is due to the fact that Hammer makes use of a much more aggressive coherence maintenance based on massive usage of broadcast messages. On the other hand, ECONO shows similar demand for network traffic when compared against Directory, which keeps track of block sharers and block owners in order to transmit just the required messages to maintain coherence. This means that, even though our protocol does not store directory information and relies on broadcast operations, diverting the broadcast messages to a dedicated PhotoBNoC saves a lot of traffic from the main interconnect. We will evaluate the efficiency of PhotoBNoC after this preliminary analysis. There is only one benchmark, CG, in which the network traffic showed by ECONO is noticeable smaller than Directory (7% less network traffic). This is because CG requires lots of multicast messages that are efficiently managed by ECONO by using a single broadcast message transmitted through the PhotoBNoC.

To understand the performance differences explained above between ECONO and the baseline coherence protocols, we explore the coherence activity for each protocol. For that, we consider two group of figures: Figure 9, which presents the amount of βroadcast messages needed by each coherence protocol; and Figure 10, which delves into the type of coherence activity depending on the three type of βroadcast messages. Recall that, as βroadcast messages we refer to the `Invalidation`, `Fwd_Write` and `Fwd_Read` coherence messages listed in Table 3.

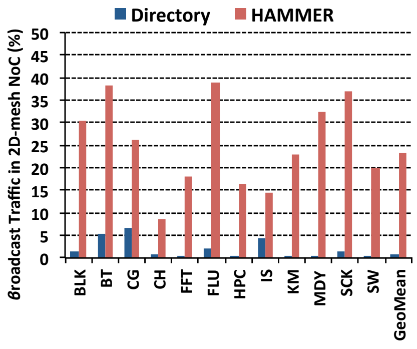
More specifically, Figure 9a illustrates the amount of βroadcast bytes per instruction (βPI) required by each coherence protocol for the benchmarks running on the target system. Even though ECONO makes use of broadcasting as in Hammer, the usage of PhotoBNoC to convey a single broadcast message to reach all L2 caches saves considerably the amount of βroadcast bytes – ECONO shows a close-to-zero βPI, whereas Hammer can obtain up to 2.7 βPI.

Now, we study the effect on performance by the achieved βPI values for the three coherence protocols. On the one hand, it is expected that benchmarks with negligible βPI and/or similar βPI among the three protocols will show little impact on performance, as they maintain coherence in a similar way. This explains why for BLK and KM benchmarks, which show very low βPI (less than 0.06), neither Directory nor ECONO improve performance against Hammer (Figure 8a). Besides, in terms of network traffic there are also marginal differences among the three protocols (Figure 8b). On the other hand, benchmarks where the βPI metric is very different among the three protocols, i.e. the coherence activity changes significantly, it is expected an impact on performance and demand for network traffic. This is the case for the remaining benchmarks. To analyze them, since our ECONO protocol can achieve similar performance improvements and savings on 2D-mesh network traffic when compared to Directory, we will start by comparing results for Directory and Hammer protocols and afterwards, we will justify why our ECONO protocol can even achieve better performance than Directory.

BT, CG and FLU show in Hammer the highest values for the βPI metric (2.7, 2.6 and 2.4, respectively). With respect to BT, although this benchmark shows 157 $\times$ more βPI in Hammer than in Directory, and much larger network traffic (Figure 8b), this translates into only 13% reduction in execution time (Figure 8a). The reason is that this benchmark does not scale well for a 256-core system (a 34 IPC shown in Figure 7a) due to a huge workload imbalance.



(a) βroadcast bytes per instruction.



(b) βroadcast bytes vs. total 2D-mesh traffic.

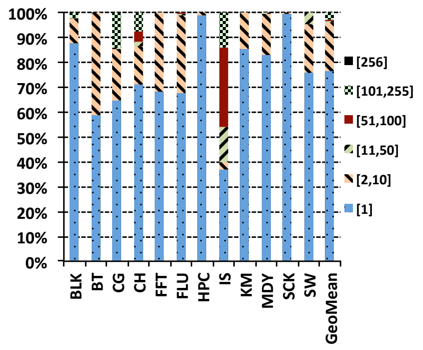
Figure 9: Analysis of βroadcast network traffic showed by the three coherence protocols for the benchmarks under study.

Only 22 out of the 256 available cores are active 90% of the time, meanwhile the remaining 234 cores have useful work just less than 1% of the total execution time. This implies that although BT shows similar difference between Hammer and Directory in terms of achieved of βPI with respect to FLU, the 150M instructions executed by both benchmarks take much longer in BT due to the large workload imbalance. Therefore, the abundant coherence activity shown in Hammer protocol for both benchmarks has much less impact on overall execution time in BT.

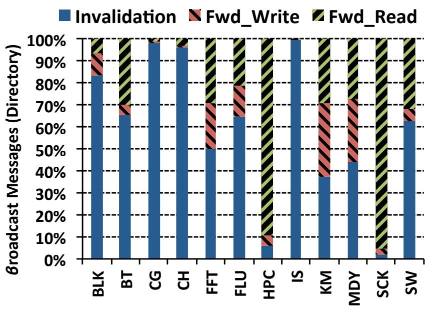
CG and FLU achieve high performance improvements when using Directory (49% and 86% lower execution times than in Hammer, respectively, as it is shown in Figure 8a). This is consequence of the large differences in βPI between Hammer and Directory (2.6 and 2.4 in Hammer, versus 0.32 and 0.02 in Directory, respectively), which leads to significant differences in terms of network traffic (2× and 5.7× larger traffic in Hammer, respectively – Figure 8b). To understand the magnitude of these performance improvements as compared to Hammer protocol, we present Figure9b. In this figure, we show the percentage of the βroadcast messages with respect to the total 2D-mesh network traffic. FLU has a larger difference between Directory and Hammer (2.1% and

