Power Reduction of CMP Communication Networks via RF-Interconnects

M-C. Frank Chang[†], *Jason Cong*[*], *Adam Kaplan*[*], *Chunyue Liu*[*], *Mishali Naik*[*], *Jagannath Premkumar*[*], *Glenn Reinman*[*], *Eran Socher*[†] *and Sai-Wang Tam*[†]
[*] *Computer Science Department, UCLA. {cong, kaplan, liucy, mishali, cpjagan, reinman}@cs.ucla.edu*
[†] *Electrical Engineering Department, UCLA. {mfchang, socher, roccotam}@ee.ucla.edu*

Abstract

As chip multiprocessors scale to a greater number of processing cores, on-chip interconnection networks will experience dramatic increases in both bandwidth demand and power dissipation. Fortunately, promising gains can be realized via integration of Radio Frequency Interconnect (RF-I) through on-chip transmission lines with traditional interconnects implemented with RC wires. While prior work has considered the latency advantage of RF-I, we demonstrate three further advantages of RF-I: (1) RF-I bandwidth can be flexibly allocated to provide an adaptive NoC, (2) RF-I can enable a dramatic power and area reduction by simplification of NoC topology, and (3) RF-I provides natural and efficient support for multicast. In this paper, we propose a novel interconnect design, exploiting dynamic RF-I bandwidth allocation to realize a reconfigurable network-on-chip architecture. We find that our adaptive RF-I architecture on top of a mesh with 4B links can even outperform the baseline with 16B mesh links by about 1%, and reduces NoC power by approximately 65% including the overhead incurred for supporting RF-I.

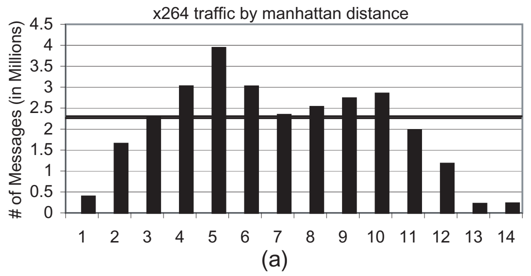
1. Introduction and Motivation

The design of the on-chip communication infrastructure will have a dramatic impact on performance, area, and power of future Chip Multi-processors (CMPs) [18]. As CMPs scale to tens or hundreds of cores, their on-chip components will be connected together with an on-chip interconnection network (or NoC: Network on Chip). Power in particular is a concern for the interconnect-dense infrastructure that would be required for high-bandwidth communication. Recent projects have estimated that this interconnect will consume about 20 to 30% of the total power budget of a CMP [8][21].

To mitigate the impact of interconnect power on future CMPs, we explore the promise of alternative interconnect technologies, based on propagation of analog waves over on-chip transmission lines. These technologies include radio frequency interconnect (or RF-I) [5], optical interconnect [17], and pulse voltage signaling [2]. Of these three, RF-I is particularly promising due to its compatibility with existing CMOS design process, thermal insensitivity, low latency and low energy consumption [5][7]. RF-I transmission lines provide single-cycle cross-chip communication, acting as a shortcut between distant endpoints on chip. Additionally RF-I can provide high aggregate on-chip bandwidth, by allowing multiple data streams to be carried simultaneously on the same medium, each stream assigned to a separate frequency band. These frequency bands can be allocated dynamically, allowing flexible adjustment of on-chip express channels [11] at compile time or runtime. For brevity, we refer to these extra links as network *shortcuts* in this work.

We observe that the patterns of inter-core and core-to-cache communication tend to vary across multi-threaded application workloads. In Figure 1, we demonstrate this on two benchmarks from the PARSEC suite [3]. These multithreaded benchmarks are executed on a 64-core CMP using a mesh interconnection network (further detailed in Section 3). In Figure 1, we plot the number of messages sent on the interconnect as a function of the number of hops they travel in the mesh. Communication demand varies by application, as the application `x264` (in Figure 1a) has a much smaller proportion of local traffic than `bodytrack` (in Figure 1b), which sends a greater proportion of traffic between nodes only a single hop apart, and almost no traffic between the most distant nodes (14 hops apart). Additionally, we have observed via manual analysis of these applications that `bodytrack` has two network *hotspots*, or routers that are the center of communication activity, whereas `x264` has only one communication hotspot. We will describe models for these and other communication patterns in Section 4.

