Low-Cost Router Microarchitecture for On-Chip Networks

John Kim
KAIST
Department of Computer Science
Daejeon, Korea
jjk12@cs.kaist.ac.kr

ABSTRACT

On-chip networks are critical to the scaling of future multicore processors. The challenge for on-chip network is to reduce the cost including power consumption and area while providing high performance such as low latency and high bandwidth. Although much research in on-chip network have focused on improving the performance of on-chip networks, they have often relied on a router microarchitecture adopted from off-chip networks. As a result, the on-chip network architecture will not scale properly because of design complexity. In this paper, we propose a low-cost,on-chip network router microarchitecture which is different from the commonly assumed baseline router microarchitecture. We reduce the cost of on-chip networks by partitioning the crossbar, prioritizing packets in flight to simplify arbitration, and reducing the amount of buffers. We show that by introducing intermediate buffers to decouple the routing in the x and the y dimensions, high performance can be achieved with the proposed, low-cost router microarchitecture. By removing the complexity of a baseline router microarchitecture, the low-cost router microarchitecture can also approach the *ideal* latency in on-chip networks. However, the prioritized switch arbitration simplifies the router but creates starvation for some nodes. We show how delaying the rate credits are returned upstream can be used to implement a distributed, starvation avoidance mechanism to provide fairness. Our evaluations show that the proposed low-cost router can reduce the area by 37% and the power consumption by 45% compared with a baseline router microarchitecture that achieves a similar throughput.

Categories and Subject Descriptors

C.1.2 [**Computer Systems Organization**]: Multiprocessors—*Interconnection architectures*

General Terms

Design, Performance

MICRO'09, December 12–16, 2009, New York, NY, USA.
Copyright 2009 ACM 978-1-60558-798-1/09/12 ...$10.00.

Keywords

on-chip network, router microarchitecture, complexity

1. INTRODUCTION

With the increasing number of transistors in modern VLSI technology, the number of cores on a single chip continues to increase in order to efficiently utilize the transistors. As a result, an efficient on-chip network is required in these manycore architectures to connect the cores together. It is projected that the on-chip network will be the critical bottleneck of future manycore processors – both in terms of performance and power [35].

Recently, on-chip network or network-on-chip (NoC) research has focused on the various aspects of on-chip networks, including topology [2, 17, 12, 8], routing [39], flow control [24], and router microarchitecture [1, 33, 22, 27]. These research efforts have focused on providing high performance and achieving power-efficient architectures. However, none of them have addressed the complexity [1] issue in designing an on-chip network. As on-chip network size increases, the design complexity can become the bottleneck that prevents the proper scaling of on-chip networks. This paper addresses the complexity issue in on-chip network by proposing a low-cost router microarchitecture that reduces router complexity and also provides high performance.

Off-chip networks provide very different constraints compared with on-chip networks, but many of off-chip network architectures have been adopted for the on-chip network. As a result, the use of the conventional off-chip network router microarchitecture results in a complex router design for on-chip networks. This increases not only the cost (area and power) of the network but also the pipeline cycle-time and design complexity. Many microarchitectural techniques have therefore been proposed to reduce on-chip network latency, but they have used the conventional off-chip router microarchitecture as the baseline, thereby involving additional complexity and cost.

Because of the complexity of proposed packet-switched, NoC architectures, simpler approach to on-chip networks have been proposed such as the use of ring topologies [14, 9]. The ring topology has been used in the IBM Cell processor [36] and has been suggested for future Intel processors [15], including the Intel Larrabee processor [38]. Because of the simplicity of the ring topology, it does not have

[1] Complexity is defined as *design* complexity. An increase in complexity results in an increase in area and verificaton complexity, and this can also lead to increase in power [3].

the complexity of other architectures. However, as the network size increases, the ring topology becomes limited in its scalability. In this work, we propose to develop a router microarchitecture that approaches the simplicity of a ring topology while providing high performance.

By reducing the complexity of the three main components of a router microarchitecture – input buffers, crossbar, and the arbitration – we present an alternative on-chip network design that attempts to simplify on-chip networks. Instead of assuming a baseline, input-queued router microarchitecture with a single crossbar switch, we propose using a dimension-sliced crossbar to partition the crossbar into an x crossbar and a y crossbar. The arbitration is simplified by providing priority to packets that are in flight and that continue to travel in the same dimension in order to enable a single-cycle router. Additional router pipeline latency is only encountered when the packets "turn" from the x dimension to the y dimension. The simplified router microarchitecture allow us to reduce the amount of buffers needed to only 2 buffer entries per input with minimal loss in performance. With the proposed router microarchitecture, the zero-load latency of the on-chip network can approach the *ideal* latency of on-chip networks [24].

In this work, we assume a 2D mesh topology for NoC. Traditionally, the topology is the critical component of an interconnection network as it determines the performance bounds [7], and different topologies for on-chip networks have been propoposed [2, 17, 12]. However, because of the tightly coupled nature of the channels and routers in NoC, we argue that the NoC router microarchitecture needs to be properly designed to fully exploit the benefits as well as the constraints of the on-chip network, and this requires us to rethink the design of NoC router microarchitecture. The main objective of this work is to achieve simplicity in the design of an on-chip network router to provide a low-cost architecture – achieving low area and power consumption in order to provide a scalable, router microarchitecture. Specifically, the contributions of this work include the following:

- We focus on achieving a simple and low-cost router microarchitecture by not adopting the commonly assumed router microarchitecture used in off-chip networks; instead we begin with a simple microarchitecture consisting of just pipeline registers and muxes.
- We propose a low-cost router microarchitecture for 2D mesh topology to provide a scalable microarchitecture. A dimension-sliced router microarchitecture is used to minimize complexity and introduce intermediate buffers to *decouple* the x-dimension and the y-dimension of the dimension-sliced router – providing high-performance and low-latency router as input buffers are reduced.
- Reducing buffers limit the throughput of the network, but we show that with prioritized allocation favoring packets in flight, we minimize the loss of throughput in the network with only two buffer entries in our proposed router microarchitecture.
- We propose a novel, local fairness scheme to prevent starvation. By managing the rate credits are returned upstream, backpressure can be created to provide fairness in the proposed architecture with minimal overhead.

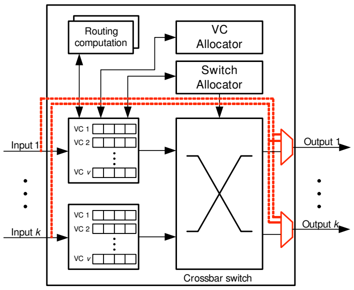


Figure 1: Conventional router microarchitecture with the components highlighted representing the additional logic needed if the router was to support bypassing to reduce router latency.

The remainder of the paper is organized as follows. In Section 2, we provide a background of the different components in a conventional on-chip network router. The proposed low-cost router microarchitecture for the 2D mesh network is described in Section 3, and the simulation results are presented in Section 4. Additional discussions on the proposed architecture are provided in Section 5. Section 6 discusses related work, and Section 7 presents conclusion.

2. CONVENTIONAL ON-CHIP NETWORK ROUTER ORGANIZATION

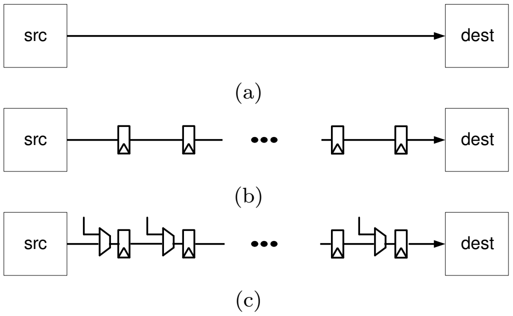
In this section, we provide a background discussion on the main components of an on-chip network router and motivation for a low-cost, complexity-effective router microarchitecture. The block diagram of a conventional router microarchitecture is shown in Figure 1. The main components include the input buffers, the crossbar switch, and the control logic which include the switch and the virtual channel allocators.