38.9%), meanwhile in CG this difference is smaller (6.7% and 26.1%, respectively). The coherence activity for these two benchmarks can be observed in Figure 10a, which presents the number of block sharers that each βroadcast message needs to deal with. This shows how many unicast messages are required for each coherence action. All βroadcast messages in FLU require less than 11 sharers, but in case of CG 15% of the messages are sent to a number of sharers in the range 101 to 255. This means that the coherence activity in CG is costlier than in FLU, and explains why Hammer that broadcast all coherence actions has less performance gap when compared with Directory in CG than in FLU. We confirm this behavior by studying Figure 10b, which shows the type of βroadcast messages in Directory for each benchmark. In FLU, more than 35% of the coherence actions are for `Fwd_Write` and `Fwd_Read` categories, which are always unicast messages and Hammer treats by employing the costlier broadcasting. To this percentage, we have to add those `Invalidation` messages sent to a single block sharer. This explains why a large number of βroadcast messages can be resolved in Hammer by means of unicast messages (67% as it is shown in Figure 10a). In contrast, in CG, less than 3% of messages belong to the `Fwd_Write` and `Fwd_Read` categories, so the negative effect of broadcasting in Hammer has less impact on performance improvement than in FLU.

HPC presents the largest difference in terms of βPI when comparing Directory and Hammer (200$\times$). However, this does not translate into the highest performance improvement. In fact, as we showed in Figure 8a, in HPC there is negligible performance differences among the three coherence protocols. As we showed in Figure 7b, this benchmark has very high LLC miss rate (close to 90%), due to very poor spatial/temporal locality of its working set. Figure 9b confirms this behavior as the percentage of broadcast traffic in Directory is close to zero, meaning that marginal coherence activity is actually necessary in HPC. Besides, Figure 10b illustrates that, this marginal coherence activity is largely due to unicast messages in Directory: 95% of the βroadcast messages belong to the `Fwd_Write` and `Fwd_Read` categories, whereas Figure 10a confirms this as 98% of the βroadcast-based coherence actions deal with a single block owner.



(a) Block sharers per βroadcast message.



(b) βroadcast messages breakdown.

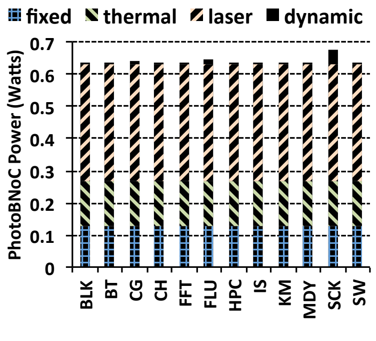
Figure 10: Analysis of coherence activity in Directory depending on the type of βroadcast message (`Invalidation`, `Fwd_Write` and `Fwd_Read`). Both Hammer and Directory employ broadcasting to transmit all these βroadcast messages.

Nevertheless, for these actions, Hammer employs the costlier broadcast and that is why the network traffic increases compared to Directory (Figure 8b).

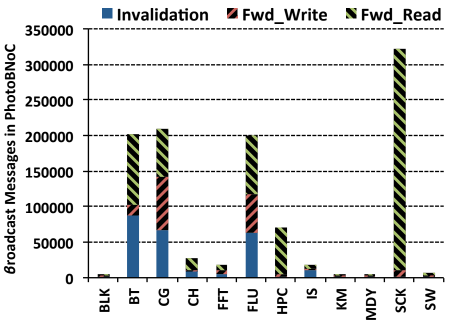
SCK and IS have moderate performance advantages when using Directory protocol (around 12% lower execution times than Hammer). As to SCK, Figure 10b shows that 94% of the βroadcast messages fall into the `Fwd_Read` category. This means that Directory employs a single message while Hammer relies on broadcasting. Even that this increments network traffic (Figure 8b), this does not implies large performance degradation because the L2 caches that do not store a copy of the memory block involved in the coherence operation, simply ignore the message. That is, the unnecessary `Fwd_Read` messages are not in the critical path of the coherence maintenance. In contrast, in IS, 99% of the βroadcast messages fall into the `Invalidation` category where there is a significant amount of block sharers (45% of the `Invalidation` messages deal with more than 51 sharers). In consequence, the large number of invalidation messages in Directory cannot obtain a huge performance improvement as compared to Hammer, which broadcast all these messages. Similarly, SW and MDY show little (5%) performance improvements when comparing Directory and Hammer. The reason is that, these benchmarks are compute intensive workloads with little demand for network traffic (Figure 8b).

The last two benchmarks, CH and FFT show very large performance improvements when comparing Directory and Hammer (40% and 66%, respectively), even that there is a moderate relative difference in coherence activity and increment in network traffic. The reason is that in these benchmarks more than 28% and 52%, respectively, of the execution time in Hammer, is spent in synchronization operations due to Lock/Unlock and Barrier primitives, which involve thousands of read/modify/write operations over global variables that are handled by Directory in much more efficient way. In Hammer, over 70% of the βroadcast traffic belongs to the `Fwd_Read` and `Fwd_Write` categories, and broadcasting these messages increases synchronization latency leading to very low IPC when using Hammer protocol (69 and 23, respectively), what explains the large performance improvements when using Directory in these two benchmarks (the IPC increases to 114 and 70, respectively).

Finally, although ECONO achieves similar performance improvements when compared to Directory, for some benchmarks ECONO can reach higher level of improvements. In particular, BT, CG, FFT, FLU and IS show an average of 5% lower execution time than ECONO. There are two major reasons for this. First, although ECONO maintains coherence by using broadcasting, we transmit all βroadcast messages through the fast and lightweight dedicated PhotoBNoC network. By using this interconnect, a single broadcast message reach all L2 caches in 12 cycles. However, the longest path for Directory is 46 cycles, thereby ECONO can optimize transmission latency for those L2 caches which are further away than 12 cycles (75% of the total number of L2 caches). Second, unlike Directory, ECONO does not need ACK messages for the `Invalidation` coherence actions. In this way, LLC banks in our ECONO protocol do not have to wait until they receive all the required ACK responses and also, we can save more network traffic from the 2D-mesh interconnect which reduces congestion of packets when competing for network resources.



(a) PhotoBNoC power dissipation.



(b) βroadcast messages in PhotoBNoC.

