Extending the Energy Efficiency and Performance With Channel Buffers, Crossbars, and Topology Analysis for Network-on-Chips

Dominic DiTomaso, *Student Member, IEEE*, Randy Morris, *Student Member, IEEE*, Avinash Karanth Kodi, *Member, IEEE*, Ashwini Sarathy, and Ahmed Louri, *Fellow, IEEE*

***Abstract*— Network-on-chips (NoCs) have emerged as a scalable solution to the wire delay constraints, thereby providing a high-performance communication fabric for future multicores. Research has shown that power, area, and performance of the NoC architecture are tightly integrated with the design and optimization of the link, router (buffer and crossbar), and topology. Recent work has shown that adaptive channel buffers (on-link storage) can considerably reduce power consumption and area overhead by reducing or replacing the power-hungry router buffers. However, channel buffer design can lead to head-of-line (HoL) blocking, which eventually reduces the throughput of the network. In this paper, we design channel buffers and router crossbars to improve the performance (latency, throughput) while reducing the power consumption. In addition, we implement the proposed channel buffers and crossbar organizations in a concentrated torus (CTorus) topology which is a dual network without the additional area overhead. We compare other dual networks with leading topologies such as mesh2X, concentrated mesh2X (CMesh2X), and flattened butterfly2X (FBfly2X), each implemented with channel buffers. Our proposed designs analyze the power–performance–area tradeoff in designing channel buffers for NoC architectures while alleviating HoL blocking through buffer organizations and crossbar optimizations. Results using Synopsys design compiler showed that the buffer and crossbar organizations for an 8 × 8 mesh architecture can reduce power consumption by 25%–40%, improve throughput and reduce latency by 525%, while occupying 4%–13% more area when compared to the baseline architecture for both synthetic as well as real benchmark traces such as Princeton Application Repository for Shared-Memory Computers (PARSEC) and Standard Performance Evaluation Corporation (SPEC). CPU2006. When the energy-efficient buffer and crossbar organization was inserted into our CTorus topology, we further reduced energy dissipation by 32% and area by 53%, on average, over mesh2X, CMesh2X, and FBfly2X.**

***Index Terms*— Channel buffers, concentrated torus, crossbars, network-on-chip.**

I. Introduction

NETWORK-ON-CHIPS (NoCs) [1], [2] design paradigm overcomes the dual problem of global wire delay and scalability in chip multiprocessors by: 1) matching or reducing the wire lengths to network topology, and 2) increasing the bandwidth with more links and switches. As NoCs architecture (combination of links for communication and routers for storage and switching) gains traction with an increasing number of cores on a chip, power dissipation combined with excess leakage currents is already a major technology constraint that affects both performance (throughput and latency) and area overhead. While the previous design of 80-core Intel TeraFlops consumed more than 28% of the total chip power [3], more recent 48-core Intel SCC design [4] reduced the overall communication power to 10% of the total power budget by implementing several power optimization techniques. Clearly, energy-efficient and high-performance NoCs architectures are required to sustain and continue the performance gains achieved by increasing the number cores on a single chip with every successive generation.

Of the several research directions that improve the energy efficiency and performance in NoCs [5], we focus on three critical interrelated components, namely: 1) buffering; 2) switching; and 3) topology. As buffers consume substantial router power, several techniques to minimize the impact of the router buffers have been proposed. These include: 1) dynamic buffer with virtual channel (VC) allocation to maximize the buffer utilization for various packet lengths [6]; 2) replacing the repeaters along the link to duplicate as hold and store (channel buffers) when desired, thereby reducing the total router buffers leading to power and area savings [7]; 3) replacing all buffers with elastic buffers along the link by replacing repeaters with flip-flops and implementing a handshaking protocol between buffers [8]; and 4) bufferless routers that either deflect or drop conflicting packets [9], [10], thereby reducing latency and power consumption. Crossbars have been the subject of evaluation for NoCs and researchers have proposed smaller segmented and split crossbars for

Manuscript received January 18, 2012; revised September 27, 2012; accepted October 18, 2012. Date of publication December 3, 2012; date of current version September 23, 2013. This work was supported by the National Science Foundation under Award ECCS-0725765, Award CCF-0915537, Award CCF-0915418, Award CCF-1054339 CAREER, and Award ECCS-1129010.

D. DiTomaso and A. K. Kodi are with the School of Electrical Engineering and Computer Science, Ohio University, Athens, OH 45701 USA (e-mail: dd292006@ohio.edu; kodi@ohio.edu).

R. Morris was with Ohio University, Athens, OH 45701 USA. He is now with Intel Corporation, Portland, OR 95630 USA (e-mail: rm700603@ohio.edu).

A. Sarathy was with the University of Arizona, Tucson, AZ 85721 USA. She is now with MIPS Technologies, Sunnyvale, CA 94085 USA (e-mail: ashwini.arizona@gmail.com).

A. Louri is with the Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ 85721 USA (e-mail: louri@ece.arizona.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2012.2227283

improved energy and area efficiency [11]. Lastly, there have been several topologies that have improved throughput and latency while reducing power. Concentrating cores has shown to be an effective way to maximize the performance by trading off serialization latency for higher radix routers [12]. Concentrated mesh (CMesh) enables building networks with lower hop counts while increasing the sharing at the core routers with more traffic directed to the center of the network [12]. Flattened butterfly (FBfly) is another high-radix NoC router architecture that reduces any extra hops along a dimension, thereby restricting the diameter of the network to two at the cost of increased router radix [13].

In this paper, we propose to extend the energy efficiency and performance of channel buffers and router crossbars with the goals of minimizing power consumption, reducing head-of-line (HoL) blocking, and further improving network performance on a concentrated Torus (CTorus) topology. HoL blocking is caused when the first packet in a queue has a conflict that blocks a subsequent packet from moving forward. Dual networks can alleviate this problem, but this requires duplicating all functionality of routing and switching. This paper takes a unique route of building duplicate paths between routers within a single network to alleviate HoL blocking and propose three channel buffer organizations; they include four repeater stages per VC (4S), two stages per VC (2S), and 1 stage per VC (1S) organizations. Each of these channel buffer organizations increases the number of inputs and thereby provides perspective into the speedup offered. While prior work has duplicated network links similar to the 4S design, none of the prior work has proposed the use of dual channels with multiinput (2S) or single channel multiinput (1S). The design of (2S) and (1S) have similar goals of reducing HoL while there is an area tradeoff.

With dual input ports, there are multiple ways of organizing the crossbars to take advantage of the speedup offered with different routing and allocation mechanisms; they include dual input single crossbar (1XB), dual crossbar (2XB), and multiple crossbar (4XB) organizations. The 1XB organization uniquely shows how a 1XB can be organized to take advantage of dual inputs to increase the matching efficiency with fairness. The 4XB organization proposed in this paper shows how to minimize the crossbar sizes while taking advantage of adaptive routing. While the 2XB design has been proposed previously [14], the 1XB and 4XB are unique contributions of this paper. Finally, we choose the most energy-efficient channel buffer (4S) and crossbar organization (4XB) and implement the same on a concentrated Torus (CTorus) topology, which balances the load better than CMesh and provides performance comparable to FBfly topology. The implementation of the dual-channel 4S and 4XB organizations has the performance of a dual network without the additional area overhead. The use of dual channels has been shown to increase throughput and lower latency [12], [15], [16]. Therefore, we compare CTorus to the following dual networks: mesh2X, CMesh2X, and FBfly2X. We used the Synopsys design compiler to evaluate the power, area, and router pipeline latencies for various configurations. Our results indicate the router pipeline to be within the design tolerances for 2-Ghz router clock at 1.0 V and consuming 25%–40% lesser power while occupying 5%–13% excess area for different designs. Cycle accurate network simulation on an 8×8 mesh network topology shows 10%–25% improvement in performance for different synthetic as well as real traffic traces when compared to the baseline with identical router buffers. Moreover, the proposed CTorus topology shows up to 56% power savings and occupies approximately 47%–64% lesser area while improving energy–delay product (EDP) from 29% to 37% over CMesh2X and FBfly2X topologies. As traffic patterns have a significant effect on the performance of the network [17], we evaluate the CTorus topology on different types of traffic loads. The major contributions of this paper are as follows.

1) We propose an adaptive channel buffer design along with uniquely identifying different channel buffer organizations that reduce HoL blocking, thereby preventing performance degradation without duplicating networks.
2) We show the design of a 1XB that can take advantage of the speedup offered while maximizing the port occupancy along with multicrossbar designs that can improve performance with minimal adaptive routing.
3) We evaluate the proposed buffer and crossbar organizations on synthetic and real applications (PARSEC [18] and SPEC CPU2006 [19] benchmarks), showing a performance improvement of 10%–25% and power savings of 25%–40% with an area overhead of 5%–13%. Using the best of channel buffer and multiple crossbar organizations in a CTorus topology, we show an average saturation throughput improvement of approximately 17%, an average power reduction of 32%, and an average total area reduction of 53%, when compared to other NoC topology such as CMesh2X and FBfly2X.

II. Related Work

Table I summarizes the proposed method compared to related work for both buffers and crossbar designs. In [20], the authors showed a design space exploration of the three different channel buffer organizations, namely, 4S, 2S, and 1S. The 4S organization increases the inputs in the traditional method, 2S further increases the design by increasing the number of buffers, and 1S trades off some of the benefits by reducing the area overhead by utilizing a single channel and then increasing the channel buffers. Along with the channel buffer organizations, three crossbar organizations are explored: 1XB, 2XB, and 4XB. Transmission gates are utilized and a 1XB is used to take advantage of the speedup offered due to dual ports. The multicrossbar design shows the twin objectives of saving power with smaller crossbars along with increasing the throughput/performance with adaptive routing techniques. Each of these organizations improves the performance through the speedup offered by the dual inputs and provides varying power savings. While this paper in [20] has shown the benefits of buffer reduction, crossbar reorganization, and topology evaluation in isolation, there has been no design space exploration of buffer and crossbar organizations combined with a topology, which accentuates the advantages of the proposed router optimizations.

