ATAC: A 1000-Core Cache-Coherent Processor with On-Chip Optical Network

George Kurian
gkurian@csail.mit.edu

Jason E Miller
jasonm@csail.mit.edu

James Psota
jim@csail.mit.edu

Jonathan Eastep
eastepjm@csail.mit.edu

Jifeng Liu
FIXME@mit.edu

Jurgen Michel
FIXME@mit.edu

Lionel Kimerling
lckim@mit.edu

Anant Agarwal
agarwal@csail.mit.edu

Massachusetts Institute of Technology
Cambridge, MA 02139

ABSTRACT

Ever since industry has turned to parallelism instead of frequency scaling to improve processor performance, multicore processors have continued to scale to larger and larger numbers of cores. Some believe that multicores will have 1000 cores or more by the middle of the next decade. However, their promise of increased performance will only be reached if their inherent scaling and programming challenges are overcome. Meanwhile, recent advances in nanophotonic device manufacturing are making CMOS-integrated optics a reality—interconnect technology which can provide significantly more bandwidth at lower power than conventional electrical analogs. Perhaps more importantly, optical interconnect also has the potential to enable new, easy-to-use programming models enabled by an inexpensive broadcast mechanism.

The contributions of this paper are two-fold. (1) It presents ATAC, a new manycore architecture that capitalizes on the recent advances in optics to design a novel opto-electronic network that addresses a number of the challenges that future manycore designs will face. (2) It introduces ACKwise, a novel directory-based cache coherence protocol that provides high performance and scalability on any large-scale optical interconnection network. Using simulations with Splash2 and synthetic benchmarks, we show that the ATAC network coupled with ACKwise out-performs a chip consisting of an electrical mesh network of similar area and conventional directory based cache coherence protocols in all scenarios.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: FIXME; D.2.8 [**Software Engineering**]: FIXME—*complexity measures, performance measures*

General Terms

FIXME

PACT'10, September 11–15, 2010, Vienna, Austria.

Keywords

FIXME, FIXME

1. INTRODUCTION

The trend in modern microprocessor architectures is clear: multicore is here. As silicon resources become increasingly abundant, processor designers are able to place more and more cores on a chip with massive multicore chips on the horizon. Many industry pundits have predicted manycores with 1000 or more cores by the middle of the next decade. But will current processor architectures (especially their interconnection mechanisms) scale to thousands of cores and will programming such systems be tractable? This paper argues that current multicore architectures will not scale to thousands of cores and introduces ATAC (pronounced ā-tack), a new processor architecture that addresses these issues. ATAC integrates an on-chip optical broadcast communication network within a mesh based tiled multicore architecture to significantly improve the performance, energy scalability, and ease of programmability of multicore processors [27, 5].

Although Moore's Law enables increasing numbers of cores on a single chip, the extent to which they can be used to improve performance is limited both by the cost of communication among the cores and off-chip memory bandwidth. Although our research is investigating the application of optical technology to both problems, this paper focuses on the on-chip interconnect challenge. As computation is spread across multiple cores on a chip, distribution of instructions to the cores, and communication of intermediate values between cores account for an increasing fraction of execution time due to both latency and contention for communication resources. The outlook is particularly dismal for applications that require a lot of global communication operations (*e.g.,* broadcasts to maintain cache coherence) because each such operation ties up many resources and consumes a lot of energy.

State-of-the-art multicore chips employ one of two strategies to deal with interconnection costs. Small-scale multicores typically interconnect cores using a bus. This simple design does not scale to large numbers of cores due to increasing bus wire length and contention. More scalable interconnection strategies use point-to-point networks. For example, the Raw microprocessor [20] uses a mesh interconnect. This avoids long global wires but communication between distant cores requires multiple hops. Furthermore, contention will become prohibitive as processors are scaled to thousands of cores.

Multicore architectures are also threatened by the programming challenge. Multicore programmers must spatially and temporally

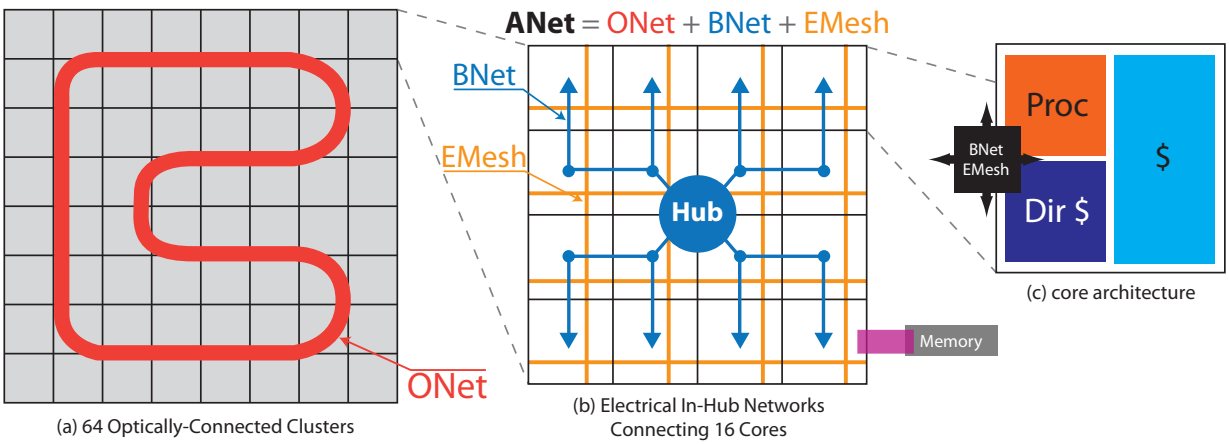

Figure 1: ATAC architecture overview

orchestrate computation and communication if they want to extract high performance from the hardware. Even simple functions like supplying instructions to all the cores is difficult even for the common, simple case where all cores run the same program (this popular approach is called SPMD, single program, multiple data, programming). Broadcast and all-to-all communication operations inherent in many coherence, instruction distribution, and synchronization protocols present even greater challenges.

The ATAC processor architecture addresses these contention and programmability issues using on-chip optical communications technologies to augment electrical communication channels. Optical communications technology is making vast strides at integrating optoelectronic components with standard CMOS fabrication processes. ATAC leverages these advances to eliminate communication contention using Wavelength Division Multiplexing (WDM). WDM allows a single optical waveguide to simultaneously carry multiple independent signals on different wavelengths. For example, a single waveguide with the same switching speed as its electrical counterpart and with 64 WDM channels would match the bandwidth of a 64-bit electrical bus. Optical waveguides, however, can also transmit data at higher speeds than electrical wires (a function of the index of refraction of the waveguide material for the optical signal; and a function of the RC delays, the dielectric material (SiO_2) surrounding the wires, and the delay of required repeaters for the electrical signal). This virtually eliminates the heterogeneous distance-dependent cost function for communication between a pair of cores on any electrical interconnect, which complicates multicore programming. Optical signaling can also use less power than electrical signaling (especially for long wires) because optical waveguides have relatively low loss and therefore do not require periodic repeaters.

ATAC is a tiled multicore chip architecture augmented with an optical broadcast network. ATAC comprises a 2D array of tiles connected with an electrical mesh interconnect. In addition, ATAC utilizes nanophotonics to implement a low-latency, contention-free global communication network. The optical network consists of a set of optical waveguides that snake through the chip and loop around on themselves to form a continuous loop as shown in Figure 1(a). Optical Hubs transmit data by modulating a laser light source and injecting it into the loop. The signal then quickly propagates around the loop and can be received by all of the other Hubs in a single operation. Thus every message on the optical network has the potential to be a highly efficient global broadcast. Filtering at the receiving Hubs can limit the scope of the message to create multicast or unicast messages.

ATAC's optical network provides the benefits of a central bus while mitigating the drawbacks. Like a bus, the optical network supports broadcast and provides uniform latency between network endpoints. Unlike a bus, it allows multiple senders to communicate simultaneously and is scalable to 1000's of cores. Unlike Corona [10], another multicore with on-chip optical network which allocates wavelengths to receivers and relies on the sender to specify each receiver, ATAC allocates wavelengths to senders and requires receivers to filter desired messages (more detail on related work is in Section 6). This paper introduces the ATAC architecture and develops its scalable cache coherence protocol.

The remainder of this paper is organized as follows. Section 2 provides context for the ATAC architecture, focusing on the constraints imposed on the architecture by the optical and electronic devices. Section 3 provides an overview of the ATAC architecture, including its processing, communication, and memory mechanisms. Section 4 introduces the ACKwise cache coherence protocol. Section 5 evaluates the ATAC architecture using the ACKwise protocol and provides a preliminary set of results using Splash2 and synthetic benchmarks focusing on how ATAC enables high performance cache coherence across 64 and 1024 cores. Section 6 follows with a detailed discussion of related work, and Section 7 concludes the paper.