2.1 Buffers

Unlike off-chip networks where bandwidth is expensive and buffers are relatively cheap, the constraints for an on-chip network are different – wires (bandwidth) are relatively cheap, while buffers are expensive [6]. Buffers are used to decouple the allocation of resources in interconnection networks and simplify the flow control by using buffered flow control such as virtual cut-through or wormhole flow control.

However, input buffers represent a significant portion of the area and power in an on-chip network router. For an SRAM buffer implementation, the input buffers can consume 46% of the total on-chip network power while occupying 17% of the total area [22]. SRAM is preferred over a register-based FIFO because of its area efficiency, as flipflop-based input buffers can occupy up to 51% of the total area [34]. However, SRAM incurs latency overhead in terms of accessing the buffers as two of the five router pipeline stages in the Intel TeraFlop are dedicated to accessing the buffers (buffer write and buffer read stages) [13].

To efficiently utilize buffers in on-chip networks, dynamic buffer management schemes have been proposed to dynamically share the buffers among different virtual channels [24, 34, 21]. However, these architectures add additional design and verification complexity and impact the router pipeline latency. To reduce the impact of buffers on performance



(a)

(b)

(c)

Figure 2: (a) Ideal on-chip network with only wire delay, (b) wire delay pipelined with registers, and (c) muxes inserted to share the wire resources with other nodes.

and cost, we propose to simplify the router microarchitecture and minimize the amount of buffers needed to create a lost-cost router microarchitecture.

2.2 Switch

The area of a crossbar switch is often the dominant area component of an on-chip router as the area is proportional to $O(p^2w^2)$, where p is the number of router ports and w is the datapath width. As compared to the datapath width (w), the number of ports (p) for on-chip routers is relatively small – e.g., $p = 5$ for a 2D mesh network and $p = 10$ for high-radix on-chip network routers [17] while $w = 128$ to $w = 256$ because of the abundant on-chip bandwidth. The wire dominated crossbar area can occupy up to 64% of the total router area [22]. As a result, to minimize the area impact of on-chip routers, the crossbar area must be minimized. We propose the use of dimension-sliced routers in on-chip networks to reduce crossbar switch area with minimal performance loss.

2.3 Arbitration

The power consumption or the area from the arbitration logic has been shown to be very minimal [41]. However, poor arbitration can limit the throughput of the router and reduce the overall performance of on-chip networks. The latency of the arbitration logic also often determines the router cycle time. Separable allocators have been proposed for on-chip networks which separates the allocation into two stages – input and output arbitration. These allocators require an efficient matching algorithm and novel switch allocation has been proposed to increase the matching efficiency for on-chip networks [22]. However, arbitration is still often in the critical path. Arbitration is needed since resources (such as channel bandwidth) are shared, but if they are reserved ahead of time, arbitration complexity can be reduced or removed completely. In this work, we present simplifying arbitration by giving priority to those packets already in the network that continue to travel in the same dimension – thus removing the switch arbitration from the critical path.

2.4 Motivation

Using baseline router microarchitecture (Figure 1), a router bypass path can be created to reduce the per-hop router latency as packets that bypass the router can avoid the internal router pipeline. However, this requires additional complexity and cost on top of the baseline router microarchitecture. Recently proposed architectures such as the concentrated mesh (CMESH) [2], flattened butterfly (FBFLY) [17], and the express virtual channel (EVC) [24] bypass intermediate routers in order to provide good performance and attempt to achieve an *ideal* latency – i.e. the wire delay from the source to its destination. However, these architectures also add complexity to the design of the on-chip network. For example, EVC requires different router designs for bypass nodes and source/sink nodes, and additional virtual channels are needed to support EVC. The CMESH and FBFLY topologies require non-uniform router designs and high-radix routers which can increase the design complexity. In addition to common evaluation metrics such as performance and cost, other metrics needs to be considered in the evaluation of on-chip network router microarchitecture as shown in Table 1. For example, the flattened butterfly topology is a scalable topology, but it is not extensible since new router designs are required as the network size increases. In this work, we present a router microarchitecture that provides good performance on design metrics.

Metric category	Metric
Performance	Latency
	Bandwidth
Cost	Area
	Power
Design	Extensibility
	Partitionability
	Regularity
	Validation/Testing
Architecture	Fault Tolerance
	Scalable

Table 1: Different evaluation metric of on-chip networks with some proposed by Azimi et al. [28].

An *ideal* on-chip network between two nodes is shown in Figure 2(a), which corresponds to the wire delay between two nodes. To improve the throughput and reduce the clock cycle, pipeline registers can be inserted (Figure 2(b)). However, key aspect of on-chip network is sharing on-chip bandwidth resource among multiple nodes and multiplexers can be inserted share the on-chip bandwidth (Figure 2(c)). In this work, instead of adopting Figure 1 as our baseline, we use the Figure 2(c) view of on-chip network as a starting point in the design of an on-chip network to reduce the cost and complexity. The proposed low-cost router microarchitecture builds on a 2D mesh topology to exploit its design regularity while attempting to provide high performance.

3. LOW-COST ROUTER MICROARCHITECTURE

In this section, we describe the microarchitecture of our proposed, low-cost router microarchitecture for the 2D mesh network. We simplify the router microarchitecture by using prioritized switch allocation, partitioning the crossbar, and reducing the amount of buffers needed in the router.

3.1 Bufferless Router for Ring Topology

The bufferless router microarchitecture has been proposed for ring topology, such as the router used in the Intel Larrabee ring network [38]. A block diagram of an 8-node ring topol-

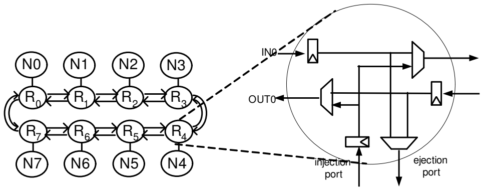


Figure 3: An 8-node ring topology with bufferless router microarchitecture.

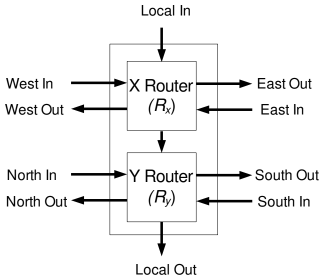


Figure 4: High-level block diagram of a dimension-sliced router in a 2D mesh network.

ogy with a bufferless router is shown in Figure 3 with only pipeline registers. Once a packet is injected into the ring network, the packet is guaranteed to make progress towards its destination by prioritizing those packets that are in flight [9]– thus, there are no contentions for network resources and input buffers are not needed. In this work, we use this bufferless router [2] microarchitecture, which is similar to the figure shown in Figure 2(c), as the starting point and extend this microarchitecture to a 2D mesh topology.

3.2 Switch

The proposed router microarchitecture block diagram for the 2D mesh is shown in Figure 4. Instead of a 5-port router used in a conventional 2D mesh topology, the router is partitioned or *sliced* into two separate routers – one for each dimension of the network – to create a *dimension-sliced* router. The dimension-sliced router was used for the Cray T3D router [16] as the router was partitioned into three separate router chips – one for each dimension of the 3D torus networks. Technology constraints prevented the router from fitting on a single chip, and it was necessary to partition the router across multiple chips. However, we leverage the same microarchitectural technique to reduce the cost of on-chip networks and simplify the router microarchitecture. The number of router ports is reduced from a single router with 5 ports to two routers with 3 ports.

A dimension-sliced router partitions the crossbar switch into two smaller crossbar switches : the x router (R_x) and the y router (R_y). R_x (R_y) is used to route packets that continue to traverse in the x (y) dimension, respectively. Since we assume dimension-ordered (X-Y) routing, a packet that needs to traverse both dimensions to reach its destination will need to change dimension once (i.e., switch routers from R_x to R_y once). However, even if the source and the destination share the same row or same column, the packet will still need to be routed from R_x to R_y because the local injection port is only connected to R_x, while the local ejection port is connected to R_y (Figure 5(a)). By using the dimension-sliced router, it enables creating *overpass* channels in on-chip network – similar to overpass in highways and roads, the dimension-sliced router does not require packets that travel in different dimensions to stop at the current router.