TABLE I
RELATED WORK COMPARED TO THIS PAPER

Buffer Design	Description	Advantages	Challenges
iDEAL [7]	Tri-state link buffers	Reduced power and area	No HoL avoidance
ECB [8]	Flip-flop link buffers	HoL avoidance	Performance limitations
FlitBLESS [9] and SCARAB [10]	Bufferless–deflects/drops packets	Reduced power and area	High-speed route computation logic
4S, 2S, and 1S [This Paper]	Multiple channel tri-state link buffers integrated in a dual CTorus network	Reduces HoL blocking, power, area, and performance	Each design explores different power, area, and performance trade-offs
Crossbar Design	**Description**	**Advantages**	**Challenges**
RoCo	Row/column crossbars	Small 2x2 crossbars	Restricted routing
Duato	Bisects output ports	2 separate crossbars	Focuses on high-radix crossbars
1XB, 2XB, and 4XB [This Paper]	Transmission gates or separate crossbars	1XB–performance 2XB, 4XB–power and area	1XB–Area 2XB, 4XB–limited routing

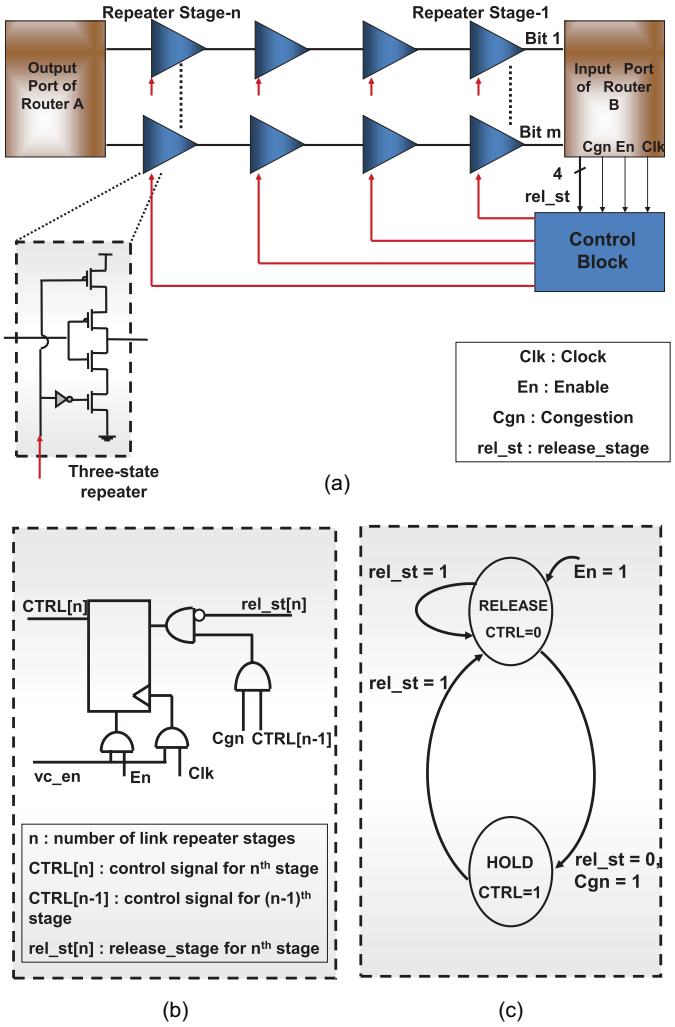


Fig. 1. (a) Link using three-state repeaters that function as channel buffers during congestion. (b) Control block implementation details. (c) State transition diagram.

III. ADAPTIVE CHANNEL BUFFERS

In this section, we detail the implementation of the adaptive dual-function links and the associated control logic. Conventional repeaters are inserted into links to quickly propagate voltage signals along the link. By adding two more transistors, the conventional repeaters become three-state repeaters as shown in the inset of Fig. 1(a). Repeaters have been shown to function with a tri-state design in [7], [8], and [21]. A single repeater stage is comprised of a three-state repeater inserted segment along all the bit wires in the link. Fig. 1(a) shows n repeater stages for an m bit link. Each stage is controlled by a control block which indicates when to repeat and when to hold a signal. When the control input to the repeater stage is low, the three-state repeaters in that stage function like the conventional repeaters transmitting data. When the control input to the repeater stage is high, the repeaters in that stage are tri-stated and hold the data bit in position. Therefore, these links have the dual function of adaptively switching between both repeating and storing a signal. The adaptive dual-function links, hence, enable a decrease in the number of buffers within the router and saves appreciable power and area. The design shown in Fig. 1(a) requires a single control block per inter-router link in order to control all the repeater stages along the link, unlike the design in [8] which uses one control block per stage along the link. Therefore, our proposed control technique is power-efficient and has a lesser area overhead compared to the design in [8]. Fig. 1(b) and (c) shows the control logic and the state diagram for one stage within the control block. The output signal, CTRL[n], is generated with only one flip-flop and four gates where n is the nth stage. The control logic determines the nth output based on the "release_stage" ("rel_st") and congestion (Cgn) signals from the router as well as the control signal from the $n-1$ stage (CTRL[$n-1$]). The control block operates with two logic states: Release and Hold. When there is congestion at the input router, each repeater is successively tri-stated to hold the data in position. The router can then request that the control block release any given repeater stage, by setting the corresponding bit in the release_stage signal. The vc_{en} signal is used in conjunction with the switching control to indicate to the control block the onset of a flit into the repeater stage.

IV. CHANNEL BUFFER ORGANIZATIONS

In this section, we propose three channel buffer organizations: four repeater stages per VC (4S); two stages per VC (2S); and 1 stage per VC (1S). Fig. 2 shows the configurations for one link between the upstream and downstream router. Each packet is composed of four flits with each flit being 128 b.

A. Four-Stage Per VC Channel Organization

Fig. 2(a) shows the 4S buffer configuration. This organization has two input ports (registers) at the downstream router. The dual-input ports act as VCs to give packets two paths that can be used to alleviate HoL blocking. The two inputs are shown as I_0 and I_0'. Each VC has four dedicated repeater stages. Once a flit is read into the register, it activates the control block (CB0) or (CB1) to indicate a full register. As explained before, the control block will then hold flits one cycle after another into different channel buffers associated with the particular control block. The control block keeps track of which channel buffers are occupied by communicating with the demultiplexer (DEMUX) at the upstream router. When all the channel buffers are occupied, it will then signal the upstream switching control to indicate a full channel or congestion. When all the channel buffers are occupied for a particular VC, the switching control will deactivate the channel buffer from receiving any more flits until the control block releases the congestion. The flit read into the register undergoes the standard router pipeline stages of route computation (RC), VC allocation, switch allocation (SA), and then switch traversal (ST), before moving on to link traversal. Here, we combine RC and VC into a single stage, giving us a four-stage router pipeline. Prior elastic buffer designs have eliminated the VC stage, thereby simplifying the channel buffer design and reducing the router pipeline. However, we retain the VC stage, as we have two channel buffer links to choose from. Moreover, this provides the opportunity to provide different classes of service for different packets. Once the flit is in the ST stage, we transmit the VC allocation information (0 or 1 as there are two VCs) along with the flit to the switching control to set the DEMUX to the appropriate channel buffer link. The 4S buffer organization reduces the HoL blocking, providing differentiated classes of service while also ensuring sufficient buffering to improve the throughput.

B. Two-Stage Per VC Channel Organization

Fig. 2(b) shows the two stages per VC (2S) organization. This organization increases throughput over the 4S organization by using four VCs each with two channel buffer stages. By giving each input port an additional VC, packets are given more paths to further alleviate HoL blocking and reduce congestion at the same input port. Each VC has a separate control block for a total of four CBs. The switching control at the upstream router allocates packets to a VC based on the congestion information from the control blocks. We use two sets of 2-to-1 DEMUXes to reduce the area overhead due to aligning the channel buffers as shown. The 2-b VC information is sent with the flits and is used to select the correct DEMUX output line. The objective of this organization is to relieve congestion while saving power and minimizing the increase in area overhead.

C. One-Stage Per VC Channel Organization

Fig. 2(c) shows the 1S organization. This organization has one repeater stage dedicated for each VC and three repeater

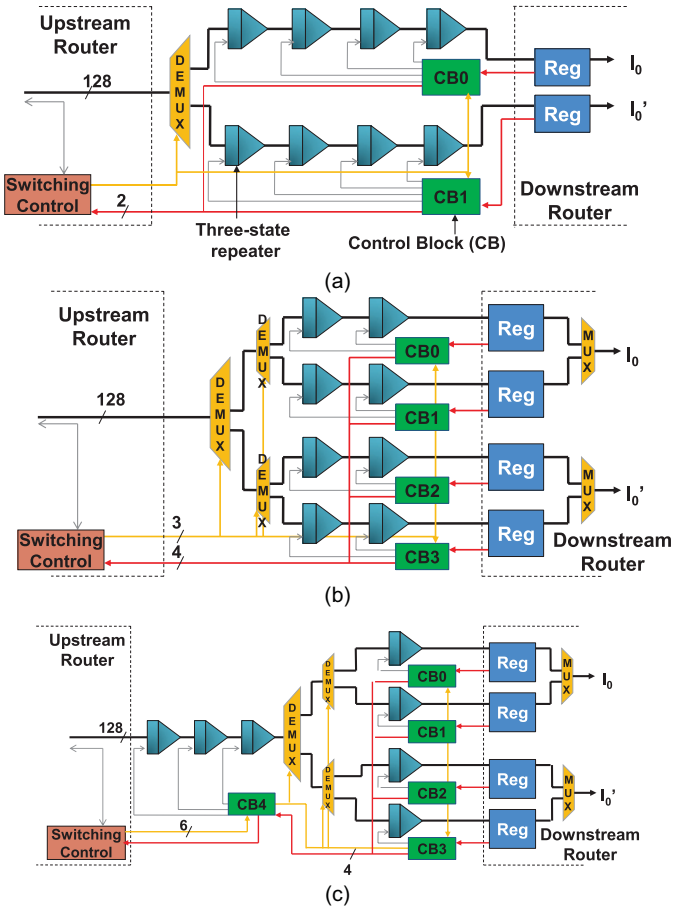


Fig. 2. (a) Four-stage (4S) channel buffer organization. (b) Two-stage (2S) channel buffer organization. (c) One-stage (1S) channel buffer organization.

stages shared between all VCs. The additional VC at the input port reduces congestion; however, the shared stages do not alleviate HoL blocking as much as 2S. The four control blocks, CB0 to CB3, are needed for the four VCs as before. An additional control block, CB4, is fed congestion information from the other four. CB4 releases the flit in the shared stages only if CB0 to CB3 release the congestion signal. This design provides a tradeoff between performance and area overhead due to the stacking of the channel buffer at the end of the link.