2. OPTICAL TECHNOLOGY BACKGROUND

Advances in electronic-photonic integration have enabled optical interconnect technologies with greater integration, smaller distances, and higher bandwidths [18], [19], [15], [25]. Further, recent research [17] has shown that optical devices can be built using standard CMOS processes, thereby allowing optics to replace global wires and on-chip buses [1].

This section presents a brief overview of these CMOS compatible devices and their constraints. The key elements in a nanophotonic network such as the one employed by the ATAC chip include: the "optical power supply" light source; waveguides to carry optical signals; filters and modulators to place signals into the waveguides; and detectors to receive signals from the waveguides. This section discusses each of these components and describes the complete path for transmitting data optically.

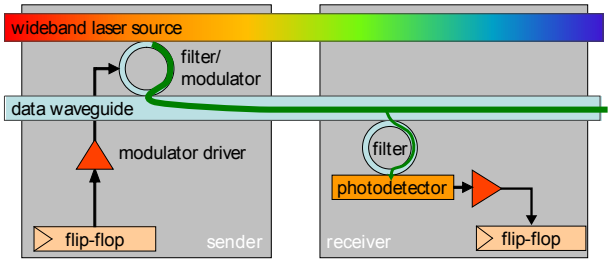

Figure 2: Optical transmission of a single bit

In ATAC the light source, or "optical power supply", is generated by off-chip lasers and coupled into an on-chip waveguide. On-chip light sources exist, but consume large quantities of precious on-chip power and area. The power consumption of an off-chip laser is roughly 1.5 W, with 0.2 W of optical power ending up in the on-chip waveguide.

Waveguides are the on-chip channels for light transmission. They guide and confine light by a combination of a high-refractive-index material on the inside of the waveguide and a low-refractive-index material on the outside (the cladding). Waveguides can be made out of either silicon (Si) or polymer. Due to the fact that Si waveguides can be packed onto a chip at much higher densities and that modulators for Si can be made much more compactly, the ATAC design employs Si waveguides. These waveguides can be manufactured in a standard CMOS process, as both the waveguide and cladding materials are commonly used elsewhere. ATAC requires waveguides with losses of less than 0.3dB/cm and total power capacity of about 10 mW, both of which are achievable with Si.

The optical filter is a ring resonator that couples only a specific wavelength from the power supply waveguide to the data waveguide. The exact wavelength, as well as the spacing between wavelengths, is determined by the ring resonator dimensions and is fixed during manufacturing. Limited tuning can be achieved by changing the ring's temperature or by injecting charge into the ring. The modulator is an optical device that imprints a digital signal on the light extracted by the filter by varying the absorption in the device. Modulators are used to translate an electrical signal (amplified by the modulator driver) into an optical signal, and can therefore be thought of as an "optical switch", placing values onto optical waveguides. The modulators are Ge based electro-absorption modulators with integrated filters. The ring resonators are not used for modulation but just for wavelength filtering. It is assumed that athermal design is implemented, so that the rings do not need to be tuned. The modulators used in the ATAC design have characteristics that are expected to be reached by designs available in 2012: insertion loss of 1dB; area less than 50 μm^2; modulation rate of 1 Gbps; energy required to switch less than 25 fJ; and power consumption of 25 μW at 1 GHz [14].

At the receiving end of a waveguide additional components are used to receive the signal and convert it to an electrical signal. An additional optical filter is used to extract light of a particular wavelength from the data waveguide and transfer it to a photodetector. The photodetector is an extremely sensitive optical device which absorbs photons and outputs an electrical signal. The photodetector proposed for ATAC has a responsitivity of greater than 1 Amp/Watt and 3dB bandwidth performance at 1 GHz. It has an area footprint of less than 20 μm^2. Furthermore, the expected capacitance of the photodetector is less than 1 fF [7]. In current technology nodes, the output of the photodetector would need to be amplified by a power-hungry TIA (transimpedance amplifier) before it could be used to drive a digital circuit. However, starting with the 22nm node, the smaller transistor input capacitances will allow the photodetector to directly drive a digital circuit, greatly reducing power consumption.

Figure 2 puts all of these elements together, showing how one bit is transmitted from a flip-flop of one core to a flip-flop of another core. In this figure, the core on the left shows the components relevant to sending and the core on the right shows the components relevant to receiving; however, in the actual chip all cores would contain both sets of components. From end to end, the process for sending a bit on the ATAC's optical network is as follows. The flip-flop signals the modulator driver to send either a 0 or a 1. The modulator driver, which consists of a series of inverter stages, drives the modulator's capacitive load. The modulator couples light at its pre-tuned wavelength λ_i from the optical power source and encodes either a 0 or 1 onto the data waveguide. The optically-encoded data signal traverses the waveguide at approximately one-third the speed of light and is detected by a filter that is also tuned to wavelength λ_i. Photons are detected by the photodetector and received by a flip-flop on the receiver side. Note that Figure 2 shows where a TIA would be needed to amplify the photodetector output, even though it would not be necessary for an ATAC chip targeting the 16nm technology node.

3. ARCHITECTURE OVERVIEW

As previously illustrated in Figure 1, the ATAC processor uses a tiled multicore architecture combining the best of current scalable electrical interconnects with cutting-edge on-chip optical communication networks. The ATAC architecture is targeted at 1000-core chips implemented in a 16nm process. However, it can also be scaled down to smaller chips. In this paper we describe and evaluate 64- and 1024-core versions. We first review the baseline electrical architecture, and then describe how it is augmented with the optical interconnect.

The underlying electrical architecture consists of a 2-D array of processing cores connected by a conventional point-to-point, packet-switched mesh network (called the *EMesh*) like those seen in other multicore processors [20, 12, 11]. Each core in ATAC contains a single- or dual-issue, in-order RISC pipeline with data and instruction caches (Figure 1(c)). ATAC uses a novel directory-based cache coherence scheme which is described later. A portion of the distributed cache-coherence directory is also located in each core.

To this electrical baseline, we add a global optical interconnect—the *ANet*—based on state-of-the-art optical technology. Whereas the EMesh is ideal for predictable, short-range point-to-point communication, the ANet provides low-latency, energy-efficient, contention-free global and long-distance communication. The key component of the ANet is the all-optical ONet shown in Figure 1(a). In the 1024-core ATAC architecture (called ANet^{1024}), cores are grouped into 64 "clusters", each containing 16 cores. Each cluster contains a single ONet endpoint called a *Hub*. The Hub is responsible for interfacing between the optical components of the ONet and the electrical components within a cluster. The ATAC architecture can be scaled down by reducing the number of cores with each cluster. A 64-core chip (called ANet^{64}) would connect each core directly to a Hub.

In ANet^{1024}, individual cores are connected to the Hub in two ways: data going from a core to the hub uses the standard EMesh (described above); data going from the Hub to the cores uses the BNet, a small electrical broadcast network (Figure 1(b)). In the 22nm node, the clusters are small enough that a flit can travel from

the Hub to all cores in a cluster within one clock cycle. Because the BNet is dedicated to broadcasts, it is essentially a fanout tree and requires no routers, crossbars, or internal buffering. It requires only a small amount of buffering and arbitration at the Hub and receiving buffers at the leaves. As a result, we estimate that a BNet requires less than one-eigth the area of a full EMesh of the same bitwidth.

The ANet^{1024} uses a 128-bit wide ONet (128 optical waveguides for data); one 128-bit wide electrical EMesh; and two parallel 128-bit wide BNets. The Hub arbitrates between the first and second BNet using a static policy: packets sent from clusters with even number IDs on the first BNet and odd number IDs on the second BNet. Together, the ONet, EMesh and BNet form the complete ANet^{1024}.

3.1 ONet Optical Network

The key to efficient global communication in a large ATAC chip is the optical ONet. The ONet provides a low-latency, contention-free connection between a set of optical endpoints called Hubs. Hubs are interconnected via waveguides that visit every Hub and loop around on themselves to form continuous rings (Figure 1(a)). Each Hub can place data onto the waveguides using an optical modulator and receive data from the other Hubs using optical filters and photodetectors. Because the data waveguides form a loop, a signal sent from any Hub will quickly reach all of the other Hubs. Thus every transmission on the ONet has the potential to be a fast, efficient broadcast.

The ONet uses wavelength division multiplexing (WDM). Each Hub has modulators tuned to a unique wavelength to use when sending and contains filters that allow it to receive signals on all of the other wavelengths. This eliminates contention and the need for arbitration in the optical network. Taken together, these features mean that the ONet is functionally similar to a broadcast bus, but without any bus contention.