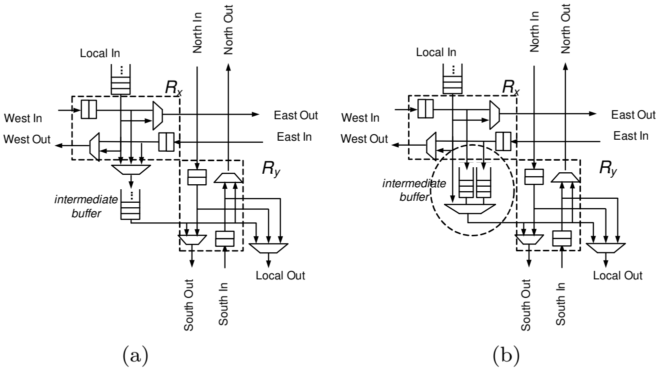


Figure 5: Detailed logic diagram of the proposed router microarchitecture using (a) a shared, intermediate buffer and (b) a dedicated, separate intermediate buffer.

Cycle	1	2	3	4	5	6	7	8	9	10	11	12
R_{00}	BW/ RC	SA/ VA	ST	LT								
R_{01}					BW/ RC	SA/ VA	ST	LT				
R_{11}									BW/ RC	SA/ VA	ST	LT

(a)

Cycle	1	2	3	4
R_{00}	ST_x/ LT			
R_{01}		ST_x/ BW	ST_y/ LT	
R_{11}				ST_y

(b)

Figure 6: Pipeline diagram of a single-flit packet in the (a) baseline, conventional router and the (b) proposed low-cost router in a 2D mesh network, routing from R00 to R11 in the network shown in Figure 7. ST_x is the switch traversal of R_x, and ST_y is the switch traversal R_y .

3.3 Buffers

With a single-cycle router, only two buffer entries are needed to cover the credit round-trip latency. The buffer management provides backpressure and avoids the need to drop packets or misroute them as in a bufferless router [30]. However, intermediate buffers are placed between R_x and R_y to decouple the flow control and routing of R_x from R_y. If a packet traversing the x dimension switches to the y dimension, the packet is buffered in this intermediate buffer before traversing in the y dimension. Thus, a packet will encounter an additional cycle when switching from R_x to R_y.

[2] A bufferless router [30] has been proposed for the 2D mesh NoC but requires deflection routing to remove the need for buffers in on-chip networks – which can increase the latency and reduce the effective bandwidth.

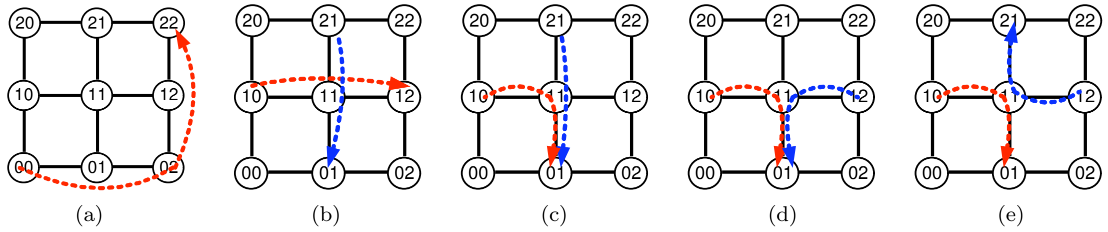


Figure 7: Example flow control in the low-cost, router microarchitecture when there is no contention (a,b) and when packets contend for the same resource (c,d,e) on a 3×3 mesh network. Different color represents different packet in the network

Two different organizations of intermediate buffer is shown in Figure 5. For the shared intermediate buffer organization (Figure 5(a)), switch arbitration (discussed in Section 3.4) is needed prior to being buffered in the intermediate buffer. Another organization requires having dedicated buffers for each router port at the intermediate buffer, as shown in Figure 5(b). For both organizations, the intermediate buffer buffers the packets and allows other packets in the x dimension to continue traversing the network. As the intermediate buffers are located locally, no complex buffer management flow control is needed. The east/west input units need to share a credit for the shared buffer organization (Figure 5(a)), or separate credits are needed with a dedicated flow buffer organization. If the intermediate buffer is full, the input buffers will hold the packets until a intermediate buffer slot becomes available. Simulations comparing the performance of the two different organization of the intermediate buffers achieve similar performance if the total amount of storage is held constant – thus, we assume Figure 5(a) organization in the rest of the paper.

3.4 Arbitration

To reduce the complexity of switch arbitration, simple priority arbitration is used where packets in flight that continue to travel in the same direction have priority over other packets. For example, if a packet arriving from the West port in R_{x} needs to be routed through the East port, it is given priority over packets injected from the local port that needs to be routed through the East port. Similarily, a packet arriving from the North port in R_{y} that needs to be routed to the South port will have priority over packets being injected from the intermediate buffer. Thus, a packet continuing to travel in one dimension will encounter delay very similar to that in Figure 2(c) shown earlier, with additional delay encoutered only when packet "turns" into the intermediate buffer. For packets injected into the network from the injection port or the intermediate buffer, if the router output port is not used by packets in flight, the packet is injected.

Switch arbitration is needed for intermediate buffers as multiple requests can be added with a shared intermediate buffer (Figure 5(a)). With a dedicated intermediate buffer (Figure 5(b)), switch arbitration is required after the packets are buffered in the intermediate buffer. However, switch arbitration can be done in parallel with writing a packet in the buffer – with the result of switch arbitraiton used in the following cycle. Thus, switch arbitration is not on the critical path of the microarchitecture.

The pipeline diagram of a conventional baseline router microarchitecture is shown in Figure 6(a) based on a 3-cycle router [23] and compared with the proposed, router microarchitecture in Figure 6(a). The dimension-sliced router microarchitecture results in the reduction of router latency, as the router pipeline assumed for the conventional router microarchitecture can be avoided, and both the router delay and link delay can be combined into a single cycle – resulting in a single pipeline stage router. An extra pipeline stage is needed when the packet needs to change dimensions as shown in Figure 6(b) because of the use of intermediate buffers. Switch arbitration is removed for packets continuing to travel in the same dimension, as packets in the network have priority over those that have not been injected.

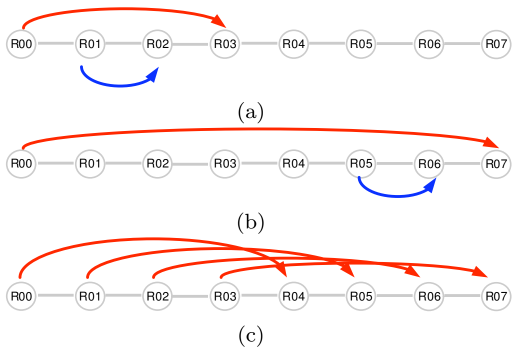
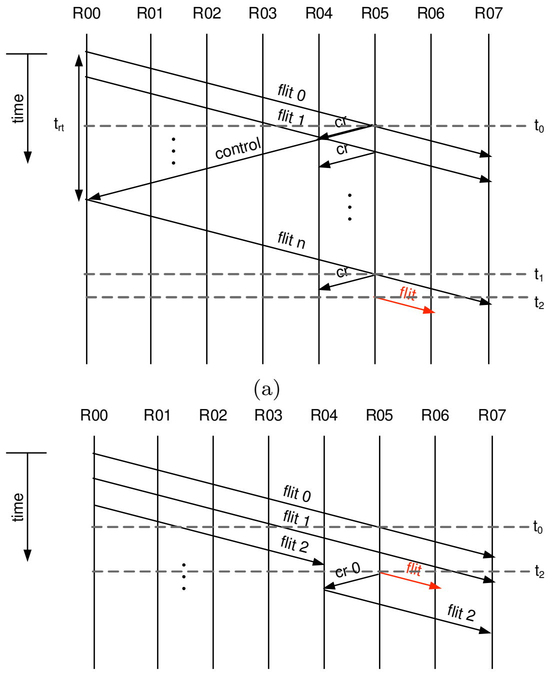


Figure 8: Illustration of (a,b) local starvation and (c) global starvation in the proposed router microarchitecture.