Figure 11: Network power dissipated by PhotoBNoC when running the benchmarks in our 256-core system using our ECONO protocol. `fixed`= clock and leakage power; `thermal`= Thermal tuning power; `laser`= Laser power; `dynamic`= Data-traffic dependent power.

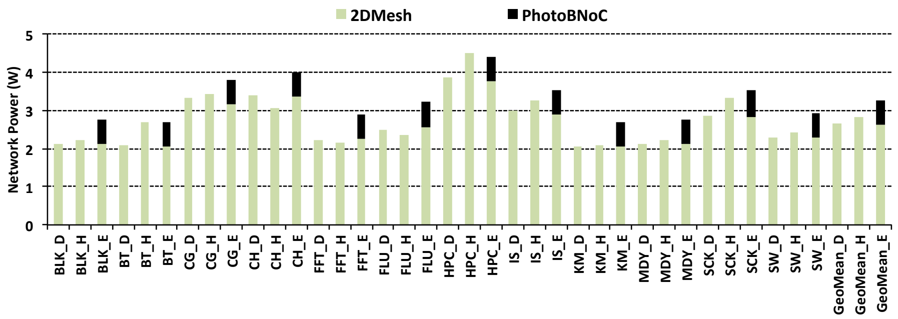
6.2. Power and Energy Analysis

Figure 11 shows the power breakdown dissipated by the PhotoBNoC when running the benchmarks with ECONO in our 256-core system. The power dissipated corresponds to the number of silicon-photonic components required by the PhotoBNoC listed in Table 7. Note that, PhotoBNoC comprises 64 SWBR channels and the number of wavelengths is equal to 64. We assign a single wavelength for each channel in order to lower the number of ring resonators so that we can achieve even lower power dissipated by PhotoBNoC, at the cost of higher serialization latency. Figure 11a shows a power breakdown for every benchmark that consists of fixed (clock and leakage power), thermal (thermal tuning power), laser (laser power) and dynamic power (data-traffic dependent power). Our PhotoBNoC design is very power efficient as it just dissipates less than 0.7 Watts for all benchmarks. In particular, this corresponds to an average energy per bit cost of 1.26 pJ/b, which is an energy efficient value considering the typical range of pJ/bit values typically explored in silicon-photonic NoC designs [9]: from 0.1 pJ/bit to 2 pJ/bit, i.e., from aggressive to conservative silicon-photonic technology designs, respectively.

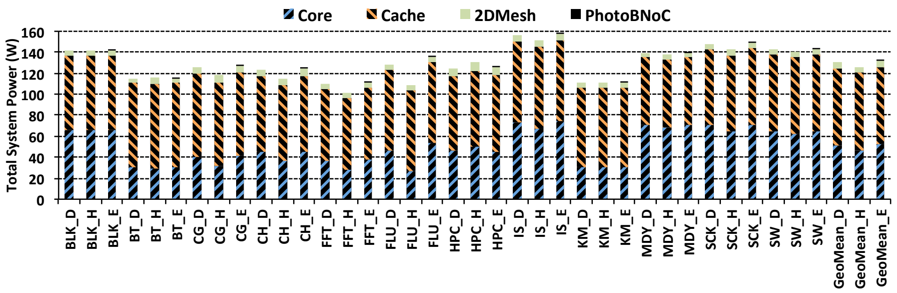
To understand the power results, Figure 11b illustrates the number of βroadcast messages transmitted through the PhotoBNoC benchmark for every benchmark. Note that, the dynamic portion of power dissipation depends on the number of βroadcast messages as well as the total time employed to execute the benchmarks. Although we do not include a figure that shows the total execution time for every benchmark (Figure 8a shows normalized times to Hammer protocol), Figure 7a allows us understand the execution times for every benchmark – the higher IPC the lower execution time. Therefore, SCK that reports high IPC and the largest number of βroadcast messages presents the highest dynamic power dissipation, which is only 0.04 Watts.

Figure 12a illustrates the small fraction of network power that our PhotoBNoC interconnect represents when added to the electrical 2D-mesh network power for ECONO. The power dissipated by the 2D-mesh NoC includes all sources of power dissipation: leakage, clock and dynamic power. PhotoBNoC just accounts for an average of 19.5% of the total network power. Besides, the design for our baseline electrical 2D-mesh NoC is very efficient too, since less than 4.5 Watts are dissipated in the worst case (HPC in Hammer). Moreover, on average our network design equipped with the PhotoBNoC for ECONO achieves comparable network power to Hammer. However, there are benchmarks such as CH, FFT and FLU, where ECONO achieves much higher IPC (Figure 7a), thus the overall network power is higher in Directory and ECONO than in Hammer.

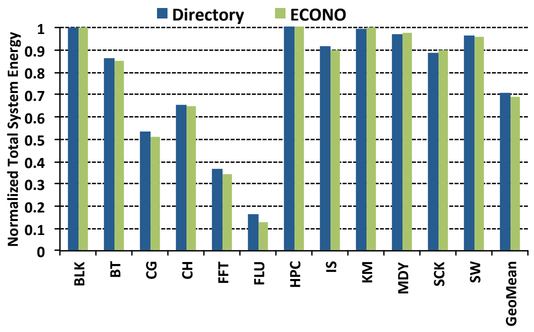
Figure 12b shows a breakdown of the overall system power considering all the main components of the many-core chip: cores, cache hierarchy and network. Benchmarks that report the highest scalability (i.e., the highest IPC illustrated in Figure 7a) such as BLK, IS, MDY, SCK and SW achieve the largest core power dissipation. On average, the core power dissipation is around 50 Watts. On the other hand, the large on-chip cache capacity provided for our target many-core system to be capable of storing the working set for every application contributes to over 70 Watts of the total system power. The overall system power is less than 160 Watts, which is affordable considering contemporary TDPs for large Data Center-based many-core systems (over 200 Watts [1]). On average, ECONO obtains similar power dissipation as compared



(a) Overall network power dissipation.



(b) Overall system power dissipation.



(c) Normalized Overall system Energy to Hammer.