This diversity of communication patterns is typically handled by providing a homogeneous level of bandwidth throughout the NoC. However, this means that in the execution of a given application, many paths have more bandwidth than needed – and moreover, that this NoC burns more power than necessary. Instead, we propose to adapt bandwidth allocation to critical communication paths as a means of reducing the area and power of the NoC. Since only certain



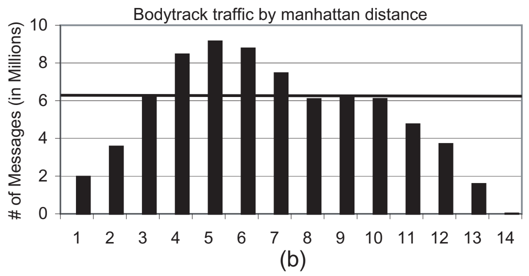


Figure 1. Traffic locality in baseline mesh, for (a) x264 and (b) bodytrack (horizontal line indicates median # msgs)

critical points in the NoC will be connected with higher bandwidth, we can reduce the overall power dissipated by using lower bandwidth in other parts of the NoC.

Furthermore, RF-I provides a natural means to perform multicast across the chip. Multicast has been demonstrated [15] as a useful technique to capture some of the communication patterns for future parallel applications. We will demonstrate how to leverage our RF-I framework for multicast, and how this integrates with the rest of our adaptive architecture.

In this work, we make the following contributions:

- We evaluate the power-efficiency of RF-I shortcuts that have been selected at design time (i.e. a fixed set of shortcuts for a given architecture) to accelerate critical communication on an underlying network topology with conventional interconnect. These shortcuts can provide an average 20% latency reduction over a baseline mesh for our probabilistic traces.
- We propose a novel interconnect scheme, which exploits the dynamic shortcut-allocation capability of RF-I to realize a reconfigurable NoC architecture. This allows the interconnection network to match the distinct traffic patterns of a given application workload. This can reduce latency by an average 32% over a baseline mesh for our probabilistic traces.
- We leverage our reconfigurable, application-specific shortcuts as a means of mitigating the performance impact of simplifying our underlying network topology. Our adaptive architecture on a 4B mesh even outperforms the baseline by about 1%, and provides an average 65% power savings.
- We explore the potential of our application-specific RF-I shortcuts to perform multicast/broadcast. Together, adaptive shortcuts and multicast enabled by RF-I can provide an average 15% performance improvement over a baseline mesh with an almost 70% reduction in NoC power on average.

2. RF Interconnect

Radio Frequency Interconnect (or RF-I) was proposed in [6], [4] as a high aggregate bandwidth, low latency alternative to traditional interconnect. Its benefits have been demonstrated for off-chip, on-board communication [4] as well as for on-chip interconnection networks [5].

On-chip RF-I is realized via transmission of electromagnetic waves over a set of transmission lines, rather than the transmission of voltage signals over a wire. When using traditional voltage signaling, the entire length of the wire has to be charged and discharged to signify either ’1’ or ’0’, consuming much time and energy. In RF-I an electro-magnetic carrier wave is continuously sent along the transmission line instead. Data is modulated onto that carrier wave using amplitude and/or phase changes.

It is possible to improve bandwidth efficiency of RF-I by sending many simultaneous streams of data over a single transmission line. This is referred to as multi-band (or multi-carrier) RF-I. In multi-band RF-I, there are N mixers on the transmitting (or Tx) side, where N is the number of senders sharing the transmission line. Each mixer up-converts individual data streams into a specific channel (or frequency band). On the receiver (Rx) side, N additional mixers are employed to down-convert each signal back to the original data, and N low-pass-filters (LPF) are used to isolate the data from residual high-frequency components.

RF-I has been projected to scale better than traditional RC wires in terms of delay and power consumption, and unlike traditional wires, it can allow signal transmission across a $400mm^2$ die in 0.3 ns via propagation at the effective speed of light [7] as apposed to less than or equal to 4 ns on a repeated bus. Chang et al. [5] used RF-I transmission lines on a 64 core CMP to realize shortcuts on a mesh NoC. They explored the potential of adaptive-routing techniques to avoid bottlenecks resulting from contention for the shortcuts. However, the work in [5] did not consider dynamic modification of the shortcuts to match the communication demands of the application. Nor did they explore any power implications of RF-I. In particular they did not consider the power savings possible when reducing the conventional interconnect bandwidth while matching performance using RF-I.

