An Efficient Hybrid-Switched Network-on-Chip for Chip Multiprocessors

Pejman Lotfi-Kamran, *Member, IEEE,* Mehdi Modarressi, *Member, IEEE,* and Hamid Sarbazi-Azad

Abstract—Chip multiprocessors (CMPs) require a low-latency interconnect fabric network-on-chip (NoC) to minimize processor stall time on instruction and data accesses that are serviced by the last-level cache (LLC). While packet-switched mesh interconnects sacrifice performance of many-core processors due to NoC-induced delays, existing circuit-switched interconnects do not offer lower network delays as they cannot hide the time it takes to set up a circuit. To address this problem, this work introduces CIMA – a hybrid circuit-switched and packet-switched mesh-based interconnection network that affords low LLC access delays at a small area cost. CIMA uses virtual cut-through (VCT) switching for short request packets, and benefits from circuit switching for longer, delay sensitive response packets. While a request is being served by the LLC, CIMA attempts to set up a circuit for the corresponding response packet. By the time the request packet is served and the response gets ready, a circuit has already been prepared, and as a result, the response packet experiences short delay in the network. A detailed evaluation targeting a 64-core CMP running scale-out workloads reveals that CIMA improves system performance by 21% over the state-of-the-art hybrid circuit-packet-switched network.

Index Terms—Network-on-chip, circuit switching, low latency, chip multiprocessor.

1 Introduction

Modern chip multiprocessors (CMPs) feature many processing cores [1], [2], [3], [4], each with instruction and data caches (i.e., L1-I and L1-D), backed up by a shared last-level cache (LLC). These processors usually consist of several tiles, wherein each tile holds a core with its instruction and data caches, a slice of the shared LLC, and a router. Tiles' routers form a network-on-chip (NoC) that is the communication fabric for inter-tile connectivity in multiprocessors.

As LLC holds the instructions and data working sets, modern workloads spend a considerable amount of their execution time waiting for the LLC [5], [6], [7]. In chip multiprocessors, a noticeable fraction of the LLC access latency is NoC-induced [5], [6] — a request for a piece of data (or an instruction) should be sent to the destined LLC slice and the response should be sent back to the requesting core. Several recent studies [8], [9], [10], [11] showed the importance of a fast NoC for improving performance.

While NoC delays negatively affect performance of chip multiprocessors, wire delays constitute just a small fraction of the total delay [1], as significant delays are due to routing, virtual channel allocation, arbitration, and reading from and writing to buffers [12]. As non-wire delays are clearly undesirable, they must be minimized to the extent possible.

One way to minimize non-wire delays is to let the routers on the path know, in advance, that a packet is coming (i.e., set up a circuit). As routers are aware of the upcoming packets, they reserve the necessary resources for the packets, which consequently leads to faster packet transmission.

While circuit switching can potentially reduce NoC delays, CMPs impose challenges for circuit-switching strategies. First, there is no dominant communication pattern in CMPs as all tiles are equally likely to be the destination of the request packet upon a private cache miss. Second, it is difficult to amortize the overhead of circuit setup time, as the communication is short — a core sends a small request to an LLC slice, and the LLC sends back a relatively small cache block. Several circuit-switched and hybrid circuit-packet-switched NoCs have been proposed, but they either cannot hide circuit setup time [13], [14], [15] or waste bandwidth [16] or increase packet serialization overhead (i.e., number of flits) [17], and consequently are not suitable to be used as the communication fabric of CMPs.

In this work, we introduce a Circuit-switched Inter-tile Memory Access (CIMA) mechanism for chip multiprocessors that takes advantage of CMP behaviors to hide circuit setup time, without bandwidth waste or serialization overhead. CIMA uses packet switching for short request packets and attempts to set up circuits for longer response packets. CIMA is based on the observation that whenever an LLC slice receives a request, if the request turns into a hit in the LLC, there will be a response to the requesting core. Based on this observation, CIMA attempts to set up a circuit for the response packet as soon as the LLC tag lookup indicates a hit.

Last-level caches of most processors benefit from a serial tag and data lookup to reduce energy usage [18], [19]. For such LLCs, the whole data lookup time is available for the circuit setup process. In case an LLC uses a parallel tag and data lookup, data lookup takes longer than the tag lookup, as the data array is much larger, and the time between the

- *P. Lotfi-Kamran is with the School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran. E-mail: plotfi@ipm.ir.*
- *M. Modarressi is with the Department of Electrical and Computer Engineering, College of Engineering, University of Tehran, and the School of Computer Science, Institute for Research in Fundamental Sciences (IPM).*
- *H. Sarbazi-Azad is with the Department of Computer Engineering, Sharif University of Technology, and the School of Computer Science, Institute for Research in Fundamental Sciences (IPM).*

1. Properly buffered wire. The delay of non-buffered wire can be significant in advanced technologies.

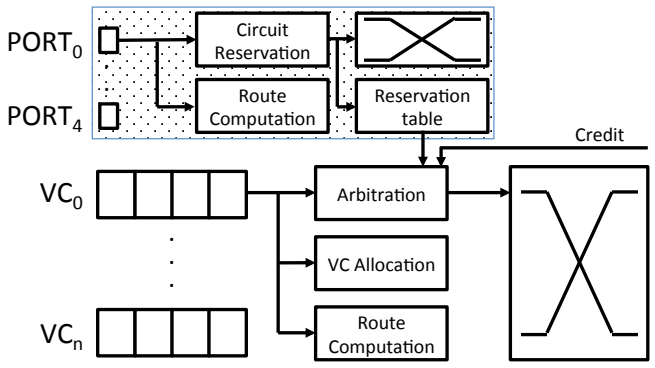

Fig. 1. Various elements of a CIMA router. Control network is shown in dotted background.

end of the tag and data lookup is available for the circuit setup process.