Figure 12: Power and energy analysis. The name of the benchmarks includes a suffix among the following: _D for Directory protocol, _H for Hammer protocol and _E for ECONO protocol.

to Directory protocol. The reasons are that our proposal can achieve slightly better performance improvements than Directory and that the power dissipated by the PhotoBNoC is very low.

Finally, Figure 12c shows the overall system energy for the three coherence protocols. We normalized the energy values with respect to Hammer. Each bar in the plot includes all sources of energy consumption. The resulting energy values are consequence of the improvements in execution time that each Directory and ECONO achieved (Figure 8a). ECONO reports similar energy improvements with respect to Directory and for some benchmarks such as CG and FLU, there can be up to 4% lower energy consumption.

6.3. On-chip Area Overhead

Previous section shows that ECONO can achieve slightly better performance and energy consumption results than Directory. Apart from these benefits, our proposal can save on-chip area as it does not need to store a list of sharers nor the owner id. Nonetheless, ECONO requires that the many-core system is equipped with the PhotoBNoC in order to efficiently transmit the βroadcast messages. In this section, we quantify the on-chip area needed by PhotoBNoC and compare it against the area required by different implementations for Directory protocol.

To estimate area, we assume the same dimensions for the silicon-photonic components utilized in [24]: the waveguides are single mode and have a pitch of 4 μm to minimize the crosstalk between neighboring waveguides. The modulator ring and filter ring diameters are $\sim$10 μm. Table 7 shows the number of different photonic components and the area occupied by those devices in the PhotoBNoC. Our calculations show that the area of the photonic devices of the PhotoBNoC equals 2.3 mm^2, which represents just 0.9% of the total chip area.

We compare the area overhead required by PhotoBNoC against different types of encoding representations for Directory, ranging from the traditional full-map bit-vector scheme (FM) to those optimized representations of *Hierarchical*(HC) [40] and *SCD* [41]. In HC, there is an exact and area-efficient representation of sharer sets, where a hierarchy of two levels of directories are

Table 6: Storage requirements for different Directory protocols against PhotoBNoC required by *ECONO* protocol (ECO) for the target 256-core system and scaled versions of the system.

Cores	***FM***	***HC***	***SCD***	***ECO***	***FM* vs *ECO***	***SCD* vs *ECO***
128	34.18%	21.09%	10.94%	4.38%	7.80×	2.50×
256	59.18%	24.22%	12.50%	6.11%	9.69×	2.05×
512	109.18%	26.95%	13.87%	8.68%	12.85×	1.63×
1024	209.18%	30.86%	15.82%	12.48%	17.51×	1.32×

used: each first-level directory encodes the sharers of a subset of caches, and the second level tracks directories of the previous one. *SCD* constitutes one of the most efficient directories to date. It encodes exact sharer sets by using variable-size representations: lines with one or few sharers use a single directory tag, while widely shared lines use additional tags. To increase performance it also leverages novel highly-associative caches as in Cuckoo Directory [42].

Table 6 shows the storage overhead depending on the three directory organizations considered (Full-map *Directory*, *Hierarchical* and *SCD*) for our target 256-core system and scaled versions of the system (128, 512 and 1024 cores) to show the scalability trend. As in [41], area overhead is given as a percentage of the total area required by aggregating all L2 caches in every case (we use the L2 cache size shown in Table 1). We base the percentage on the study carried out in [41]. Last column accounts for *ECONO*. In this case, the percentages stem from the *ECONO* on-chip area overhead considering the PhotoBNoC system. The table illustrates that our *ECONO* protocol considerably reduces the on-chip area overhead of a *Full-map Directory* (9.69×), and save more than 2× area with respect to the most efficient an scalable *SCD* protocol. In addition, we can affirm that our solution scales better than the SCD protocol for all considered system sizes[3]. From this study, we conclude that *ECONO* is the most scalable protocol in terms of area overhead.

[3]For the scaled many-core systems, we size the PhotoBNoC's segments lengths according to their chip dimensions estimated with the methodology explained in Section 2.1.

Table 7: Photonic components required by distinct silicon-photonic NoC designs. Waveguide lengths for the four PhotoBNoC's segments are: 15 mm, 33 mm, 51mm and 68 mm. PhotoBNoC_v0 and PhotoBNoC_v1 are those preliminary designs described in Section 4 with waveguide lengths of 68 mm. PhotoNoC is equipped with 16 64-Byte MWMR 68-mm channels based on [9]. We assume 8 Gbit/second/wavelength; 16 wavelengths/waveguide/direction and 1 wavelength/SWBR channel. PCH = Number and type of photonic channels, WL = Wavelengths, MD = Modulators, FL = Filters, WG = Waveguides. PDA = Photonic device area in mm^2 assuming 10μm-radius rings, 4μm waveguide pitch. In brackets, we show the magnitude of this area w.r.t. total chip area (256 mm^2). PJB = pJoules/bit.

	PCH	WL	MD	FL	WG	PDA (%)	PJB
PhotoBNoC_v0	16×SWBR	16	16	4,096	16	15.23 (5.95)	11.74
PhotoBNoC_v1	16×SWBR	16	16	1,024	6	5.71 (2.23)	3.74
PhotoBNoC	64×SWBR	64	64	1,024	4	2.33 (0.91)	1.26
PhotoNoC [9]	16×MWMR	1,024	6,144	6,144	64	60.92 (23.8)	1.58

7. Alternate NoC Designs for our ECONO protocol

In this Section, we quantify efficiency of our hybrid NoC solution for ECONO protocol (2D-mesh electrical and PhotoBNoC) as compared to alternate NoC proposals. Particularly, we study the preliminary PhotoBNoC_v0 and PhotoBNoC_v1 infrastructures explained in Section 4 and, apart from these two alternate NoCs for broadcasting the ECONO's ACN messages, we consider a state-of-the-art silicon-photonic multi-bus NoC namely PhotoNoC [9], which is a fully-photonic NoC to transfer all type of network traffic.