V. Crossbar Organizations

The dual inputs from the buffer should be utilized to further increase the throughput of the network. To that end, we propose three crossbar organizations with different routing and allocation mechanisms: the dual-input 1XB, the 2XB, and the 4XB.

A. Dual-Input 1XB

Fig. 3(a) shows the 1XB crossbar (1-b). The 1XB crossbar allows twice as many inputs as outputs. The input lines of a conventional matrix crossbar are modified to allow electrical signals to travel in two directions. Transmission gates are placed on the input lines as shown in Fig. 3(b). Two three-state repeaters facing opposite directions comprise one transmission

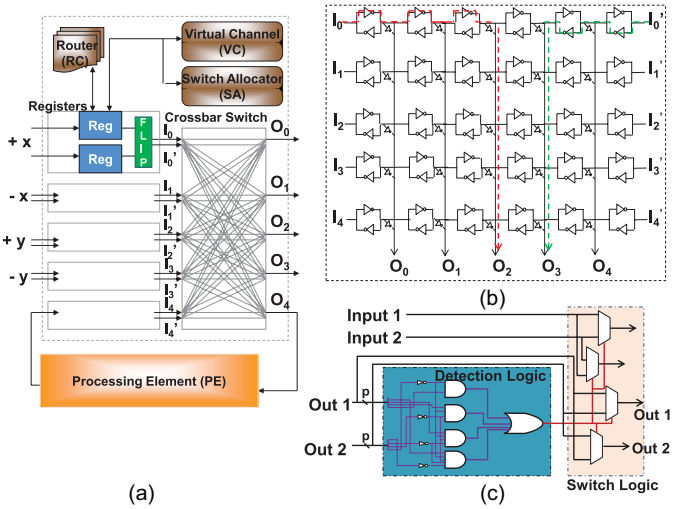


Fig. 3. (a) Dual-input single matrix crossbar (1XB) organization. (b) Example communication from dual inputs. (c) Flip logic.

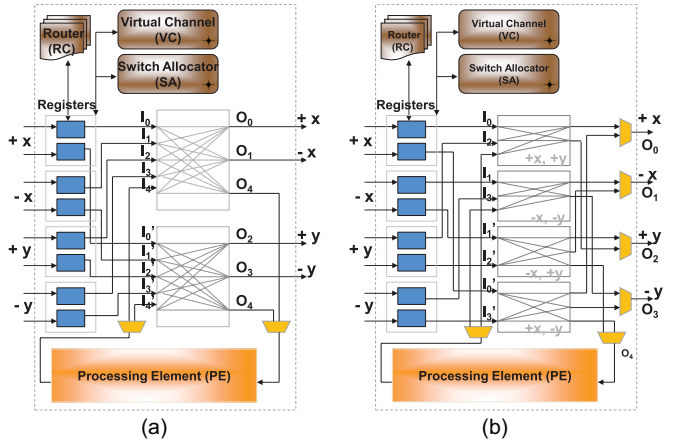


Fig. 4. (a) 2XB organization. (b) 4XB organization.

gate. The control lines of the three-state repeaters, therefore, allow or block an electrical signal from crossing from one side to the other. For example, if a high voltage signal is placed on the transmission gate, there is a conduction path from one side to another. On the other hand, if a low voltage signal is placed on the transmission gate, the electrical current is blocked by creating a segmentation of the crossbar input. For example, Fig. 3(b) shows multiple flits traversing the crossbar at the same time. A flit is allowed to traverse from I_0 to O_2 by turning on the three transmission gates on that path. At the same time, a flit can traverse from $I_0^{'}$ to O_2 by turning on the two transmission gates along that path. The transmission gate between O_2 and O_3 is deactivated so that the two flits do not interfere.

Since each input port has the potential for two different packets traversing across a crossbar, the standard switch allocation found in most routers must be augmented. In a separable output-first switch allocator, flits will proceed through two stages of arbitration [22]. During the first stage, the output ports are arbitrated. All the output ports requested from each pair of input ports (e.g., I_0 and $I_0^{'}$) are sent to separate arbiters. Each arbiter independently selects which input port is granted the right to traverse across the crossbar to one of the five output ports. In the second stage, the pair of input ports (e.g., I_0 and $I_0^{'}$) compete to see who will traverse the crossbar. Next, a third arbiter is used to select an additional packet for a different output port if the given input port was granted to two or more output ports. This arbiter allows multiple packets from the same pair of input ports to traverse to different output ports, as in the example in Fig. 3(b).

Since the arbiters can select a combination of output ports, this may cause a conflict. For example, if one arbiter selects O_3 for I_0 and the other arbiter selects O_3 for $I_0^{'}$, then there will be a conflict. To compensate for these situations, we add extra logic after the switch allocation to detect whether a conflict arises. If a conflict arises, we switch the two packets so that the packet originating from I_0 is switched to $I_0^{'}$ input and vice versa, thereby enabling forward progress by both the packets. Fig. 3(c) shows the logic used to evaluate and detect a conflict between the two inputs from the same port. As can be seen from the figure, the conflict detection logic is divided into two different stages. In the first stage, a conflict is detected by comparing the requested outputs of the two input ports (I_0 and $I_0^{'}$). After the detection logic, the signal will be an input for four multiplexors which will select the corrected conflict-free combination. The 1XB design has more overhead: power as well as latency for additional logic. However, 1XB design with dual input can provide consistently better performance, and different routing algorithms can be easily implemented because of full connectivity.

B. 2XB Organization

The 2XB, shown in Fig. 4(a), splits the monolithic crossbar into two, each with a smaller number of output ports. This proposed 2XB has been well researched in several architectures [14], [23]–[25]. The dual 2×2 crossbar used in RoCo is aligned along the x and y dimensions, thereby reducing the area and power consumption. Another high-radix router [23] has similar functionality with the dual-input port feeding into two separate crossbars. The 2XB organization shown here is slightly different from the previous work, as we have a single register connected to the crossbars. This makes the VC allocation more restrictive with the direction in which we expect the packet to turn. For example, with dimension order routing (DOR), a flit being routed in the x dimension will always be allocated to the upper VC until a turn is needed. During and after the turn, the flit will be allocated to the lower VC so that it can traverse the crossbar with the y outputs. The 2XB organization reduces the power consumption and area overhead while delivering performance proportional to the dual-input crossbar. Because of the single register storage, this design limits the VC allocation during turns.

C. Multicrossbar Organization

Fig. 4(b) shows the multicrossbar organization which splits the crossbar into four smaller crossbars to reduce area and power consumption. The division of the four crossbars are along the four quadrants: $(+x, +y)$ [North–East], $(-x, -y)$

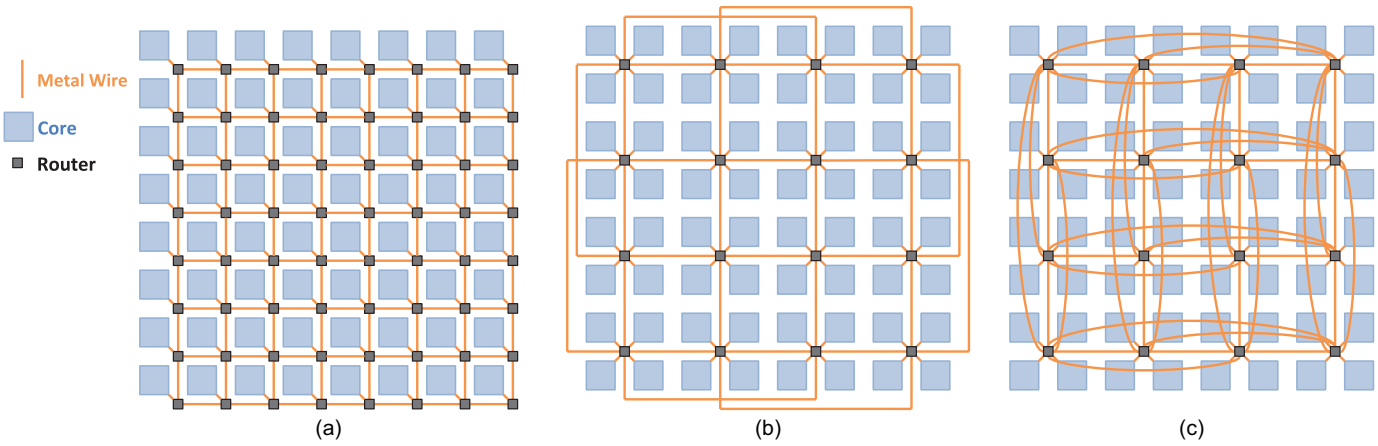


Fig. 5. (a) Mesh. (b) CMesh. (c) FBFly topologies.

[South–West], $(-x, +y)$ [North–West], and $(+x, -y)$ [South–East]. The four quadrants represent the four directions with dedicated channels such that adaptive packet flow can be implemented. Communication along the quadrant chooses the crossbar designed for the direction. Suppose the packet arrives from the $+x$ direction into I_0. This packet can be routed to either O_0 ($+x$ direction) or O_2 ($+y$ direction) using the North–East crossbar. Similarly, if the packet arrives from the $+x$ direction from I_0' into the South–East crossbar, then the possible outgoing directions will be O_0 and O_3. Therefore, by limiting the crossbar connections and combining select crossbar outputs, we provide more opportunities for the output ports to be occupied than a conventional crossbar. The VC allocation is more flexible than the previous approach. The VC allocation is based on how many hops away the packet is from the destination. If the packet is more than one hop away from the destination in either dimension, then the packet can be allocated to either VC. If the packet is exactly one hop away from the destination in a particular dimension, then the lower VC should always be allocated. With this simple restriction, we can use both the VCs and connect using different crossbars to get to the same direction. The availability of VC guarantees that the load will be lower in the specified direction. This also allows the packet to be adaptively routed along the minimal dimension. Deadlocks are naturally avoided, as there are always two VCs available. Furthermore, as the packets traverse specific quadrants $(+x, +y)$, $(-x, -y)$, $(-x, +y)$, and $(+x, -y)$ to reach the destination, there are no circular dependencies that could potentially lead to deadlocks. XY DOR is used to restrict the turns a packet can make by first routing in the x-dimension, then the y-dimension. Much research, orthogonal to that in this paper, has studied protocol deadlocks avoidance by using escape VCs or routing/VC allocation restrictions [26]. For this paper, we simply assume protocol deadlock is avoided by splitting into two sets of VCs: one set is allocated for requests and the other for responses. Therefore, the multicrossbar configuration provides the best of the three worlds: lower area due to split crossbars, lower power dissipation due to shorter path lengths, and higher throughput due to selective merging of different output ports.