3.5 Routing/Flow Control Examples

To illustrate the behavior of the proposed architecture, we describe examples of routing and flow control on a 3×3 mesh in Figure 7. As shown in Figure 7(a), without any congestion in the network, the network can approach *ideal* latency using the proposed low-cost router microarchitecture. If traffic from the two different dimensions cross over a single router (Figure 7(b)), each packet does not affect the other packets because of the dimension-sliced architecture and achieves minimal latency as well.

When one packet continues to travel in a dimension while another packet turns from a different dimension into the same dimension (Figure 7(c)), contention will occur for the same channel resource. By prioritizing packets continuing in the same dimension, the packet that is "turning" will be buffered in the intermediate buffer until the output channel becomes available. When two packets arrive at the same router node from the same dimension and want to turn to the *same* direction of a different dimension (Figure 7(d)),



(a)

(b)

Figure 9: Time diagram of using (a) explicit control signals to avoid starvation and (b) stalling credits to provide local starvation avoidance scheme for the traffic illustrated in Figure 8(b).

arbitration is needed to access the shared intermediate buffer before traversing in the new dimension. The winner accesses the intermediate buffers, while the packet that does not have its access granted remains buffered in the input buffer.

The contention of resource shown in Figure 7(c,d) is similar to that observed in a conventional microarchitecture; the only difference is how the contention is resolved. However, Figure 7(e) is a contention that is unique to our proposed architecture with a shared intermediate buffer. Two packets arrive at a router from the same dimension and want to turn to a *different* direction of the new dimension. Because of the limited connections in a dimension-sliced crossbar, the intermediate buffer becomes a shared resource and creates a bottleneck. Thus, in the worst-case scenario, the throughput of the network can be degraded by 1/2 compared with a network using a conventional on-chip network router.

3.6 Fairness/Starvation

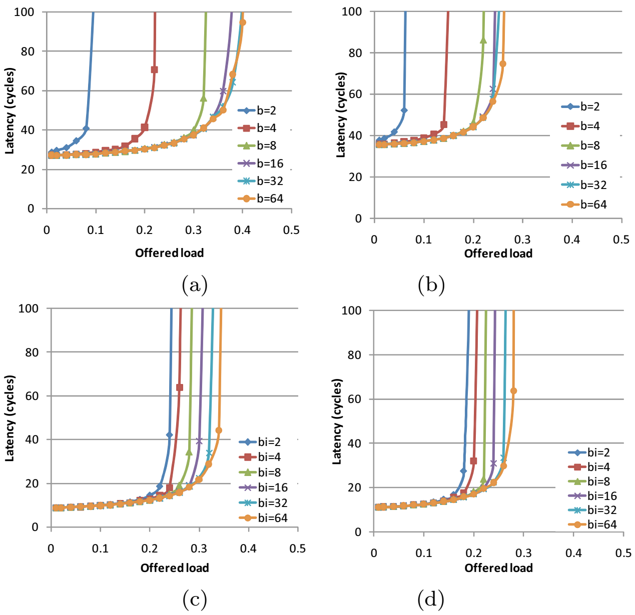
By simplifying the arbitration in the proposed lightweight router microarchitecture, fairness can become an issue. Fairness is not an issue For uniform random traffic near zero-load, but as the load increases and approaches saturation, fairness will become a problem as the packets being injected from the edge of the network will always have priority. Nonuniform traffic patterns will also cause starvation. For example, with the traffic pattern shown in Figure 8(a), R01 can be starved indefinitely if R00 continues to inject packets into the network. Similarly, R05 can be starved in Figure 8(b), as well as R01, R02, and R03 in Figure 8(c).

To overcome this limitation, we can send an explicit control signal upstream to prevent starvation. For example, R01 in Figure 8(a) can wait n cycles and if it is still starved, it can send a control signal to R00 to halt its transmission of packets. Once R00 stops transmission and R01 does not see any more packets, it can inject its packet. A similar scheme has been proposed in the EVC flow control [24] for starvation avoidance as express VCs can starve normal VCs. Thus, *tokens* were proposed in the EVC to prevent starvation. However, using control signals can be very complex and add latency and complexity. For the traffic pattern shown in Figure 8(a), control signals can be sent relatively quickly because the injecting node is only 1 hop away. However, in Figure 8(b), sending an explicit control signal or token can be very time consuming, as illustrated in a time diagram in Figure 9(a). If R05 decides to send an explicit control signal at t_0, in the worst-case scenario (when R00 continues to inject packets), R05 would not be able to inject packets into the network until t_1; thus, the $(t_1 - t_0)$ corresponds to the round-trip delay from R05 to R00. For traffic patterns such as the one shown in Figure 8(c) where R01, R02, and R03 are all starved from R00 traffic, each node sending a control signal (or tokens) will complicate the starvation avoidance scheme.

To prevent starvation, we propose a simplified distributed approach where starvation is prevented by manipulating local router credits. A credit is decremented when a flit is sent downstream. Once the flit departs from the downstream node, a credit is sent back upstream and the appropriate credit count is incremented. However, by stalling the return of credits, an artificial backpressure can be created and prevent upstream nodes from injecting packets into the network, allowing the current node to inject packets into the network.

In this scheme, each router maintains a starvation count n, and once it reaches the maximum value (n_{max}), the router stops the transmission of credits upstream. By delaying the credit upstream, the backpressure ensures that upstream nodes do not inject any more packets into the network. Once the source router injects its packet, n is reset to 0. Each router needs to maintain four separate values of n: n_E and n_W for the east and the west port of R_x, respectively and n_N and n_S for the north and the south port of R_y, respectively. In each cycle, if the injection port of the router has a packet that it is trying to inject into the network, the appropriate n value is incremented each cycle if the packet is unable to inject the packet into the network. When n reaches n_{max}, it means that the n_{max} continuous stream of flits has flowed through the current router node in one particular direction while the router waited and thus, credits are not immediately returned.

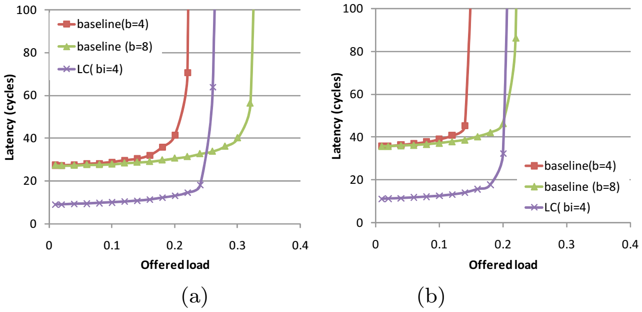
Figure 9(b) shows the time diagram of the starvation avoidance scheme. Assume R00 continues to inject packets into the network destined for R07 (Figure 8(b)), and for simplicity, assume that $n_{max} = 1$. Initially, $n_E = 0$ at R05 and at t_0, R05 wants to inject a packet to R06. In t_0, flit 0 also arrives from R04. Due to the prioritized allocation, flit 0 is granted access to the output port and transmitted to R06 in the next cycle while n_E is incremented to 1. Since there is a packet waiting at R05 to be injected into the network and $n_E = n_{max}$, instead of immediately sending a credit back upstream at t_0, the credit return is stopped. As another packet can be in flight, R05 will need to wait another cycle for flit 1 to pass through. At this point, R04 does not have any credits and is thus required to stop sending packets to R05 – allowing R05 to inject flits into the network at t_2 and



(a) (b)

(c) (d)

Figure 10: Latency vs. load comparison of (a,b) the conventional router microarchitecture and (c,d) the low-cost router for (a,c) UR and (b,d) TOR traffic as the amount of buffer is varied.