Depending on the available time and the distance between the LLC tile and the requesting core, CIMA reserves part of the path (or even the entire path) to the destination. The response packet benefits from the delay and energy of circuit switching for the part that is reserved, and for the rest of the path, normal virtual cut-through (VCT) switching will be used. Through detailed full-system simulations of a 64-core CMP running scale-out workloads, we show that CIMA improves system performance by 15% over packet-switched NoCs and by 21% over the state-of-the-art hybrid circuit-packet-switched NoC.

2 The Proposed NoC

CIMA implements a hybrid circuit-packet switched interconnection fabric for low-latency on-chip communications in chip multiprocessors. CIMA leverages several insights that allow it to minimize interconnect delays.

First, while there are many types of messages passing through the on-chip network of a cache-coherent chip multiprocessor (e.g., invalidation, write back, etc.), the message types that have significant impact on the performance of workloads are *requests* for pieces of data or instructions originating from cores, as a result of L1 cache misses, and *responses* to the requests originating from LLC slices, should the requests turn into hits in the LLC [20]. Other types of messages either constitute a tiny fraction of on-chip traffic (e.g., coherence activity [21], [22]) or require significant processing delay, much larger than the delay of the interconnect (e.g., off-chip memory accesses), or no other event depends on them (e.g., write backs), and as a result, accelerating their transfer in the on-chip network has little impact on system performance. Based on this observation, CIMA focuses on accelerating delay sensitive *request-response* messages.

Second, while it is not known in advance when and where a request packet is going to be sent (i.e., requests are generated as a result of L1 cache misses), for a response packet, not only we know where the packet has to be sent to (i.e., the core that originated the request), but also we know when it is going to happen (i.e., end of data lookup), because the delays of the tag and data lookup are known to CMP designers. Due to this behavior, CIMA focuses on accelerating response packets.

Third, when a request reaches the destination, it is queued until it becomes its turn for tag lookup. Right after the tag lookup, if the lookup indicates a hit, we know that there will be a response packet. However, there is still some time until the requested piece of data becomes ready (i.e., end of data lookup), and CIMA takes advantage of this time difference to set up a circuit for the response packet. Many last-level caches benefit from a serial tag and data lookup. In such cases, the whole data lookup time is available to CIMA. Even if an LLC uses parallel tag and data lookup, data lookup takes longer because the data array is much larger than the tag array, and the time difference between the end of the tag and data lookup will be available to CIMA.

Forth, due to good L1 cache performance in server and commercial workloads [20], [23], the traffic in the network is low. Consequently, it has been shown that when a request reaches the destined LLC slice, most of the time, either there is no other request or there is just one request that is being served by the LLC [24]. CIMA takes advantage of this feature of server workloads to simplify the protocol without a considerable loss of opportunity as we will see later in this section.

Based on these insights and observations, CIMA tries to take advantage of the time difference between the end of tag and data lookup to set up a circuit (i.e., prepare the path) for the upcoming response packet. In the rest of this section, we detail the organization of a CIMA router.

2.1 CIMA Router Microarchitecture

CIMA uses VCT switching for request packets, and attempts to use circuit switching for response packets. A response packet benefits from circuit switching for part of the path on which a circuit is established. For the rest of the path, VCT switching will be used.

A CIMA router consists of two related networks: a *data* network for sending and receiving packets (e.g., requests and responses), and a *control* network for setting up circuits. The data network has five ports: four network ports and one local port. Key elements of the data network include virtual channel buffers, route computation unit, VC allocation logic, arbitration logic, and crossbar.

The control network, which is included for setting up circuits, has five (narrow) ports: four network ports and one local port. The elements of the control network include circuit reservation logic (which includes arbitration), circuit reservation table, route computation unit, and crossbar. The control network does not have buffers (i.e., it supports bufferless routing) and virtual channels. Figure 1 shows the main elements of a CIMA router.

Request packets are routed using VCT switching: no element unique to CIMA is involved. On arriving a request packet to the destination, it is queued within the LLC and waits for its lookup time. If the tag lookup indicates a hit, LLC controller will notify the network interface (NI). The NI creates a control packet, which is 1-flit long, and places it in the local register of the control network if the register is empty. In case the register is not empty, the control packet will be dropped immediately, as it is obvious that the circuit cannot be established. A control packet simply consists of the *destination* address and the *lag* between the control and

the response packet. The destination is the source of the *request* packet, and the lag is the number of cycles between the end of tag and data lookup in the LLC.

On receiving a control packet in a router, the packet is passed through route computation unit and circuit reservation logic in parallel (i.e., look-ahead routing). The circuit reservation logic determines if the requested circuit can be granted according to conditions that will be discussed shortly. If the circuit cannot be granted, the control packet will be dropped and no other action is necessary. Otherwise, the granted circuit will be recorded in the circuit reservation table. As a control packet experiences longer delays as compared to the response packet it precedes, we also need to adjust the *lag* in each router to account for this difference. If the *lag* is not zero and current router is not the destination, the control packet will go through the crossbar and the link in the following cycle.

The time interval between two consecutive control packets sent over an output port (which must be equal to or greater than the transmission time of a response packet) is longer than the time required to process a control packet. Consequently, when a router sends a control packet, with certainty, the previous control packet sent by this router is processed in the downstream router (i.e., either forwarded or dropped). Therefore, CIMA does not require a credit network for the control network (See Figure 1).

CIMA relies on VCT switching to simplify the circuit reservation logic. In CIMA, the size of the buffers in the data network is equal to the size of a response packet. The size of the buffer enables CIMA routers not to interrupt the transmission of a response packet after the header flit gets transferred. If a router transfers the header flit of a response packet at time t, the subsequent flits will be transferred at $t+1$, $t+2$, etc. (this is the expected behavior of a circuit). As a result, there is a guarantee that if there is one empty slot in the downstream buffer, one full response packet can be transferred to the downstream router on successive cycles, and there will be no buffer overflow.