VI. Topology

Some leading topologies for NoCs include mesh, concentrated mesh (CMesh), and FBfly. These topologies are shown in Fig. 5. The mesh network topology has a router at each processing core. The routers are connected in a grid fashion in which each router is connected to four neighboring routers. Each router, except those around the edges of the grid, has one input and output port for the cores as well as four input and output ports for the four directions $+x, -x, +y$, and $-y$. The mesh topology allows for quick communication between neighboring cores, but there is a high hop count which increases the network diameter [27]. The CMesh topology has four cores concentrated to one router. The routers are also connected in a grid fashion, but there are extra links around the edges which skip over one router. The CMesh routers have four ports for the four cores as well as four ports for the four cardinal directions. CMesh offers a lower hop count, allowing lower packet latency. However, the multiple cores connected to the same router may cause contention as packets enter and leave the same input and output ports. The FBfly topology also uses a concentration of four cores, although routers in the same x and y dimension are fully connected. Concentration is a technique in which multiple cores are connected to the same router to reduce router and buffer overhead [12]. This further reduces the hop count of the network. However, the router area does not scale well since there are four ports for the cores and ports for each of the other routers in the x and y dimensions. In addition to this high-radix router, the cost in wires increases area and power dissipation.

We propose a CTorus topology using the 4S buffer and 4XB organizations. Torus topology balances the traffic load better than a mesh because of wrap-around links, allowing packets to travel in both directions and, thereby, reducing the traffic contention at the center of the network. Concentration of the cores provides the added advantage of reduced hop count, thereby leading to savings in power and area overhead. Moreover, due to the reduced crossbar complexity (4XB configuration), we can further reduce the router complexity when compared to FBfly topology. As explained in the next section, we use the combination of 4S+4XB, as this is the most

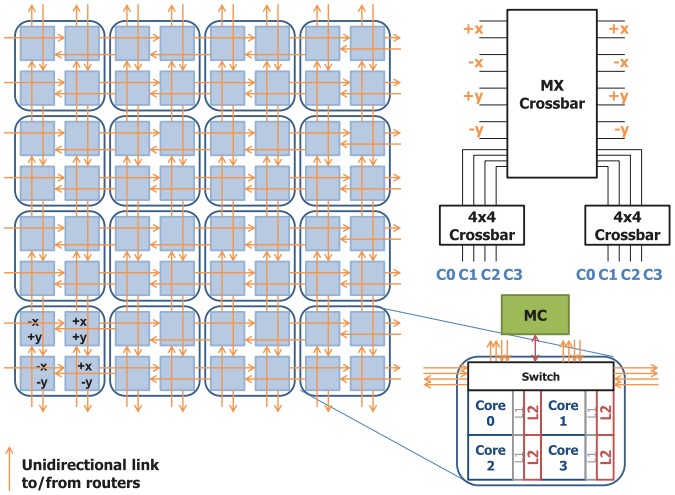


Fig. 6. CTorus topology and router design using the 4XB crossbar organization.

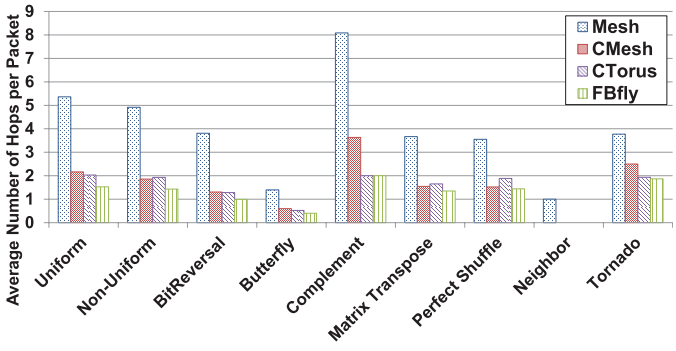


Fig. 7. Average hop count per packet for different synthetic traffic loads.

area- and power-efficient combination. The CTorus topology is shown in Fig. 6 and uses a concentration of four cores. The arrows around the edges of the topology are links which wrap around the opposite edge and are shown like this for simplicity. We have found that the longest distance we can transmit a flit within our 2-GHz clock frequency is 5 mm. Therefore, we assume that a flit takes three cycles to propagate along the long 15-mm wrap-around links by splitting the link into three sections of 5 mm. The EDA flow of this design can be challenging, but similar links are used in the FBfly architecture [13]. Each router has four inputs and outputs for each of the four directions: $+x$, $-x$, $+y$, and $-y$. Since the 4S buffer organization is used, each router has two links for each direction. To accommodate for the concentration of four cores, the 4XB crossbar organization changes slightly. Instead of two 3×2 and two 2×3 crossbars, we now need four 3×3 crossbars so that each core can go to a crossbar. Additionally, instead of 2 to 1 multiplexers and demultiplexers at the cores, we must use two 4×4 crossbars at the cores, as shown in Fig. 6. The figure also shows the logical connection between the cores. Each core in the concentration is given one of four quadrants: $(+x, +y)$ [North–East], $(-x, -y)$ [South–West], $(-x, +y)$ [North–West], and $(+x, -y)$ [South–East]. Each quadrant has a dedicated channel between each router. Fig. 7 shows the average hop count for a packet to reach its destination. Mesh has the highest average hop count for all traffic traces, due to the network diameter of 14. CMesh and CTorus have similar average hop counts, with CTorus slightly lower overall due to the long wrap-around links. The high-radix routers of FBfly cause this topology to have the lowest hop count. Note that, for neighbor traffic, the hop count is 0 for concentrated networks because all communication happens between the concentrated cores. The low average hop count of CTorus along with the implementation of the 4S and 4XB organization will cause CTorus to have high performance and low power with minimal area overhead. CTorus represents a dual network in that it has two redundant links between routers. Dual networks are created by duplicating the NoC routers and links so that packets have more resources. For this reason, we compare CTorus to mesh2X, CMesh2X, and FBfly2X, which duplicate routers and links. CTorus represents a dual network, but without the need to duplicate routers. The inset of Fig. 6 shows how the cores, caches, and memory controllers (MCs) are connected. Each core has a private L1 and private L2 cache. Each L2 cache is connected to the switch. From the switch, communication can go to the MCs, which are located around the edges of the chip, or to other core routers.

VII. Performance Evaluation

In this section, we evaluate our proposed channel buffer and router crossbar organizations in terms of power dissipation, area overhead, and overall network performance, and compare them with a baseline VC router. We consider each router with a four-stage router pipeline (baseline and all proposed approaches) as discussed before. Each router has $P = 5$ input ports (four for each cardinal direction: North, South, East, and West, and one for the PE). For a fair comparison, we consider two baseline designs with two VCs and four VCs per input port with each VC having four flit buffers in the router for a total of 40 and 80 flit buffers, respectively. For synthetic traffic, each packet consists of four flits where each flit is 128 b for a total of 512 b per packet. In synthetic traffic, messages are passed from point to point. For real applications, a mixture of short and long packets were used. A one-flit request packet was sent from the source to a destination core based on the traffic traces and a four-flit response packet was sent back. In real application traffic, cache coherent messages are directed by the application, and the hardware simply ensures that the messages travel from the source to the destination. We implement a sequential memory consistency model where flits are ordered based on application demand. Sequential consistency is maintained at the processor/memory level and is outside the scope of this paper. Every combination of channel buffer and crossbar organization as well as the links and routers were synthesized and optimized using the Synopsys design compiler tool using the TSMC-LPBWP 65-nm technology library with a nominal supply voltage of 1.0 V and an operating frequency of 2 GHz. We also evaluate our CTorus topology using channel buffers and the 4XB crossbar and compare it with mesh2X, CMesh2X, and FBfly2X. For equal comparison, the bisectional bandwidth was maintained equal for all designs by adjusting the link width

TABLE II
BISECTIONAL BANDWIDTH OF EACH ARCHITECTURE SIMULATED IN THE RESULTS

Architecture	Number of Unidirectional Links	Link Width (bits)	Bisectional Bandwidth (bits)
Mesh2X	32	128	4096
CMesh2X	32	128	4096
FBfly2X	64	64	4096
CTorus	32	128	4096

in bits. Table II shows the bisectional bandwidth and the link widths of each architecture. Since FBfly2X has twice as many links, the width of each link is halved. This is simulated by adding cycle delays in FBfly2X links. Moreover, router to router distance was assumed to be 5 mm. Additional cycle delays were added to account for links of 10 and 15 mm in length. Addresses are mapped to cores in a Cartesian grid with the origin starting in the lower left corner. DOR is used in which packets are first sent in the x direction then in the y direction. This type of routing avoids deadlocks, as it restricts the types of turns that packets can make. For all networks, there are 64 cores with private L1 instruction and data caches. There are 16 memory controllers, each attached to the core router.

A. Power, Timing, and Area Estimation

The power per segment of the repeater-inserted link is given by $P_{\text{segment}} = P_{\text{dynamic}} + P_{\text{leakage}} + P_{\text{short-ckt}}$, where P_{dynamic} is the switching power, P_{leakage} is the power due to the subthreshold leakage current, and $P_{\text{short-ckt}}$ is the power due to the short-circuit current. The power per segment is multiplied by the number of segments and the link width to obtain the total link power dissipation for a flit traversal. When a conventional repeater is replaced by a three-state repeater, there is an additional capacitance due to the added transistors, as shown in Fig. 1. The increase in the switching capacitance increases the total power consumed by the links. Power is also dissipated in the control blocks controlling the dual-function repeater stages, when they are enabled during congestion. In calculating the power values, the inter-router links are assumed to be 5 mm long. The buffer organizations considered are 4S, 2S, and 1S; the crossbar organizations considered are 1XB, 2XB, and 4XB. Therefore, this provides us with nine different architectures with different naming conventions (e.g., 4S-2XB implies four-stage channel buffers with 2XB), which are compared to the baseline, which is the two VC router. This keeps the number of buffers the same across different designs. Table III shows the power and area overhead of each router design in 65 nm-technology.

