A High-Performance Hierarchical Ring On-Chip Interconnect with Low-Cost Routers

Chris Fallin cfallin@cmu.edu
Xiangyao Yu yxythu@gmail.com
Gregory Nazario gnazario@cmu.edu
Onur Mutlu onur@cmu.edu

Computer Architecture Lab (CALCM)
Carnegie Mellon University

SAFARI Technical Report No. 2011-007

September 6, 2011

Abstract

Energy consumption of routers in commonly used mesh-based on-chip networks for chip multiprocessors is an increasingly important concern: these routers consist of a crossbar and complex control logic and can require significant buffers, hence high energy and area consumption. In contrast, an alternative design uses ring-based networks to connect network nodes with small and simple routers. Rings have been used in recent commercial designs, and are well-suited to smaller core counts. However, rings do not scale as efficiently as meshes.

In this paper, we propose an energy-efficient yet high performance alternative to traditional mesh-based and ring-based on-chip networks. We aim to attain the scalability of meshes with the router simplicity and efficiency of rings. Our design is a hierarchical ring topology which consists of small local rings connected via one or more global ring. Routing between rings is accomplished using bridge routers that have minimal buffering, and use deflection in place of buffered flow control for simplicity.

We comprehensively explore new issues in the design of such a topology, including the design of the routers, livelock freedom, energy, performance and scalability. We propose new router microarchitectures and show that these routers are significantly simpler and more area and energy efficient than both buffered and bufferless mesh based routers. We develop new mechanisms to preserve livelock-free routing in our topology and router design. Our evaluations compare our proposal to a traditional ring network and conventional buffered and bufferless mesh based networks, showing that our proposal reduces average network power by 52.4% (30.4%) and router area footprint by 70.5% from a buffered mesh in 16-node (64-node) configurations, while also improving system performance by 0.6% (5.0%).

1 Introduction

Interconnect is a first-order concern in the design of future CMPs (chip multiprocessors). As we move beyond the current commercial 2- to 8-core multicores, scalability and energy efficiency demand a better alternative to conventional shared bus, crossbar, or other central interconnect structures. For tiled CMPs, networks-on-chip [5] are the most commonly explored solution to efficiently and scalably allow communication between cores with reasonable cost as the number of cores grows into tens or hundreds.

As interconnect begins to play a more central role in large-CMP design, the die area of interconnect components, and the energy that the interconnect consumes to move a given block of data, matter more in the system-level design tradeoffs of such a system. Previous works have shown that interconnect can consume a large portion of system power: for example, in the Intel Terascale chip, the mesh interconnect consumes 30% of chip power [13]. Die area is also a concern: in the Intel Terascale, the NoC consumes 25% of die area [13]. Previous work has addressed both power and area by eliminating router buffers [21, 8, 12, 9, 28] or using a simplified microarchitecture [17], switching to a lower-power mode when possible while retaining the ability to perform traditional buffered mesh routing [15]. However, mesh-based designs have fundamental complexities in routing and crossbar overhead that cannot be eliminated.

In contrast, *ring*-based networks allow for much simpler router design. Rings are a well-known network topology [4], and the idea behind a ring topology is very simple: all routers are connected by a loop that carries network

traffic. At each router, new traffic can be injected into the ring, and traffic in the ring can be removed from the ring when it reaches its destination. When traffic is traveling on the ring, it continues uninterrupted until it reaches its destination. A ring router is thus very simple in comparison to a mesh router: control logic is vastly simpler, and the crossbar is replaced by one MUX per router (to allow new traffic to enter). Router latency is also typically only one cycle, because unlike in a mesh, a router does not need to compute next-hop routes for each flit and arbitrate multiple flits. Rather, it simply allows traffic in the ring to continue, removes that traffic if it has arrived at its destination, or injects new traffic if there is space in the ring.

Because on-die wiring overhead is low, rings are often very wide [13] (e.g., 256 bits in Intel Sandy Bridge [14]): once the fixed cost of control logic is paid, high bandwidth is a relatively smaller incremental cost. Rings are thus competitive with meshes up to tens of nodes [18]. As a result, several prototype and commercial multicore processors have utilized ring interconnects: among them, The Intel Larrabee [25], IBM Cell [22], and more recently, the Intel Sandy Bridge [14]. Unfortunately, rings suffer from a fundamental scaling problem, because a ring's bisection bandwidth does not scale with the number of nodes in the network. Instead, rings saturate once enough nodes are added. Although a wide ring or multiple rings can reduce this bottleneck, the ring router die area and power increase accordingly.

We wish to obtain the advantages of both mesh- and ring-based designs while avoiding their shortcomings. At one end, ring networks use very simple routers, which reduces energy and die area, and eases design and validation. However, performance suffers with scaling. At the other end, mesh networks scale relatively well, but require large, complex, energy-inefficient routers. To bridge the gap, we begin with a ring-based approach, and observe that to reduce ring contention caused by higher node counts, we can split the network into several smaller *local rings* composed of simple ring routers. Then, using slightly more complex *bridge routers* that join two rings, but using a smaller number of them, we join the local rings with one or more *global rings* which transfer traffic between local rings. By adjusting the number of global rings and their width, we can retain the bisection bandwidth of the more complex mesh network without adopting its complex routers. The operation of the network is only slightly more complex than a single ring. When traffic is sent between two nodes in the same local ring, the routers in that ring communicate as before, injecting traffic into the local ring whenever there is a free slot and removing the traffic at its destination. However, when traffic is sent between nodes in different local rings, the bridge router in the sender's local ring transfers the traffic to the global ring, and the bridge router in the receiver's local ring sees the traffic on the global ring and transfers it into its own local ring.

While hierarchical-ring topologies have been proposed before [23], we adopt design principles from deflection (hot-potato) routing [1] in order to make the routers, and especially the bridge routers that connect two rings, as simple and energy-efficient as possible. We use minimal buffering at these bridge routers, and no flow control, instead deflecting flits (i.e., leaving them in the original ring to make another round) when a bridge router cannot transfer a flit. To guarantee forward progress, we design explicit mechanisms to guarantee eventual delivery of every flit.

Such a design has multiple advantages. First, the router at each processor or cache node is a very simple ring router, significantly reducing complexity, die area and energy cost relative to a mesh. Second, the hierarchical structure is a good match for traffic patterns that have some locality. Even when there is no locality (which is unlikely as core counts continue to increase[1]), it is more effective to add more global rings, or widen existing global rings, than it is to widen the ring on which every local router sits, as we show in §8 by comparing to a single-wide-ring design. Third, the global ring can act as an express network for long-distance traffic by directly connecting two local rings without impacting the rings between them. Hence, relative to a traditional ring design, a hierarchical ring architecture attains better scalability and lower cost for a given performance target. It also uses routers that are significantly simpler and more energy-efficient than conventional mesh routers, as supported by our extensive experimental evaluations.

In this work, **our contributions** are:

- We propose a new, low-cost, hierarchical ring NoC design based on very simple router microarchitectures that achieve single-cycle latencies. This design places an ordinary ring router at every network node, connects local rings with global rings, and joins local and global rings using *bridge routers*, which have minimal buffering, and use deflection rather than buffered flow control for inter-ring transfers.
- We provide new mechanisms for *guaranteed delivery of traffic* ensuring that inter-ring transfers do not cause livelock or deadlock, even in the worst case.
- We qualitatively and quantitatively compare our hierarchical ring-based NoC to three state-of-the-art NoC designs:

[1] As CMPs grow larger, locality-aware mechanisms that reduce on-chip communication overhead are becoming more critical to maintain effective scalability [19, 6, 29, 2].

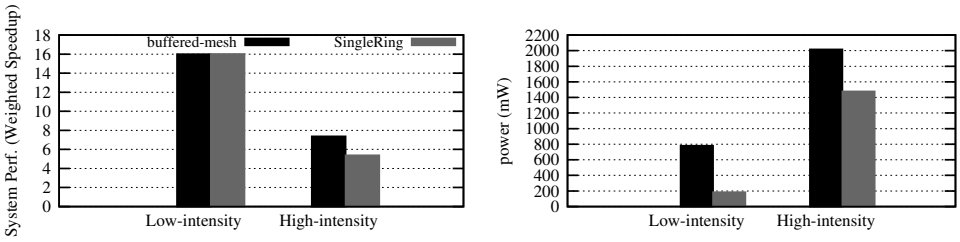


Figure 1: Performance and average network power for 16 nodes, in mesh and ring interconnects.

(i) a buffered-router 2D-mesh interconnect, (ii) a single-ring interconnect, and (iii) a bufferless deflection network (CHIPPER [8]). We show average network power reduction in a 4x4 network of 52.4%, 39.1% and 23.6% compared to the buffered mesh, single ring and CHIPPER, respectively. Despite these large reductions, the hierarchical ring design performs 0.6% better (4x4) or 5.0% better (8x8) than the conventional buffered mesh network over a set of 105 multiprogrammed SPEC CPU2006 workloads, and attains further gains over the other interconnect designs. In addition, evaluations with synthetic traffic patterns show that performance results are robust to workload traffic characteristics.

- We perform a number of sensitivity and scalability analyses on our proposed NoC design. We show that a hierarchical ring interconnect can be configured across a wide range of the performance-power trade-off, and that unlike a single-ring design, a hierarchical design scales comparably to traditional mesh designs as the core count increases.

2 Motivation

2.1 Energy Efficiency in Ring-Based Interconnects

Rings achieve very good energy efficiency at low-to-medium core counts [18], or when network load is low enough that the ring does not become a bottleneck. The routing logic for a router in a single-ring system needs to handle only two tasks: injecting traffic into a ring when there is space, and ejecting traffic when it is addressed to the current router. Most of the energy consumed in the interconnect is thus due to link traversal – in other words, the router energy is minimal because the routers are very simple. The simplicity, and resulting low cost of design and validation, are further appealing advantages.

In order to understand the behavior of rings relative to meshes more concretely, let us examine data from application workloads on a medium-sized (16 node) CMP. In Fig. 1, we show application performance and average network power for a set of 105 multiprogrammed (SPEC CPU2006 [27]) workloads on a 16-node system with either a single ring or a conventional 2D-mesh (4x4) buffered interconnect. (Our methodology and other details are presented in §7). The workloads are split into two categories: **L**, or "low-intensity", which have a low average traffic injection rate into the network, and **H**, or "high-intensity," which stress the network with much higher injection rates on average.