3. An Adaptable RF-I Enabled NoC

While RF-I has dramatic potential to reduce on-chip communication latency, we will further demonstrate how the flexibility of RF-I can dramatically reduce NoC power through adaptive reconfiguration. Adaptive RF-I shortcuts allow us to selectively provide bandwidth to an application's critical communications, enabling us to retain a high level of performance with a much simpler underlying conventional RC wire mesh.

3.1. Baseline Architecture

The baseline architecture we consider in this paper is shown in Figure 2(a). This architecture is comprised of 64 processor cores, 32 cache banks, and 4 memory ports, interconnected by a 10x10 mesh of routers, each router with a local port connecting it to a computing or memory element. Inter-router mesh links can transfer 16B every network clock cycle in our baseline architecture (although we will vary this link bandwidth in subsequent sections for power and performance comparison). In the figure, routers are represented by squares. Cores are mapped to white colored squares, caches are mapped to gray colored squares, and memory controllers are mapped to black colored squares. The cores and cache banks use a 4 GHz system clock, whereas the interconnect network operates at 2 GHz. The mesh uses wormhole routing, and each router in the mesh has a 5-cycle pipelined latency comprised of stages: route-computation, virtual-channel allocation, switch-allocation, switch-traversal, and link-traversal. The first two stages are only entered by the head flit of each network message. Remaining body and tail flits use the route and virtual channels computed/reserved by the head flit, and thus incur only 3 cycles per router [10]. We choose a 2D mesh as our reference topology, as mesh networks allow for regular implementation in silicon, and are simple to lay out.

3.2. RF-I Enabled Shortcuts

RF-I can be integrated on top of a network-on-chip, providing single-cycle shortcuts that accelerate communication from a set of source routers to a set of destination routers. We refer to this set of source and destination routers as *RF-enabled* routers. In a mesh topology for example, standard routers have five input/output ports, which carry traffic to/from their north, south, east, and west neighbors, as well as to a local computing element like a cache or core (attached to the fifth port). To add RF-I shortcuts into a mesh, each RF-enabled router must be given a sixth port, which connects it to either an RF-I transmitter (if it is a source router), an RF-I receiver (if it is a destination router), or both (if it is both sending and receiving on RF-I shortcuts). As RF-I transmission lines can carry multiple signals at once, each on a different frequency band, the RF-enabled routers may share this medium simultaneously, and transmit/receive independent messages in the same clock cycle. When our mesh is extended to include RF-I shortcuts, we switch from XY routing to shortest path routing.

To realize a reconfigurable network-on-chip, the set of shortcuts present in the network must be changed such that the set of source and destination routers is modified to match the current communication demand of network traffic. The basic idea to achieve this is to tune the selected transmitter and receiver at each RF-enabled router to send and listen on the same frequency band to establish a shortcut connection. One fundamental question is how many RF-enabled routers we need to adapt to network communication patterns. We will explore this requirement in section 5. The flexibility of reconfiguration does come with two costs: arbitration for frequency bands among transmitters and receivers, and subsequent integration of the resulting shortcuts into network packet routing. In this paper, we assume a coarse-grain approach to arbitration, where shortcuts are established for the entire duration of an application's execution. This allows us to amortize the cost of reconfiguration over a large number of cycles. A reconfiguration of our architecture involves the following steps:

1) **Shortcut Selection** - We must decide which shortcuts will augment the underlying topology. This can be done ahead of time by the application writer or compiler, or at run time by the operating system, a hypervisor, or in the hardware itself. In section 3.2.2 we will detail our algorithm to select shortcuts.
2) **Transmitter/Receiver Tuning** - Based on shortcut selection, each transmitter or receiver in the topology will be tuned to a particular frequency (or disabled entirely) to implement our shortcuts.
3) **Routing Table Updates** - New routes must be established and programmed into the routing tables at all network routers, to match the new available paths. If all network routers are updated in parallel, and each routing table has a single write port, it would take 99 cycles to update all the routes in the network (1 cycle for each other router in the network). Since we consider per-application reconfiguration of an NoC, this cost can easily be overlapped with other context switch activity, and will not increase the delay to start executing an application.