The circuit reservation logic checks the following conditions to determine if a circuit can be reserved: (1) no circuit is already reserved in the reservation table for the requested output port, as CIMA only allows one reservation per output port at any time, (2) no currently in-transfer response packet holds the output port in the requested timeslots, and (3) there is (would be) at least one empty slot in the downstream buffer if it is idle (after the transmission of the current packet). The circuit reservation logic then arbitrates on the control packets for which the three conditions hold to decide which ones will be granted. A control packet will be dropped (i.e., bufferless routing) if any of the three conditions does not hold or if it does not win the arbitration, because under these conditions the requested circuit cannot be granted (for the response packet, circuit switching will be used to this point, and VCT switching for the rest of the path).

All three conditions are evaluated in a look-ahead manner: at each cycle, conditions are evaluated for the next cycle. Conditions 1 and 3 can fully be evaluated. As Condition 2 requires the *lag* field of the control packet, its evaluation is partial: every cycle and for each output port, the router calculates for how many cycles the port is busy. When a control packet arrives, the circuit reservation logic just checks if the *lag* is greater than or equal to this number. As the *lag* field has only few bits (e.g., three bits), the comparison can be done quickly.

The arbitration unit of the data network in our proposal is slightly different from its conventional counterpart. In CIMA, the arbitration unit should also consider the reserved circuits and not grant the output port to a response packet (request packets are fine because they are just one flit) if a circuit is reserved on the output port (even for a few cycles later).

At the end of data lookup in the LLC, a response packet will be sent to the requesting core. If a circuit is reserved for this packet in a router (i.e., there is an entry in the reservation table for this packet), as route computation, VC allocation, and arbitration have already been done as part of the circuit reservation, the header flit bypasses these steps and directly goes to the crossbar and link, greatly speeding up the transfer. After the transmission of the header flit, the entry in the reservation table gets discarded. If a circuit is not reserved for a response packet, the header flit should go through all units as in a VCT-switched router.

3 METHODOLOGY

Table 1 summarizes the key elements of our methodology, with the following sections detailing the specifics of the evaluated designs, technology parameters, workloads, and simulation infrastructure.

3.1 CMP Parameters

Our target is a many-core CMP implemented in 32 nm technology. We use the Scale-Out Processor methodology [24], [25] to derive the optimal core count, number of memory controllers, and LLC capacity for the assumed technology and microarchitectural parameters. The resulting processor features 64 cores, 8 MB of last-level cache, and four DDR3-1667 memory channels. Core microarchitecture is modeled after an ARM Cortex-A15, a three-way out-of-order design with 32KB L1-I and L1-D caches. Cache line size is 64 bytes. The *request* packet length is one flit, while the length of the *response* packet is five flits.

We consider three system organizations, as follows:

Mesh: Our baseline for the evaluation is a mesh-based tiled CMP. The 64 tiles are organized as an 8-by-8 grid, with each tile containing a core, a slice of the LLC and a directory node.

At the network level, a mesh hop consists of a single-cycle link traversal followed by a two-stage router pipeline for a total of three cycles per hop at zero load. The router performs routing, VC allocation, and speculative crossbar (XB) allocation in the first cycle, followed by XB traversal in the next cycle. Each router port has three VCs to guarantee deadlock freedom across three message classes: data requests, snoop requests, and responses. Each VC is five flits deep, which is the minimum necessary to cover the round-trip credit time.

HCS: The state-of-the-art Hybrid Circuit Switching (HCS) [17] network is implemented on top of the mesh baseline. Each channel of every router is divided into two

TABLE 1
Evaluation parameters.

Parameter	Value
Technology	32 nm, 0.9 V, 2 GHz
Processor features	64 cores, 8 MB NUCA LLC, four DDR3-1667 memory channels
Core	ARM Cortex-A15-like: 3-way out-of-order, 64-entry ROB, 16-entry LSQ, 2.9 mm^2, 1 W
Cache	per MB: 3.2 mm^2, 500 mW
NoC Organizations:	
Mesh	Router: 5 ports, 3 VCs/port, 5 flits/VC, 2-stage speculative pipeline. Link: 1 cycle
HCS	Data network: 5 ports/router, 3 VCs/port Packet-switching mode: 2-stage speculative pipeline. Link: 1 cycle Circuit-switching mode: Bypassing pipeline stages, Link: 1 cycle Control network: 5 ports/router, NO VCs, 1-stage pipeline. Link: 1 cycle
CIMA	Data network: 5 ports/router, 3 VCs/port Packet-switching mode: 2-stage speculative pipeline. Link: 1 cycle Circuit-switching mode: Bypassing pipeline stages, Link: 1 cycle Control network: 5 ports/router, NO VCs, 1-stage pipeline. Link: 1 cycle

half-sized physical channels. Physical channels are used for carrying packets in the network. A circuit for a source-destination pair can be set up on physical channels using a dedicated setup network. When a source sends a packet to a destination, if a circuit is already established for the source-destination pair, the packet will be sent using circuit switching, and flits pass each hop in one cycle. Otherwise, VCT switching will be used, and at the same time, a circuit for the source-destination pair will be set up (potentially by tearing down existing circuits).

CIMA: Implemented on top of the mesh baseline. For *request* packets, VCT switching will be used. Control packets are injected into the network right after the end of tag lookups to set up circuits for *response* packets. A control packet passes each hop in two cycles. For part of the path to the destination on which a circuit is established, the header flit of a response packet passes routers in just one cycle. For the rest of the path, baseline VCT switching will be used. Using CACTI, we estimated the tag and data lookup delays of the LLC to be one and four cycle(s), respectively.

3.2 Technology Parameters

We use publicly available tools and data to estimate the area and energy of the various network organizations. Our study targets a 32 nm technology node with an on-die voltage of 0.9 V and a 2 GHz operating frequency.