First, as the data in the figure demonstrate, for a medium-sized (16-core) system configuration, the single ring slightly exceeds the performance of a 2D mesh at low load, because the smaller router latency (1 cycle rather than 2 cycles) enables higher performance. With the set of high-network-intensity workloads, in which the NoC is the system bottleneck, performance degrades 27.1% relative to the mesh. Hence, although the network begins to hit scaling limits when stressed, it performs very well at low load, and forms the basis of a feasible system.

The true advantage of the ring design is seen by examining total interconnect power. The average network power shown in the figure indicates that the ring consumes 76.3% less power on average with workloads that have low network intensity, and 26.7% less power at high network intensity. Hence, even at saturation, the ring consumes less power on average. Furthermore, because power reduction (as a fraction of the baseline mesh) is greater than the performance degradation even at the highest load, the ring is the more energy-efficient design when only network energy is considered.

2.2 Scaling Limits of a Single-Ring Interconnect

Unfortunately, as network size increases, ring-based designs scale poorly, relative to mesh-based designs. This is fundamentally due to *ring contention*: a ring saturates more quickly as load increases because all traffic passes through every node between the source and destination. This traffic then prevents any node along the path from injecting additional traffic. In contrast, a mesh is able to handle higher load per network node, because it has more links per node and more path diversity.

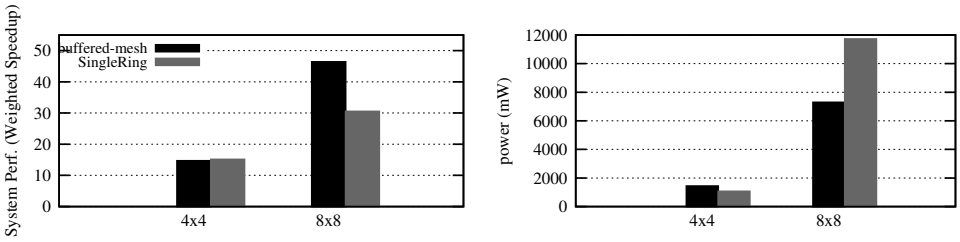
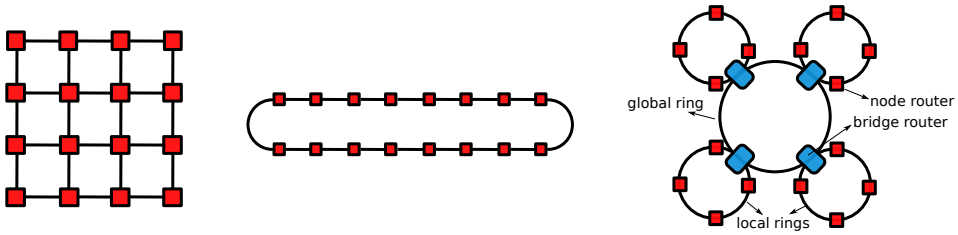


Figure 2: Application performance and average network power as network grows from 16 to 64 nodes in mesh and ring interconnects. Bisection bandwidth is equalized between mesh and ring designs.



(a) mesh (b) ring (c) hierarchical ring

Figure 3: Topology for (a) mesh, (b) ring, and (c) hierarchical ring networks.

We demonstrate the scaling limits of a single-ring design in Fig. 2. The figure shows average application performance and network energy for a set of medium-network-intensity workloads in both 16-core and 64-core configurations, for a single ring and a 2D-mesh buffered interconnect. The width of the ring interconnect is increased so that the bisection bandwidth is equivalent to the mesh design's bisection bandwidth at both network sizes, for fair comparison. As the figure shows, in the 16-node configuration, the ring performs slightly better than the mesh interconnect, again because of lower router latency. In contrast, when the network size increases, the ring performs very poorly, degrading performance by 34.2% relative to the mesh while increasing average power by 60.7% (due to increased link traversals).

We conclude that while single-ring interconnect, in which all network nodes are on the same ring, has excellent energy-efficiency and satisfactory performance for low-to-medium network load in medium-sized CMPs, performance and energy efficiency drop significantly when the network load or network size increases. Despite this shortcoming, the simplicity and efficiency of a ring-based router are attractive. **Our goal** in this paper is to devise a new NoC design that achieves the energy-efficiency of a ring router with the performance and scalability of mesh-based designs.

3 Hierarchical Ring Design for Scalability

To achieve better scalability in a ring-based interconnect than a single ring can offer, we make use of the basic principle of *hierarchy*. We combine multiple *local* rings, each of which connects a subset of the nodes in a system, and connect them with a single *global* ring. Local and global rings connect via *bridge routers*, which attach to two rings (one global and one local). This two-level structure is illustrated alongside conventional mesh and ring topologies in Fig. 3.

Advantages of Hierarchical Rings: Building an interconnect topology with hierarchy has several advantages. First, by placing each node on a local ring, its immediate impact on ring utilization is limited to only its local ring. Each local ring has fewer nodes than the single ring in a non-hierarchical design. Second, the latency to cross the chip (i.e., to send a packet between the nodes with the greatest distance between them) is reduced, because the global ring connecting each local ring acts as an "express ring." That is, the global ring has fewer routers on it (only the bridge routers), and so can move traffic from one local ring to another local ring relatively quickly. Traffic thus transferred does not contend with local traffic in other parts of the network, as it would with a single-ring design or a mesh-based design, but only contends with other long-distance traffic, and can have reduced latency relative to both single-ring and mesh-based networks. Third, when there is some locality to traffic – that is, when a node is more likely to communicate with a nearby node than a far-away node – then the hierarchical structure is better adapted to the workload than a design with a single, large ring. Even when a node is equally likely to communicate with any node in the network, a node's local ring can handle some fraction of the packets sent by that node without transferring them to the global ring (when the packets are sent to other nodes on the same local ring), reducing contention relative to the single-ring design.

Bisection Bandwidth: Note that the fundamental bisection bandwidth of a hierarchical-ring network is still constrained by the single global ring. However, in cases where there is no locality, the bandwidth of the global ring can be increased proportional to demand. This is less costly than building a high-bandwidth ring that reaches all nodes in the system, because only the bridge routers require the wider and/or faster datapaths. As we describe in detail below, we adopt a higher-bandwidth global ring to improve network performance in our evaluations.

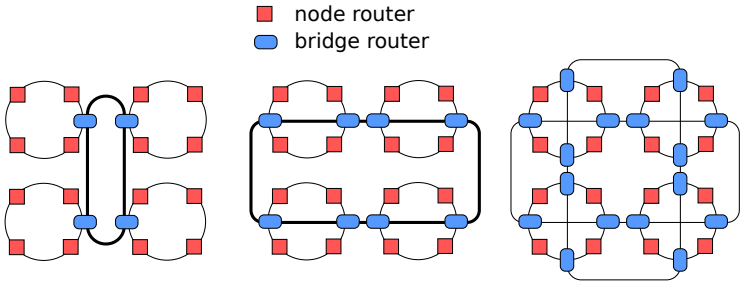


Figure 4: 4-bridge, 8-bridge and 16-bridge Hierarchical Ring topology

Bridge Router and Global Ring Configurations: When building a two-level topology, there are many different arrangements of global rings and bridge routers that can efficiently link the local rings together. The size (bandwidth) and number of global rings, as well as the number of bridges between each local and global ring, must be tuned like any other system design tradeoff. In this work, we study three particular configurations, shown in Fig. 4. We refer to these designs by the number of bridge routers in total: 4-bridge, 8-bridge, and 16-bridge. The first two designs contain a single global ring, with either one or two bridge routers for each local ring, respectively. The last, 16-bridge, design contains *two* global rings with two bridge routers per local-global ring combination. As we will show in our evaluations, these three topologies trade off between system performance and network power/area.

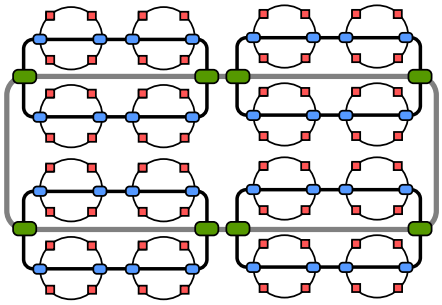


Figure 5: Three-level hierarchy for an 8x8 network.

Generalized Hierarchical Rings: Multiple Levels: Finally, note that the hierarchical structure that we propose can be extended to more than two levels; in general, rings can be composed hierarchically with bridge routers. We use a 3-level hierarchy, illustrated in Fig. 5, to build a 64-node network, and a 4-level hierarchy (not illustrated, but constructed using the same principle) to build a 256-node network.

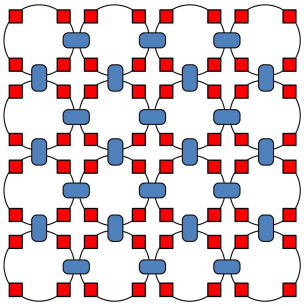


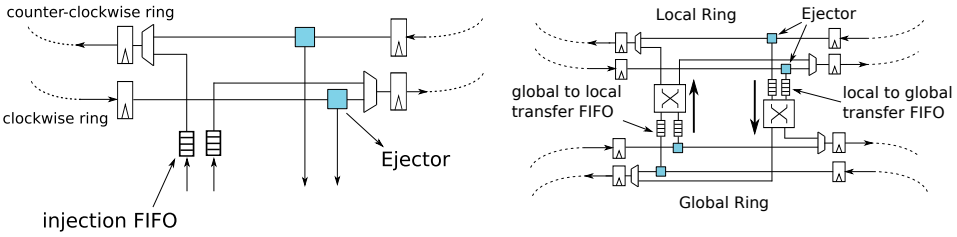
Figure 6: mesh of rings topology

Mesh Of Rings Topology: Another approach to make ring topology scalable is to place local rings in a mesh style as shown in Fig. 6. Such a topology has the same scalability as a mesh topology, both having complexity proportional to the total number of nodes in the network. The mesh of rings topology also has the benefit of simple router design since the basic building blocks: node router and bridge router have the same structure as those in a hierarchical ring network.

In §5.4, we discuss the deadlock and livelock avoidance in mesh of rings topology and we evaluate the performance and power consumption of mesh of rings in §8.