(a) (b)

Figure 11: Performance comparison of the baseline and LC router for (a) UR and (b) TOR traffic.

also start returning credits back upstream. As long packets in the network require multiple flits, we prevent the stalling of the credits until the tail flit is received at the current node. This ensures that packets will not be interleaved at the flit granularity and allows the body and tail flits to follow the head flit to its destination.

4. EVALUATION

In this section, we evaluate the proposed, low-cost router microarchitecture against the conventional input-queued microarchitecture using a 64 node, 8×8 2D mesh network. We evaluate the proposed architecture using a cycle-accurate interconnection network simulator [7]. To evaluate the latency-throughput, the simulator is warmed up under load without taking measurements until steady-state is reached. Then, a sample of injected packets is labeled during a measurement interval. The simulation is run until all labeled packets exit the system. Synthetic traffic pattern results from uniform random and tornado traffic are presented. Simulations show that other permutation traffic such as bit complement and transpose follow a trend very similar to tornado traffic and are not included due to page constraint.

In addition to comparing latency/throughput curves, synthetic workloads using closed-loop simulations are used for comparison as well. Synthetic workloads are used to model the memory coherence traffic of a shared memory with each node or processor generating 1K remote memory operations requests [2]. Once requests are received, responses are generated from the destination, and the total completion time for entire network is measured. We allow r outstanding requests per router node to mimic the effect of MSHRs – thus, when r outstanding requests are injected into the network, new requests are blocked from entering the network until the response packets are received. We use $r = 4$ in the results presented in this paper.

We also use network traces from a shared memory multiprocessor. Network traces have also been collected from a 64-processor directory-based Transactional Coherence and Consistency multiprocessor simulator [4] using SPLASH2 benchmarks [42] (methodology is described in [17]). The power and area model is based on the 65nm technology used in [2], and the conventional router has a 3 cycle router pipeline [22, 23]. We assume a datapath width of 128 bits. For short packets, we assume 1-flit packets while for long packets such as cache line, we assume 4-flit packets. For the latency-throughput analysis, we assume a bimodal distribution of packets with 50% of the packets being short, 1-flit packets and the rest being long, 4-flit packets.

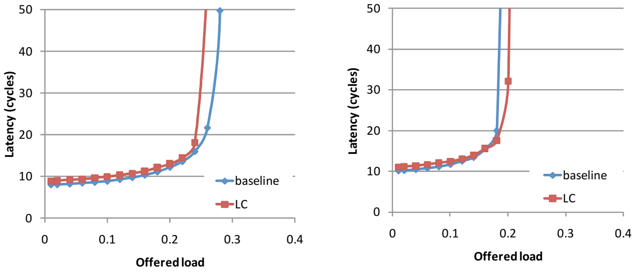
4.1 Performance Comparison

4.1.1 Latency/Throughput Comparison

The latency-throughput of the conventional, baseline router and the proposed low-cost (LC) router is shown in Figure 10 varying amounts of buffers for uniform random (UR) and tornado traffic (TOR) traffic patterns. For the baseline microarchitecture, we vary the number of entries in the input buffer (b), and for the LC router we keep only 2 buffer entries per router input port ($b = 2$) and vary the number of entries in the intermediate buffer (b_i). In general, deeper buffers improve network throughput by decoupling the behavior of neighboring routers. For the baseline router, with $b = 2$ or $b = 4$, there is a severe throughput degradation because of the insufficient buffer entries to cover the credit round-trip latency [3], resulting in up to 75% loss in throughput. However, beyond $b = 16$, there is little increase in throughput as the amount of buffer is increased further. For the LC router, continuing to increase the amount of intermediate buffers (b_i) results in a slight increase in throughput for both UR and TOR traffic. The increase in b_i helps decouple the allocation of $\mathrm{R_x}$ and $\mathrm{R_y}$. Thus larger amount of b_i improves the overall network throuhgput. However, beyond $b_i = 64$, there is minimal increase in throughput.

In Figure 11, we compare the performance of the baseline with the LC router and show that the LC router reduces zero-load latency by up to 67% while providing an increase in throughput by up to 15%, compared with the baseline with $b = 4$. For the TOR traffic (Figure 11(b)), the LC router with $b_i = 4$ is able to nearly match the throughput of the baseline router ($b = 8$), which has approximately 2.6×

[3] Based on Figure 6(a), the credit round-trip latency is 6 cycles.



(a) (b)

Figure 12: Impact of arbitration for (a) UR and (b) TOR traffic as the amount of buffers are varied.

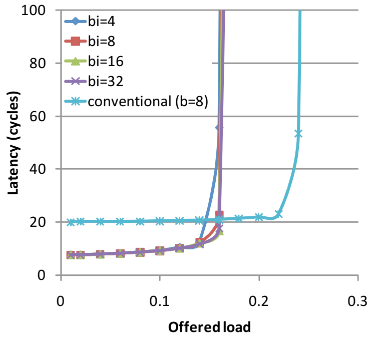


Figure 13: Adversarial traffic patttern evaluation.

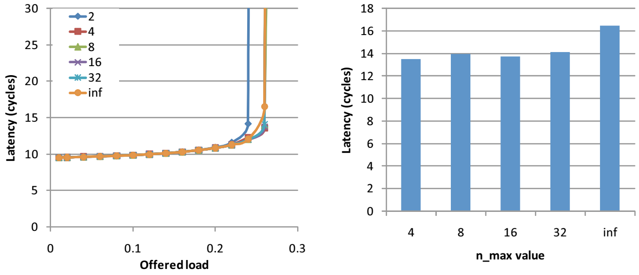
additional amount of storage. With a simplified pipeline and arbitration, the proposed architecture is able to achieve similar throughput and illustrate how simplifying router microarchitecture and pipeline improves the efficiency of the on-chip network.

4.1.2 Impact of Prioritized Arbitration

In order to evaluate the impact of prioritized arbitration, we compare the performance of the baseline and LC router while holding the total amount of buffer constant and assuming the router delays to be constant as well. Thus, we assume $b = 3$ in the baseline and $b = 2, b_i = 4$ in the LC router, and assume the baseline is also a single-cycle router including the link-traversal. The results are compared in Figure 12. The LC router has a slightly higher zero-load latency as we still assume the additional cycle delay to switch dimensions. The throughput of the two routers are very similar. On UR traffic, the baseline exceeds the LC router by approximately 9%, while on TOR traffic, the LC exceeds the baseline by approximately 11%. As a result, even with a simplified, prioritized arbitration, high throughput can be achieved with the LC router compared with the baseline router, which requires all packets to go through a centralized arbitration at each router.

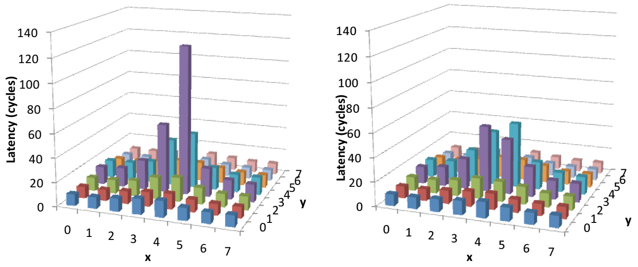
4.1.3 Adversarial Traffic Pattern

As described earlier in Section 3.5, because of the limited bandwidth between $\mathrm{R_x}$ and $\mathrm{R_y}$, the throughput of the LC router can be degraded. For the evaluated synthetic traffic patterns such as UR and TOR, this was not the case as we observed minimal loss in throughput. However, we also simulated an adversarial traffic pattern for the LC router and its results are shown in Figure 13. The traffic pattern attempts to create packet contentions similar to the example shown in Figure 7(e). For example, with the router nodes represented as $\mathrm{R}(x, y)$ in a 2D mesh, $\mathrm{R}(x_1, 0)$ sends its traffic to $\mathrm{R}(x_2, 1)$ where $x_1 \geq x_2$, and $\mathrm{R}(x_3, 2)$ sends its traffic to $\mathrm{R}(x_4, 1)$ where $x_3 \leq x_4$. As b_i increases, unlike other traffic patterns, the throughput does not increase accordingly with the limited bandwidth between $\mathrm{R_x}$ and $\mathrm{R_y}$. Compared with the baseline architecture, the throughput degrades by approximately 30% for this particular traffic pattern. Thus, the simplicity of the switch architecture can result in performance degradation. However, for the other workloads evaluated, we saw this limitation to have a small impact on the overall performance.