We use custom wire models, derived from a combination of sources [26], [27], to model links and router switch fabrics. For links, we model semi-global wires with a pitch of 200 nm and power-delay-optimized repeaters that yield a link latency of 125 ps/mm. On random data, links dissipate 50 fJ/bit/mm, with repeaters responsible for 19% of link energy. For area estimates, we assume that link wires are routed over logic or SRAM and do not contribute to network area; however, repeater area is accounted for in the evaluation.

Our buffer models are taken from DSENT [28]. We model flip-flop based buffers, as all networks have relatively few buffers per port. Cache area, energy, and delay parameters are derived via CACTI 6.5 [29]. A 1 MB slice of the LLC has an area of 3.2 mm^2 and dissipates on the order of 500 mW of power, mostly due to leakage.

Finally, parameters for the ARM Cortex-A15 core are borrowed from Microprocessor Report and scaled down from the 40 nm technology node to the 32 nm target. Core area, including L1 caches, is estimated at 2.9 mm^2. Core power is 1.05 W at 2 GHz. Core features include 3-way decode/issue/commit, 64-entry ROB, and 16-entry LSQ.

3.3 Workloads

We evaluate CIMA using two synthetic traffic patterns: *hotspot* and *uniform random*. These traffic patterns provide insight into the relative strengths and weaknesses of the different networks. These traffic patterns determine how a *request* gets injected into the network. A *response* to the requester gets injected into the network five cycles (which is the delay of the LLC) after the arrival of the request packet to the LLC.

Moreover, we use scale-out workloads from CloudSuite [30]. The workloads include Data Serving, MapReduce, Media Streaming, Web Frontend, SAT Solver, and Web Search. We consider two MapReduce workloads – text classification (MapReduce-C) and word count (MapReduce-W). Two of the workloads – SAT Solver and MapReduce – are batch, while the rest are latency-sensitive and are tuned to meet the response time objectives.

3.4 Simulation Infrastructure

We use Booksim network simulator [31] to measure latency of the three NoCs under the synthetic traffic patterns. For scale-out workloads, we estimate the performance of the various processor designs using Flexus full-system simulator [32]. Flexus extends the Virtutech Simics functional simulator with timing models of cores, caches, on-chip protocol controllers, and interconnect. Flexus models the SPARC v9 ISA and is able to run unmodified operating systems and applications.

We use the SimFlex multiprocessor sampling methodology [32]. Our samples are drawn over an interval of 10 seconds (30 seconds for Media Streaming) of simulated time. For each measurement, we launch simulations from

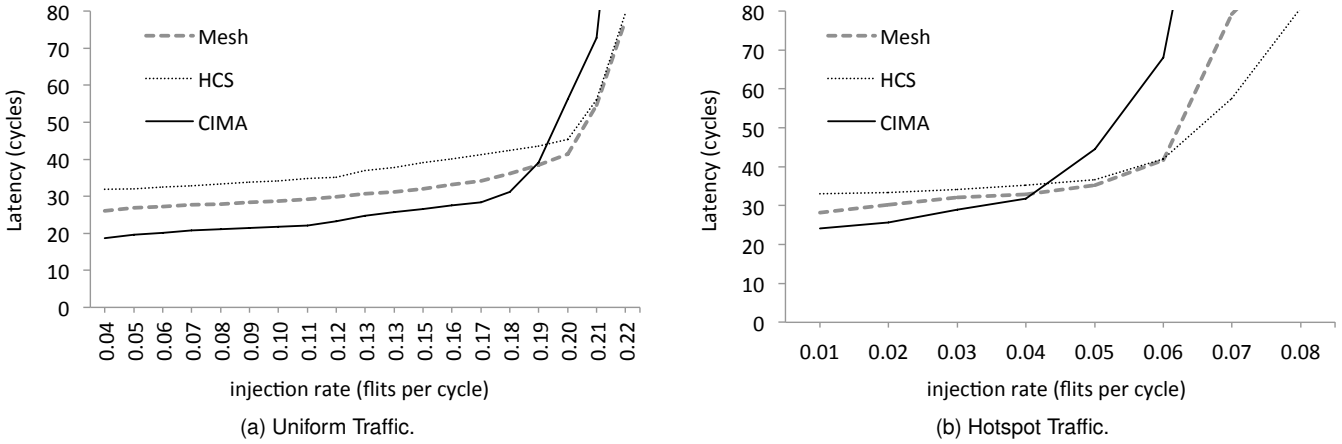

Fig. 2. Latency versus flit injection rate on synthetic traffic patterns.

checkpoints with warmed caches and branch predictors, and run 100 K cycles (2 M cycles for Data Serving) to achieve a steady state of detailed cycle-accurate simulation before collecting measurements for the subsequent 50 K cycles. We use the ratio of the number of application instructions to the total number of cycles (including the cycles spent executing operating system code) to measure performance; this metric has been shown to accurately reflect overall system throughput [32]. Performance measurements are computed with 95% confidence with an average error of less than 5%.

4 Evaluation

We examine system performance and area efficiency of Mesh, HCS, and CIMA designs given a fixed 128-bit link bandwidth, followed by a discussion of power trends.

4.1 NoC Delay on Synthetic Traffic

Figure 2 shows the average communication delay as a function of the average packet injection rate for the two synthetic traffic patterns. As expected, CIMA achieves the lowest latency under both traffic patterns in normal network condition. Under uniform traffic pattern, CIMA improves network latency over Mesh by up to 39% and over HCS by up to 70%. The non-uniform nature of the Hotspot traffic pattern reduces the benefit of CIMA as compared to Mesh and HCS. With Hotspot traffic pattern, CIMA improves network latency over Mesh by up to 16% and over HCS by up to 36%. It is important to note that the traffic patterns in CMPs resemble uniform distribution, and consequently, we expect significant benefit from using CIMA in chip multiprocessors (See Section 4.2 for evaluation of CIMA on realistic workloads).