4 Hierarchical Ring Router Design

Two types of routers exist in the hierarchical ring interconnects: *node routers* connect a node (e.g., a processor) with a local ring, and *bridge routers* connect two different rings. High-level block diagrams of both routers are shown in Fig. 7. Note that both designs have very simple datapaths (in addition to the simple control logic): each router consists of one pipeline register per ring datapath, plus MUXes to inject traffic into the ring and eject traffic from the ring, and some FIFO buffering in the bridge routers.



(a) node router (b) bridge router

Figure 7: High-level block diagrams of of (a) a node router and (b) a bridge router

4.1 Node Router Operation

At each node on a local ring, we place a single node router, shown in Fig. 7a. A node router is extremely simple, because it performs a very simple function: it passes through circulating traffic, allows new traffic to enter the ring through a MUX, and allows traffic to leave the ring when necessary. Each router contains one pipeline register for the router stage, and one pipeline register for link traversal, so the router latency is exactly one cycle and the per-hop latency is two cycles. Such a design as this is very common in ring-based and ring-like designs, such as [17].

As flits enter the router on the ring, they first travel to the ejector. Because we use bidirectional rings, each node router has two ejectors, one per direction[2]. We assume that any traffic addressed to the local node can be consumed by that node immediately, so traffic is never denied ejection at a node router. After locally-destined traffic is removed from the ring, the remaining traffic travels to the injection stage. At this stage, the router looks for "empty slots," or cycles where no flit is present on the ring, and injects new flits into the ring whenever they are queued for injection. The injector is even simpler than the ejector, because it only needs to find cycles where no flit is present and MUX-in new flits in these slots. Note that we implement two separate injection buffers (FIFOs), one per ring direction; thus, in the best case, two flits can be injected into the network in a single cycle. The routing behavior for a node router is summarized in Ruleset 1.

Ruleset 1 Node Router Rules

1. Flits addressed to the local node are ejected by the router and passed to the local node.
2. All other flits arriving on the ring continue through the router.
3. When no flit continues through the router, and the FIFO for the corresponding ring direction is not empty, the router sends the flit at the FIFO head to the ring output and advances the FIFO.

4.2 Bridge Routers

Second, the *bridge routers* connect a local ring and the global ring, and exchange traffic between them. Bridge router design is important for system performance, because a poor design can bottleneck the global flow of traffic across the network. In addition, bridge router design is non-trivial, because bridge routers transfer traffic to a new ring, and traffic that wishes to enter a ring must yield to traffic already in the ring: if a flit in one ring requires a transfer when it reaches the bridge router, it is not guaranteed that the other ring will have an open slot at that instant. Bridge routers can either *deny a transfer* if no such slot is available, or *buffer the transferring data* until space becomes available in the receiving ring. Note that even if a bridge router contains transfer buffers, it may still refuse transfers when the buffer becomes full. When a flit is refused transfer, it simply remains in the origin ring and continues to circulate; it will try again when it again reaches the bridge. This is analogous to a *deflection* in hot-potato routing [1], also leveraged in recent mesh interconnect designs [21]. In our design, we make use of buffered bridge routers (which buffer first, then deflect

[2]For simplicity, assume that up to two ejected flits can be accepted by the processor or reassembly buffers in a single cycle. For a fair comparison, we also implement two-flit-per-cycle ejection in our mesh-based baselines.

if full) rather than deflection-only bridge routers so that deflections do not cause the network to saturate. We will show sensitivity to this buffer size in §8.5.

A high-level block diagram of a bridge router is shown in Fig. 7b. At an abstract level, a bridge router resembles two node routers, one on the local ring and one on the global ring. These two ring interfaces are joined by FIFO buffers in both directions, used as described above to allow for flit transfer.

Note that the bandwidth ratio between local and global ring is a system parameter that can be tuned for performance. If the global ring has higher bandwidth, it is wider than the local ring (assuming both rings run in the same clock domain). A wide ring can be considered as several independent rings, each with a width equivalent to a single local ring. A bridge router must choose into which global ring slice to transfer a flit, and must be able to transfer a flit from any of the global ring slices out into the local ring. We use a transfer crossbar to allow such transfers into or out of any global ring slice[3]. Bridge router operation is summarized in Ruleset 2.

Ruleset 2 Bridge Router Operation

1. **Local Pass-through:** Flits arriving on the local ring that are addressed to a destination on that local ring remain on the ring.
2. **Global Pass-through:** Flits arriving on the global ring interface that are addressed to a destination not on this bridge router's local ring remain on the global ring.
3. **Local Ejection:** Flits arriving on the local ring that are addressed to any destination on another local ring are ejected from the local ring and placed into the global ring injection FIFO, if it is not full. If the FIFO is full, the flit is *deflected.*
4. **Global Injection:** When a flit is present in the global ring injection FIFO, and no flit arrives on the global ring interface, the flit waiting to inject may be injected.
5. **Global Ejection:** Flits arriving on the global ring that are addressed to a destination on the local ring are ejected from the global ring and placed into the local ring injection FIFO, if it is not full. If the FIFO is full, the flit is *deflected.*
6. **Local Injection:** When a flit is present in the local ring injection FIFO, and no flit arrives on the local ring interface, the flit waiting to inject may be injected.
7. **Dual-Transfer Swap:** As a special case, when a flit on the local ring and a flit on the global ring wish to eject simultaneously, and hence create empty slots for other flits to be injected, they may do so even if both FIFOs are full. This rule eliminates full-buffer deadlock.

Of particular note is the *Dual-Transfer Swap* rule. This rule allows flits from the global and local ring to swap places without "temporary space," or empty FIFO slots, available. It avoids the deadlock situation that could otherwise occur if both FIFOs were full and both rings were full with flits that required a transfer to the other ring. In this hypothetical deadlock scenario, no flit would be able to eject, because the transfer FIFOs are full, even though by ejecting, they would immediately create an empty slot in the network for another flit to inject and make forward progress. This deadlock is avoided by allowing ejection when it is known that the other end of the FIFO will be able to inject. We will discuss the implications of this rule in more detail in §5 as we discuss the livelock-free forward progress guarantee that our mechanism provides.

5 Guaranteed Delivery: Correctness in Hierarchical Ring Interconnects

For the interconnect to operate correctly, it must guarantee that every flit in the network is eventually delivered. Failure to deliver a flit (or packet) might occur for two reasons: *deadlock* or *livelock*. Deadlock occurs when the network reaches a state where some traffic cannot move at all. In contrast, livelock occurs when traffic continues to move, but never reaches its destination.

In a single-ring system, neither of these conditions can occur: once flits enter the ring, they continue to move (thus, never deadlock), and always move closer to their destination (thus, never livelock), arriving in a finite amount of time. However, in a multi-ring system, *inter-ring transfers* create both problems. First, a flit in a inter-ring transfer buffer, waiting to inject into a new ring, might be stuck forever if the new ring is fully utilized and never has space for another flit to inject. This is a case of *deadlock*, because the transferring flit never moves. Second, a flit may never enter the inter-ring transfer buffer in the first place, if the buffer is always full (or periodically empties but is filled

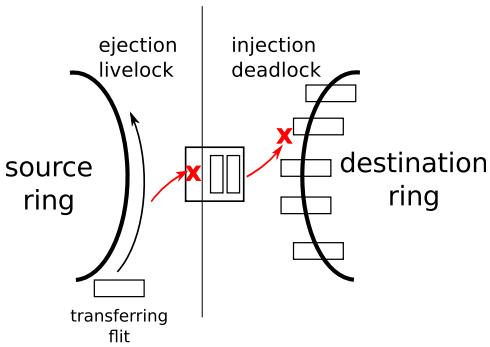


Figure 8: Ejection livelock and injection deadlock in a multi-ring system.

[3]Note that although such a router design uses a crossbar, as a conventional mesh router does, (i) the crossbar in our design is typically smaller, e.g. a 2x4 crossbar to join a one-lane bidirectional local ring to a two-lane bidirectional global ring, compared to a 5x5 crossbar in a 2D-mesh router, and (ii) a hierarchical ring design will typically have fewer bridge routers than the number of network nodes, so the cost of the crossbars is amortized over more nodes.

again before the unlucky flit arrives at the bridge router). This is a case of *livelock*, because the flit that wishes to transfer rings will circle its original ring forever, denied a transfer each time it reaches the bridge router.

To address both of these problems, we propose two mechanisms, the *Injection Guarantee* and the *Ejection Guarantee*. The injection guarantee ensures that when flits are in a transfer buffer, waiting to inject, they will eventually be able to do so. It thus prevents injection deadlock. The ejection guarantee ensures that when flits are circling in a ring, waiting to eject into a transfer buffer, they will eventually be able to eject (as long as the transfer buffer makes forward progress). It thus prevents ejection livelock. An abstract view of both problems is shown in Fig. 8.

5.1 Preventing Injection Deadlock: Injection Guarantee

The *injection guarantee* ensures that every router on a ring can eventually inject a flit, as long as the flits on the ring are eventually consumed (to produce space for new traffic to enter). We observe that within a single ring of reasonable size, it is feasible to globally coordinate among all routers to solve injection deadlock. When a single router cannot inject after a certain number of contiguous cycles, this router sends a signal using dedicated wires to the rest of the ring to stop injecting. Flits already in the ring will eventually drain from the ring, creating space for the deadlocked flits to enter. This coordination is controlled by a central "injection coordinator" in each ring, a very small logic structure that requires only two wires to each router on the ring: a *injection request line* to the coordinator, and an *inject-inhibit line* from the coordinator to each router. Each injection request line requests the special injection-guarantee mode, and each inhibit line prevents a router from injecting, even if it could otherwise inject. Finally, to simplify implementation, one coordinator can be used for the entire network. By inhibiting all new traffic injection (except for the injection that had been blocked), the entire network quickly drains and creates space for the blocked traffic to be injected.