1) Power: As Fig. 8 shows, the majority of the power consumption is in the links. This power is equal in all designs because of the fixed wire length of 5 mm. The baseline input buffers were implemented with 128-b FIFO registers that were found to have a power of 2.78 mW using Synopsys. Overall, the channel buffers consumed approximately 24% less power because of the low power three-state repeaters which were found to have a power of 0.1325 mW each. This

TABLE III
POWER AND AREA ESTIMATION USING SYNOPSYS DESIGN COMPILER FOR 65-nm TECHNOLOGY NODE AT 1.0 V AND 2-GHZ CLOCK

Design	Power (mW) Buf + xbar	% Diff	Area (mm^2) Buf + xbar	% Diff
Baseline	91.30 + 13.56	–	0.248 + 0.0356	–
4S-1XB	66.60 + 16.10	−21	0.272 + 0.0471	+12
4S-2XB	66.60 + 8.19	−29	0.272 + 0.0246	+4
4S-4XB	66.60 + 5.95	−31	0.272 + 0.0237	+4
2S-1XB	66.40 + 16.10	−21	0.274 + 0.0471	+13
2S-2XB	66.40 + 8.19	−29	0.274 + 0.0246	+5
2S-4XB	66.40 + 5.95	−31	0.274 + 0.0237	+5
1S-1XB	66.56 + 16.10	−21	0.274 + 0.0471	+13
1S-2XB	66.56 + 8.19	−29	0.274 + 0.0246	+5
1S-4XB	66.56 + 5.95	−31	0.274 + 0.0237	+5

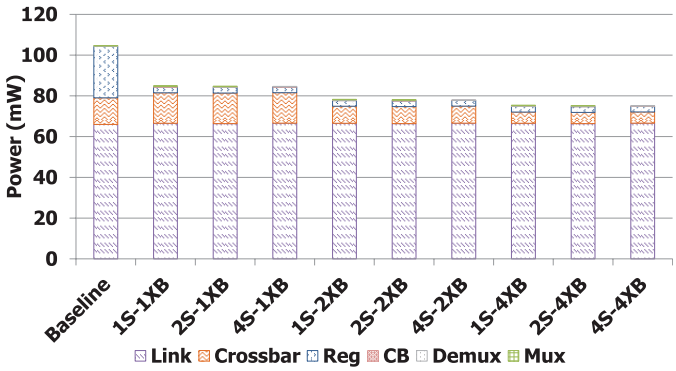


Fig. 8. Dynamic power breakdown for different design choices.

difference in power, shown as registers (reg) in Fig. 5, is the cause of the large power savings of the channel buffer designs. The 4S design with the 4XB crossbar showed the best reduction at 28.2% compared to the baseline, whereas 1S with the 1XB crossbar had the least power reduction at 18.9%. The small difference in power between the different channel buffer designs is due to the different number of multiplexers and demultiplexers used. For the crossbars, the power values calculated by Synopsys were lower for the 4XB crossbar because the total distance for a flit to travel is smaller in the 4XB compared to the larger 2XB and 1XB crossbars. In smaller crossbars, the number of wire taps is less, so the input and output lines are shorter compared to large crossbars. Therefore, the distance the flit travels from input to output is less. The large savings in power allowed the channel buffers to have more flexibility with the crossbars while maintaining a significantly lower overall power compared to the baseline.

The network energy can be described as $E_{\text{net}} = E_{\text{ch}} + E_r$, where E_{net} is the network energy, E_{ch} is the channel energy, and E_r is the router energy. E_r consists of the energy dissipation for each of the routing stages: buffer write, VC allocation, SA allocation, RC computation, and switch traversal, i.e., $E_r = E_{\text{bw}} + E_{\text{VC}} + E_{\text{SA}} + E_{\text{RC}} + E_{\text{ST}}$. As shown in [12], E_{ch} can be further described as $E_{\text{ch}} = wM(E_{\text{sq}} + NE_w)$, where w is the channel width in bits, M is the sequencing elements, N is repeaters per segment, E_{sq} is the energy dissipation in

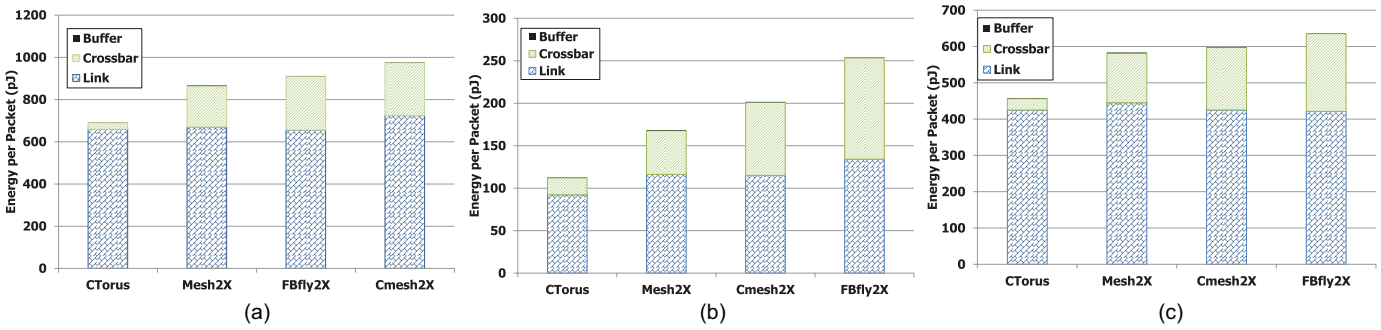


Fig. 9. Energy per packet for different traffic loads. (a) Complement. (b) Butterfly. (c) Uniform random.

the sequencing element, and E_w is the energy dissipation of the wire. Likewise, from [12], the width of a channel can be described as $W_{\text{ch}} = w \times p_M \times s_w$, where w is the width of the channel, p_M is the metal pitch, and s_w is the average wire spacing. The width (W_R) and height (H_R) of a 5×5 router can be calculated from the size of the input module (IM), output module (OM), and spacing (x) by $W_R = W_{\text{ch}} + 5H_{\text{IM}} + 2H_{\text{OM}} + W_x$ and $H_R = 3H_{\text{OM}} + H_x$.

For topology, Fig. 9 shows the energy per packet for a certain traffic pattern for the CTorus, mesh2X, CMesh2X, and FBfly2X topologies. For a fair comparison, all architectures use channel buffers. The 4XB crossbar is implemented only in the CTorus design because it cannot be implemented in high-radix routers such as FBfly2X. The energy is broken down to link, crossbar, and buffer energy dissipation. The three traffic patterns shown are complement, butterfly, and uniform random. For each load, the link energy per packet is similar across all topologies. This is because the distance a packet must travel from the source to the destination is independent of the topology. The slight differences are due to concentration and the total number of packets sent. Each link consumes 6.65 pJ/mm for a 128-b link in 65-nm technology. In each traffic pattern, CTorus has lower total energy dissipation per packet. This savings is due to the smaller crossbars used and long wrap-around links which skip over intermediate routers. As the crossbar size increases, the number of wire taps as well as the distance the packet must travel increases for both the input and output lines. Therefore, using two smaller crossbars can be more energy efficient than one large crossbar. For example, CTorus uses one 3×3 crossbar at intermediate routers and one 3×3 crossbar plus one 4×4 crossbar when the packet is at the source and destination. This corresponds to a crossbar power of 7.0 pJ per packet at intermediate routers and 17.4 pJ at the source and destination. The energy dissipation for an equivalent 8×8 crossbar is 28.44 pJ. Therefore, a four-hop packet in CTorus saves 86.3 pJ in crossbar traversals compared to the 8×8 crossbar in CMesh2X. Additionally, channel buffers contribute to the energy savings. The energy of four 128-b tri-state buffers used as channel buffers is 0.265 mW. The buffer energy is significantly lower than the crossbar and link in all topologies due to the low energy channel buffers implemented in each design. CTorus saves energy in traffic patterns with long-distance communication such as complement traffic because the long wired link allows packets to skip over immediate routers. While the FBfly2X topology also has long links, it also has a large high-energy 10×10 crossbar. Overall, the torus also saves energy for short and medium distance communication as seen in the butterfly and uniform traffic patterns because there are savings at each router.

2) Timing: The latency for the baseline, 4S, 2S, and 1S designs was found to be 0.47, 0.37, 0.44, and 0.46 ns, respectively. These latencies, which were due to the buffering, were all within our specified clock period of 0.50 ns. The small differences in the critical paths of the channel buffer designs were due to the different number of repeaters, demultipexers, and multiplexers that a flit had to travel through in each design. The latency of four three-stage repeaters was found to be 0.20 ns and the latency of the demultipexer and multiplexers was found to be 0.08 ns each. Additionally, the latency for the baseline, 2XB, and 4XB crossbars alone were 0.35, 0.39, and 0.39 ns, respectively. These were due to the critical path of the logic in the VA stage. The latency for the 1XB crossbar was largest at 0.47 ns due to the extra logic needed to switch the VC input flits. This accounts for switching on all the transmission gates that are needed to connect the input and the corresponding output ports.