7.1. Silicon-Photonic Resources

Table 7 lists the amount of photonic resources required by all the four networks. Recall that PhotoBNoC_v0 includes 16 SWBR channels where for each channel there is one sender (an L3 bank) and 256 receivers (the number of L2 caches). PhotoBNoC_v1 lowers the amount of photonic devices and signal losses by leveraging network concentration at the destination nodes. In this way, instead of 256 receivers (the number of L2 caches), we consider 64 receivers where each receiver is a router (each router interfaces 4 L2 caches). As we can see in the Table, our PhotoBNoC is a more efficient design in terms of number of waveguides as compared to PhotoBNoC_v0 and PhotoBNoC_v1 (4 vs. 6 and 16, respectively). This indicates that just 0.91% overhead is required to implement our PhotoBNoC infrastructure. Also, PhotoBNoC is a more energy efficient design and is within the reasonable energy/bit consumption range explained in Section 6.2 (1.26 pJ/bit vs 11.74 pJ/bit and 3.74 pJ/bit, respectively).

On the other hand, we chose PhotoNoC as an example of a competitive silicon-photonic NoC as it has been demonstrated to be efficient for application performance in a 1024-core system, and it can be easily optimized to reduce laser power dissipation dynamically. To implement PhotoNoC in our 256-core target system, we empirically estimated the best-performing network configuration that saturates application performance for all benchmarks under study. As a result, our PhotoNoC design comprises 16 64-Byte silicon-photonic Multiple-Write-Multiple-Read (MWMR) buses (8 in L2-to-L3 direction and 8 in L3-to-L2 direction), where each bus is attached to an Access Point (AP) at L2 side (4 cores share one AP), and 8 APs at L3 side (1 Memory Controller and 2 L3 shared banks share one AP). Following the recommendations given in [9], we also use concentrations at the L3 side and the L2 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. A logical view of our PhotoNoC design is illustrated in Figure 13 where, to simplify the representation of the network layout, all MCs and shared L3 banks of the 256-core system are placed along one edge of the chip.

Table 7 shows the photonic devices required by PhotoNoC. As we can see, even though we utilize network concentrators, the number of photonic devices is much larger as compared to the other NoCs. This is necessary as PhotoNoC conveys all network traffic while the other designs utilize the concentrated 2D-mesh electrical network for all traffic but the ACN messages. Moreover, as we can see in the PJB column, this design is very energy efficient as it reports 1.58 pJ/b which is very close to the 1.26 pJ/b required by PhotoBNoC. However, the on-chip area overhead of PhotoNoC is not acceptable for a monolithic integration of the photonic devices, because it is larger than 10% of total chip area (23.8%).

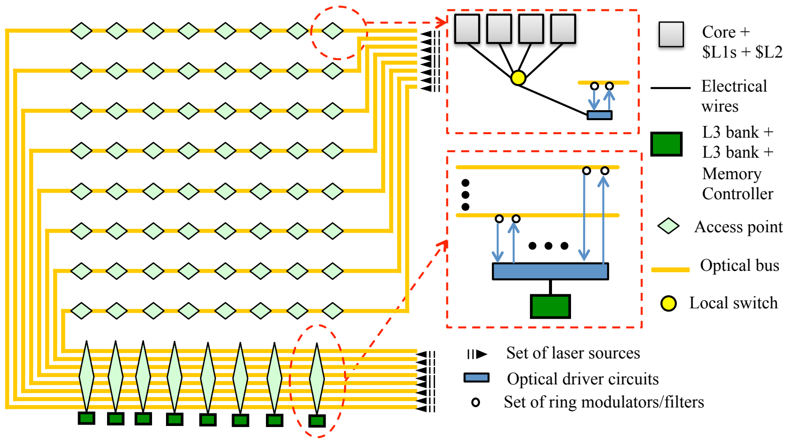


Figure 13: Logical layout for the Photonic Multi-bus NoC (PhotoNoC) design.

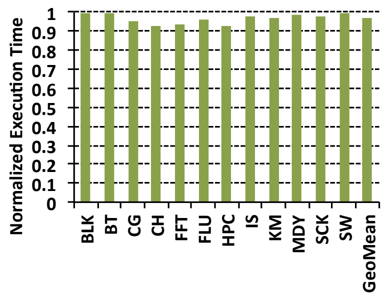
So, to implement PhotoNoC in our 256-core system, a much costlier and complex 3D integrated design would be necessary (further details in Section2.2).

7.2. Application Performance and Energy Efficiency

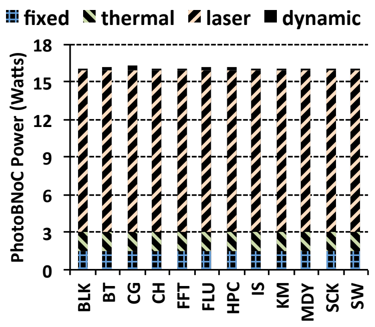
In this Section, we compare our hybrid NoC solution to implement ECONO protocol with PhotoNoC in terms of performance, power dissipation and Energy-Delay-Product (EDP) metric for the benchmarks under study (see Figure 14).

Figure 14a presents the execution times achieved by PhotoNoC for the benchmarks normalized to PhotoBNoC resulting times. As we can see, PhotoNoC lowers execution time by a maximum of 9% (3.3% on average) especially for those benchmarks with the highest demand for network traffic (see CH, CG and HPC in Figure 8b). The reason is that in PhotoNoC all network traffic is transferred through the low-diameter high-radix MWSR buses, which helps shorten the amount of network hops required by PhotoBNoC when using the high-diameter low-radix electrical 2D-mesh NoC for non-ACN messages.