Algorithm 3 Injection Guarantee Mechanism

```
Each router keeps a counter for each ring injection point
counter = 0
At router, at each cycle:
if router has a flit to inject but was not able to inject the flit then
    counter ← counter + 1
else
    counter ← 0
end if
if counter ≥ starvation_threshold then
    assert injection request line to coordinator
else
    de-assert request line
end if
At a router, when inhibit wire from coordinator is asserted:
do not inject new traffic until line is de-asserted
At coordinator, when any injection request wire is asserted (i.e., system is in injection-guarantee mode):
select one requesting router among all that are asserting their respective lines, breaking ties with round-robin arbitration
assert injection inhibit line to all routers except selected router
wait until selected router de-asserts injection request line
de-assert all injection inhibit lines
```

Algorithm 3 captures the injection-guarantee implementation in each ring router and in the central injection coordinator. The main idea behind this mechanism is to allow the ring to operate freely until a problem occurs. Thus, in the common case, the mechanism has no impact. Furthermore, the implementation of the coordinator does not need to be optimized for speed, because the injection-guarantee mode will be invoked rarely. As long as flits on the ring are eventually consumed, this mechanism guarantees that any node will eventually inject: if a node cannot inject for more than a certain number of cycles, it asserts its request line, and the round-robin arbitration at the coordinator guarantees that any request will be chosen if asserted long enough. Once the system enters injection-guarantee mode, all node routers except the chosen router immediately stop injecting, and do not inject until the chosen router is able to inject, at which point the chosen router de-asserts its injection request wire and the system leaves injection-guarantee mode.

5.2 Preventing Ejection Livelock: Ejection Guarantee

The second guarantee that must be provided is that a bridge router can eventually eject any flit that requires a transfer. In other words, the network should ensure that a flit does not circle forever in a ring waiting for a transfer. Recall that a bridge router cannot eject a flit for transfer if its buffer is full. For the purpose of this guarantee, we assume that injection deadlock does not occur (as we described above), and so the buffer is able to inject flits and create space to

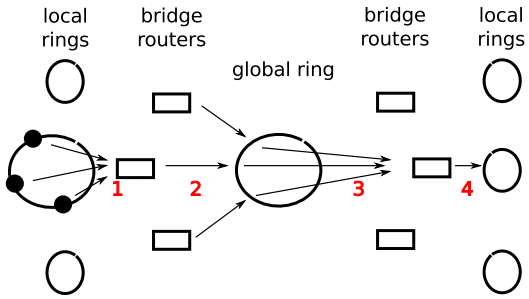


Figure 9: End-to-end delivery guarantee: (1) the ejection guarantee ensures that flits injected from nodes on a local ring are ejected for transfer at the bridge to the global ring. (2) The injection guarantee ensures that flits in the bridge router's FIFO are injected into the global ring. Similarly, the ejection guarantee (3) and injection guarantee (4) ensure traffic moves from the global ring to its destination local ring. The Dual-Transfer Swap rule breaks any circular dependencies.

accept new flits. Given that the buffer will continue to have free space for new transferring flits, we must only ensure that the buffer *fairly chooses* which flit in the source ring to eject when it has an opening available.

We provide this guarantee of fair ejection by implementing an *observer mechanism* at every bridge router. This observer mechanism "observes" one flit at a time, by examining the flit in the bridge router's ring pipeline register at regular intervals (once per full-ring latency). The observer counts the number of times the same flit passes through the router (a flit will pass through more than once if the flit is denied a transfer). The observer takes explicit action by *reserving a bridge buffer space* if this count exceeds a threshold value. The observer chooses which flit to track in such a way that if any flit stays in the ring long enough, it will eventually become the observed flit; if it continues to be denied ejection, it will have a reserved transfer-buffer space and will make forward progress. Algorithm 4 describes this behavior in more detail.

Algorithm 4 Ejection Guarantee: at each bridge router

Given: N_{slots} slots in the ring
Initialization:
$retry\_counter \leftarrow 0$
start observing the flit on the ring pipeline register in the router, if any
At the cycle when the observed flit should pass by on the ring:
if no flit is present in the ring pipeline register, **or** the flit does not wish to eject here, **or** the flit successfully ejects **then**
  start observing the flit that comes in the next cycle
  $retry\_counter \leftarrow 0$
else
  $retry\_counter \leftarrow retry\_counter + 1$
  if $retry\_counter \geq retry\_threshold$ **then**
    prevent any other flit from ejecting at this router until observed flit ejects
  end if
end if

5.3 Putting it Together: Guaranteed Delivery

We demonstrate that the combination of the *ejection guarantee* and *injection guarantee* mechanisms presented in this section provides an end-to-end delivery guarantee. To see why, we will argue why forward progress must occur at every step of a flit's journey from source to destination. Fig. 9 shows an abstract view of the traffic flow through a two-level hierarchical ring network.

Note that at each stage of a flit's journey, forward progress is guaranteed by either the ejection guarantee (when entering a bridge router) or the injection guarantee (when leaving a bridge router). Furthermore, at each stage, forward progress depends on the flits ahead of a given flit also making progress. For example, the ejection guarantee *into* a bridge only works as long as the injection guarantee *out of* that bridge can continue to move flits into the destination ring. Likewise, the injection guarantee into a ring only works as long as the ejection guarantee ensures that all flits currently on the ring will eventually leave it, creating space for new traffic.

This chain of guarantees naturally ensures forward progress as long as no dependency cycles are created. However,

in the situation where two rings A and B are joined by a bridge router, and both rings are filled with flits that require a transfer to the other ring, the ejection and injection guarantees alone cannot ensure forward progress, because each depends on the other to create empty space for flits to move forward. In this case, the *Swap Rule* (in Ruleset 2) implemented by the bridge router resolves the deadlock: this rule allows two flits at either end of the bridge to trade places directly, without progressing through the bridge FIFOs in each direction, when both flits require a bridge transfer. Note that in a hierarchical topology, where flits can only transfer from local to global rings and vice versa, the only such transfer cycles occur between two directly-connected rings. This rule thus breaks dependence cycles and allows forward progress.

5.4 Deadlock and Livelock in Mesh-of-Rings Topology

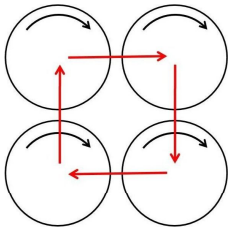
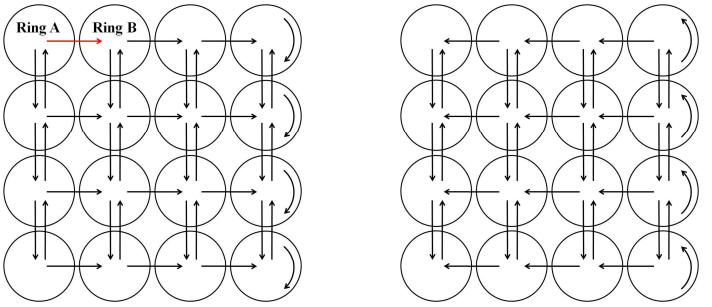


Figure 10: A livelock situation in Mesh of Ring topology

In the mesh-of-rings topology, since each local ring can be connected to multiple other local rings, deadlock can occur due to circular dependencies between rings[4]. One example is shown in Fig. 10. In this situation, all flits in each local ring need to enter another local ring to reach their destination, but none of the flits can enter as the target local ring is already full. These flits are thus forever stuck in the network and can never make forward progress. In contrast, this kind of deadlock can never happen in a hierarchical ring topology where hierarchies of rings form a tree structure and no circular dependency between rings exists.



(a) Clockwise Ring Network (b) Counter-Clockwise Ring Network

Figure 11: allowed data tranfer directions in mesh of rings topology

To prevent such a deadlock problem, one possibility is to forbid certain inter-ring transfers. Notice that each because each local ring is bidirectional, it effectively consists of two independent rings, the clockwise and counter-clockwise directions. To ensure that no deadlock occurs, we only need to guarantee that no circular dependencies exist between individual uni-directional rings (analogous to virtual channels). For our evaluations of mesh-of-rings, we use the scheme shown in Fig. 11. The arrows in the figure represent the directions that flits are allowed to move. For example, the red arrow in Fig. 11a means that if a flit is traveling along a ring in the clockwise direction, it can only transfer from ring A to ring B but not from ring B to ring A. Notice that there are no circular dependencies, so no deadlock can occur.

[4]Though flits are still moving in local rings in this situation, we call it deadlock rather than livelock for it is similarity to the deadlock situation in the baseline buffered mesh topology.

To guarantee livelock freedom (injection and delivery), the same mechanisms in Algorithm 3 and Algorithm 4 can still be used in the mesh of rings topology. Generally, for all deflection and ring-to-ring communication based networks, the two algorithms introduced above can always be used to guarantee livelock freedom.

One drawback noticed in Fig. 11 is that the bisection bandwidth of the mesh of ring topology is different between horizontal and vertical directions. The horizontal bisection bandwidth is only half of the vertical bisection bandwidth. Thus the horizontal communication bottlenecks the whole network. In §8, we will further discuss the performance of the mesh of ring topology compared to other topologies.

6 Scalability

6.1 Scaling Approaches in Hierarchical Ring Networks

Using a hierarchical ring topology allows for multiple ways to scale the network to larger sizes. In this section, we discuss scalability of this topology, and show that it is more scalable than conventional 2D-mesh networks or single-ring networks.

In general, there are three possible ways to scale the network: (i) increase the number of nodes connected to the local ring, (ii) increase the number of local rings connected to the global ring, or (iii) increase the number of levels in the hierarchy. The first two methods yield the same scalability as a single-ring topology, because they simply increase the number of routers in the ring. As we have already motivated, this type of scaling hits a performance bottleneck after a certain node count is reached. Rather, in this work, we use *hierarchy* to allow for greater scalability; hence, we focus on the third mode of scaling, adding more levels to the hierarchy. This approach is demonstrated by the example configuration in Fig. 5, which shows an 8x8 (64-node) network consisting of three levels of hierarchy.

For the rest of this section, we discuss and compare the scalability of mesh, single ring, hierarchical ring and mesh of rings topology respectively. Specifically, we talk about area and power scalability in §6.2 and latency scalability in §6.3.

6.2 Area and Power Scalability

Router Scalability

Because we expect the simple ring routers to be wire-dominated, we use crossbar area to model router area in this section. We thus assume that control logic is small in all routers, which is a conservative assumption for our work, since ring-based routers tend to be simpler than the baseline mesh-based routers against which we compare. We also assume that the total power of a router is proportional to the router area, which is true as long as the activity factor remains the same as the network scales up. Hence, the results we show here apply both to area scalability and power scalability equally.