CIMA establishes circuits for response packets, so the improvement in network latency as compared to Mesh comes from faster routing of response packets. Both CIMA and HCS establish circuits (CIMA only for response packets and HCS for both request and response packets). HCS relies on many narrow physical channels to establish multiple circuits per output port. Therefore, it requires many flits per packet, and as such, its latency increases due to serialization. Moreover, as the number of LLC tiles is 64, but HCS reserves only few circuits (two for the shown results, but we also tried four and witnessed worst latency due to higher serialization), the chances of using the circuits are low. Consequently, HCS gives the highest network latency.

At very high loads, HCS and Mesh achieve the best results due to higher utilization of bandwidth. However, this behavior has little practical significance, as we do not run a network-on-chip in saturation points.

4.2 System Performance on CloudSuite

Figure 3 shows full-system performance, normalized to Mesh, for various NoC organizations. As can be seen, CIMA achieves the highest performance. Compared to Mesh, the proposed CIMA network improves performance by 6%-27%, with a geomean of 15%. Compared to the HCS network, CIMA improves performance by 6%-30%, with a geomean of 21%. The highest performance gain is registered on the Media Streaming application, which is characterized by very low instruction-level parallelism (ILP) and memory-level parallelism (MLP), making it particularly sensitive to the LLC access latency.

The difference between Mesh and CIMA is that Mesh uses standard packet switching, but CIMA attempts to forward response packets using circuit switching. Consequently, CIMA's observed improvement in system performance over Mesh is due to faster response-packets' delivery. Compared to HCS, CIMA requires fewer number of flits per packet (i.e., lower serialization overhead) and can also use established circuits more effectively. As a result, system performance of CMPs with CIMA is larger than that of CMPs with HCS.

It is important to note that for these workloads the majority of LLC accesses turn into hits, and consequently, CIMA has opportunity to reduce the transmission time of responses to such LLC accesses. The reduction in network access latency has caused the performance improvement shown in Figure 3.

We conclude the performance assessment by noting that while the bisection bandwidths of the various topologies are different, the networks are not congested. Differences in latency, not bandwidth, across the topologies are responsible for the performance variations.

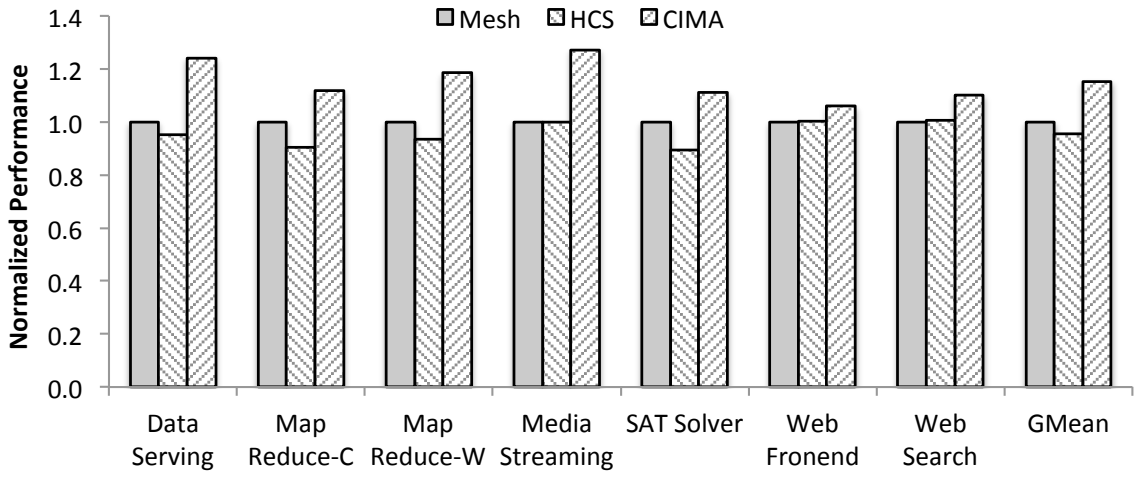

Fig. 3. System performance, normalized to a mesh-based design.

4.3 NoC Area

Figure 4 breaks down the NoC area of the three organizations by links, buffers, and crossbars. Only repeaters are accounted for in link area, as wires are assumed to be routed over tiles.

We observe that the NoC area of all three networks-on-chip is small. At over 3.7 mm^2, HCS has the highest NoC area, exceeding that of Mesh by 6%. CIMA's footprint of 3.5 mm^2 is slightly larger than that of a mesh, requiring 3% less area than the HCS network. The savings are due to the use of a narrow bufferless control network. CIMA requires a bufferless control network with 11-bit links for setting up and maintaining circuits. As most of the area of a NoC is due to buffers and crossbars, the difference between the area of Mesh (which does not have a control network) and CIMA is negligible.

4.4 Power Analysis

Our analysis shows that the NoC is not a significant consumer of power at the chip level (also corroborating prior work [20], [33]). We estimate the power usage of Mesh, HCS, and CIMA to be 1.8 W, 1.5 W, and 1.7 W, respectively. For all three organizations, NoC power is below 2 W. In contrast, cores alone consume in excess of 60 W. Low ILP and MLP of scale-out workloads is the main reason for the low power consumption at the NoC level.

5 Related Work

The need for low-latency and scalable on-chip communication mechanisms is pointed out in prior work. Most existing network-on-chip design methods target either hop count reduction or per-hop latency reduction to achieve low-latency. The former methods mainly consist of mapping and topology optimization designs, which try to minimize hop count to increase performance. In particular, high-radix routers have received increasing attention in recent years [34]. Famous high-radix router designs include Fat Tree [35], Flattened Butterfly [9], Black-Widow [36], and MECS [37]. High-radix routers divide the router bandwidth into larger number of narrow ports. Consequently, routers have more links to connect to other routers, thereby average hop count is reduced. These designs, however, complicate VLSI layout due to long links that connect remote routers. They also suffer from higher serialization delay due to narrower links. SMART (Single-cycle Multi-hop Asynchronous Repeated Traversal) introduces an alternative approach to adding physical links by using virtual long links [38]; it allows flits traverse multi-hop paths within a single cycle by virtually bypassing all routers along the route. As the bypassing is implemented on regular NoC links and crossbars without adding any physical channels to the data-path, a path reservation and arbitration logic is implemented to resolve conflict among flits sharing a common link at the same time.