WDM is a key differentiator of the ATAC architecture from a performance scalability perspective. WDM allows a single waveguide to simultaneously carry bits of many overlapping communications. To contrast, an electrical wire typically carries a single bit. Whereas ATAC may share a single waveguide medium between a large number of simultaneous communication channels, implementing multiple simultaneous communication channels in the electrical domain requires additional physical wires. For network operations that are expensive to implement in the electrical domain (such as broadcast), the ATAC approach greatly improves efficiency.

The broadcast mechanism of the ATAC architecture is another key differentiator. Optical technology provides a way to build fast, efficient broadcast networks whereas electrical mechanisms do not. When using optical components instead of electrical components, signals may travel farther and be tapped into by more receivers before they need be regenerated. With electrical components, regeneration is accomplished via buffers or sizing-up of transistors for increased drive strength. When these electrical mechanisms are extensively employed, as they would be in a large electrical broadcast network, it leads to high energy consumption and poor scaling.

Besides broadcasts, optical technology also allows efficient long-distance point-to-point communication. Initiating an optical signal (*i.e.,* switching the modulator) requires more energy than switching a short wire electrically. However, once generated, the optical signal can quickly travel anywhere on the chip without the need for repeaters. According to our estimates of future optical technology, the on-chip energy required to send data on the ANet is approximately 300 fJ/bit and the signal could be received by all Hubs within 3ns. An electrical signal, on the other hand, would require approximately 94 fJ/bit/mm and about 1 ns per hop in a mesh network. Therefore we estimate that it will be more efficient to send an electrical signal if the destination is less than four hops away and an optical signal otherwise.

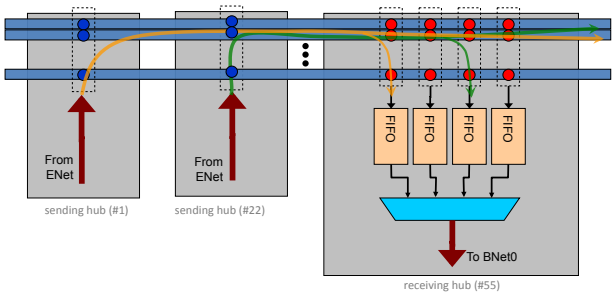

Figure 3: Hub-to-hub communication over the ONet

The ATAC architecture was carefully designed taking into account the physical limitations and constraints of both the optical (see Section 2) and electronic devices. Based on these constraints, the ONet as described above should scale to at least 64 (and possibly as many as 100) Hubs. This limit is based on several factors: 1) the total range of wavelengths over which the optical devices can be tuned divided by the minimum spacing between wavelengths, 2) the total amount of optical power a waveguide can carry divided by the minimum amount that each photodetector needs to receive to reliably register a signal, and 3) the maximum length of a waveguide based on the acceptable propagation losses.

These limits can be overcome using multiple waveguides and dividing the communication channels between them. However, eventually the area needed for the optical components will become the limiting factor. The ONet's optical components and photonic interconnect can be placed on a separate layer in the CMOS stack, and can therefore overlap the electrical components to which they connect. However, for a 400 mm^2 chip, the entire area would be consumed by an ONet with approximately 384 Hubs. Since we believe that chips will eventually grow to thousands of cores, some sharing of Hubs will certainly be needed. Therefore, for the purposes of this paper, we take the simple approach and assume that the ONet is limited to 64 Hubs.

Sending data using the ONet is shown in more detail in Figure 3. To provide adequate on-chip bandwidth, the ONet uses a bundle of waveguides, each containing 64 wavelengths. Multiple waveguides are used to transmit multiple bits of a word simultaneously. As mentioned previously, wavelengths are unique to each sender. This allows the two Hubs shown to send their data simultaneously without interference. The receiving Hub captures both of the values simultaneously into sender-Hub-specific FIFOs. These values are then propagated to the cores using the BNet. The ONet contains 128 waveguides for data, one for backwards flow control, and several for metadata. The metadata waveguides are used to indicate a message type (*e.g.,* memory read, barrier, raw data) or a message tag (for disambiguating multiple messages from the same sender).

3.2 Cache Subsystem

The data caches across all cores on the ATAC chip are kept coherent using a variant of a directory-based MOESI coherence protocol [23] termed *ACKwise* and described in more detail in Section 4. The directory is distributed evenly across the cores. Furthermore, each core is the "home" for a set of addresses (the allocation policy of addresses to homes is statically defined).

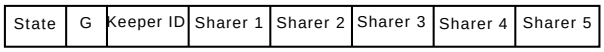

Figure 4: Structure of a directory entry

3.3 External Memory Subsystem

When cores need to communicate with external memory, they do so via several on-chip memory controllers. Each cluster has one core replaced by a memory controller. After receiving requests through the ANet, the memory controller communicates with external DRAM modules through I/O pins. Replies are then sent back to the processing cores through the ANet. By varying the number of cores replaced by memory controllers, different ATAC chips with different ratios of compute power to memory bandwidth can be produced.

The primary task of the memory controller is to translate requests from the processing cores into transactions on a memory I/O bus. The choice of I/O bus technology is independent of the on-chip network architecture since the memory controller is performing a translation. However, to support the large number of memory controllers needed for a 1000-core chip, we assume that the connection to memory is optical as well.

A detailed design for an optical memory subsystem is left to future work. However, we can assume that an optical memory bus would consist of some number of on-chip waveguides that are coupled to external fibers, effectively creating optical "pins." Each optical pin could carry up to 64 wavelengths of light at speeds of up to 20 GHz. The actual transmission speed would likely be limited by design trade-offs in the electrical circuits driving the optical components. We estimate that optical I/O pins operating at 5 GHz (yielding 40 GB/s of bandwidth) should be practical. Thus each off-chip memory bus can be implemented using a single optical pin. This makes it practical to integrate the large number of memory controllers needed to meet the bandwidth needs of future 1000-core chips.

4. CACHE COHERENCE PROTOCOL

This section presents *ACKwise*, a cache coherence protocol derived from a MOESI-directory based protocol [23]. Each directory entry in this protocol, as shown in Figure 4 is similar to one used in a limited directory scheme [2] but with a few modifications. The 4 fields in each directory entry are as follows: (1) **State**: This field specifies the state of the cached block(s) associated with this directory entry (one of the MOESI states); (2) **Global(G)**: This field states whether the number of sharers for this data block exceeds the capacity of the sharer list. If so, a broadcast is needed to invalidate all the cached blocks corresponding to this address when a cache demands exclusive ownership of this data block; (3) **KeeperID**: This field holds the ID of a core which contains an up-to-date copy of this data block. This copy could be clean (as in Exclusive(E) and Shared(S) state) or could be dirty (as in Modified(M) and Owned(O) state); and (4) $\textbf{Sharers}_{1-5}$: This field represents the sharer list. It can hold upto 5 core IDs.

When the number of sharers exceeds 6 (including the keeper), the **Global(G)** bit is set so that any number of sharers beyond this point can be accommodated. Once the global (G) bit is set, the sharer list ($Sharers_{1-5}$) holds the total number of sharers of this data block. In other words, once the global(G) bit is set, the directory has only the following information about a data block: (a) KeeperID; and (b) Number of sharers.

4.1 Operation of the ACKwise Protocol

When a request for a shared copy of a data block is issued, the directory controller first checks the state of the data block in the directory cache. (a) If the state is *Invalid(I)*, it forwards the request to the memory controller. The memory controller fetches the data block from memory and sends it directly to the requester. It also sends an acknowledgement to the directory. The directory changes the state of the data block to *Exclusive(E)* and sets the *KeeperID* field to the ID of the requester. (b) If the state is one of the valid states (*i.e.,* one of *MOES*), it forwards the request to the *Keeper*. The *Keeper* forwards the data block directly to the requester and sends an acknowledgement to the directory. Appropriate state changes happen in the cache block of the *Keeper* and the directory according to the rules of the *MOESI* protocol [23]. The directory controller also tries to add the ID of the requester to the sharer list. This is straightforward if the *global(G)* bit is clear and the sharer list has vacant spots. If *global(G)* bit is clear but the sharer list is full, it sets the *global(G)* bit and stores the total number of sharers (in this case, 7 (= 6+1)) in the sharer list. If the *global(G)* bit is already set, then it increments the number of sharers by one.