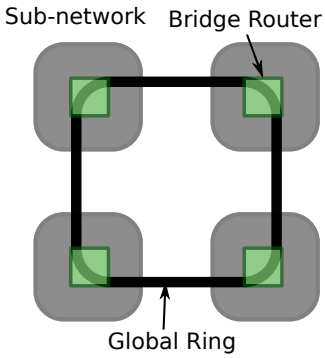


Figure 12: Hierarchical ring topology scaled to 64 cores

First, we show the induction step that takes a given size hierarchical-ring network and builds the next larger size, which is doubled in width and height (quadrupled in node count). Fig. 12 adds one more level of hierarchy to the hierarchical ring topology. The gray blocks represent a sub-network which is an n-by-n hierarchical ring network.

Thus, the whole network in the figure is thus a $2n$-by-$2n$ hierarchical ring network. The highest level global ring connects these sub networks through bridge routers. Since we maintain bisection bandwidth that is equivalent to the same-sized mesh network, where bisection bandwidth is proportional to the number of nodes in a row or column, the global ring bandwidth must double with each level of hierarchy.

We now model the total router area (and thus power) in such a network with a recurrence relation. Assume the total router area in each sub-network (gray blocks in Fig. 12) is $S(n)$. The router area of the whole network is the sum of each sub-network router area and the area of the bridge routers connected to the global ring.

$$S(2n) = 4S(n) + S_{global\ ring} \quad (1)$$

As mentioned above, the global ring bandwidth is always proportional to the network dimension, $2n$ in this case. And the number of bridge routers on the global ring is a constant which does not change with the size of the network. The total area of all the bridge routers on the global ring should be proportional to n^2. So we have

$$S(2n) = 4S(n) + Kn^2 \quad (2)$$

From this, we derive that $S(n) = Cn^2 \log n$, where C is a constant number determined by the number of bridge routers on each level of global rings. With similar methods, we can also derive total router area of other topologies as shown in Table 1.

	Mesh	Single Ring	Hierarchical Ring	Mesh of Rings
Scalability	n^2	n^4	$n^2 \log n$	n^2

Table 1: Complexity Scalability Comparison Between Different Topologies

The mesh topology has the best scalability of n^2 since router size does not change when the network scales. So the total router area of a mesh network is proportional to the number of routers on the whole chip, which is n^2. The same analysis also works for the mesh of rings topology. For a singe ring topology, since we keep the bisection bandwidth a constant across all the topologies, the bandwidth of the ring is proportional to the network dimension n, so each router area scales as n^2, the total router area of the network is thus n^4.

Dimension	Mesh	Single Ring	Hierarchical Ring	Mesh of Rings
4	$400w^2$	$256w^2$	$192w^2$	$104w^2$
8	$1600w^2$	$4096w^2$	$1280w^2$	$496w^2$
16	$6400w^2$	$65536w^2$	$7168w^2$	$2144w^2$

Table 2: Scaling of Router Area for Different Topologies

The total area of routers in different topologies as a function of the network dimension is shown in Table 2, again assuming that router area is dominated by crossbar area. w is the physical width of a link, so the area for a 1x1 crossbar is w^2. For a small network of 4x4, both the single ring and hierarchical ring have a smaller crossbar area than a mesh topology. However, when the network becomes larger, single ring topology soon becomes too complex to implement. For hierarchical ring topology, the total router area grows only a little faster than a mesh topology. But for smaller network sizes, the total area is smaller or very close to a mesh topology. Among all the topologies, mesh of rings has the smallest router area and the area of each router does not increase when the network scales.

Link Scalability

We use the total link area to model the link complexity. The link area is calculated as the link length times link physical width. The total link area for different topologies as a function of network dimension is shown as Table 3. s is the area for a 2.5 mm long and 128-bit wide link. Network configurations for different topologies are in Table 6. It is clear that the hierarchical ring topology has a much lower link area than a single ring topology.

Dimension	Mesh	Single Ring	Hierarchical Ring	Mesh of Rings
4	48s	64s	44.8s	64s
8	224s	512s	339.2s	256s
16	960s	4096s	2060.8s	1024s

Table 3: Scaling of Link Area for Different Topologies

6.3 Latency Scalability

	Mesh	Single Ring	Hierarchical Ring	Mesh of Rings
Scalability	n	n^2	n	n

Table 4: Zero-load Latency Scalability Comparison Between Different Topologies

When the network scales, the radix of the network may also change, making the zero load latency longer. Zero load latency describes the network latency when there is no contention in the network, and is important for the performance of the whole chip. The scalability of zero load latency for mesh, single ring, hierarchical ring topologies and Mesh of Rings are listed in table 4. The maximum zero-load latency in a mesh or mesh of ring topology is proportional to the dimension n of the network. For a hierarchical ring, the latency on the global ring is proportional to n. The maximum total latency is the sum of latency on all level of rings, which is still proportional to n, as shown in Eq. 3.

$$total\ latency \propto 1+2+\cdots+n/2+n+n/2+\cdots+2+1 \propto n \quad (3)$$

For a single ring topology, the zero-load latency is proportional to the number of nodes in the network which is n^2. So hierarchical ring has the same scalability as mesh in terms of zero-load latency while both are better than single ring topology.

7 Methodology

We perform our evaluations using a cycle-accurate simulator to provide application-level performance results. The simulator is an in-house x86 CMP simulator. Each CPU core faithfully models stalls due to memory latency, and a cache hierarchy with a realistic cache coherence protocol is modeled on top of the cycle-accurate interconnect model. Instruction traces for the simulator are taken using a Pintool [20] on representative portions of SPEC CPU2006 workloads [27]. System parameters for our evaluations are given in Table 5. Parameters for each interconnect design are given in Table 6.

We compare mesh, single-ring and hierarchical-ring topologies. To fairly compare topologies, we hold bisection bandwidth constant across all designs. In addition, the total buffer capacity of the three different hierarchical ring topologies is held constant.

Workloads: We evaluate each system design with a set of 105 multiprogrammed workloads. Each workload consists of one single-threaded instance of a SPEC CPU2006 [27] benchmark on each core, for a total of either 16 (4x4) or 64 (8x8) benchmark instances per workload. Multiprogrammed workloads such as these are representative of many common workloads for large CMPs. Workloads are constructed at varying network intensities as follows: first, benchmarks are split into three classes (Low, Medium and High) by L1 cache miss intensity (which correlates directly with network injection rate), such that benchmarks with less than 5 misses per thousand instructions (MPKI) are "Low-intensity," between 5 and 50 are "Medium-intensity," and above 50 MPKI are "High-intensity." Workloads are then constructed by randomly selecting a certain number of benchmarks from each category. We form workload sets with seven intensity mixes: High, High-Medium, High-Medium-Low, High-Low, Medium, Medium-Low, and Low, with 15 workloads in each. We also evaluate all network designs using synthetic traffic patterns: uniform random, bit-complement, and transpose [4]. With each pattern, we sweep injection rate from zero to saturation.

While some other works use multithreaded workloads, there is little fundamental difference at the network level between coherence traffic and traffic from threads with independent working sets. In addition, synthetic traffic shows fundamental enhancements in network performance regardless of workload.

Energy and Area Modeling: We model router and link energy and area using both ORION 2.0 [31] as well as synthesized Verilog models with a commercial 65nm design library. To model routers, we individually model the crossbar, pipeline registers, and/or other datapath components, buffers, and control logic. We synthesize Verilog models of control logic to obtain area and power/energy results for these components. ORION provides the remainder of the modeling.

We assume a 2.5 mm link length for mesh and single ring topologies. For the hierarchical ring design, we assume 1 mm links between local-ring routers, since the four routers on a local ring can be placed at four corners that meet in a tiled design; global-ring links remain at 2.5 mm on average.

Application Evaluation Metrics: We present application performance results using the commonly-used Weighted Speedup metric [26] (representing system throughput [7]), which calculates performance of each application in a multi-programmed mix relative to how well the application performs when running alone on the system: $WS = \sum_{i=1}^{N} \frac{IPC_i^{shared}}{IPC_i^{alone}}$. We take an application's performance running on a baseline CHIPPER network as a common baseline when computing weighted speedup on all topologies, in order to allow for fair comparison across topologies. To demonstrate that our results are robust across metrics, we also report average system IPC.

Parameter	Setting
System topology	CPU core and shared cache slice at every node
Core model	Out-of-order, 128-entry ROB, 16 MSHRs (simultaneous outstanding requests)
Private L1 cache	64 KB, 4-way associative, 32-byte block size
Shared L2 cache	perfect (always hits), cache-block-interleaved mapping
Cache coherence	directory-based protocol, directory entries colocated with shared cache blocks
Simulation length	5M cycle warm-up, 25M cycle active execution

Table 5: Simulation and system configuration parameters.

Parameter	Network	Setting
Interconnect Links	buffered	1-cycle latency, 64-bit width
	CHIPPER	
	single ring	**4x4:** 128-bit width, **8x8:** 256-bit width
	Hierarchical Ring	**4x4:** 2-cycle (local), 3-cycle (global) per-hop latency; 64-bit local ring, 128-bit global ring; **8x8:** four copies of 4x4 design, with second-level rings connected by 256-bit third-level ring (Fig. 5); **16x16:** four copies of 8x8 design, joined by 512-bit fourth-level ring.
router	buffered	3-cycle per-hop latency, 8 VCs, 8 flits/VC, buffer bypassing [30], age-based arbitration
	CHIPPER	3-cycle per-hop latency [8]
	single ring	1-cycle per-hop latency (as in [18])
	HR 4-bridge	local-to-global buffer depth 2, global-to-local buffer depth 8
	HR 8-bridge	local-to-global buffer depth 1, global-to-local buffer depth 4
	HR 16-bridge	

Table 6: Network parameters.