(a) (b)

Figure 14: Performance impact of varying n_{max} on TOR traffic – (a) latency/throughput curve and (b) latency comparison near saturation.

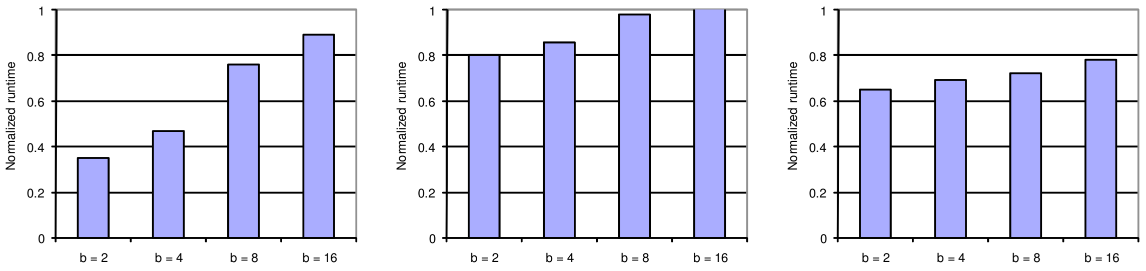


(a) (b)

Figure 15: Latency distribution near saturation with (a) $n_{max} = \infty$ and (b) $n_{max} = 4$.

4.1.4 Fairness and Impact of n_{max}

With the simplified, prioritized arbitration in the LC router, some nodes can be continuously starved without any fairness support. To evaluate the impact of the proposed fairness mechanism described in Section 3.6, different values of n_{max} are compared in Figure 14 for TOR traffic. For $n_{max} > 2$, the throughput of the network as n_{max} is increased is nearly identical to $n_{max} = \infty$, which corresponds to the LC router without any support for fairness. However, if we look into the details of the average latency near saturation (Figure 14(b)), having no support for fairness can result in an increase in average latency by approximately 17% because of the unfairness. The latency distribution of packets near the saturation throughput is shown in Figure 15. We plot the average latency of packets injected from the different nodes in an 8x8 mesh. With $n_{max} = \infty$, the nodes in



(a) (b) (b)

Figure 16: Synthetic workload comparison using closed-loop simulation comparison using (a) uniform random, (b) bit complement, and (c) transpose traffic patterns. The results are normalized to the runtime of the proposed router microarchitecture with $b_i = 2$.

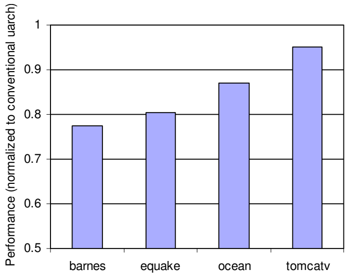
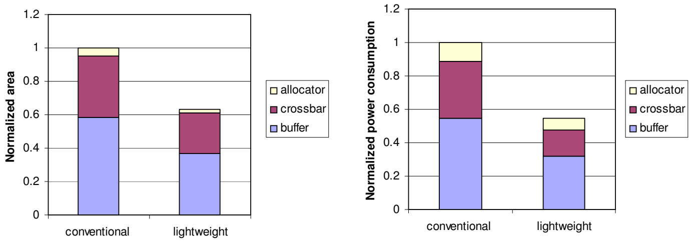


Figure 17: Simulation results using SPLAHS2 traces.



(a) (b)

Figure 18: (a) Area and (b) Power comparison of the conventional microarchitecture to the proposed lightweight router microarchitecture.

the middle are continuously starved, resulting in very high latency as the packets need to wait for the outgoing channels to be idle before injecting their packets. By using $n_{max} = 4$, we can reduce the peak latency by over 50% while achieving lower overall, average latency. However, for $n_{max} = 2$, the throughput of the network degrades slightly as shown in Figure 14(a). By holding back credits, the effective credit round-trip latency is increased – thus, with only 2 buffer entries, there are not enough buffers to cover the credit round-trip latency.

4.1.5 *Synthetic Workload and Traffic Pattern*

Closed-loop simulations comparing the two microarchitectures are shown in Figure 16 for synthetic traffic patterns, and the SPLASH2 benchmark results are shown in Figure 17. When the amount of buffers are held constant ($b = 2$), the LC router provides up to 65% reduction in execution time and up to 22% when $b = 16$ using synthetic workloads and closed-loop simulations. For the different SPLASH2 benchmarks, the LC router with $b_i = 2$, also provides up to 20% reduction in execution time compared with the baseline with $b = 8$. Because of the reduction in zero-load latency, the LC router is able to achieve an improvement in overall performance.

4.2 Cost

To evaluate the cost of the low-cost router microarchitecture, we compare the area and power of the baseline and the LC router as shown in Figure 18. For the baseline router microarchitecture, we assume an input buffer depth of 8 ($b = 8$) and for the LC router,$b = 2$ and the depth of the intermediate buffer is assumed to be 4 ($b_i = 4$) to approximately match the throughput of the two router microarchitecture.

The area comparison of the two routers is shown in Figure 18(a). With only 2 buffers at the inputs in the LC router, the amount of storage bits required is reduced by approximately 2.6×, resulting in the router area consumed by the buffer to be reduced by approximately 40%. The area of the router is also dominated by the crossbar, as it is quadratically proportional to the number of router ports and channel width. By dimension slicing the router structure, the crossbar area is reduced by approximately 33%, and the reduction in the amount of buffer along with dimension-sliced crossbar results in an overall reduction of 37% area. The power consumption comparison shown in Figure 18(b) follows the same trend as the area comparison. The only significant difference is that the area consumed by the allocator is nearly negligible, but it does consume some amount of power. However, with the LC router, the power consumption of the allocator is significantly reduced as well.

In addition, with the low-cost router, the critical path of the router changes. The critical path through the router is often the control signals (e.g., allocator outputs) that drive the datapath (e.g., mux select), resulting in a large delay due to the high fanout with a wide datapath. However, with the LC router, this critical path is removed as the packets in flight have priority and thus, pre-determined allocation results. Our estimate shows that the LC router is able to achieve a cycle time of approximately 13 FO4 – with the critical path consisting of register read, FIFO mux (2-to-1 mux as we only have two entries in the input buffers),

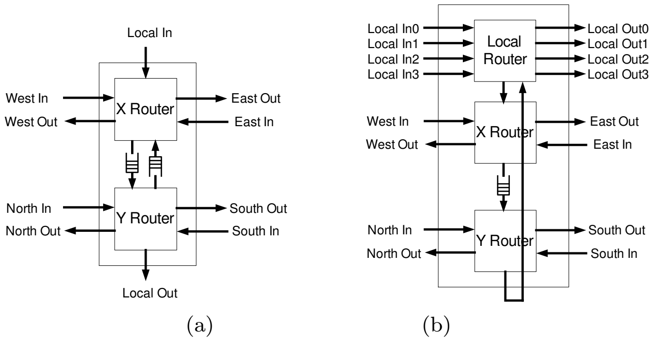


Figure 19: Alternative router microarchitecture to (a) support alternative routing algorithms and (b) support 4-way concentration.

crossbar mux (3-to-1 mux with a dimension-sliced crossbar) and the channel link traversal of 1mm in 65nm technology and finally, the register write. Compared to the baseline router microarchitecture, we are able to achieve approximately 2.5× reduction in latency in the LC router if we assume that the baseline router is built as a single-cycle router and all the pipeline stages are serialized.

5. DISCUSSIONS

In this section, we provide discussions on different possible variations of the proposed architecture to support different on-chip networks.