When a request for an exclusive copy of a data block is issued, the directory controller first checks the state of the data block in the directory cache. (a) If the state is *Invalid(I)*, the sequence of actions followed is the same as that above except that the state of the data block in the directory is set to *Modified(M)* instead of *Exclusive(E)*. (b) If the state is one of the valid states (*i.e.,* one of *MOES*), then the directory controller performs the following 2 actions: (i) It forwards the request to the *Keeper*. (ii) If the global bit is clear, it sends unicast invalidation messages to each core in the sharer list. Else, if the global bit is set, it broadcasts an invalidation message (to all the cores on the chip). Now, the *Keeper*, on receiving the forwarded request sends the data block directly to the requester, acknowledges the directory and invalidates its cache block. The other sharers invalidate their cache blocks and acknowledge the directory. The directory controller expects as many acknowledgements as the number of sharers (encoded in the sharer list if the *global(G)* bit is set and calculated directly if the *global(G)* bit is clear). After all the acknowledgements are received, the directory controller sets the state of the data block to *Modified(M)*, the *global(G)* bit to 0 and the *KeeperID* field to the ID of the requester. Due to the broadcast capabilities of ATAC as described in section 3, the sending of broadcast messages can be achieved easily.

Silent evictions are not supported in the ACKwise protocol since the directory should always have an accurate count of the number of sharers of a cache line for correct operation. However, disallowing silent evictions is not found to be detrimental to the performance of the ACKwise protocol due to the following two reasons: (1) the additional coherence messages do not lie on the critical path of load or store misses and hence do not directly affect the average memory latency and thereby processor performance; (2) these messages do not include data and hence contribute only a small percentage to the overall network traffic and thereby do not really affect the network latency.

5. EVALUATION

The purpose of this section is to demonstrate: (1) The capabilities of the ATAC network (*ANet*) over a pure electrical mesh network (*EMesh*), and (2) The performance advantages of using the $ACKwise_k$ protocol over limited directory based cache coherence protocols denoted by Dir_kB and Dir_kNB according to the nomenclature in [2]. Dir_kB is a limited directory based protocol which broadcasts once the capacity of the sharer list is exceeded and collects acknowledgements from all the cores in the system. $ACKwise_k$ on the other hand, intelligently tracks the number of

sharers once the capacity of the sharer list is exceeded and needs acknowledgements from only the actual sharers of the data on a broadcasted invalidation. *Dir$_k$NB* always ensures that the number of sharers of a cache line is less than the capacity of the sharer list. k denotes the number of hardware sharers in all the above three coherence protocols. In this section, the performance of Splash2 benchmarks as well as synthetic applications are evaluated on 64 and 1024 cores using the following six combinations of on-chip networks and cache coherence protocols: (a) *ANet-ACKwise$_k$*, (b) *ANet-Dir$_k$B*, (c) *ANet-Dir$_k$NB*, (d) *EMesh-ACKwise$_k$*, (e) *EMesh-Dir$_k$B* and (f) *EMesh-Dir$_k$NB*. Results demonstrate the advantages of using *ANet* over *EMesh* due to its higher bandwidth, lower latency and broadcast capabilities as well as the performance benefits of the *ACKwise$_k$* protocol over the *Dir$_k$B* and *Dir$_k$NB* protocols.

5.1 Methodology

The Graphite [16] distributed multicore simulator is used for all evaluations done in this section. Applications from the Splash2 benchmark suite as well as synthetic benchmarks are used for the experiments conducted. Splash2 benchmarks are used for 64 core simulations while synthetic benchmarks are used for 64 and 1024 core evaluations.

For 64 core simulations, the ANet^{64} network described in Table 1 is compared to a 64-bit wide electrical mesh network. Since optical components can be placed on a separate layer, the cost of adding an optical network is negligible compared to doubling the width of an electrical mesh network. In ANet^{64}, short unicast messages (less than four hops away) are sent on the EMesh (due to energy considerations as described in Section 3) while broadcasts and long unicast messages are sent on the ONet. For 1024 core simulations, the ANet^{1024} network described in Section 3 is compared to a 256-bit wide electrical mesh network. The routing protocol of ANet^{1024} is similar to that of ANet^{64} . The detailed target system architecture studied is shown in Table 1. Small private L2 cache sizes were assumed due to the small working set sizes of Splash2 benchmarks. All the references to EMesh in the remaining part of the evaluation section refer to the pure electrical mesh network against which the ANet^{64} or ANet^{1024} networks are compared.

5.2 Splash2 benchmarks

Ten applications in the Splash2 benchmark suite are simulated on 64 cores using the 6 combinations of cache coherence protocols and networks mentioned previously. The two configurations *ANet-Dir$_{64}$NB* and *ANet-Dir$_{64}$B* are expected to show the same performance as *ANet-ACKwise$_{64}$* since the directory type of the cache coherence protocol does not play a role when the number of hardware sharers is equal to the number of cores simulated. Similarly, the performance of *EMesh-ACKwise$_{64}$*, *EMesh-Dir$_{64}$NB* and *EMesh-Dir$_{64}$B* are expected to be the same. Here, ANet refers to ANet^{64} described in Section 5.

The ACKwise protocol running on the ANet network is observed to show nearly the same performance irrespective of the number of hardware sharers on all the ten Splash2 benchmarks used for simulation, i.e, the runtime of the application when using ANet-ACKwise$_2$ differs from that when using ANet-ACKwise$_{64}$ by $<$ 1%. For the ACKwise protocol running on EMesh, the performance when using 2 hardware sharers differs from that when using 64 sharers for only one out of the ten benchmarks (namely `ocean_contig`). For all the other nine benchmarks, the performance difference is $<$ 2%. The reason for this is discussed later in this section when interpreting the performance numbers obtained with the Dir$_k$B protocol. On an average, EMesh-ACKwise$_k$ encounters a slowdown of 17% when compared to ANet-ACKwise$_k$.

In Figures 5 & 6, runtime of Splash2 benchmarks observed when running with the Dir$_k$NB cache coherence protocol on the two networks ANet and EMesh is plotted as a function of the number of hardware sharers. Results are normalized against the runtime observed when using the ANet-ACKwise$_{64}$ combination. The performance is extremely poor at low numbers of hardware sharers and gradually improves as the number of sharers is increased. With two hardware sharers, the average slowdown of the Dir$_k$NB protocol is observed to be 1.5x with the ANet network and 2x with the EMesh network. The max slowdown is as high as 2.2x with the ANet network and 2.7x with the EMesh network. When the number of hardware sharers is 64, the performance on ANet equals that of ANet-ACKwise$_{64}$ as expected whereas an average slowdown of 15% is observed on EMesh. These results clearly indicate that ANet-ACKwise$_k$ clearly outperforms both ANet-Dir$_k$NB and EMesh-Dir$_k$NB.

Figure 7 shows the cache miss rates of the benchmarks when run with the Dir$_k$NB protocol on the ANet network. The cache miss rate of the benchmarks when run with the ACKwise$_k$ protocol is observed to be almost independent of the number of hardware sharers (k). The numbers clearly illustrate that the slowdown observed with the Dir$_k$NB protocol is due to the occurrence of frequent invalidations caused by large numbers of cores trying to simultaneously read some shared data and evicting each other in the process due to the presence of limited number of hardware sharers. The ACKwise$_k$ protocol, on the other hand does not limit the number of cores that can simultaneously read a cache line, and hence encounters a much lower miss rate than the Dir$_k$NB protocol. The above results indicate the presence of a large amount of frequently read and sparsely written data in the ten Splash2 benchmarks studied which is corroborated in [26].

Figure 8 shows the runtime of six Splash2 benchmarks when run using the Dir$_k$B protocol with the ANet and EMesh networks. The results here are also normalized to that of ANet-ACKwise$_{64}$. The other four Splash2 benchmarks also show the same behavior. The performance of Dir$_k$B closely resembles the performance obtained with ACKwise$_k$ on both the ANet and EMesh networks except for one benchmark, `ocean_contig`. For all other benchmarks, the runtime obtained with the Dir$_k$B protocol lies within 5% of the runtime obtained with the ACKwise$_k$ protocol. The `ocean_contig` benchmark was also the benchmark that demonstrated speedup with the EMesh-ACKwise$_k$ combination when the number of sharers was increased from 2 to 3. The reason for this increase in performance (10% for the Dir$_k$B protocol and 5% for the ACKwise$_k$ protocol) is due to the significant number of broadcast invalidations observed with `ocean_contig` when the number of sharers is 2. For all other benchmarks, the percentage of invalidation broadcasts is $<$ 1%. The above observations can be made when looking at Figure 9. The difference between the ACKwise$_k$ and Dir$_k$B protocols is that ACKwise intelligently tracks the number of sharers of a cache line once the capacity of the sharer list is exceeded and hence, on a subsequent exclusive request for that cache line, it only needs to get acknowledgements from only the actual sharers of the cache line and not from all the cores in the system as in Dir$_k$B. Since the percentage of invalidations broadcasts is low, the number of exclusive results to a cache line when it is in Global(G) mode (i.e, assuming that all cores are sharers in the Dir$_k$B protocol) is also very low. Hence, the unicast messages generated as acknowledgements to invalidation broadcasts by the Dir$_k$B protocol do not raise congestion levels of the on-chip network by a significant amount to affect the overall system throughput.