8 Evaluation

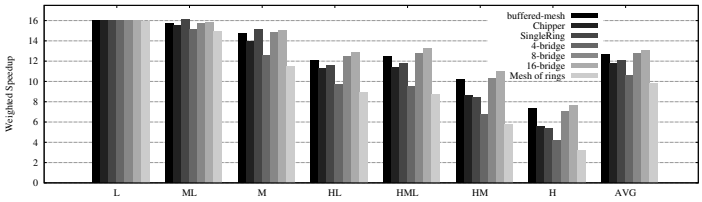
In this section, we demonstrate that a hierarchical ring-based design effectively improves the performance and scalability of a single-ring interconnect, and matches the performance of conventional 2D mesh interconnects, while significantly reducing network power. First, in §8.1, we present application performance results to show the impact of each interconnect design on system performance. Then, in §8.2, we use a network power model that takes both static and dynamic (network load-dependent) power into account, and show how a hierarchical ring design preserves much of the power reduction of single-ring routers over larger, more complex mesh routers. In §8.3, we show behavior with synthetic traffic patterns. In §8.4, we make use of hardware models to show that the simpler routers in ring-based designs lead to higher operating frequency and significant area savings. Finally, in §8.5, we examine the sensitivity of these results to several parameters.

8.1 Performance

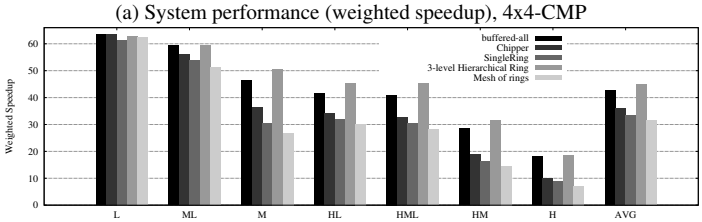
Fig. 13a shows system application performance (measured as weighted speedup) for a 4x4 tiled CMP with seven interconnect configurations: a buffered mesh baseline, a bufferless deflection mesh network, CHIPPER [8], a baseline single-ring design, three variants of a two-level ring hierarchy (4-, 8-, and 16-bridge-router variants) and a mesh of ring design. Results are split by workload category, as described in §7, with less network-intensive workloads to the left and more network-intensive to the right. At the lowest network intensity, all topologies perform with perfect weighted speedup, i.e., there is no network contention. However, as network load increases, differences in network capacity and performance begin to appear. Relative to a conventional buffered mesh interconnect, the CHIPPER low-cost bufferless network design, which significantly reduces the cost of a mesh interconnect but also significantly degrades application performance in the worst case, leads to 7.1% performance reduction on average, and up to 23.8% in the worst case (all-High workloads).

A single-ring topology performs very well at low network loads. The ring's increased performance relative to a mesh network is due to the much lower per-hop latency: as indicated by the microarchitecture that we described in §4, each hop takes only a single cycle. In contrast, a router in a mesh must calculate a routing decision for each incoming flit, allow all incoming flits to arbitrate for outputs, and transfer flits across a large crossbar with significant wire-dominated delay.

However, the single-ring design begins to degrade in performance past Medium-intensity workloads (third bar-group). As discussed in previous sections, the bisection bandwidth of a ring becomes the interconnect bottleneck when load rises past a certain threshold. In High-Medium intensity workloads and above (the second-highest-intensity category), a single ring interconnect performs slightly worse than the low-cost bufferless mesh design, CHIPPER. Hence, even though the single-ring design allows for attractive hardware reductions relative to any mesh design, it loses significant performance.



(a) System performance (weighted speedup), 4x4-CMP



(b) System performance (weighted speedup), 8x8-CMP

Figure 13: Application performance comparison for 4x4 and 8x8 CMPs.

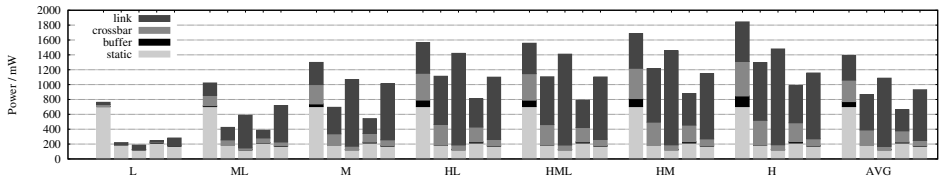
In contrast, the hierarchical ring design shows very strong performance results for all workload intensities. In all categories except High-intensity, the 8-bridge hierarchical design performs as well as or better than the buffered mesh design; in the highest-intensity workload category, it reduces performance by only 4.7% on average. Over all workloads, the average performance is increased by 0.6% relative to a buffered mesh and 8.3% relative to a bufferless mesh, and the design shows significant average power and die area reduction, as we show in the next section. The performance gains seen over both meshes and single-ring designs are due to the "express network" effect of the global ring: sending long-distance traffic across the network on the global ring eliminates contention with nodes along the traffic's path, a problem that exists in meshes and even more so in single-ring designs. In addition, the reduced per-hop latency of a simple ring router gives the hierarchical ring design an advantage over mesh-based interconnects across the load spectrum, just as it does for a single-ring design at low loads.

To ensure that our results are robust across metrics, we also examine average IPC. We find that the hierarchical ring design improves IPC by 0.6% relative to the baseline buffered result, and 7.8% relative to the bufferless mesh (CHIPPER).

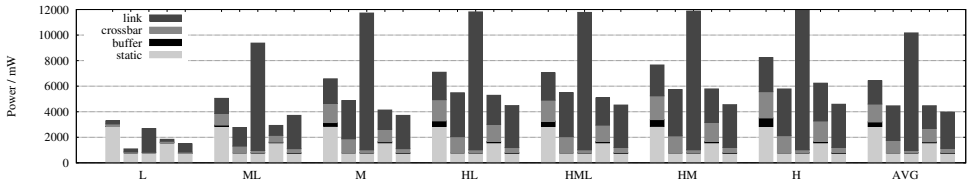
Finally, we compare three variants of the hierarchical ring design, altering the number of bridge routers to trade off performance and hardware cost. The results in Fig. 13a clearly show that an increased number of bridge routers leads to better performance. Performance reduces when there are too few bridge routers because the ring-transfer point becomes a network bottleneck. Even if the global ring is provisioned appropriately to allow the network the same bisection bandwidth as the equivalent 4x4 mesh design, forcing all traffic that leaves a local ring to transfer through one point can cause local congestion. In contrast, allowing two transfer points between each local ring and the global ring, which corresponds to our 8-bridge-router design, performs very well, as described above, exceeding the performance of the buffered mesh in all but one workload intensity category. Finally, adding 4 bridge routers per local ring, or 16 total, allows 7.9% performance gain over the buffered mesh network in the best case, the High-Medium-intensity workload category.

Fig. 13b shows the same set of experiments run in an 8x8 network. The 8x8 hierarchical ring design uses 3 levels of hierarchies and each global ring has 8 bridge routers connecting to its local rings. The buffer size in level-2 bridge routers is twice as the buffer in level-1 bridge routers. The performance reduction from the buffered mesh to the low-cost bufferless deflection mesh design is increased, as also observed in the original CHIPPER proposal [8], because network congestion increases as the bisection bandwidth of the mesh becomes more of a bottleneck. The bisection bandwidth bottleneck is even more severe in the single ring, though, which performs worse than the bufferless mesh even in the Low-intensity category, and reduces performance by 51.4% relative to the buffered mesh and 11.6% relative to the bufferless mesh in the High-intensity category. In contrast, with a hierarchical-ring design, the system again performs better than the buffered mesh in all but the High category. On average in these 8x8 workloads, the hierarchical ring increases performance by 5.0% relative to the buffered mesh.

In both 4x4 and 8x8 networks, the mesh of rings topology has the worst overall performance among all the designs. One reason for this poor performance is the unbalanced (horizontal vs. vertical) bisection bandwidth due to deadlock-freedom ring transfer constraints, as mentioned in §5.4. Because of this, the bandwidth of such a topology is not fully utilized. Another reason for the poor performance is more contention in the local rings at the center of the network: higher deflection rates lead to similar contention issues as bufferless mesh designs when cross-chip traffic



(a) Average Network Power, 4x4-CMP



(b) Average Network Power, 8x8-CMP

Figure 14: Network power for buffered and bufferless mesh, single and hierarchical ring and mesh of rings networks.(from left to right)

moves through rings at the center of the network. In contrast, a hierarchical design moves cross-chip traffic across the network independently of the local rings in the center of the network; the global ring(s) allow such traffic to "fly over" most of the network and eventually land in a local ring near its destination.

8.2 Network Power Consumption

Fig. 14a shows average network power consumption for each of the topologies and designs that we examined above. Each bar is broken down into its constituent components: static power, buffer read/write power, crossbar (or MUX) power, and link power. The buffered mesh interconnect consumes the greatest average power for a given network load. Static power (lowest bar section) is high because buffers constitute a large portion of such a router and have significant leakage at 65nm. At very low load, almost all of the increased power in a buffered mesh relative to other designs is due to buffers. The baseline bufferless mesh design and both ring-based designs have lower power at near-zero load because the simpler router designs are smaller (hence have lower leakage). As load increases, dynamic router and link energy becomes significant. The baseline buffered mesh has increased buffer power: buffer accesses are relatively energy-intensive, and at each hop, a flit must be written to, and later read from, a buffer slot. Crossbar power increases in both buffered and bufferless mesh designs, and this constitutes significant overhead over ring-based designs: because crossbar overhead scales with the square of the number of inputs or outputs, the crossbar/MUX logic in a mesh router with a 5x5 crossbar (4 neighbors and 1 local port) is over four times as expensive as the 2-input, 2-output inject/eject MUX arrangement in a ring design. Note that bridge routers require larger crossbars to allow transfer between all ring directions on both sides of the bridge; however, a bridge traversal occurs at most twice per flit in a two-level hierarchy, so the impact of these crossbars is reduced. For all of these reasons, the simplicity of a ring router relative to a mesh router leads to significant power reduction in the router itself.

Finally, we note that at high load, single-ring designs have a large average network power consumption, nearly equivalent to a buffered mesh router. An examination of the power breakdown for each router shows that *link* power dominates in the single-ring design at high load. Increased link power is both due to wider links (which we implement in order to maintain constant bisection bandwidth for fair performance comparison) and a higher hop count, on average, than a mesh. (Link power also increases in the deflection network, due to extra hops at when the deflection rate increases at high load, but not to the degree seen in the ring.) Adding *hierarchy* significantly reduces dynamic link power relative to the single-ring design, because the global ring acts as an "express network," moving flits long distances with fewer hops. When moving from a single-ring design to a hierarchical-ring design, this link power reduction is much greater than the increase in crossbar power (at bridge routers) and static power.