5.1 Adaptive Routing Algorithms

In this work, we assumed the use of dimension-ordered routing (DOR) and compared our proposed architecture to a conventional microarchitecture that also implements DOR. However, for load-balancing adversarial traffic patterns, alternative routing algorithms such as randomized DOR (O1turn [39]) or adaptive routing can be used. The microarchitecture described in this work does not support other routing algorithms because of the dimension-sliced microarchitecture. However, if increased performance and load-balancing are required, other routing algorithms can be supported by adding additional complexity to the crossbar switch as shown in Figure 19(a), enabling the Y router to send packets to the X router. This allows the packets to make both X-Y turns as well as Y-X turns. In addition, by using a router microarchitecture with a very shallow buffer, stiff backpressure can be provided such that it can possibly enable a better adaptive routing decision without the need for explicit global congestion notification [10].

5.2 Concentration

Concentration in on-chip networks has been proposed [2, 17] to reduce the cost of the network. The proposed router architecture can be used with concentration to exploit local traffic and further reduce network cost. For example, the proposed router architecture with a concentration factor of 4 can be implemented as shown in Figure 19(b) by creating another slice and creating a *local* router for the local injection and ejection ports. This organization isolates the local traffic among the local nodes from any global traffic traversing the X or the Y router. However, the bandwidth between the local router and the X router can become the bottleneck, and additional internal speedup might be required to provide good performance.

6. RELATED WORK

Many different designs of single-cycle on-chip network routers have been proposed. Mullins et. al [33] proposed a single cycle router, which uses precomputation to remove the control logic from the critical path to create a single-cycle router. Although it was initially assumed to achieve a clock period of 12 FO4, because of the complexity of their microarchitecture, an implementation of their router resulted in a router pipeline with 35 FO4 [32]. The on-chip network router used in the TRIPS processor was built as a single-cycle router which included the link traversal delay [11]. The router pipeline serialized the different pipeline stages, but a single cycle was achieved because of a non-aggressive cycle time (366Mhz using 130nm technology). Kumar et al. [22] proposed a single-cycle router that achieved 3.6GHz in 65nm technology. However, to achieve single-cycle, additional *advanced bundle* signals were required to set up the path of the packet in low load. In addition, the capability to bypass routers to achieve the ideal zero-load latency was proposed on top of a conventional router microarchitecture, thereby adding complexity to the router design.

The complexity and tradeoff of arbitration and scheduling have been previously studied [31, 22]. Mukherjee et. al [31] showed that a simpler, low-latency but sub-optimal arbiter design outperformed the traditional, complex arbiter in an Alpha 21364 router. They proposed the *Rotary Rule* which provides priority to those packets already in the network. However, the rotary rule was proposed for a network when it approaches saturation. In addition, there is still significant complexity in the proposed arbiter as multiple priority rules are required, and like other iterative arbiters, it requires multiple steps to arbitrate between the inputs and the outputs. SPAROFLO [22] proposes using a separable allocator and gives priority to past requests over new requests in the network. However, SPAROFLO also requires three different priority rules which complicate the allocation and it is a simplified matching algorithm. In comparison, the proposed low-cost microarchitecture does not require a separable allocator but relies on a simple arbiter to prioritize packets in flight.

Many crossbar switch designs have been proposed which partition the crossbar into a smaller crossbars such as using smaller, faster subcrossbars to exploit traffic characteristics [5], as well as using subswitches to scale the router to high-radix [37]. However, these architectures were focused on off-chip networks where the constraints are different. Recently, partitioning on-chip router microarchitecture designs have been proposed. Lee et al. [26] proposed an router microarchitecture that partitioned the router into a *left* and a *right* router which are disjoint – the left router handles traffic coming from the left (west) port, while the right router handles traffic coming from the right (east) port. However, this requires partitioning the north and south port bandwidth in half to accommodate the outputs of each of the disjoint router. Kim et al. [20] partitioned a router microarchitecture for a 2D mesh network into two 2×2 crossbar. Although the crossbar design was simplified, additional buffers and virtual channels were needed in front of the crossbars, which resulted in additional complexity. The on-chip network router for crossbar structure in a 3D architecture has been proposed which was partitioned according to the dimensions [19]. This reduces the area occupied by the crossbar but adds wire and routing complexity in front of the dimension decomposed crossbar.

Using buffers within channels has been proposed to modify repeaters and use them as a storage element [29, 21]. This approach reducess the amount of buffer needed at the routers and creates a more efficient router. However, this does not reduce the need for buffers but only distributes them across the channels. *Bypass channels* were proposed to increase the performance of an on-chip flattened butterfly network for non-minimal routing [17]. This work uses a similar approach of bypass channels as each router behaves similar to a bypass channel in each dimension but does not require the complexity of high-radix routers in on-chip networks. Credit round-trip latency was used in the dragonfly topology to stiffen backpressure and thus improve the performance of adaptive routing [18]. The fairness mechanism described in Section 3.6 is similar because the rate of credit return is modified; however, we do not rely on credit round-trip latency but only on the local management of credits. The proposed fairness mechanism is not intended to provide global fairness such as other proposed schemes (i.e., GSF [25]) but only an attempt to provide local fairness – similar to a conventional router microarchitecture. The motivation described in Section 2.4 is similar to the motivation of using an operand network in the MIT RAW processor [40]. However, we extend this motivation to simplify the router microarchitecture for creating a scalable 2D mesh network.

7. CONCLUSION AND FUTURE WORK

In this work, we present an alternative approach to designing on-chip network routers to achieve a low-cost and *complexity-effective* router microarchitecture. By eliminating the amount of buffers, simplifying the switch arbitration, and using dimension-sliced router microarchitecture, a low-cost router microarchitecture is developed that can provide single-cycle router latency and approach ideal on-chip network latency. To support a scalable 2D mesh network, we introduce intermediate buffers internal to the router to decouple the two dimensions of the dimension-sliced router. By giving priority in switch arbitration to packets continuing to travel in the same dimension, the router pipeline delay is also minimized and reduces network contention to provide high throughput with limited amount of buffers. However, simplified switch arbitration causes starvation, and we show how delaying credits can provide a simple mechanism for starvation avoidance. Evaluations show that the proposed lightweight architecture can reduce the area by 37% and the power consumption by 45% compared with a conventional router microarchitecture that achieves the same throughput.

Our low-cost router does not include the many functionalities that have been proposed for on-chip networks, including fault tolerance, QoS, support for different traffic classes, and alternative routing algorithms. Our future work will focus on incorporating these functionalities into the proposed low-cost router without adding any significant cost. We also assumed a conventional, credit-based flow control but with only two buffer entries and the prioritized arbitration, other flow controls may be more appropriate to minimize cost. In addition, improvement in the fairness mechanism is needed to provide better fairness while still attempting to minimize the complexity as the network size continues to increase.

Acknowledgments

We would like to thank the anonymous reviewers for their comments. This work was supported in part by the KAIST-Microsoft Research Collaboration Center (KMCC) at KAIST, Korea.

8. REFERENCES

[1] P. Abad, V. Puente, J. A. Gregorio, and P. Prieto. Rotary router: An efficient architecture for cmp interconnection networks. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, San Diego, CA, June 2007.

[2] J. Balfour and W. J. Dally. Design tradeoffs for tiled cmp on-chip networks. In *ICS '06: Proceedings of the 20th annual international conference on Supercomputing*, pages 187–198, 2006.

[3] P. Bose, D. H. Albonesi, and D. Marculescu. Guest editors' introduction: Power and complexity aware design. *IEEE Micro*, 23(5):8–11, 2003.

[4] H. Chafi, J. Casper, B. D. Carlstrom, A. McDonald, C. Cao Minh, W. Baek, C. Kozyrakis, and K. Olukotun. A scalable, non-blocking approach to transactional memory. In *International Symposium on High-Performance Computer Architecture (HPCA)*. Feb 2007.