The results of the above experiments clearly indicate the Splash2

Core Model		In-order core, 32 KB private L2 Cache	
1024 cores		**64 cores**	
$ANet^{1024}$ = ONet + EMesh + 64x2 BNets		$ANet^{64}$ = ONet + EMesh	
ONet	128-bit wide	ONet	64-bit wide
EMesh	128-bit wide	EMesh	64-bit wide
BNet	128-bit wide		
EMesh for comparison	256-bit wide	EMesh for comparison	64-bit wide
Memory Bandwidth	64 memory controllers, 5 GBps per controller	Memory Bandwidth	4 memory controllers, 5 GBps per controller

Table 1: Target System Architecture Configuration Parameters

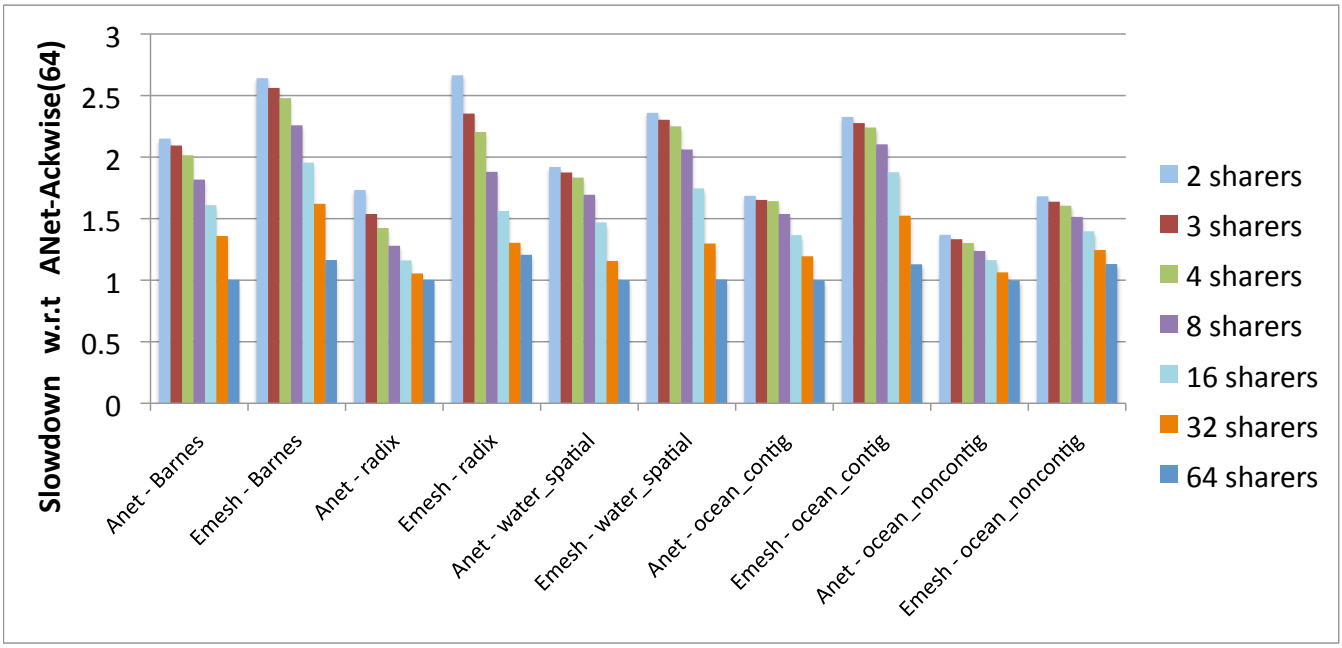

Figure 5: Runtime of five Splash2 benchmarks when using the Dir_kNB protocol with the ANet and EMesh networks. Results are normalized to the runtime of ANet-$ACKwise_{64}$. The number of hardware sharers are varied as 2, 3, 4, 8, 16, 32 and 64

benchmarks contain a lot of frequently read and sparsely written shared data. Overall, it is observed that the ANet-$ACKwise_k$ and ANet-Dir_kB protocols exhibit the best performance on Splash2 benchmarks.

5.3 Synthetic benchmarks

The Splash2 benchmarks are highly structured applications that exhibit extremely good cache behavior as observed by the experiments conducted in the previous section. They are not representative of future multicore workloads that widely share data and exhibit highly unstructed access to them. In this section, we evaluate the performance of a synthetic benchmark that emulates different types of workloads when run with the six combinations of cache coherence protocols and networks mentioned previously. Experiments are done on 64 and 1024 cores.

The characteristics of the synthetic benchmark used are shown in Table 2. The benchmark is constructed by assigning probabilities to instructions and memory access types. Data accessed by this synthetic benchmarks is divided into three types: (a) private data, (b) shared data that can only be read (read-only shared data) and (c) shared data that can both be read and written (read-write shared data). Among the instructions that access private data and read-write shared data, the fraction of reads to the fraction of writes is assumed to be 2:1 (This is a valid assumption because most workloads read data from two memory locations, do some computation on it and store the result in a third memory location). The only variables in the synthetic benchmark are the fraction of instructions that access read-only shared data and the degree of sharing of the shared data. For read-only shared data, a sharing-degree d denotes that this data can be read by a total of d sharers and for read-write shared data, degree d denotes that this data can be read/written by a total of d sharers. The amount of private data each core can access is 16KB and the total amount of shared data is 64 KB and 1 MB for 64-core and 1024-core simulations respectively.

5.3.1 64 Cores

The network architectures $ANet^{64}$ and *EMesh* studied are the same as discussed in the previous section. The cache coherence protocols studied are $ACKwise_k$, Dir_kB and Dir_kNB. In this evaluation, k is fixed at 4 (k is the number of hardware sharers).

The percentage of instructions that access read-only shared data among those that access shared data is fixed to be 25% and 75% in the experiments conducted below. The number of application sharers are varied from 1 to 64 as powers of 2 in all the experiments conducted in this section. The following paragraphs explain the results obtained.

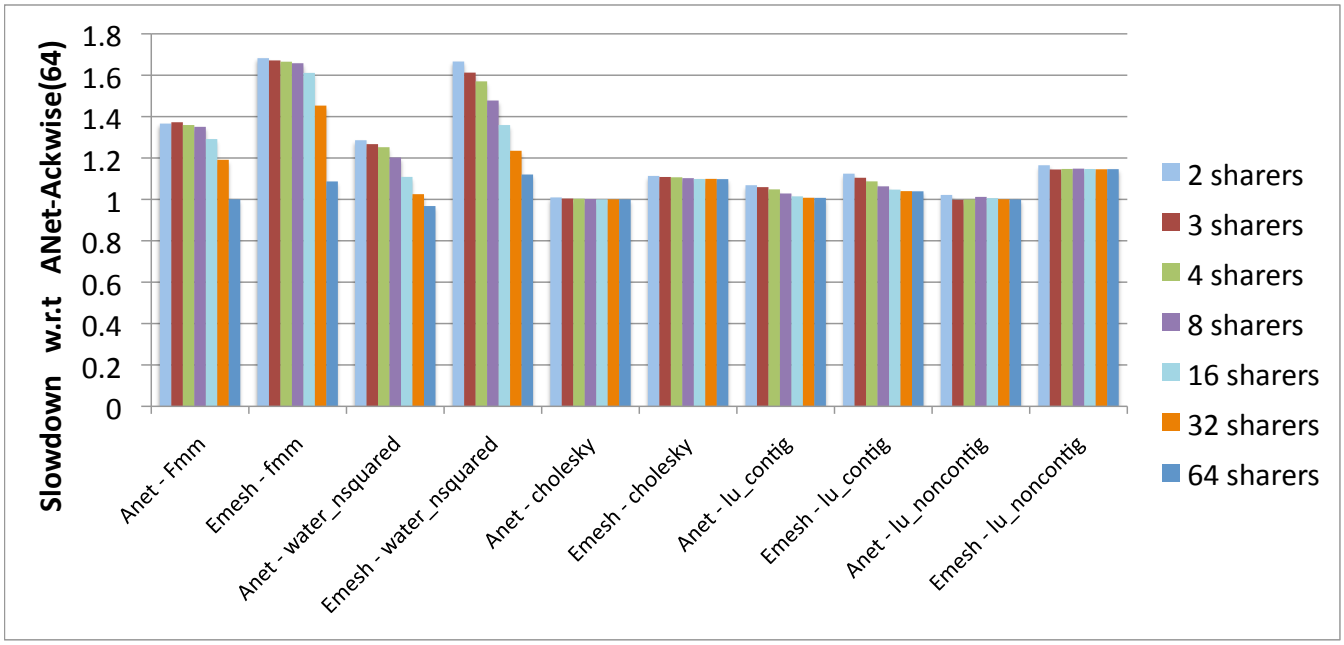

Figure 6: Runtime of five Splash2 benchmarks when using the Dir$_k$NB protocol with the ANet and EMesh networks. Results are normalized to the runtime of ANet-ACKwise$_{64}$. The number of hardware sharers are varied as 2, 3, 4, 8, 16, 32 and 64

.