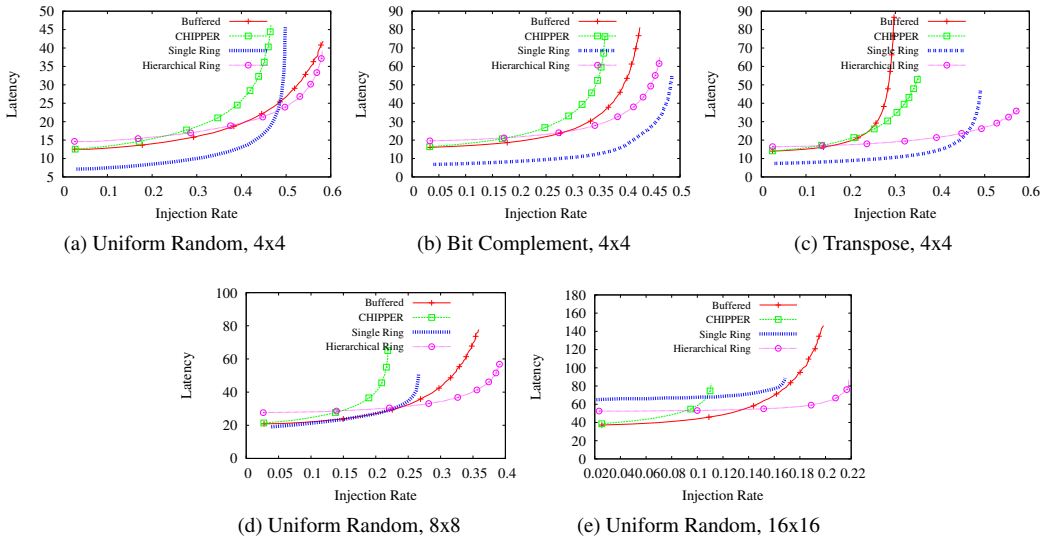
Fig. 14b shows network power consumption for 8x8 (64-node) designs. While trends are similar in general to our results in 16-node networks, a notable increase is seen in single-ring networks, largely due to link power. This increase is due to wider links necessary to preserve equivalent bisection bandwidth to the mesh, which in turn is necessary to maintain acceptable performance. Thus, we conclude that single-ring designs do not scale to large node counts in an energy-efficient way.

On the other hand, the mesh of rings network has very simple node routers and bridge routers. Unlike hierarchical ring networks, all the bridge routers can be very small, and do not need to scale larger as the network scales to accommodate more cores. This results in very low router energy which is close to that in a single ring network. However, the link energy in mesh of rings topology is considerable due to more deflections in the network.

To summarize, hierarchical rings draw less power on average than either a buffered mesh, reducing by 52.3% (30.4%) in a 4x4 (8x8) network. A bufferless mesh draws less power on average, but hierarchical rings still reduce power by 23.6% in a 4x4 network. In an 8x8 network, our design increases power over a bufferless mesh by 0.2%, but this is negligible when taken in context with the significant performance improvement over bufferless networks. The power reductions shown by a hierarchical ring are largely due to eliminating crossbar power and buffer power. Furthermore, hierarchical rings draw less power on average than a single-ring design, reducing by 39.1% (56.1%) in a 4x4 (8x8) network, because dynamic link power is significantly reduced. Combined with the competitive application-level performance shown in the previous subsection, we conclude that a hierarchical ring is a significantly better point in the tradeoff space between performance and power than any of the other designs that we evaluate. We will explicitly show this tradeoff space in §8.6 below.

8.3 Synthetic-Traffic Network Behavior

Fig. 15 shows latency as a function of injection rate for buffered and bufferless mesh routers, a single-ring design, and a hierarchical ring design in 16-, 64-, and 256-node systems. Results in 16-node systems (4x4) are shown for three different synthetic traffic patterns: uniform random traffic, bit complement (in which each node sends traffic only to the node whose address is the bit-complement of its own address), and transpose (in which each node sends traffic to the node reflected across the tiled CMP's diagonal). In each case, injection-rate sweeps terminate when the network saturates (injection rate ceases to rise when presented load is increased).



(a) Uniform Random, 4x4 (b) Bit Complement, 4x4 (c) Transpose, 4x4

(d) Uniform Random, 8x8 (e) Uniform Random, 16x16

Figure 15: Synthetic-traffic evaluations for 4x4, 8x8 and 16x16 networks.

The uniform random traffic sweeps are the most representative of the application workloads presented above. As shown, the hierarchical-ring design performs very closely to the buffered mesh design in 4x4, saturating at nearly the same injection rate but maintaining a slightly lower network latency at high load. In contrast, the single-ring design saturates at a lower injection rate, closer to the bufferless mesh design, CHIPPER. In contrast, in the bit-complement traffic pattern, the single-ring and hierarchical ring designs both saturate at nearly the same point, with roughly a 10% higher network injection rate than the buffered mesh router. Mesh-based designs suffer under this traffic pattern because it sends traffic over longer distances and across the middle of the network, which becomes congested. Ring-based designs do not suffer this effect. Finally, under a transpose traffic pattern, the buffered mesh baseline saturates at a very low point, below even the bufferless mesh design, because its non-adaptive dimension-ordered routing is a poor match for the worst-case traffic pattern. In contrast, the adaptivity of mesh deflection routing in CHIPPER is able to make better use of the network's bisection bandwidth. However, both ring-based designs, but especially the

Metric	Buffered	CHIPPER	Single-Ring	Node Router	Bridge Router
Critical path length (ns)	1.21	0.93	0.33	0.33	0.61
Normalized area	1	0.232	0.233	0.060	0.473

Table 7: Area and critical path of different routers in a 65nm process

hierarchical ring, are better suited to this traffic pattern because they do not suffer a bottleneck in the center of the network as mesh topologies do.

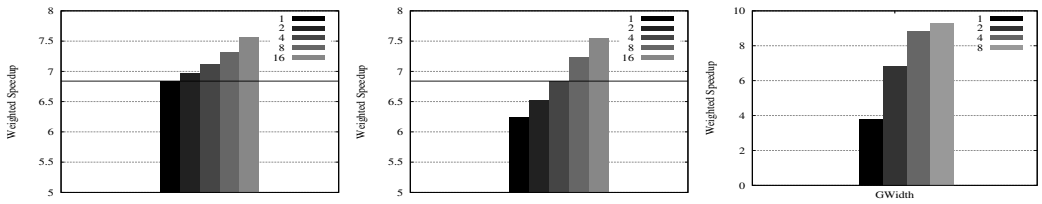
As network size scales to 8x8 and 16x16 (64 and 256 nodes respectively), results are even more encouraging. Relative to the mesh-based designs, the hierarchical-ring design performs significantly better, because the hierarchy reduces the cross-chip latency while preserving bisection bandwidth. In a 64-node network, injection rate at saturation with uniform random traffic is increased by 9.0% from buffered mesh to hierarchical ring networks; with 256 nodes, this peak-capacity injection rate is increased by 9.9%. Hence, we conclude that hierarchical ring designs scale better than mesh-based networks when bisection bandwidth is kept constant across the two options.

8.4 Router Area and Timing

We show both critical path length and normalized die area for buffered and bufferless mesh routers, single-ring routers, and the node and bridge routers of our proposed design, in Table 7. As shown, both routers in our design improve significantly on the critical path seen in previous mesh-based designs. As expected, the bridge router is nearly twice as slow as the node router, because of its higher complexity, but is still faster than either mesh-based router, because routing and arbitration is much simpler. Area also reduces: a hierarchical-ring interconnect in a 4x4 network with 8 bridge routers requires 70.5% less die area than a buffered mesh, and only 27.1% more than the bufferless mesh.

8.5 Sensitivity Studies

Bridge Router Buffer Size: The size of the FIFO queues that are used to transfer flits between local and global rings can have an impact on performance if they are too small (and hence are often full and deflect transferring flits) or too large (and hence increase bridge router power and die area). We show the effect of local-to-global and global-to-local FIFO sizes in Figs. 16a and 16b respectively, for the 8-bridge 4x4-node topology. In both cases, increased buffer size leads to increased performance. However, performance is more sensitive to global-to-local buffer size (20.7% gain from 1 to 16-flit buffer size) than local-to-global size (10.7% performance gain from 1 to 16 flits), because in the 8-bridge configuration, the whole-loop latency around the global ring is slightly higher than in the local rings, so a global-to-local transfer retry is more expensive. For our evaluations, we use a 4-flit global-to-local and 1-flit local-to-global buffer per bridge router, which result in transfer deflection rates of 28.2% (global-to-local) and 34% (local-to-global) respectively.



(a) Local-to-global bridge buffer (b) Global-to-local bridge buffer (c) Global:Local ring BW ratio

Figure 16: Performance sensitivity to buffer sizes and the global ring bandwidth in a 4x4 network.

Global Ring Bandwidth: Previous work on hierarchical-ring designs did not examine the impact of global-ring bandwidth on performance, but instead, assumed equal bandwidth in local and global rings [24]. In Fig. 16c, we examine the sensitivity of system performance to global ring bandwidth relative to local ring bandwidth, for the all-High category of workloads (in order to stress bisection bandwidth). Each point in the plot is described by this bandwidth ratio. The local ring design is held constant while the width of the global ring is adjusted. If a ratio of 1:1 is assumed (leftmost bar), performance is significantly worse than the best possible design. Our main evaluations in 4x4 networks use a ratio of 2:1 (global:local) in order to provide equivalent bisection bandwidth to a 4x4 mesh baseline.

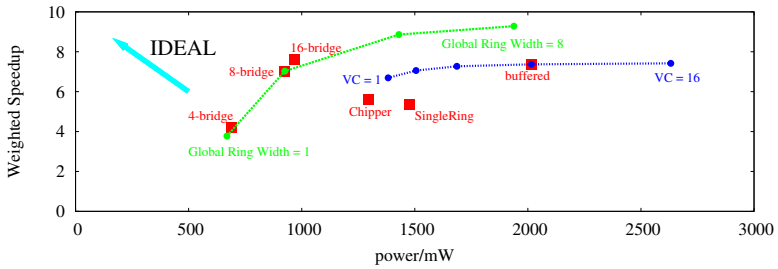


Figure 17: Performance and power in a 4x4-CMP for High-intensity workloads.