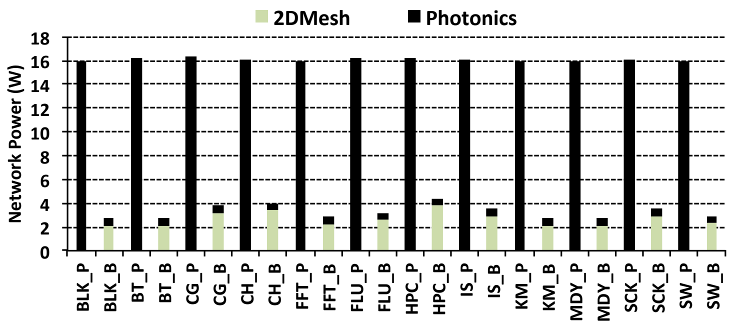
Figure 14b presents a power breakdown of PhotoNoC with similar categories as illustrated in Figure 12a for our PhotoBNoC. As we can see, PhotoNoC requires 40× higher power dissipation than PhotoBNoC (0.45 vs 16 Watts) because of the much larger amount of wavelengths, ring resonators and waveguides



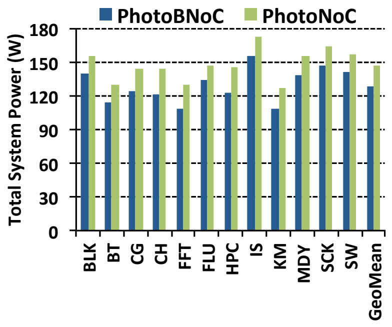
(a) Execution time of ECONO with PhotoNoC normalized to ECONO with our PhotoBNoC.



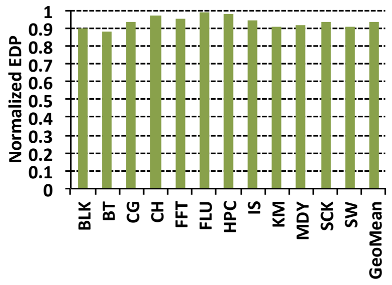
(b) Network power in PhotoNoC.



(c) Overall network power. The name of the benchmarks includes a suffix among the following: _P for PhotoNoC and _B for our PhotoBNoC.



(d) Overall system power.



(e) Overall EDP of ECONO with PhotoNoC normalized to ECONO with our PhotoBNoC.

Figure 14: Performance, power and EDP comparisons of our proposed ECONO implementation with PhotoBNoC against an alternate ECONO implementation with PhotoNoC [9].

(see Table 7). As our PhotoBNoC needs an electrical 2D-mesh NoC, Figure 14c includes the overall network power considering both sub-networks. We can see that, even considering the electrical sub-network, our PhotoBNoC design still requires 4× less network power (4.3 W vs. 16.2 W).

Finally, we analyze the overall system power and energy efficiency when running the benchmarks for the two implementations of ECONO. First, as shown in Figure 14d, the power dissipation for the whole system is larger in case of PhotoNoC (an average of 14% more power), and second, Figure 14e reveals that, despite achieving an average of 3.3% lower execution times when using PhotoNoC, our solution with the hybrid NoC is the most energy efficient NoC design for all cases (PhotoBNoC gets an average of 7% lower EDP).

8. Related Work

In this Section, we carry out a literature review about the most relevant approaches to the cache coherence challenge.

Snooping-based coherence protocols maintain coherence in a simple way and work well in small-scale systems, but their performance is highly compromised beyond a handful of cores, due to their prohibitive bandwidth overhead, even with improvements such as snoop filters [43, 44]. On the other hand, directory-based protocols drastically reduce the network requirements at the cost of introducing a directory that stores the sharing information and results in a non-scalable on-chip area overhead. Multiple solutions have been proposed to minimize directory overhead, such as compressed sharing codes [45, 46, 47, 48, 49], hashing optimizations [42, 41], hierarchical designs [50, 51], or deactivating coherence tracking at different system levels [52, 53, 54].

Directory-free coherence has also been proposed in order to fully remove the directory. One example is the hammer protocol evaluated in this paper, which has been proved to be a non scalable solution for large many-core systems. To take advantage of the goodness of these broadcast-based coherence protocols, there are several proposals that leverage novel technologies, such as silicon-photonic link technology, to optimize the broadcast transmissions. For instance, *ATAC* [55] that operates in the same way as a conventional limited directory, but when the capacity of the sharer list is exceeded, it resorts to very efficient optical-based broadcasts to invalidate all possible block sharers. Besides, *Atomic Coherence* [26] simplifies considerably protocol design and verification by approximating to the maintenance of coherence in bus-based machines. All coherence requests perform atomically and hence, all possible racing requests are removed. To do so, a very efficient optical-based mutex is implemented. Contrarily, *ECONO* enables races from the requesting cores to home tiles, and serialization is only employed to atomically broadcast express coherence notifications, thus increasing concurrency. Moreover, we do not inject an important amount of coherence-related traffic into the CMP's interconnect, hence saving on-chip traffic and energy.

ECONO meets the trade-off between these two solutions by employing a very efficient photonic network to deliver broadcast messages. Therefore, unlike snoopy protocols, *ECONO* is not bandwidth intensive since it releases the main interconnect from an important amount of coherence-related traffic and, unlike directory-based protocols, it does not devote any storage overhead to track information about sharers, thereby saving on-chip area and energy. In Section 6, we have demonstrated that ECONO+PhotoBNoC is an energy efficient co-design that can achieve the highest performance, and the most scalable solution in terms of on-chip area overhead when compared to SCD, which is considered one of the most scalable directory-based coherence protocols to date.

9. Conclusions

In this work, we propose and discuss the design of an efficient photonic-based broadcast network design called PhotoBNoC. PhotoBNoC represents the substrate for the ECONO coherence protocol. Particularly, PhotoBNoC guarantees both atomic and fast transmissions for the ECONOs atomic broadcast coherence messages. To do so, it leverages a novel network layout with distance-dependent segmented SWBR waveguides to save laser power and lower on-chip area overhead. Atomic transmissions are guaranteed through SWBR channels, and both lightweight local arbiters and buffer queues at destination nodes.