Our method tries to reduce per-hop latency and is orthogonal to most of the above methods that target hop count reduction. Router latency reduction is an attractive option for overcoming the high NoC latency. These methods range from old yet efficient designs like look-ahead routing and speculative switch allocation [39] to more complicated router microarchitecture optimization, hybrid switching, and bypassing methods [17], [40], [41], [42].

As an example, in [42] a bufferless NoC is proposed that aims at reducing NoC latency by always forwarding received packets to some output port in a single cycle. If the preferred port of a packet is busy, it will be deflected to some idle, but not necessarily profitable, port. This single-cycle operation, however, will come at the cost of deflecting some packets and although improves performance in low traffic, increases network latency under heavy traffic.

Circuit switching is an efficient scheme to reduce on-chip communication latency, when compared to packet switching, since packets need not go through buffering, routing, hop-by-hop flow control, and arbitration at each hop once circuits are established. However, the baseline circuit switching suffers from long circuit setup delay and poor bandwidth utilization.

Equipping a baseline packet-switched NoC with circuit switching to benefit from the best of both switching mecha-

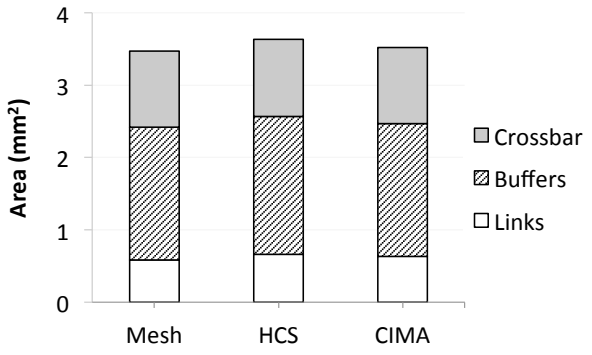

Fig. 4. NoC area breakdown.

nisms has been addressed in several prior work [17], [40], [41]. Express Virtual channels (EVCs) [40] allow packets to virtually bypass a router to accelerate packet transfer. EVCs are virtual straight paths between some fixed source-destination pairs in a conventional mesh NoC along a single dimension and cannot turn from one dimension to another. They are set up on virtual channel 0 at each physical channel of a router; this virtual channel bypasses the entire router pipeline. Virtual point-to-point connections (VIPs) [41] are low-latency dedicated end-to-end virtual paths between any two source-destination pairs in a NoC by bypassing the pipeline of intermediate routers (one router per cycle). VIPs exploit on-chip traffic locality and hence, are beneficial to the applications with temporal and spatial locality in communication. They are reconfigurable and can dynamically adapt to traffic pattern of applications.

Circuit-switched coherence mechanism [17] establishes and reuses a circuit for a sequence of messages between a source-destination pair. A new circuit can tear down existing circuits, so it is guaranteed that a new circuit can always be established successfully. The packets of a torn down circuit will move to the packet-switched network and stay there for the rest of their travel.

In token flow control method [43], each router advertises its empty ports as tokens to the nodes at the neighborhood of n hops. Tokens can be used by packets to bypass empty ports and shorten their path. NOC-Out is a network-on-chip for scale-out server applications that benefits from both high-radix topologies to reduce hop count and low-latency routers to reduce per-hop latency [20]. It relies on flattened butterfly as a backbone that connects LLC slices. Each flattened butterfly router is also connected to a simple and fast tree-based NoC that connects a cluster of cores to the backbone. Requests are first directed from the core to the flattened butterfly via the low-latency tree-based network and then to the target LLC slice via the flattened butterfly. Then, the response packet traverses the same path in the backward direction.

To benefit from circuit switching without suffering from long setup time, proactive circuit switching [16] hides circuit setup latency by pre-allocating circuits. It adopts a control plane that carries data request packets with higher voltage and frequency (and hence faster), and a low-power data plane that handles data packets. In this design, request packets reserve circuits in backward direction, as they move toward the destination, for their anticipated response packets, because response packets in upcoming cycles should be sent in the opposite direction of request packets. Although pre-allocation considerably reduces the latency for packets that travel over circuits, early reservation of circuits can result in underutilization of network resources.

As discussed in this paper, our method equips a baseline packet-switched router with a prediction-based circuit-switching scheme. It differs from most existing circuit-switching mechanisms, in that it eliminates circuit setup overhead by pre-allocation of circuits for the exact transfer time of packets. Working on long response packets sent by LLC slices, this method takes advantage of the LLC information about the exact time of data preparation and packet destination to reserve circuits before the actual transfer. Like other single-cycle routers, response packets benefit from a single-cycle transfer per hop up to the point where circuit ends, and conventional packet switching from that point on.

6 Conclusion

In this paper, we introduced CIMA, a hybrid circuit-packet-switched NoC for accelerating memory accesses in CMPs. CIMA routes request packets using packet switching (i.e., VCT), but tries to route response packets using circuit switching. For this goal, CIMA benefits from a narrow control network to announce future arrival of a response packet to downstream routers.

CIMA takes advantage of the time between a tag and data lookup in the LLC to inform routers of the upcoming response packets. Right after a tag lookup that results in a hit, CIMA sends a control packet to notify downstream routers of the upcoming response packet. Having been informed of the response packet, routers can pass it through quickly. With negligible overhead, CIMA improves system performance by 15% (21%) over state-of-the-art packet-switched (hybrid circuit-packet switched) routers.

Acknowledgments

The authors would like to thank Armin Ahmadzadeh, Hatef Madani, and Mahmood Naderan for their assistance with setting up and maintaining the cluster that is used to conduct the experiments. This work was supported in part by a grant from IPM.