Performance increases by 81.3% from a 1:1 ratio to the 2:1 ratio that we use. After a certain point, the global ring is no longer the bottleneck, and further global-ring bandwidth increases have little effect.

Delivery guarantee parameters: In §5, we introduced injection guarantee and ejection guarantee mechanisms to ensure every flit is eventually delivered. The injection guarantee mechanism takes a threshold parameter that specifies how long an injection can be blocked before action is taken. Setting this parameter too low can have an adverse impact on performance, because the system throttles too aggressively and thus underutilizes the network. Our main evaluations use a 100-cycle threshold. For High-intensity workloads, performance drops 21.3% when using a threshold of only 1 cycle. From 10 cycles upward, variation in performance is at most 0.6%: the mechanism is invoked rarely enough that the exact threshold does not matter, only that it is finite. In addition, we evaluate the impact of communication latency between routers (which report injection blockage) and the coordinator (which takes appropriate actions). We find less than 0.1% variation in performance for latencies from 1 to 30 cycles, confirming that low-cost, slow wires may be used to implement this mechanism.

The second mechanism, the ejection guarantee, takes a single threshold parameter: the number of times a flit is allowed to circle around a ring before action is taken. We find less than 0.4% variation in performance with a threshold from 1 to 16. Thus, the mechanism provides correctness but is unimportant for performance.

8.6 Performance/Power Trade-off

To summarize the tradeoff between system performance and network power, we plot these two key statistics for all evaluated designs in Fig. 17. The ideal NoC would appear at the upper-left corner, providing the highest system performance (X axis) with near-zero power (Y axis). In the plot, we show one point each for bufferless mesh (CHIPPER) and Single-Ring designs, a sweep over buffer size (VC count) in a buffered mesh, and sweeps over the number of bridge routers and the global ring width in our hierarchical ring design. It is clear that while a wide tradeoff is possible in several designs, hierarchical ring-based interconnects provide the best tradeoff of power and performance out of all designs shown.

9 Related Work

Hierarchical Interconnects: Hierarchical ring-based interconnect was proposed in a previous line of work [23, 32, 11, 24, 10]. The major difference between our proposal and this previous work is that we propose deflection-based bridge routers with minimal buffering, and node routers with no buffering; in contrast, all of these previous works use routers with in-ring buffering, and use wormhole switching and flow control. This is analogous to the difference between buffered mesh routers [4] and bufferless deflection mesh routers [21, 8], and our design leads to significantly simpler router design with reduced area and network power. Udipi et al. proposed a hierarchical topology using global and local buses [29]. While this work recognizes the benefits of hierarchy, using buses limits scalability in favor of simplicity. In contrast, our design that scales well, in exchange for using flit-switching routers.

Bufferless Mesh-based Interconnects: While we focus on ring-based interconnects to achieve simpler router design and lower power, other work modifies conventional buffered mesh routers by removing the buffers and using deflection [21, 8]. As we show in our evaluations for CHIPPER [8], while such designs successfully reduce power and area, they retain the fundamental complexity of mesh-based routers, and hierarchical ring designs are a better overall performance/power tradeoff for most workloads.

Other Ring-based Topologies: Spidergon [3] proposes a bidirectional ring augmented with links that directly connect nodes opposite each other on the ring. These additional links reduce the average hop distance for traffic. However, the cross-ring links become very long as the ring grows, preventing scaling past a certain point, whereas our design has no such scaling bottleneck. Octagon [16] forms a network by joining Spidergon units of 8 nodes each. Units are joined

by sharing a "bridge node" in common. Such a design scales linearly. However, it it does not make use of hierarchy, while our design makes use of global rings to join local rings.

Other Low Cost Router Designs: Finally, Kim [17] proposes a low-cost router design that is superficially similar to our proposed node router design: routers convey traffic along rows and columns in a mesh without making use of crossbars, only pipeline registers and MUXes. Once traffic enters a row or column, it continues until it reaches its destination, as in a ring. Traffic also transfers from a row to a column analogously to a ring transfer in our design, using a "turn buffer." However, because a turn is possible at any node in a mesh, every router requires such a buffer; in contrast, we require these intermediate transfer buffers only at bridge routers, and their cost is amortized over all nodes.

10 Conclusion

On-chip interconnect is becoming a primary concern for large-CMP designs: chip power and die area have become first-order concerns for the design of every component, and the performance of the interconnect can bottleneck the system if it is not adequately provisioned for load. In this work, we propose a *hierarchical-ring* design for on-chip interconnect, with both reduced die area and network power, and increased performance, over previously-proposed designs. Based on a simple deflection-based ring router microarchitecture that we introduce, out design reduces average network power and router area significantly, while showing similar or slightly improved performance to buffered-mesh routers in 4x4 and 8x8 networks, and scaling better than meshes to a 16x16 network. Furthermore, the reduced-complexity router design leads to very fast and compact routers, with short critical paths and low area. Based on our evaluations, we conclude that hierarchical ring-based interconnects provide a significantly-improved power/performance tradeoff relative to previously-proposed NoCs.

11 Acknowledgments

We thank members of CALCM (Computer Architecture Lab at Carnegie Mellon) for their insightful feedback and contribution of ideas to this work. We gratefully acknowledge the support of NSF CAREER Award CCF-0953246, NSF Grant CCF-1147397, Gigascale Systems Research Center, Intel Corporation ARO Memory Hierarchy Program, and Carnegie Mellon CyLab. We also acknowledge equipment and gift support from Intel.

References

[1] P. Baran. On distributed communications networks. *IEEE Trans. on Comm.*, 1964. 2, 6

[2] S. Cho and L. Jin. Managing distributed, shared l2 caches through os-level page allocation. *MICRO-39*, 2006. 2

[3] M. Coppola et al. Spidergon: a novel on-chip communication network. *Proc. Int'l Symposium on System on Chip*, Nov 2004. 19

[4] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004. 1, 13, 19

[5] W. J. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. *DAC-38*, 2001. 1

[6] R. Das et al. Design and evaluation of hierarchical on-chip network topologies for next generation CMPs. *HPCA-15*, 2009. 2

[7] S. Eyerman and L. Eeckhout. System-level performance metrics for multiprogram workloads. *IEEE Micro*, 28:42–53, May 2008. 13

[8] C. Fallin, C. Craik, and O. Mutlu. CHIPPER: A low-complexity bufferless deflection router. *HPCA-17*, 2011. 1, 3, 14, 15, 19

[9] C. Gómez et al. Reducing packet dropping in a bufferless noc. *Euro-Par-14*, 2008. 1

[10] R. Grindley et al. The NUMAchine multiprocessor. *In Proc. 29th Int'l Conf. on Parallel Processing*, pages 487–496, 2000. 19

[11] V. C. Harmacher and H. Jiang. Hierarchical ring network configuration and performance modeling. *IEEE Transaction on Computers*, 50(1):1–12, 2001. 19

[12] M. Hayenga, N. E. Jerger, and M. Lipasti. Scarab: A single cycle adaptive routing and bufferless network. *MICRO-42*, 2009. 1

[13] Y. Hoskote et al. A 5-GHz mesh interconnect for a teraflops processor. *IEEE Micro*, 2007. 1, 2

[14] Intel Corporation. Intel details 2011 processor features. http://newsroom.intel.com/community/intel_newsroom/blog/2010/09/13/intel-details-2011-processor-features-offers-stunning-visuals-built-in. 2

[15] S. A. R. Jafri, Y.-J. Hong, M. Thottethodi, and T. Vijaykumar. Adaptive flow control for robust performance and energy. *MICRO-43*, 2010. 1

[16] F. Karim, A. Nguyen, S. Dey, and R. Rao. On-chip communication architecture for oc-768 network processors. *Proceedings of 38th Design Automation Conference*, pages 678–683, June 2001. 19

[17] J. Kim. Low-cost router microarchitecture for on-chip networks. *MICRO-42*, 2009. 1, 6, 20

[18] J. Kim and H. Kim. Router microarchitecture and scalability of ring topology in on-chip networks. *NoCArc*, 2009. 2, 3, 14

[19] H. Lee, S. Cho, and B. Childers. Cloudcache: Expanding and shrinking private caches. *HPCA-17*, 2011. 2

[20] C.-K. Luk et al. Pin: building customized program analysis tools with dynamic instrumentation. *PLDI*, 2005. 13

[21] T. Moscibroda and O. Mutlu. A case for bufferless routing in on-chip networks. *ISCA-36*, 2009. 1, 6, 19

[22] D. Pham et al. Overview of the architecture, circuit design, and physical implementation of a first-generation cell processor. *J. Solid-State Circuits*, 41(1):179–196, Jan 2006. 2

[23] G. Ravindran and M. Stumm. A performance comparison of hierarchical ring- and mesh-connected multiprocessor networks. *HPCA-3*, 1997. 2, 19

[24] G. Ravindran and M. Stumm. On topology and bisection bandwidth for hierarchical-ring networks for shared memory multiprocessors. *HPCA-4*, 1998. 18, 19

[25] L. Seiler et al. Larrabee: a many-core x86 architecture for visual computing. *SIGGRAPH*, 2008. 2

[26] A. Snavely and D. M. Tullsen. Symbiotic jobscheduling for a simultaneous multithreaded processor. *ASPLOS-9*, 2000. 13

[27] Standard Performance Evaluation Corporation. SPEC CPU2006. http://www.spec.org/cpu2006. 3, 13

[28] S. Tota et al. Implementation analysis of NoC: a MPSoC trace-driven approach. *GLSVLSI-16*, 2006. 1

[29] A. Udipi et al. Towards scalable, energy-efficient, bus-based on-chip networks. *HPCA-16*, 2010. 2, 19

[30] H. Wang, L. Peh, and S. Malik. Power-driven design of router microarchitectures in on-chip networks. *MICRO-36*, 2003. 14

[31] H. Wang, X. Zhu, L. Peh, and S. Malik. Orion: a power-performance simulator for interconnection networks. *MICRO-35*, 2002. 13

[32] X. Zhang and Y. Yan. Comparative modeling and evaluation of cc-numa and coma on hierarchical ring architectures. *Parallel and Distributed Systems, IEEE Transactions on*, 6(12):1316 –1331, Dec 1995. 19