3) Area: Area overhead of the baseline VC2 router obtained from Synopsys is 0.283 mm^2, which includes the buffer and crossbar. All proposed designs occupy slightly more area compared to the baseline due to the increase in link width as shown in Table III. The total area for each channel buffer design is due to the wires, registers, and control blocks because the repeaters and wires use different metal layers [28]–[31]. For area optimization of the channel buffers, the link will not be split into separate channels or inputs until the end of the link. This optimization causes the wire to remain a single 128-b wire for most of the link. However, an increase in the number of repeaters on the link will occur. This will slightly add to the overall power, but allows a significant reduction in area. The 4S was assumed to be a single 128-b wire for 0.5 mm, and then split into two parallel channels for the remaining 0.5 mm, causing the total wire length be 1.5 mm. Similarly, the 2S and scM were assumed to be single 128-b wire for the first 0.875 mm. The lengths were determined in order to offer the best area optimization while also limiting the additional power added by the repeaters. In the 4S buffer, the two registers and control blocks on the two channels reduced

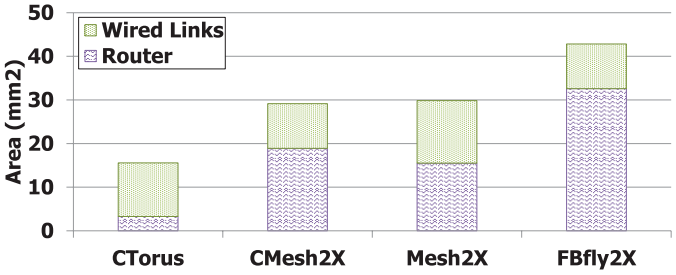


Fig. 10. Router and link area overhead of each topology.

the area overhead. The combination of this channel buffer and the 4XB crossbar had the least area overhead of only 0.295 mm^2. The smaller 2×3 and 3×2 crossbars in the 4XB crossbar result in a lower area for all channel buffer designs. The multiple inputs in the 2S along with the size of the 1XB crossbar resulted in an area of 0.322 mm^2, which was the largest. The area overhead of the 2S and scM designs are the same because the total wire lengths are the same.

For topology, Fig. 10 shows the router and link area overhead of each topology. The torus saves approximately 47% area over CMesh2X, 48% area over mesh2X, and 64% area over FBfly2X. The 8×8 crossbar used in CMesh2X was estimated from Synopsys to have an area of 0.590 mm^2. Since CMesh2X is a dual network, the network requires two 8×8 crossbar. Using four 3×3 crossbars and two 4×4 crossbars in CTorus reduces the crossbar area overhead by 83%. This large savings is due to the dual links in the 4S channel buffer design and the 4XB crossbar, which creates a dual network without the overhead of doubling router and link components. With equal bisectional bandwidth, each topology has similar link area overheads. Each 128-b link occupies 0.0256 mm^2 for every 1 mm length, estimated from Synopsys. The dual high-radix router as well as the many wired links in the FBfly2X topology cause the total network area to occupy over 40 mm^2 of the 400 mm^2 chip. While the mesh2X topology has lower radix routers, the area is still almost 30 mm^2 since a router is needed at every core, which increases the total number of routers compared to concentrated networks. CMesh2X occupies approximately 29 mm^2; however, the area is still over $2\times$ higher because of the large crossbar needed in the duplicate routers.

B. Simulation Methodology

A cycle-accurate on-chip network simulator was used to conduct a detailed evaluation of the proposed channel buffer and router crossbar designs in a 8×8 mesh network. We consider five designs out of nine, as they represent the best design choices: 4S-2XB, 4S-4XB, 2S-4XB, 4S-1XB, and 1S-4XB. The proposed designs were compared to a 2 VC and 4 VC router buffer with a standard 5×5 crossbar. For open-loop measurement, the packet injection rate was varied from 0.1 to 0.9 of the network capacity, and packets were injected according to the Bernoulli process based on the given network load. The simulator was warmed up under load without taking measurements until steady state was reached. Then a sample of injected packets was labeled during a measurement interval.

TABLE IV
CORE AND CACHE PARAMETERS USED FOR PARSEC AND SPEC2006 APPLICATION SUITE SIMULATION

Parameter	Value
L1/L2 coherence	MOESI
L2 cache size/accoc	4 MB/16-way
L2 cache line size	64
L2 access latency (cycles)	4
L1 cache/associativity	64 KB/4-way
L1 cache line size	64
L1 access latency (cycles)	2
Core frequency (GHz)	3
Threads (core)	2
Issue policy	In-order
Memory size (GB)	4
Memory controllers	16
Memory latency (cycle)	160
Directory latency (cycle)	80

The simulation was run for 10 000 cycles for each network. All designs were tested with different synthetic traffic traces such as: 1) uniform random, where each node randomly selects its destinations with equal probability, and 2) permutation patterns, where each node selects a fixed destination based on the permutations. For permutation traffic, we evaluated the performance on bit-reversal, butterfly, matrix transpose, complement, and perfect shuffle. Evaluating synthetic traffic gives us a peek into the behavior of the network when encountering real benchmarks, as a mix of synthetic traffic will comprise a real benchmark.

For closed-loop measurement, we collected traces from real applications using the full execution-driven simulator SIM-ICS from WindRiver [32], with the memory package GEMS enabled [33]. We evaluated the performance on PARSEC [18] and SPEC CPU2006 [19] workloads. Table IV shows the core and cache parameters used for PARSEC and SPEC CPU2006 workloads. We assume a 2-cycle latency to access the L1 cache, a 4-cycle latency to access the L2 cache, and a 160-cycle latency to access the main memory. In addition, there are 16 memory controllers used to access the main memory and each processor can issue two threads. These parameters can be varied. However since the same parameters are used for each network, any variation will cause the results to change equally, yielding the same relative results. We consider seven PARSEC applications with medium inputs (blackscholes, facesim, ferret, fluidanimate, freqmin, streamcluster, ferret, and swaptions) and three workloads from SPEC2006 (bzip, gcc base, and hmmer).

C. Simulation Results and Discussion

1) Buffer and Crossbar Organization (Synthetic and Real Traffic Results): Fig. 11(a) shows the throughput plot for UR traffic. From the figure, 2S-4XB (two-stage channel buffer with 4XB) is the best performing network with a saturation throughput of about 0.37 or a 15% improvement over the

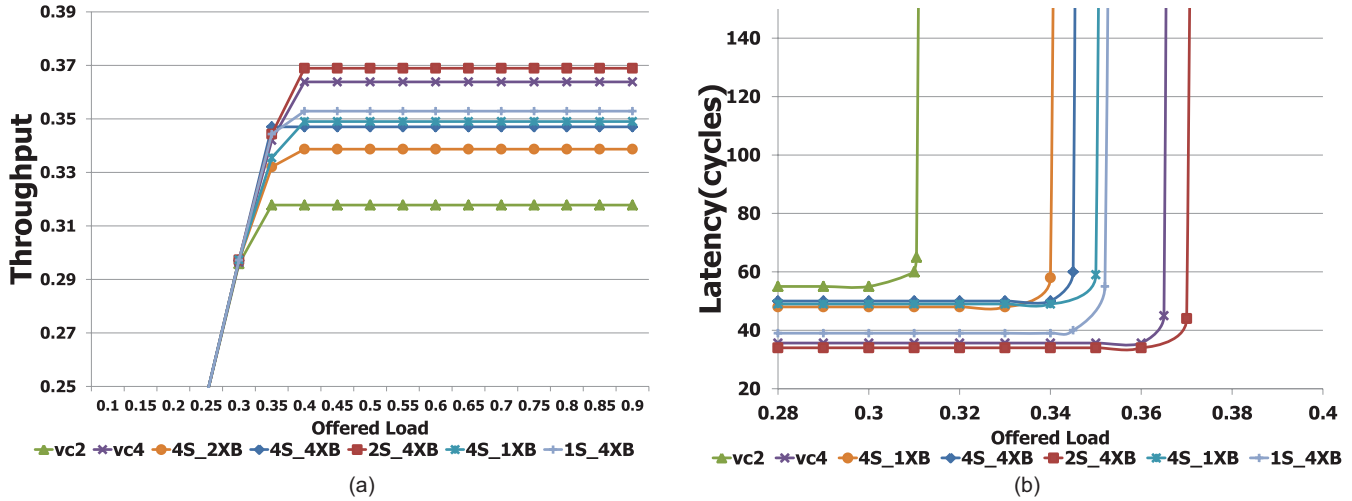


Fig. 11. Throughput and latency for different designs for uniform traffic.

baseline VC2. This results from the dual input of 2S where multiple flits from the same input port can traverse to separate output ports. In addition, there are four potential flits that are available to traverse the crossbar instead of only two flits found in the VC2 design. Also, 2S slightly outperforms the baseline VC4 design, where VC4 has two times more buffer space. The increase in performance is due to the dual-input nature of 2S, as both networks have the same number of flits available (four flits) to traverse the crossbar. 1S, 1XB, and 4S network designs have a saturation throughput of about 0.35 and have a performance improvement of about 10% over VC2 due to the dual inputs found in each router design. Lastly, 2XB has the least increase in performance over VC2 with a 6% improvement in performance. This reduction in overall performance over the other designs is due to the restricted dual-input crossbar found in 2XB. In 2XB, two flits can traverse the crossbars from the same input if the two flits are required to traverse to two different crossbars. Fig. 11(b) shows the latency plot for UR traffic. From the figure, 2S has the lowest zero load latency of about 34 clock cycles, followed by ScM with a latency of 39 clock cycles.

Fig. 12 shows the execution time speedup when normalized to VC2 configuration for PARSEC and SPEC CPU2006, respectively. From Fig. 12, the majority of PARSEC benchmarks (blackscholes, facesim, fluidanimate, ferret, and swaptions) show performance improvement of 10%–12% speedup when compared to VC2 baseline. It should be noted that the performance jump obtained from the real benchmarks is equal to, and in some cases even more than, a VC4 configuration. This clearly shows that with half the number of buffers (and VCs) and smaller crossbars, we can obtain the performance equivalent to what can be obtained with twice the number of buffers. For SPEC CPU2006 benchmarks, the performance jump from most of the combinations is above 10% and outperforms the baseline VC2. Clearly, the combined effects of channel buffer organizations and crossbar designs improve the performance for both synthetic as well as real applications.