Through detailed simulations of a 256-core, we demonstrate that the combination of ECONO+PhotoBNoC outperforms directory-based protocols and broadcast-based protocols with a single solution to the cache coherence problem in future many-core CMPs. Particularly, ECONO+PhotoBNoC can obtain similar performance levels to the non-scalable and area-hungry Directory protocol. At the same time, it improves performance compared with Hammer, but maintains its advantage in area. Regarding power dissipation, our detailed study reveals that PhotoBNoC barely has impact on it, and ECONO+PhotoBNoC can also mimic the results of Directory in terms of energy efficiency. Finally, when compared to a competitive state-of-the-art fully-photonic NoC, namely PhotoNoC, our hybrid NoC design does not require a complex and expensive 3D stacking integration of the photonic devices due to its much lower area overhead (less than 30×) and power dissipation (less than 4×), while achieving similar performance results (an average of 3.3% higher execution times).

Acknowledgments

The authors would like to thank the anonymous reviewers and the editor for their valuable comments and suggestions to improve the quality of the manuscript. This work is supported by the Spanish MINECO under grant “TIN2016-78799-P” and by the Fundación Séneca-Agencia de Ciencia y Tecnología de la Región de Murcia under grant “19295/PI/14”.

References

[1] A. Sodani, R. Gramunt, J. Corbal, H.-S. Kim, K. Vinod, S. Chinthamani, S. Hutsell, R. Agarwal, Y.-C. Liu, Knights landing: Second-generation intel xeon phi product, IEEE Micro 36 (2) (2016) 34–46.

[2] B. Bohnenstiehl, A. Stillmaker, J. Pimentel, T. Andreas, B. Liu, A. Tran, E. Adeagbo, B. Baas, A 5.8 pj/op 115 billion ops/sec, to 1.78 trillion ops/sec 32nm 1000-processor array, in: Proc. of 2016 Symposium on VLSI Technology and Circuits, 2016, pp. 1–2.

[3] A. Ahmed et al., AMD Opteron Shared Memory MP Systems, in: Proc. HotChips Symposium, 2002.

[4] L. A. Barroso et al., Piranha: A Scalable Architecture Based on Single-Chip Multiprocessing, in: Proc. IEEE International Symposium on Computer Architecture, 2000.

[5] M. B. Taylor et al., The Raw Microprocessor: a Computational Fabric for Software Circuits and General-Purpose Programs, IEEE Micro 22(2) (2002) 25–35.

[6] M. Lodde, J. Flich, M. E. Acacio, Heterogeneous noc design for efficient broadcast-based coherence protocol support, in: Proc. of the 6th IEEE/ACM International Symposium on Networks-on-Chip, 2012, pp. 59–66.

[7] J. L. Abellán, A. Ros, J. Fernández, M. E. Acacio, Econo: Express coherence notifications for efficient cache coherency in many-core cmps, in: Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIII), 2013 International Conference on, IEEE, 2013, pp. 237–244.

[8] T. Krishna, A. Kumar, L.-S. Peh, J. Postman, P. Chiang, M. Erez, Express virtual channels with capacitively driven global links, IEEE Micro 29 (4) (2009) 48–61.

[9] J. Abellán, C. Chao, A. Joshi, Electro-photonic noc designs for kilocore systems, ACM Journal on Emerging Technologies in Computing (2016) –.

[10] Mellanox Technologies, Seventy two core processor soc with 8x 10gb ethernet ports, pcie and networking offloads, `http://www.mellanox.com/related-docs/prod_multi_core/PB_TILE-Gx72.pdf` (2016).

[11] B. Daya, et al., Scorpio: A 36-core research chip demonstrating snoopy coherence on a scalable mesh noc with in-network ordering, in: Computer Architecture (ISCA), 2014 ACM/IEEE 41st International Symposium on, 2014, pp. 25–36.

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

[13] A. Sodani, Knights landing (knl): 2nd generation intel xeon phi processor, in: Hot Chips, Vol. 27, 2015.

[14] S. Beamer, et al., Re-architecting DRAM memory systems with monolithically integrated silicon photonics, in: 37th Annual International Symposium on Computer Architecture, ISCA 2010, 2010, pp. 129–140.

[15] S. Li, et al., McPAT: An integrated power, area, and timing modeling framework for multicore and manycore architectures, in: Proc. MICRO-42, 2009, pp. 469–480.

[16] C. Sun, C.-H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, V. Stojanovic, Dsent-a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling, in: Networks on Chip (NoCS), 2012 Sixth IEEE/ACM International Symposium on, IEEE, 2012, pp. 201–210.

[17] M. J. Cianchetti, J. C. Kerekes, D. H. Albonesi, Phastlane: A Rapid Transit Optical Routing Network, SIGARCH Computer Architecture News 37 (3).

[18] Y. Pan, et al., Firefly: Illuminating Future Network-on-chip with Nanophotonics, SIGARCH Computer Architecture News 37 (3).

[19] A. Joshi, et al., Silicon-photonic clos networks for global on-chip communication, in: Networks-on-Chip, 2009. NoCS 2009. 3rd ACM/IEEE International Symposium on, 2009, pp. 124–133.

[20] L. Ramini, D. Bertozzi, L. Carloni, Engineering a bandwidth-scalable optical layer for a 3d multi-core processor with awareness of layout constraints, in: Proc. International Symposium on Networks-on-Chip (NOCS), 2012, pp. 185–192.

[21] M. Georgas, et al., A Monolithically-Integrated Optical Receiver in Standard 45-nm SOI, IEEE Journal of Solid-State Circuits 47.

[22] B. Moss, et al., A 1.23pj/b 2.5gb/s monolithically integrated optical carrier-injection ring modulator and all-digital driver circuit in commercial 45nm soi, in: ISSCC, 2013, pp. 18–20.

[23] J. S. Orcutt, et al., Nanophotonic integration in state-of-the-art cmos foundries, Opt. Express.

[24] J. Abellan, A. Coskun, A. Gu, W. Jin, A. Joshi, A. B. Kahng, J. Klamkin, C. Morales, J. Recchio, V. Srinivas, T. Zhang, Adaptive tuning of photonic devices in a photonic noc through dynamic workload allocation, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on XX (2016) pp–pp. `doi:10.1109/TCAD.2016.2600238`.