Percentage of Non Memory Instructions	70%
Percentage of Instructions Accessing Shared Data	10%
Percentage of Instructions Accessing Private Data	20%
Percentage of Read-only Data in Shared Data	{25%, 75%}
Private Data per Thread	16 KB
Total Shared Data	{64 KB, 1 MB}
Degree of Application Sharing	{1, 2, 4, 8, 16, 32, 64}
Number of Instructions Simulated per Thread	{1 million, 100,000}

Table 2: Synthetic Benchmark Characteristics

25% Read-Only, 75% Read-Write: From Figure 10(a), it can be observed that the ACKwise$_4$ protocol performs best on ANet and the Dir$_4$NB protocol performs best on EMesh. The ACKwise$_4$ and Dir$_4$B protocols perform poorly on EMesh. The performance worsens as the degree of application sharing increases. This is because an increase in the degree of sharing increases the number of broadcast invalidations and a pure electrical mesh performs poorly with a lot of broadcast traffic. The Dir$_4$NB protocol, on the other hand does not produce any broadcast traffic. Moreover, the performance penalty of evicting a sharer in order to accommodate another sharer is small for 75% of the data because exclusive requests arrive frequently for cache lines in that data space.

The ANet network on the other hand supports broadcast traffic efficiently and hence ACKwise$_4$ has the best performance. The Dir$_4$B protocol still suffers due to the many unicast acknowledgements that have to be sent as a result of a broadcasted invalidation. The Dir$_4$NB protocol on ANet is found to perform slightly worser than ACKwise$_4$.

75% Read-Only, 25% Read-Write: From Figure 10(b), it can be observed that the ACKwise$_4$ protocol performs best on the both ANet and EMesh. With 75% read-only shared data, the Dir$_4$NB protocol performs poorly on both networks because all sharers of a read-only shared cache line cannot have the data in their private caches at the same time. Hence, the cores accessing read-only shared data keep invalidating each other frequently. The performance of the Dir$_4$B protocol lies between that of ACKwise$_4$ and Dir$_4$NB protocol. Even though the Dir$_4$B protocol achieves the same performance as ACKwise$_4$ on read-only shared data, it still suffers when there are a sufficient number of broadcast invalidation requests because it has to collect acknowledgements from all the cores for each broadcasted invalidation. This configuration produces results extremely similar to those produced by the Splash2 benchmarks.

5.3.2 1024 Cores

The network architectures ANet^{1024} and EMesh studied are as discussed at the start of the evaluation section. Recall that ANet^{1024} uses one 128-bit optical waveguide for the ONet; one 128-bit wide electrical mesh and two parallel 128-bit wide BNets per cluster. The pure electrical mesh network EMesh to which we compare ANet^{1024} is 256-bit wide. This comparison is justified because optical components can be placed on a separate layer and the area of a 128-bit wide BNet is roughly one-eigth the area of a 128-bit wide electrical mesh (see Section 3). All the evaluations in this section are done using the synthetic benchmark that has 25% read-only data out of the total shared data. For the synthetic benchmark that has 75% read-only shared data, it is fairly obvious from Section 5.3.1 that ANet-ACKwise will have the best performance.

From Figure 11, it can be easily observed that the ACKwise$_4$ protocol coupled with the ANet network provides the best results. The Dir$_4$B protocol performs extremely poorly on ANet due to its lack of network bandwidth for the large number of unicast acknowledgements generated by the protocol. This fact is corroborated by the extremely large queueing delays observed at the send-

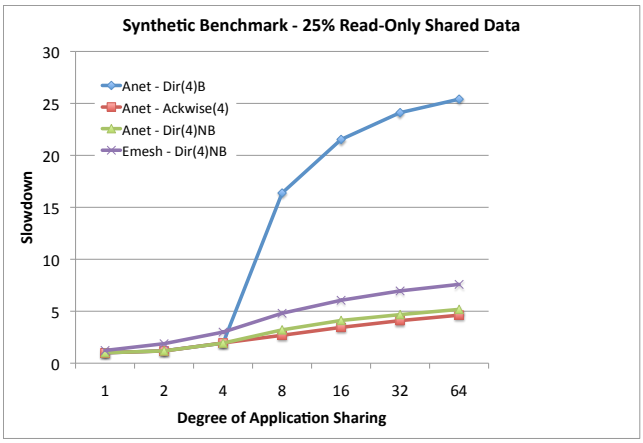

Figure 11: Completion time of the synthetic benchmark running on 1024 cores with 4 different combinations of networks and cache coherence protocols. The x-axis values denote the degree of application sharing. The completion times are normalized to that of ANet-ACKwise$_4$ with an application sharing degree of 1. Smaller values on the y-axis indicate higher performance. Only Dir$_4$NB coherence protocol is evaluated on the electrical mesh baseline(EMesh). ACKwise$_4$ and Dir$_4$B perform poorly on a pure electrical mesh with this synthetic benchmark as illustrated in Section 5.3.1

.

ing Hub with the Dir$_4$B protocol. The gap between the performance of the ACKwise$_4$ and the Dir$_4$NB protocol widens as the percentage of read-only shared data increases as demonstrated in the earlier section. Overall, ANet-ACKwise outperforms the best cache coherence protocol on EMesh (Dir$_4$NB in this case) by 50%. ACKwise$_4$ and Dir$_4$B protocols perform poorly on a pure electrical mesh with this synthetic benchmark as illustrated in Section 5.3.1.

6. RELATED WORK

CMOS-compatible nanophotonic devices are an emerging technology. Therefore there have only been a few architectures proposed that use them for on-chip communication: Corona [10], the optical cache-coherence bus of Kirman et al [22], and the switched optical NoC of Shacham et al [4].

The Corona architecture primarily differs from ATAC in the way that it assigns communication channels. While Corona assigns a physical channel to each receiver and uses WDM to send multiple bits of a dataword simultaneously, ATAC assigns a physical channel to each sender and uses WDM to carry multiple channels in each waveguide, thereby eliminating contention and the need for arbitration.

Kirman et al [22] design a cache-coherent hierarchical opto-electronic bus, consisting of a top-level optical broadcast bus which feeds into small electrical networks connecting groups of cores. The design of their network is similar to ATAC but is limited to snooping cache coherence traffic whereas ATAC is composed of a network supporting a general communication mechanism and a coherence protocol (i.e, *ACKwise*) designed to scale to hundreds of cores.

Shacham et al [4] propose a novel hybrid architecture in which they combine a photonic mesh network with electronic control packets. Their scheme is still partially limited by the properties of electrical signal propagation since they use an electronic control network to setup photonic switches in advance of the optical signal transmission. It only becomes efficient when a very large optical payload follows the electrical packet. ATAC, on the other hand, leverages the efficiencies of optical transmission for even a single word packet.

Pan et al. [24] proposed Firefly, a hybrid electrical-optical network architecture. Similar to ATAC, Firefly breaks the chip into clusters of cores which are interconnected using electrical links. Clusters communicate via a single-writer multiple-reader optical network. Unlike ATAC, Firefly's photonic links use an optical crossbar which must be configured by a handshake between the sender and receiver. Firefly partitions its crossbar into multiple smaller logical crossbars to eliminate the need for global arbitration.

Batten et al. [6] take a different approach and use integrated photonics to build a high-performance network that connects cores directly to external DRAM. However, their design does not allow for optical core-to-core communication. An ATAC processor could leverage their design to connect its memory controllers to DRAM.

Previous techniques for reducing directory storage space in cache coherence protocols include using hierarchical directories [28], coarse vectors [3], sparse directories [3], chained directories [8], and maintaining limited directories with broadcasting capabilities [2] or software support [9]. The *ACKwise* protocol, on the other hand, augments a limited directory based protocol by tracking the number of sharers once the capacity of the sharer list is exceeded. It also borrows the strategy of maintaining a clean owner for reducing the offchip miss rate from *cooperative caching* [13]. Recent proposals for a cache organization combining the low hit latency of a private L2 cache and the low miss rate of a shared L2 cache [13, 21] are orthogonal to *ACKwise* and could be used along it.