References

[1] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, L. Bao, J. F. Brown, M. Mattina, C.-C. Miao, C. Ramey, D. Wentzlaff, W. Anderson, E. Berger, N. Fairbanks, D. Khan, F. Montenegro, J. Stickney, and J. Zook, "TILE64 - Processor: A 64-Core SoC with Mesh Interconnect," in *Proceedings of the IEEE International Solid-State Circuits Conference*, Feb. 2008, pp. 88–598.

[2] Cavium Networks, "Cavium Announces Availability of ThunderX™: Industry's First 48 Core Family of ARMv8 Workload Optimized Processors for Next Generation Data Center & Cloud Infrastructure," http://www.cavium.com/newsevents-Cavium-Announces-Availability-of-ThunderX.html, Dec. 2014.

[3] Intel, "Intel® Xeon Phi™ Product Family," http://www.intel.com/content/www/us/en/processors/xeon/xeon-phi-detail.html.

[4] E. Ozer, K. Flautner, S. Idgunji, A. Saidi, Y. Sazeides, B. Ahsan, N. Ladas, C. Nicopoulos, I. Sideris, B. Falsafi, A. Adileh, M. Ferdman, P. Lotfi-Kamran, M. Kuulusa, P. Marchal, and N. Minas, "EuroCloud: Energy-Conscious 3D Server-on-Chip for Green Cloud Services," in *Proceedings of the Workshop on Architectural Concerns in Large Datacenters in conjunction with ISCA*, Jun. 2010.

[5] B. M. Beckmann, M. R. Marty, and D. A. Wood, "ASR: Adaptive Selective Replication for CMP Caches," in *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, Dec. 2006, pp. 443–454.

[6] J. Chang and G. S. Sohi, "Cooperative Caching for Chip Multiprocessors," in *Proceedings of the 33rd Annual International Symposium on Computer Architecture*, Jun. 2006, pp. 264–276.

[7] M. Zhang and K. Asanović, "Victim Replication: Maximizing Capacity while Hiding Wire Delay in Tiled Chip Multiprocessors," in *Proceedings of the 32nd Annual International Symposium on Computer Architecture*, Jun. 2005, pp. 336–345.

[8] J. Kim, D. Park, T. Theocharides, N. Vijaykrishnan, and C. R. Das, "A Low Latency Router Supporting Adaptivity for On-chip Interconnects," in *Proceedings of the 42nd Annual Design Automation Conference*, Jun. 2005, pp. 559–564.

[9] J. Kim, W. J. Dally, and D. Abts, "Flattened Butterfly: A Cost-Efficient Topology for High-Radix Networks," in *Proceedings of the 34th Annual International Symposium on Computer Architecture*, Jun. 2007, pp. 126–137.

[10] C.-H. O. Chen, S. Park, T. Krishna, S. Subramanian, A. P. Chandrakasan, and L.-S. Peh, "SMART: A Single-cycle Reconfigurable NoC for SoC Applications," in *Proceedings of the Conference on Design, Automation and Test in Europe*, Mar. 2013, pp. 338–343.

[11] P. Lotfi-Kamran, A.-M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, and Z. Navabi, "EDXY - A Low Cost Congestion-Aware Routing Algorithm for Network-on-Chips," *Journal of Systems Architecture*, vol. 56, no. 7, pp. 256–264, Jul. 2010.

[12] A. Kumar, L.-S. Peh, P. Kundu, and N. K. Jha, "Express Virtual Channels: Towards the Ideal Interconnection Fabric," in *Proceedings of the 34th Annual International Symposium on Computer Architecture*, Jun. 2007, pp. 150–161.

[13] S. Liu, A. Jantsch, and Z. Lu, "Parallel Probing: Dynamic and Constant Time Setup Procedure in Circuit Switching NoC," in *Proceedings of the Conference on Design, Automation and Test in Europe*, Mar. 2012, pp. 1289–1294.

[14] A. Hansson, M. Coenen, and K. Goossens, "Channel Trees: Reducing Latency by Sharing Time Slots in Time-multiplexed Networks on Chip," in *Proceedings of the 5th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis*, Oct. 2007, pp. 149–154.

[15] P.-H. Pham, J. Park, P. Mau, and C. Kim, "Design and Implementation of Backtracking Wave-Pipeline Switch to Support Guaranteed Throughput in Network-on-Chip," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 2, pp. 270–283, Feb. 2012.

[16] A. Abousamra, A. K. Jones, and R. Melhem, "Proactive Circuit Allocation in Multiplane NoCs," in *Proceedings of the 50th Annual Design Automation Conference*, Jun. 2013, pp. 35:1–35:10.

[17] N. D. E. Jerger, L.-S. Peh, and M. H. Lipasti, "Circuit-Switched Coherence," in *Proceedings of the Second ACM/IEEE International Symposium on Networks-on-Chip*, Apr. 2008, pp. 193–202.

[18] D. Sanchez and C. Kozyrakis, "The ZCache: Decoupling Ways and Associativity," in *Proceedings of the 43rd Annual IEEE/ACM International Symposium on Microarchitecture*, Dec. 2010, pp. 187–198.

[19] M. Ferdman, P. Lotfi-Kamran, K. Balet, and B. Falsafi, "Cuckoo Directory: A Scalable Directory for Many-core Systems," in *Proceedings of the 17th IEEE International Symposium on High-Performance Computer Architecture*, Feb. 2011, pp. 169–180.

[20] P. Lotfi-Kamran, B. Grot, and B. Falsafi, "NOC-Out: Microarchitecting a Scale-Out Processor," in *Proceedings of the 45th Annual IEEE/ACM International Symposium on Microarchitecture*, Dec. 2012, pp. 177–187.

[21] A. Moshovos, G. Memik, B. Falsafi, and A. Choudhary, "JETTY: Filtering Snoops for Reduced Energy Consumption in SMP Servers," in *Proceedings of the 7th IEEE International Symposium on High-Performance Computer Architecture*, Jan. 2001, pp. 85–96.