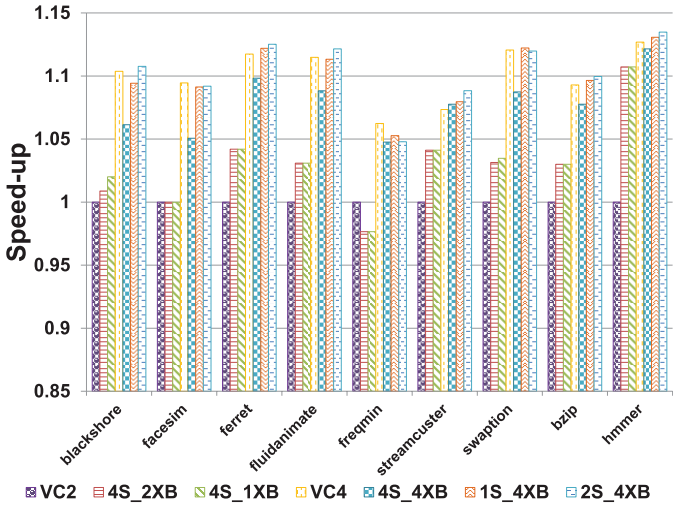


Fig. 12. PARSEC and SPEC2006 speedup when normalized to the execution of VC2 router design.

2) CTorus Results: Fig. 13(a) shows the saturation throughput on the synthetic traffic patterns uniform random, nonuniform random, bit reversal, butterfly, complement, matrix transpose, perfect shuffle, neighbor, and tornado on concentrated Torus (CTorus) architecture with 4S-4XB organization (dual-channel and 4XB) as shown in Fig. 6. CTorus has a saturation throughput approximately 1.5× higher than CMesh2X and FBfly2X for the complement traffic pattern. The dual links in the torus and long wrap-around links reduce contention for packets traveling all the way across the chip as in complement traffic. In traffic such as butterfly, where traffic travels halfway across the chip, the 10-mm links in CMesh2X and FBfly2X allow for a similar saturation throughput compared to CTorus. In bit reversal, the bits in the address of the core are switched and many cores will communicate with themselves because of symmetric bit addresses. This type of communication does not use the network as much, which results in CMesh2X and CTorus having a similar saturation throughput. However, the communication that does use the network can take advantage

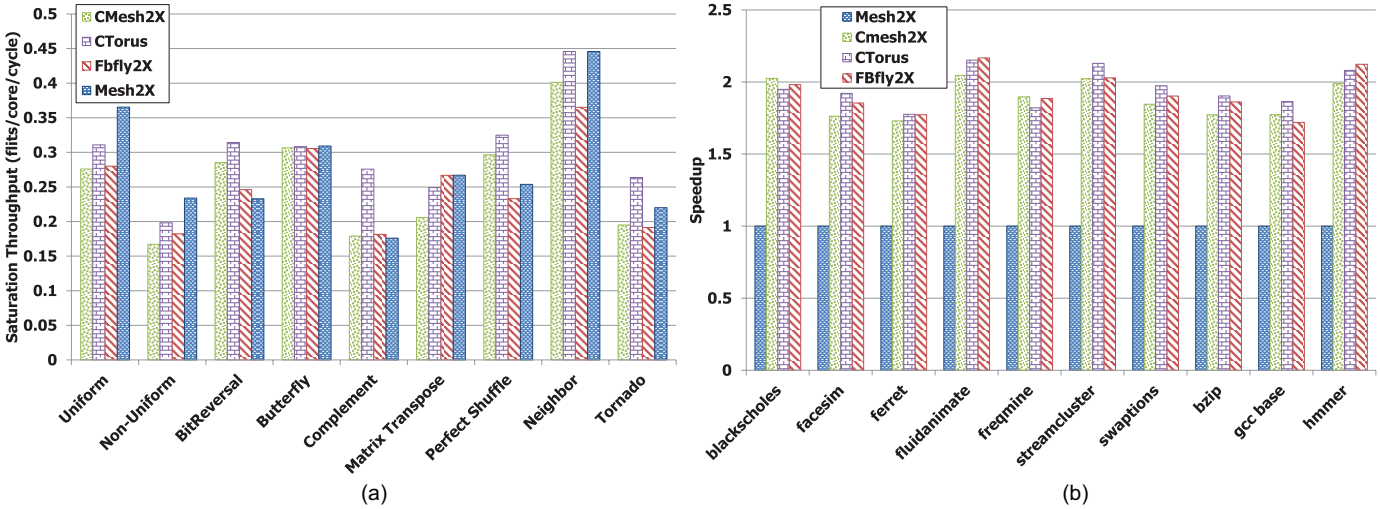


Fig. 13. (a) Saturation throughput of 64 cores for synthetic traffic patterns. (b) Speedup of 64 core topologies for SPEC2006 and PARSEC benchmarks.

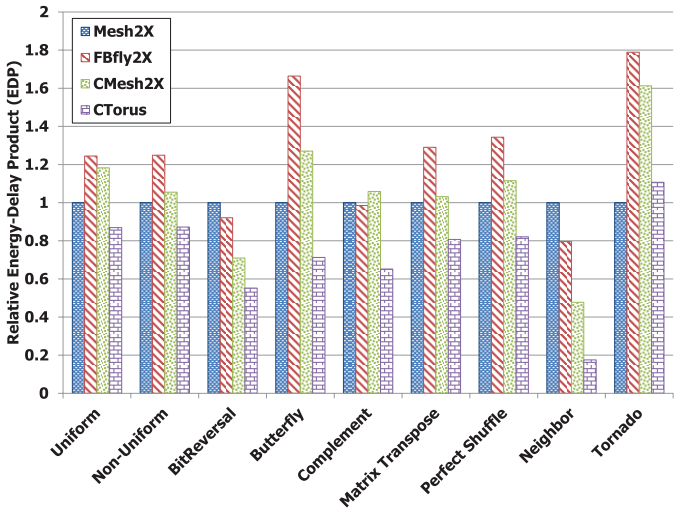


Fig. 14. Relative EDP of 64 core topologies for all synthetic traffic.

of the long links in CTorus. These links in CTorus improve the saturation throughput by an average of approximately 20% over CMesh2X, 21% over FBfly2X, and 11% over mesh2X. Other traffic patterns with a mix of short-, medium-, and long-range communication have comparable results between CTorus, CMesh2X, and FBfly2X, with CTorus having a slight advantage in certain traffic patterns such as nonuniform random and tornado.

Different topologies were evaluated on the PARSEC and SPEC CPU2006 benchmarks. Fig. 13(b) shows the speedup of the total number of clock cycles compared to mesh2X. For PARSEC benchmarks, the CTorus improvement over mesh2X ranges from 1.78 for ferret to 2.15 for fluidanimate benchmarks. For SPEC CPU2006, the communication pattern of gcc base gives an improvement of 1.86, whereas the communication in hmmer allows for a speedup of 2.08 over mesh2X. This improvement over mesh2X is due to the 14 hop diameter of mesh2X compared to 4 hops for CMesh2X and CTorus and 2 hops for FBfly2X. The low hop count of FBfly2X decreases latency, which allows this topology to perform the best for all benchmarks. CMesh2X and CTorus have the same network diameter; however, the long wrap-around links of CTorus allows packets to have a lower average hop count for most traffic traces. Therefore, skipping over more immediate routers in CTorus lowers the average packet latency. The dual network in each concentrated topology lowers the contention for most applications, which is why each concentrated topology saturates at similar loads.

Fig. 14 shows the average EDP per packet. The EDP allows us to analyze how both the latency and power affect each network. Since each topology uses the same number of bits and clock frequency, the power and energy are directly related for each topology. The results shown are normalized to the mesh2X topology. Mesh2X has a high EDP for some cases, but not all, with an average EDP 30% higher than CTorus and an average EDP 18% lower than FBfly2X. This is due to the large network diameter causing high latency and to the high power of mesh2X. On the other hand, FBfly2X has lower latency due to the small network diameter, but the large crossbar power increases the EDP in traffic patterns such as butterfly (66% over mesh2X) and matrix transpose (29% over mesh2X). CMesh2X shows a slightly smaller EDP than FBfly2X for most traffic patterns. This is due to the balance of CMesh2X in terms of power and latency. The CMesh2X topology has a smaller network diameter than Mesh2X, which decreases the latency, and a smaller router than FBfly2X, decreasing the power. Additionally, the longer links around the edges of CMesh2X allow this topology unique benefits for certain cases. CTorus has the same network diameter as CMesh2X, but the low power from the crossbar design allows CTorus to have a lower EDP, for many cases with an average of 29% less than CMesh2X. The average lower EDP is due to the low-power 4XB crossbar and 4S channel buffers, which alleviate HoL blocking and decrease packet latency. Traffic patterns such as complement, where cores communicate with their bitwise complement, will allow packets to use the long wrap-around links to move quickly with low power. For complement, CTorus has an EDP 38% less than CMesh2X.

VIII. Conclusion

In this paper, we evaluated different organizations of channel buffers and crossbars with the objectives of reducing HoL blocking, reducing power dissipation, and simultaneously improving performance at the cost of slight area increase. Our best designs showed power savings of 31% while improving performance from 10%–20% at the cost of 4%–13% area overhead for synthetic as well as real benchmarks (PARSEC and SPEC CPU2006). The 4S design combined with 4XB organization showed that we can achieve high throughput and minimize power while expending some area. The 1XB design consumed more area and power while yielding better performance across all traffic patterns. Our dual-link designs reduced the HoL blocking of traditional channel buffers and increased throughput with restrictive VC allocation with 4XB. Our results concluded that it is possible to improve performance of channel buffers with some area overhead while saving substantial power when compared to the VC router based NoC architectures. In addition, we compared leading topologies such as mesh2X, CMesh2X, and FBfly2X to a CTorus topology which utilizes the proposed channel buffers and crossbar organizations. Our design indicated that the proposed channel buffer and crossbar organizations in the CTorus topology improve the EDP by 29%–37% over CMesh2X and FBfly2X topologies.

References

[1] L. Benini and G. D. Micheli, "Networks on chips: A new SoC paradigm," *IEEE Comput.*, vol. 35, no. 1, pp. 70–78, Jan. 2002.

[2] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proc. Design Autom. Conf.*, Jun. 2001, pp. 684–689.

[3] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar, "A 5-GHz mesh interconnect for a Teraflops processor," *IEEE Micro*, vol. 27, no. 5, pp. 51–61, Sep.–Oct. 2007.

[4] S. Borkar, "Will interconnect help or limit the future of computing?" in *Proc. IEEE Int. 19th Annu. Symp. High-Perform.*, Nov. 2011, pp. 1–3.