7. CONCLUSION

The recent advances of optical technology have certainly inspired confidence in computer architects that optics may very well continue to make its way into smaller and smaller packages; just as optical interconnect has moved from connecting cities to connecting data centers, it seems likely that it will soon connect chips and on-chip components.

Overall, this paper presented a novel manycore architecture that scales to 1000 cores by embracing new technology offered by recent advances in nanophotonics. This paper also introduced *ACKwise*, a novel directory-based cache coherence protocol that takes advantage of the special properties of the ATAC network to achieve high performance. The paper provides significant insight into the performance implications of using different directory based cache coherence protocols with various memory access patterns on on-chip networks. From the evaluations of the Splash2 and synthetic benchmarks, it is observed that using the ACKwise protocol on the ANet network outperforms all other combinations of networks and cache coherence protocols on both 64-core and 1024-core architectures. On average, ANet-ACKwise$_k$ outperforms EMesh-ACKwise$_k$ by 17%, ANet-Dir$_k$NB by 1.5x and EMesh-Dir$_k$NB by 2x on 64-core Splash2 benchmarks. With synthetic benchmarks on 1024 cores, ANet-ACKwise$_k$ outperforms the best cache coherence protocols on an electrical mesh by 50%.

Acknowledgement

This work was partially funded by the National Science Foundation under Grant No. 0811724.

8. ADDITIONAL AUTHORS

9. REFERENCES

[1] The International Technology Roadmap for Semiconductors (ITRS) Technology Working Groups, 2008. http://public.itrs.net.

[2] A. Agarwal et al. An evaluation of directory schemes for cache coherence. In *ISCA*, 1988.

[3] A. Gupta et al. Reducing Memory and Traffic Requirements for Scalable Directory-Based Cache Coherence Schemes. In *ICPP*, 1990.

[4] A. Shacham et al. Photonic NoC for DMA Communications in Chip Multiprocessors. In *Hot Interconnects*, Aug 2007.

[5] Authors withheld. ATAC: All-to-All Computing Using On-Chip Optical Interconnects. In *BARC*, 1/2007.

[6] C. Batten et al. Building manycore processor-to-dram networks with monolithic silicon photonics. In *Hot Interconnects*, pages 21–30, Aug 2008.

[7] D. Ahn et al. High performance, waveguide integrated Ge photodetectors. In *Optics Express 15, 3916*, 2007.

[8] D. Chaiken et al. Directory-Based Cache Coherence in Large-Scale Multiprocessors. In *IEEE Computer, Vol 23, p 49-58*, June 1990.

[9] D. Chaiken et al. Limitless Directories: A scalable cache coherence scheme. In *ASPLOS*, 1991.

[10] D. Vantrease et al. Corona: System Implications of Emerging Nanophotonic Technology. In *ISCA*, 2008.

[11] D. Wentzlaff et al. On-chip interconnection architecture of the Tile Processor. *IEEE Micro*, 27(5):15–31, 2007.

[12] Intel Corporation. Intel's Teraflops Research Chip. http://techresearch.intel.com/articles/Tera-Scale/1449.htm.

[13] J. Chang et al. Cooperative Caching for Chip Multiprocessors. In *ISCA*, 2006.

[14] J. F. Liu et al. Waveguide-integrated, ultra-low energy GeSi electro-absorption modulators. In *Nature Photonics 2, 433*, 2008.

[15] J. Michel et al. Advances in Fully CMOS Integrated Photonic Circuits. In *Proc. of the International Society for Optical Engineering (SPIE) 6477, p64770P-1-11*, 2007.

[16] J.Miller et al. Graphite: A Distributed Parallel Simulator for Multicores. 2009.

[17] R. Kirchain and L. Kimerling. A roadmap for nanophotonics. In *Nature Photonics, 1 (6): 303-305*, 2007.

[18] J. F. Liu and J. Michel. High Performance Ge Devices for Electronic-Photonic Integrated Circuits. In *ECS Transactions, Vol 16, p 575-582*, 2008.

[19] M. Beals et al. Process flow innovations for photonic device integration in CMOS. In *Proc. of the International Society for Optical Engineering (SPIE) 6898, 689804*, 2008.

[20] M. Taylor et al. Evaluation of the Raw Microprocessor: An Exposed-Wire-Delay Architecture for ILP and Streams. In *ISCA*, 2004.

[21] M. Zhang et al. Victim Replication: Maximizing Capacity while Hiding Wire Delay in Tiled Chip Multiprocessors. In *ISCA*, 2005.

[22] N. Kirman et al. Leveraging Optical Technology in Future Bus-based Chip Multiprocessors. In *MICRO*, 2006.

[23] P. Sweazey et al. A Class of Compatible Cache Consistency Protocols and their Support by the IEEE Futurebus. In *ISCA*, 1986.

[24] Y. Pan, P. Kumar, J. Kim, G. Memik, Y. Zhang, and A. N. Choudhary. Firefly: illuminating future network-on-chip with nanophotonics. In *ISCA*, pages 429–440, 2009.

[25] C. Schow. Optical Interconnects in Next-Generation High-Performance Computers. OIDA 2008 Integration Forum, 2008.

[26] S.Woo et al. The SPLASH-2 Programs: Characterization and Methodological Considerations. 1995.

[27] N. withheld. Improving performance and programmability with on-chip optical networks. In *ISCAS*, 2010.

[28] Y. Maa et al. Two economical directory schemes for large-scale cache coherent multiprocessors. In *ACM SIGARCH Computer Arch News, Vol 19*, Sept 1991.

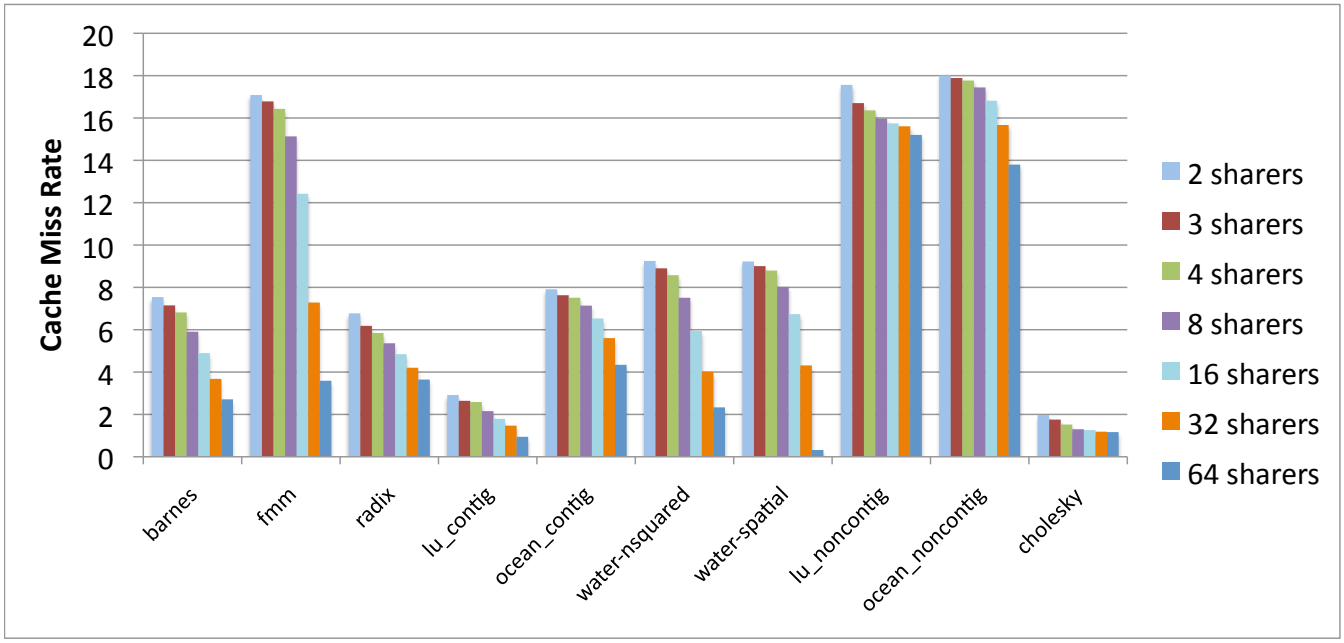

Figure 7: Cache miss rate observed when the Splash2 benchmarks are run with ANet-Dir$_k$NB. The number of hardware sharers are varied as 2, 3, 4, 8, 16, 32 and 64

.