[22] P. Lotfi-Kamran, M. Ferdman, D. Crisan, and B. Falsafi, "TurboTag: Lookup Filtering to Reduce Coherence Directory Power," in *Proceedings of the 16th ACM/IEEE International Symposium on Low Power Electronics and Design*, Aug. 2010, pp. 377–382.

[23] P. Ranganathan, K. Gharachorloo, S. V. Adve, and L. A. Barroso, "Performance of Database Workloads on Shared-Memory Systems with Out-of-Order Processors," in *Proceedings of the 8th International Conference on Architectural Support for Programming Languages and Operating Systems*, Oct. 1998, pp. 307–318.

[24] P. Lotfi-Kamran, B. Grot, M. Ferdman, S. Volos, O. Kocberber, J. Picorel, A. Adileh, D. Jevdjic, S. Idgunji, E. Ozer, and B. Falsafi, "Scale-Out Processors," in *Proceedings of the 39th Annual International Symposium on Computer Architecture*, Jun. 2012, pp. 500–511.

[25] B. Grot, D. Hardy, P. Lotfi-Kamran, B. Falsafi, C. Nicopoulos, and Y. Sazeides, "Optimizing Data-Center TCO with Scale-Out Processors," *IEEE Micro*, vol. 32, no. 5, pp. 52–63, Sep. 2012.

[26] J. D. Balfour and W. J. Dally, "Design Tradeoffs for Tiled CMP On-Chip Networks," in *Proceedings of the 20th Annual ACM International Conference on Supercomputing*, Jun. 2006, pp. 187–198.

[27] "International Technology Roadmap for Semiconductors (ITRS), 2011 Edition," http://www.itrs.net/ITRS%201999-2014%20Mtgs,%20Presentations%20&%20Links/2011ITRS/Home2011.htm.

[28] C. Sun, C.-H. O. Chen, G. Kurian, L. Wei, J. Miller, A. Agarwal, L.-S. Peh, and V. Stojanovic, "DSENT - A Tool Connecting Emerging Photonics with Electronics for Opto-Electronic Networks-on-Chip Modeling," in *Proceedings of the 2012 IEEE/ACM Sixth International Symposium on Networks-on-Chip*, May 2012, pp. 201–210.

[29] N. Muralimanohar, R. Balasubramonian, and N. P. Jouppi, "Optimizing NUCA Organizations and Wiring Alternatives for Large Caches with CACTI 6.0," in *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, Dec. 2007, pp. 3–14.

[30] CloudSuite 1.0, http://parsa.epfl.ch/cloudsuite.

[31] N. Jiang, D. U. Becker, G. Michelogiannakis, J. Balfour, B. Towles, J. Kim, and W. J. Dally, "A detailed and flexible cycle-accurate network-on-chip simulator," in *Proceedings of the 2013 IEEE International Symposium on Performance Analysis of Systems and Software*, Apr. 2013, pp. 86–96.

[32] T. F. Wenisch, R. E. Wunderlich, M. Ferdman, A. Ailamaki, B. Falsafi, and J. C. Hoe, "SimFlex: Statistical Sampling of Computer System Simulation," *IEEE Micro*, vol. 26, no. 4, pp. 18–31, July-August 2006.

[33] A. Kumar, P. Kundu, A. P. Singh, L.-S. Peh, and N. K. Jha, "A 4.6Tbits/s 3.6GHz Single-cycle NoC Router with a Novel Switch Allocator in 65nm CMOS," in *Proceedings of the 25th International Conference on Computer Design*, Oct. 2007, pp. 63–70.

[34] A. Jain, R. Parikh, and V. Bertacco, "High-Radix On-chip Networks with Low-Radix Routers," in *Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design*, Nov. 2014, pp. 289–294.

[35] D. Ludovici, F. Gilabert, S. Medardoni, C. Gómez, M. E. Gómez, P. López, G. N. Gaydadjiev, and D. Bertozzi, "Assessing Fat-tree Topologies for Regular Network-on-chip Design Under Nanoscale Technology Constraints," in *Proceedings of the Conference on Design, Automation and Test in Europe*, Apr. 2009, pp. 562–565.

[36] S. Scott, D. Abts, J. Kim, and W. J. Dally, "The BlackWidow High-Radix Clos Network," in *Proceedings of the 33rd Annual International Symposium on Computer Architecture*, Jun. 2006, pp. 16–28.

[37] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu, "Express Cube Topologies for On-Chip Interconnects," in *Proceedings of the 15th IEEE International Symposium on High-Performance Computer Architecture*, Feb. 2009, pp. 163–174.

[38] T. Krishna, C.-H. O. Chen, W. C. Kwon, and L.-S. Peh, "Breaking the On-chip Latency Barrier Using SMART," in *Proceedings of the 19th IEEE International Symposium on High-Performance Computer Architecture*, Feb. 2013, pp. 378–389.

[39] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*, 1st ed. Morgan Kaufmann Publishers Inc., 2003.

[40] T. Krishna, A. Kumar, L.-S. Peh, J. Postman, P. Chiang, and M. Erez, "Express Virtual Channels with Capacitively Driven Global Links," *IEEE Micro*, vol. 29, no. 4, pp. 48–61, Jul. 2009.

[41] M. Modarressi, A. Tavakkol, and H. Sarbazi-Azad, "Virtual Point-to-point Connections for NoCs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 6, pp. 855–868, Jun. 2010.

[42] T. Moscibroda and O. Mutlu, "A Case for Bufferless Routing in On-Chip Networks," in *Proceedings of the 36th Annual International Symposium on Computer Architecture*, Jun. 2009, pp. 196–207.

[43] A. Kumar, L.-S. Peh, and N. K. Jha, "Token Flow Control," in *Proceedings of the 41st Annual IEEE/ACM International Symposium on Microarchitecture*, Nov. 2008, pp. 342–353.