[5] J. D. Owens, W. J. Dally, R. Ho, D. N. Jayasimha, S. W. Keckler, and L. S. Peh, "Research challenges for on-chip interconnection networks," *IEEE Micro*, vol. 27, no. 5, pp. 96–108, Sep.–Oct. 2007.

[6] C. A. Nicopoulos, D. Park, J. Kim, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "ViChaR: A dynamic virtual channel regulator for network-on-chip routers," in *Proc. 39th Annu. Int. Symp. Microarch.*, Dec. 2006, pp. 333–344.

[7] A. K. Kodi, A. Sarathy, and A. Louri, "Ideal: Inter-router dual-function energy- and area-efficient links for network-on-chip (NoC)," in *Proc. 35th Int. Symp. Comput. Arch.*, Jun. 2008, pp. 241–250.

[8] G. Michelogiannakis, J. Balfour, and W. J. Dally, "Elastic-buffer flow control for on-chip networks," in *Proc. 15th Int. Symp. High-Perform. Comput. Arch.*, Feb. 2009, pp. 151–162.

[9] T. Moscibroda and O. Mutlu, "A case for bufferless routing in on-chip networks," in *Proc. 36th Annu. Int. Symp. Comput. Arch.*, Jun. 2007, pp. 196–207.

[10] M. Hayenga, N. E. Jerger, and M. Lipasti, "SCARAB: A single cycle adaptive routing and bufferless network," in *Proc. 42nd Annu. IEEE/ACM Int. Symp. Microarch.*, Dec. 2009, pp. 244–254.

[11] F. Gilabert, M. Gomez, S. Medardoni, and D. Bertozzi, "Improved utilization of NoC channel bandwidth by switch replication for cost-effective multi-processor systems-on-chip," in *Proc. 4th ACM/IEEE Int. Symp. Netw.-Chip*, May 2010, pp. 165–172.

[12] J. Balfour and W. J. Dally, "Design tradeoffs for tiled cmp on-chip networks," in *Proc. 20th ACM Int. Conf. Supercomput.*, Jun. 2006, pp. 187–198.

[13] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: Cost-efficient topology for high-radix networks," in *Proc. 34th Annu. Int. Symp. Comput. Arch.*, Jun. 2007, pp. 126–137.

[14] J. Kim, C. A. Nicopoulos, D. Park, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "A gracefully degrading and energy-efficient modular router architecture for on-chip networks," in *Proc. 33rd Annu. Int. Symp. Comput. Arch.*, Jun. 2006, pp. 4–15.

[15] E. Carara, F. Moraes, and N. Calazans, "Router architecture for high-performance NoCs," in *Proc. 20th Annu. Conf. Integr. Circuits Syst. Design*, 2007, pp. 111–116.

[16] S. Lin, J. Shi, and H. Chen, "Designing cost-effective network-on-chip by dual-channel access mechanism," *J. Syst. Eng. Electron.*, vol. 22, no. 4, pp. 557–564, Aug. 2011.

[17] R. Marculescu, U. Ogras, L.-S. Peh, N. Jerger, and Y. Hoskote, "Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 1, pp. 3–21, Jan. 2009.

[18] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The PARSEC benchmark suite: Characterization and architectural implications," in *Proc. 17th Int. Conf. Parallel Arch. Compilat. Tech.*, Oct. 2008, pp. 72–81.

[19] J. L. Henning, "SPEC CPU suite growth: An historical perspective," *SIGARCH Comput. Arch. News*, vol. 35, no. 1, pp. 65–68, Mar. 2007.

[20] A. K. Kodi, R. Morris, D. DiTomaso, A. Sarathy, and A. Louri, "Co-design of channel buffers and crossbar organizations in NoCs architectures," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, Nov. 2011, pp. 219–226.

[21] A. K. Kodi, A. Sarathy, and A. Louri, "Adaptive channel buffers in on-chip interconnection networks: A power and performance analysis," *IEEE Trans. Comput.*, vol. 57, no. 9, pp. 1169–1181, Sep. 2008.

[22] D. U. Becker and W. J. Dally, "Allocator implementations for network-on-chip routers," in *Proc. Conf. High Perform. Comput. Netw., Storage Anal.*, 2009, pp. 1–12.

[23] G. Mora, J. Flich, J. Duato, P. Lopez, E. Baydal, and O. Lysne, "Toward an efficient switch architecture for high-radix switches," in *Proc. ACM/IEEE Symp. Arch. Netw. Commun. Syst.*, Dec. 2006, pp. 11–20.

[24] J. Kim, "Low-cost router microarchitecture for on-chip networks," in *Proc. 42nd Annu. IEEE/ACM Int. Symp. Microarch.*, Dec. 2009, pp. 255–266.

[25] J. Kim, D. Park, C. Nicopoulos, N. Vijaykrishnan, and C. Das, "Design and analysis of an NoC architecture from performance, reliability and energy perspective," in *Proc. ACM/IEEE Symp. Arch. Netw. Commun. Syst.*, Oct. 2005, pp. 173–182.

[26] Y. Ho Song and T. M. Pinkston, "A progressive approach to handling message-dependent deadlock in parallel computer systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 14, no. 3, pp. 259–275, Mar. 2003.

[27] U. Ogras and R. Marculescu, "It's a small world after all: NoC performance optimization via long-range link insertion," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 7, pp. 693–706, Jul. 2006.

[28] M. L. Mui, K. Banerjee, and A. Mehrotra, "A global interconnect optimization scheme for nanometer scale VLSI with implications for latency, bandwidth, and power dissipation," *IEEE Trans. Electron Devices*, vol. 51, no. 2, pp. 195–203, Feb. 2004.

[29] H. S. Wang, X. Zhu, L. S. Peh, and S. Malik, "ORION: A power-performance simulator for interconnection networks," in *Proc. 35th Annu. ACM/IEEE Int. Symp. Microarch.*, Nov. 2002, pp. 294–305.

[30] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A fast and accurate NoC power and area model for early-stage design space exploration," in *Proc. Conf. Design, Autom. Test Eur.*, 2009, pp. 423–428.

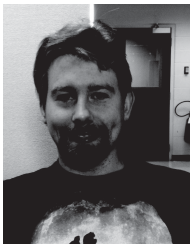
[31] K. Banerjee and A. Mehrotra, "A power-optimal repeater insertion methodology for global interconnects in nanometer designs," *IEEE Trans. Electron Devices*, vol. 49, no. 11, pp. 2001–2007, Nov. 2002.

[32] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hllberg, J. Hgberg, F. Larsson, A. Moestedt, and B. Werner, "Simics: A full system simulation platform," *Computer*, vol. 35, no. 2, pp. 50–58, Feb. 2002.

[33] M. Martin, D. Sorin, B. Beckmann, M. Marty, M. Xu, A. Alameldeen, K. Moore, M. Hill, and D. Wood, "Multifacet's genreal execution-driven multiprocessor simulator (GEMS) toolset," *ACM SIGARCH Comput. Arch. News*, vol. 33, no. 4, pp. 92–99, Nov. 2005.

Dominic DiTomaso (S'06) received the B.S. and M.S. degrees from Ohio University, Athens, in 2010 and 2012, respectively, where he is currently pursuing the Ph.D. degree, all in electrical engineering and computer science.

His current research interests include wireless interconnects, network-on-chips, and computer architecture.



Randy Morris (S'10) received the B.S., M.S., and Ph.D. degrees in electrical engineering and computer science from Ohio University, Athens, in 2007, 2009, and 2012, respectively.

He is currently with Intel Corporation, Portland, OR. His current research interests include optical interconnects, network-on-chips, and computer architecture.

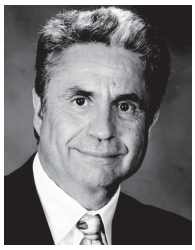
Avinash Karanth Kodi (M'07) received the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Arizona, Tucson, in 2006 and 2003, respectively.

He is currently an Associate Professor of electrical engineering and computer science with Ohio University, Athens. His current research interests include computer architecture, optical interconnects, chip multiprocessors, and network-on-chips.

Dr. Kodi was a recipient of the National Science Foundation CAREER Award in 2011. His papers have been nominated for the Best Paper Award in IEEE/ACM Symposium on Network-on-Chips in 2010 and the IEEE Symposium on ASP-DAC in 2009. He was on the Technical Program Committee of Hot Interconnects 2010 and is a reviewer of several IEEE and the Optical Society of America (OSA) conferences and journals. He is a member of the OSA.

Ashwini Sarathy received the B.E. degree in telecommunications engineering from Visvesvaraya Technological University, Belgaum, India, and the M.S. degree in electrical and computer engineering from the University of Arizona, Tucson, in 2004 and 2008, respectively.

She is currently with MIPS Technologies, Sunnyvale, CA. Her current research interests include computer architecture and network-on-chips (NoCs), with particular emphasis on modeling and simulation of power-efficient NoC architectures.



Ahmed Louri (F'85) received the M.S. and Ph.D. degrees in computer engineering from the University of Southern California, Los Angeles, in 1984 and 1988, respectively.

He joined the University of Arizona, Tucson, in 1988, where he is currently a Professor of electrical and computer engineering and the Director of the High Performance Computing Architectures and Technologies Laboratory. He was a Visiting Scientist with the Communications Research Laboratory, Tokyo, Japan, the Laboratoire d'Informatique du Parallelism, Lyon, France, the University of Tsukuba, Tsukuba, Japan, and the Universite de Paul Sabatier, Toulouse, France. His research is sponsored by the National Science Foundation, DOE, AFOSR, and a number of industrial organizations. He was engaged in bringing optical interconnects into mainstream research in interconnection networks and bridging the gap between the computer architecture and optics research communities. His current research interests include computer architecture, parallel processing, interconnection networks, optical interconnects for parallel computing systems, network-on-chips for multicore architectures, and embedded systems.

Dr. Louri was the General Chair of the 13th Annual Symposium of the High Performance Computer Architecture, Phoenix, AZ, in 2007. He is a Regular Member of the Optical Society of America. He is a member of the International Society for Optical Engineering Working Group on Optical Computing, the IEEE Society Technical Committee on Computer Architecture, and the IEEE Technical Committee on Parallel Processing.