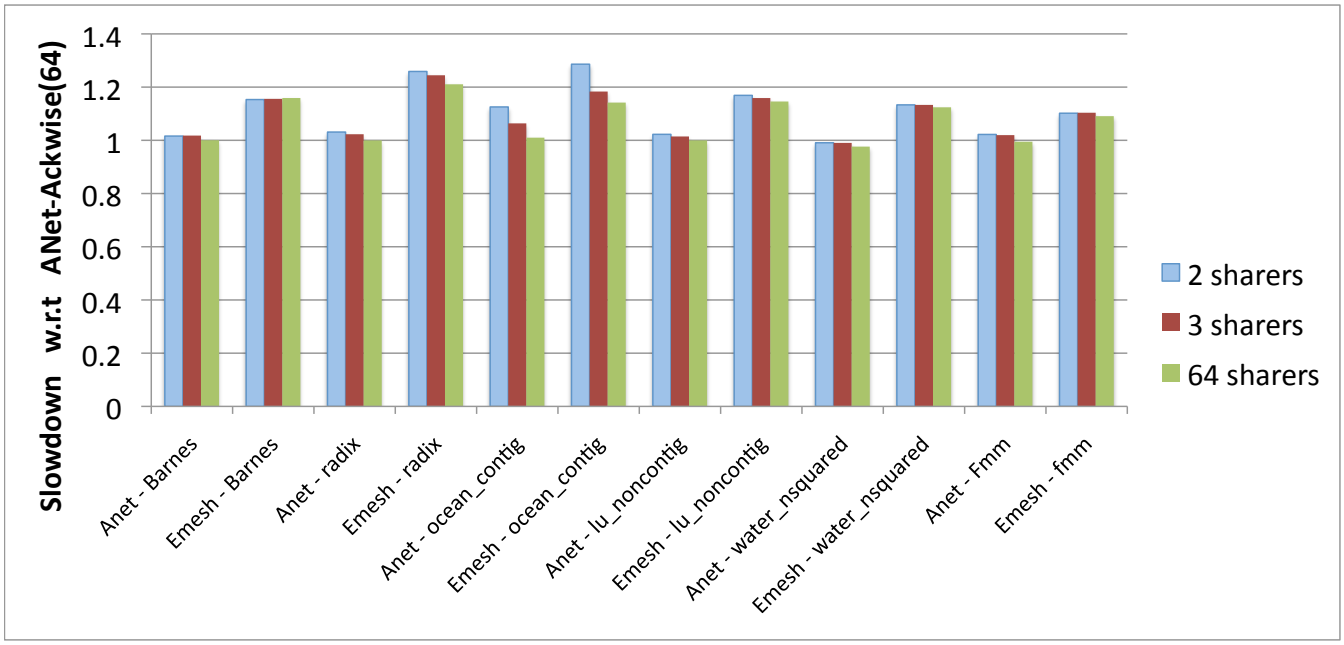

Figure 8: Runtime of six Splash2 benchmarks when using the Dir$_k$B protocol with the ANet and EMesh networks. Results are normalized to the runtime of ANet-ACKwise$_{64}$. The number of hardware sharers are varied as 2, 3 and 64

.

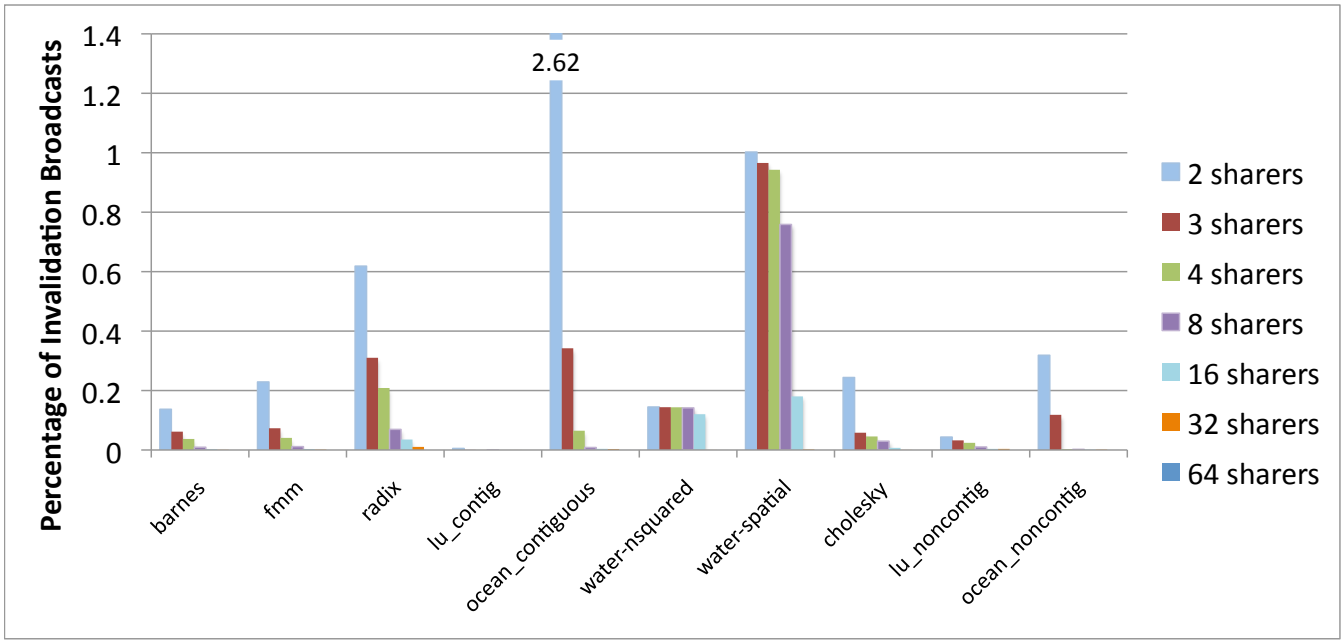

Figure 9: Percentage of invalidation broadcasts generated due to memory requests at the directory of a broadcast enabled cache coherence protocol (ACKwise$_k$ or Dir$_k$B)

.

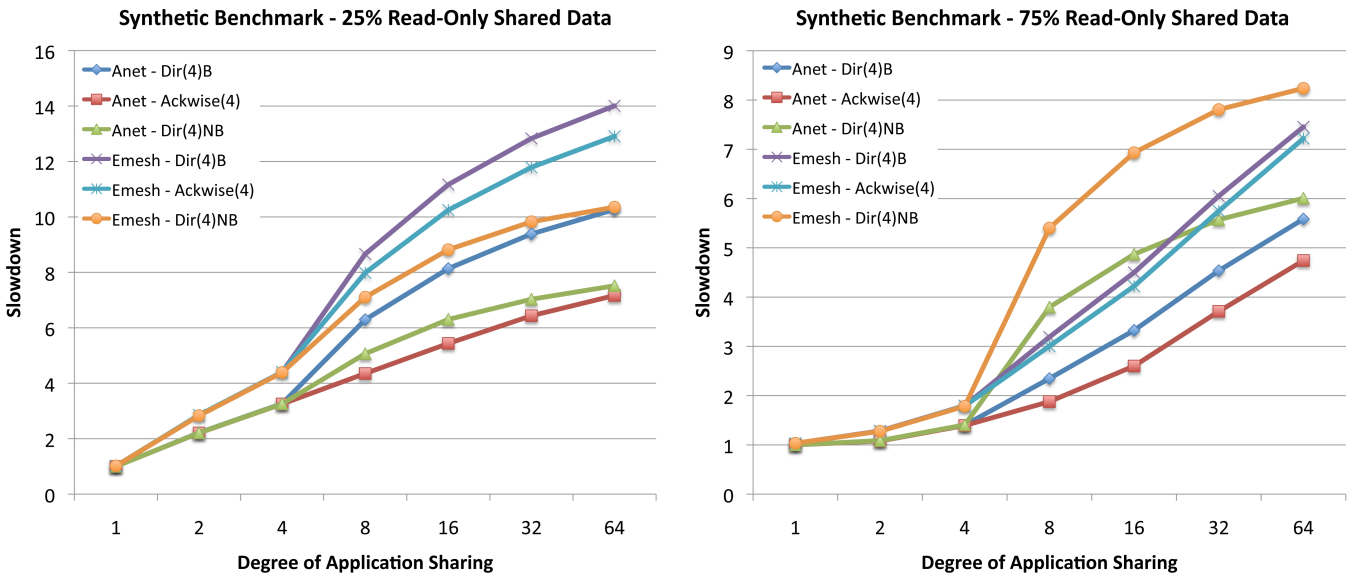

Figure 10: Completion time of the synthetic benchmark running on 64 cores with six different combinations of networks and cache coherence protocols. The x-axis values denote the degree of application sharing. The completion times are normalized to that of ANet-ACKwise$_4$ with an application sharing degree of 1. Smaller values on the y-axis indicate higher performance.