[25] D. J. Sorin et al., A Primer on Memory Consistency and Cache Coherence, Synthesis Lectures on Computer Architecture#16, 2011.

[26] D. Vantrease et al., Atomic Coherence: Leveraging Nanophotonics to Build Race-Free Cache Coherence Protocols, in: Proc. IEEE International Symposium on High-Performance Computer Architecture, 2011.

[27] T. Krishna et al., Express Virtual Channels with Capacitively Driven Global Links, IEEE Micro 29(4) (2009) 48–61.

[28] J. L. Abellán, J. Fernández, M. E. Acacio, Efficient and scalable barrier synchronization for many-core cmps, in: Proceedings of the 7th ACM international conference on Computing frontiers, ACM, 2010, pp. 73–74.

[29] J. L. Abellán et al., GLocks: Efficient Support for Highly-Contended Locks in Many-Core CMPs, in: Proc. IEEE International Parallel & Distributed Processing Symposium, 2011.

[30] C. Batten, A. Joshi, V. Stojanovć, K. Asanović, Designing chip-level nanophotonic interconnection networks, in: Integrated Optical Interconnect Architectures for Embedded Systems, Springer, 2013, pp. 81–135.

[31] T. E. Carlson, et al., Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulations, in: Proc. SC, 2011, pp. 1–12.

[32] J. Meng, C. Chen, A. K. Coskun, A. Joshi, Run-time energy management of manycore systems through reconfigurable interconnects, in: Proceedings of the 21st edition of the great lakes symposium on Great lakes symposium on VLSI, ACM, 2011, pp. 43–48.

[33] K. Gharachorloo, M. Sharma, S. Steely, S. Van Doren, Architecture and design of alphaserver gs320, in: Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems, 2000, pp. 13–24.

[34] D. Bailey, et al., The NAS parallel benchmarks, Tech. Rep. RNR-94-007 (mar 1994).

[35] S. C. Woo, et al., The SPLASH-2 programs: characterization and methodological considerations, in: Proc. ISCA, 1995, pp. 24–36.

[36] C. Bienia, et al., The PARSEC benchmark suite: Characterization and Architectural Implications, in: Proc. PACT, 2008, pp. 72–81.

[37] W. keng Liao, Parallel k-means data clustering. (2005).
URL `http://users.eecs.northwestern.edu/~wkliao/Kmeans/index.html`

[38] M. A. Heroux, et al., Improving performance via mini-applications, Sandia National Laboratories, Tech. Rep.

[39] D. Campbell, et al., Ubiquitous high performance computing: Challenge problems specification, Tech. Rep. HR0011-10-C-0145, Georgia Institute of Technology (2012).

[40] S. Guo et al., Hierarchical Cache Directory for CMP, Journal of Computer Science and Technology 25(2) (2010) 246–256.

[41] D. Sanchez and C. Kozyrakis., SCD: A Scalable Coherence Directory with Flexible Sharer Set Encoding, in: Proc. IEEE International Symposium on High-Performance Computer Architecture, 2012.

[42] M. Ferdman et al., Cuckoo Directory: A Scalable Directory for Many-Core Systems, in: Proc. IEEE International Symposium on High-Performance Computer Architecture, 2011.

[43] A. Moshovos, G. Memik, B. Falsafi, A. N. Choudhary, JETTY: Filtering snoops for reduced energy consumption in SMP servers, in: 7th Int'l Symp. on High-Performance Computer Architecture (HPCA), 2001, pp. 85–96.

[44] J. H. Kelm et al., WAIYPOINT: Scaling Coherence to 1000-core Architectures, in: Proc. IEEE International Conference on Parallel Architectures and Compilation Techniques, 2010.

[45] A. Agarwal, R. Simoni, J. L. Hennessy, M. A. Horowitz, An evaluation of directory schemes for cache coherence, in: 15th Int'l Symp. on Computer Architecture (ISCA), 1988, pp. 280–289.

[46] A. Gupta, W.-D. Weber, T. C. Mowry, Reducing memory traffic requirements for scalable directory-based cache coherence schemes, in: Int'l Conf. on Parallel Processing (ICPP), 1990, pp. 312–321.

[47] S. S. Mukherjee, M. D. Hill, An evaluation of directory protocols for medium-scale shared-memory multiprocessors, in: 8th Int'l Conf. on Supercomputing (ICS), 1994, pp. 64–74.

[48] M. E. Acacio, J. González, J. M. García, J. Duato, A new scalable directory architecture for large-scale multiprocessors, in: 7th Int'l Symp. on High-Performance Computer Architecture (HPCA), 2001, pp. 97–106.

[49] A. Ros, M. E. Acacio, Dasc-dir: a low-overhead coherence directory for many-core processors, Journal of Supercomputing (SUPE) 71 (3) (2015) 781–807.

[50] M. R. Marty, M. D. Hill, Virtual hierarchies to support server consolidation, in: 34th Int'l Symp. on Computer Architecture (ISCA), 2007, pp. 46–56.

[51] M. M. K. Martin, M. D. Hill, D. J. Sorin, Why on-chip cache coherence is here to stay, Commun. ACM 55 (7) (2012) 78–89.

[52] B. Cuesta, A. Ros, M. E. Gómez, A. Robles, J. Duato, Increasing the effectiveness of directory caches by deactivating coherence for private memory blocks, in: 38th Int'l Symp. on Computer Architecture (ISCA), 2011, pp. 93–103.

[53] A. Esteve, A. Ros, M. E. Gómez, A. Robles, J. Duato, Efficient tlb-based detection of private pages in chip multiprocessors, IEEE Transactions on Parallel and Distributed Systems (TPDS) 27 (3) (2016) 748–761.

[54] A. Ros, A. Jimborean, A hybrid static-dynamic classification for dual-consistency cache coherence, IEEE Transactions on Parallel and Distributed Systems (TPDS) 27 (11) (2016) 3101–3115.

[55] G. Kurian et al., ATAC: A 1000-Core Cache-Coherent Processor with On-Chip Optical Network, in: Proc. IEEE International Conference on Parallel Architectures and Compilation Techniques, 2010.