[5] Y. Choi and T. M. Pinkston. Evaluation of Crossbar Architectures for Deadlock Recovery Routers. *J. Parallel Distrib. Comput.*, 61(1):49–78, 2001.

[6] W. J. Dally and B. Towles. Route packets, not wires: on-chip inteconnection networks. In *Proc. of the 38th conference on Design Automation (DAC)*, pages 684–689, 2001.

[7] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, San Francisco, CA, 2004.

[8] R. Das, S. Eachempati, A. K. Mishra, N. Vijaykrishnan, and C. R. Das. Design and evaluation of a hierarchical on-chip interconnect for next-generation cmps. In *International Symposium on High-Performance Computer Architecture (HPCA)*, pages 175–186, Raleigh, North Carolina, 2009.

[9] T. Fossum. Keynote : On-die interconnect and other challenges for chip-level multi-processing. In *Proc. of Hot Interconnects*, Stanford, CA, Aug. 2007.

[10] P. Gratz, B. Grot, and S. Keckler. Regional congestion awareness for load balance in networks-on-chip. In *International Symposium on High-Performance Computer Architecture (HPCA)*, Feb. 2008.

[11] P. Gratz, C. Kim, R. McDonald, S. Keckler, and D. Burger. Implementation and Evaluation of On-Chip Network Architectures. In *International Conference on Computer Design (ICCD)*, 2006.

[12] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu. Express cube topologies for on-chip interconnects. In *International Symposium on High-Performance Computer Architecture (HPCA)*, pages 163–174, Raleigh, North Carolina, 2009.

[13] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar. A 5-ghz mesh interconnect for a teraflops processor. *IEEE Micro*, 27(5):51–61, 2007.

[14] C. J. Hughes, R. Grzeszczuk, E. Sifakis, D. Kim, S. Kumar, A. Selle, J. Chhugani, M. J. Holliman, and Y.-K. Chen. Physical simulation for animation and visual effects: parallelization and characterization for chip multiprocessors. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, pages 220–231, San Diego, CA, 2007.

[15] Intel. Platform 2015: Intel Processor and Platform Evolution for the Next Decade. Mar. 2005.

[16] R. Kessler and J. Schwarzmeier. Cray t3d: a new dimension for cray research. *Compcon Spring '93, Digest of Papers.*, pages 176–182, 22-26 Feb 1993.

[17] J. Kim, J. Balfour, and W. J. Dally. Flattened Butterfly for

On-Chip Networks. In *Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 172–182, Chicago, IL, Dec. 2007.

[18] J. Kim, W. J. Dally, S. Scott, and D. Abts. Technology-driven, highly-scalable dragonfly topology. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, pages 77–88, Beijing, China, 2008.

[19] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, M. S. Yousif, and C. R. Das. A novel dimensionally-decomposed router for on-chip communication in 3d architectures. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, pages 138–149, San Diego, CA, 2007.

[20] J. Kim, C. Nicopoulos, D. Park, V. Narayanan, M. S. Yousif, and C. R. Das. A gracefully degrading and energy-efficient modular router architecture for on-chip networks. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, pages 4–15, Boston, MA, June 2006.

[21] A. K. Kodi, A. Sarathy, and A. Louri. iDEAL: Inter-Router Dual-function Energy and Area-efficient Links for Network-on-Chip. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, Beijing, China, June 2008.

[22] A. Kumar, P. Kundu, A. Singh, L.-S. Peh, and N. Jha. A 4.6tbits/s 3.6ghz single-cycle noc router with a novel switch allocator in 65nm cmos. In *International Conference on Computer Design (ICCD)*, October 2007.

[23] A. Kumar, L.-S. Peh, and N. K. Jha. Token flow control. In *Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 342–353, Lake Como, Italy, 2008.

[24] A. Kumar, L.-S. Peh, P. Kundu, and N. K. Jhay. Express virtual channels: Towards the ideal interconnection fabric. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, San Diego, CA, June 2007.

[25] J. W. Lee, M. C. Ng, and K. Asanovic. Globally-synchronized frames for guaranteed quality-of-service in on-chip networks. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, Beijing, China, June 2008.

[26] S. E. Lee, J. H. Bahn, and N. Bagherzadeh. Design of a feasible on-chip interconnection network for a chip multiprocessor (cmp). *International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*, 0:211–218, 2007.

[27] H. Matsutani, M. Koibuchi, H. Amano, and T. Yoshinaga. Prediction router: Yet another low latency on-chip router architecture. In *International Symposium on High-Performance Computer Architecture (HPCA)*, pages 367–378, Raleigh, North Carolina, 2009.

[28] M.Azimi, N.Cherukuri, D.N.Jayasimha, A. Kumar, P. K. andS. Park, I. Schoinas, and A. Vaidya. Integration Challenges and Tradeoffs for Tera-scale Architectures. In *Intel Technology Journal*, 2006.

[29] M. Mizuno, W. Dally, and H. Onishi. Elastic interconnects: repeater-inserted long wiring capable of compressing and decompressing data. *Solid-State Circuits Conference, 2001. Digest of Technical Papers. ISSCC. 2001 IEEE International*, pages 346–347, 464, 2001.

[30] T. Moscibroda and O. Mutlu. A case for bufferless routing in on-chip networks. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, Austin, TX, 2009.

[31] S. S. Mukherjee, F. Silla, P. Bannon, J. Emer, S. Lang, and D. Webb. A comparative study of arbitration algorithms for the alpha 21364 pipelined router. pages 223–234, 2002.

[32] R. Mullins, A. West, and S. Moore. The design and implementation of a low-latency on-chip network. In *Proc. of the 2006 conference on Asia South Pacific design automation (ASP-DAC)*, pages 164–169, Yokohama, Japan, 2006.

[33] R. D. Mullins, A. West, and S. W. Moore. Low-latency virtual-channel routers for on-chip networks. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, pages 188–197, 2004.

[34] 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 *Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Orlando, FL, 2006.

[35] 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*, pages 96–108, Sept./Oct. 2007.

[36] D. Pham, T. Aipperspach, D. Boerstler, M. Bolliger, R. Chaudhry, D. Cox, P. Harvey, P. Harvey, H. Hofstee, C. Johns, J. Kahle, A. Kameyama, J. Keaty, Y. Masubuchi, M. Pham, J. Pille, S. Posluszny, M. Riley, D. Stasiak, M. Suzuoki, O. Takahashi, J. Warnock, S. Weitzel, D. Wendel, and K. Yazawa. Overview of the architecture, circuit design, and physical implementation of a first-generation cell processor. *Solid-State Circuits, IEEE Journal of*, 41(1):179–196, Jan. 2006.

[37] S. Scott, D. Abts, J. Kim, and W. J. Dally. The BlackWidow High-radix Clos Network. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, Boston, MA, June 2006.

[38] L. Seiler, D. Carmean, E. Sprangle, T. Forsyth, P. Dubey, S. Junkins, A. Lake, R. Cavin, R. Espasa, E. Grochowski, T. Juan, M. Abrash, J. Sugerman, and P. Hanrahan. Larrabee: A many-core x86 architecture for visual computing. *IEEE Micro*, 29(1):10–21, 2009.

[39] D. Seo, A. Ali, W.-T. Lim, N. Rafique, and M. Thottethodi. Near-optimal worst-case throughput routing for two-dimensional mesh networks. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, pages 432–443, 2005.

[40] M. B. Taylor, W. Lee, S. P. Amarasinghe, and A. Agarwal. Scalar operand networks. *IEEE Transactions on Parallel and Distributed Systems*, 16(2):145–162, 2005.

[41] H. Wang, L. S. Peh, and S. Malik. Power-driven Design of Router Microarchitectures in On-chip Networks. In *Proc. of the 36th Annual IEEE/ACM Int'l Symposium on Microarchitecture*, pages 105–116, 2003.

[42] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta. The SPLASH-2 programs: Characterization and methodological considerations. In *Proc. of the International Symposium on Computer Architecture (ISCA)*, pages 24–36, Santa Margherita Ligure, Italy, 1995.