Figure 2(a) demonstrates a conventional mesh topology

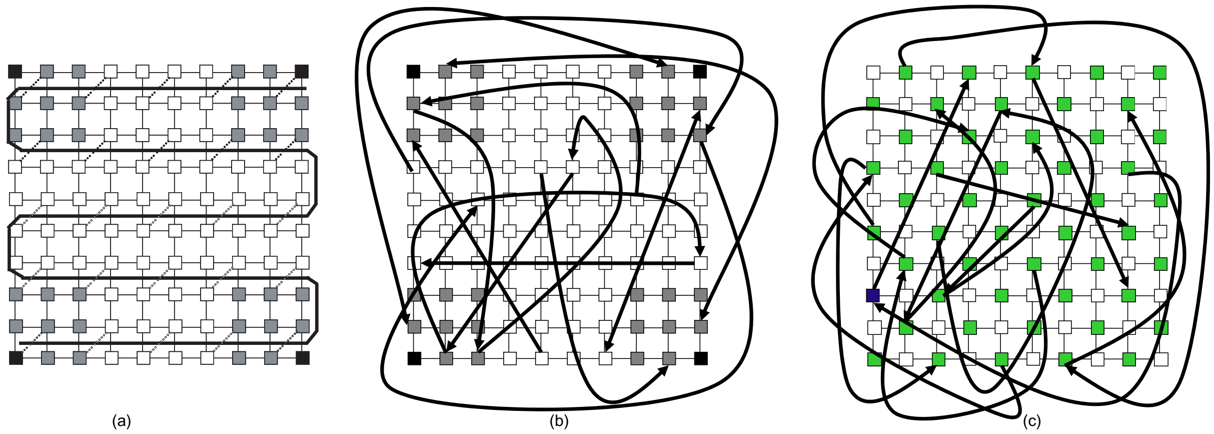


Figure 2. (a) RF-I Transmission Lines (b) Static Shortcuts (c) Adaptive Shortcuts for **1Hotspot** trace

with a set of overlaid RF-I transmission lines. Here, we constrain the number of RF-enabled routers to half of the total routers (50 routers). In Figure 2(a) these routers appear to have a small diagonal connection to the set of RF-I transmission lines, which is represented as a single thick line winding through the mesh. Although physically this RF-I appears to connect the entire set of RF-enabled routers, it logically behaves as a set of *N* unidirectional single-cycle shortcuts (where *N* is the number of available RF-I frequency bands), each of which may be used simultaneously. An example of this logical organization is shown in Figure 2(b) which represents the static set of shortcuts generated by our shortcut selection algorithm in section 3.2.2. Note that the set of RF-enabled routers is different in Figures 2(a) and 2(b), but that Figure 2(a)'s RF-enabled routers match those (shaded) in Figure 2(c).

To limit the design space, we will consider a total aggregate RF-I bandwidth of 256B from these transmission lines. This aggregate bandwidth (4096 Gbps on a 2GHz network) will require 43 parallel transmission lines, where each can carry a bandwidth of 96 Gbps [5]. Furthermore, we choose to only consider 16B shortcuts. This means that we have a total budget *B* = 16 uni-directional shortcuts which we can allocate between pairs of RF-enabled routers.

3.2.1. Selecting Architecture-Specific RF-I Shortcuts. If shortcuts are to be selected at architecture design time, designers of general-purpose processors need a cost metric that allows them to minimize communication overhead. This cost metric cannot reflect any information about the communication patterns of a given application (or set of applications). Rather, we select a cost function $W_{x,y}$ which estimates the cost of communicating between a pair of routers x and y in a given network. For architecture-specific RF-I shortcuts, we set $W_{x,y}$ to be the length of shortest-path between x and y. For our shortcut selection, we consider the baseline mesh topology to be a directed grid graph G, whose vertices are the routers of the mesh. The RF-I shortcuts we select are simply directed edges which will be added to this graph.

Given a budget B of directed edges that we can add to the mesh, we aim to select these edges such that $\sum_{all(x,y)} W_{x,y}$ is minimized over all router pairs (x,y). To ensure that the resulting topology is restricted to routers with a maximum of 6 ports, we add the restriction that at most one inbound shortcut and one outbound shortcut may be added to a given router. Additionally, we restrict shortcuts such that they may not start or end on any of the 4 corner routers which are attached to memory interfaces, as those interfaces will only be communicating with nearby cache-banks in our architecture.

In Figure 3a, we show one heuristic to minimize $\sum_{all(x,y)} W_{x,y}$ on a directed grid graph G. This heuristic constructs a permutation graph (G') for each candidate edge (i, j), and calculates the total cost of each permutation graph. The candidate edge leading to the best cost improvement is added to graph G. This continues until the budget B is exhausted. A simple implementation of this heuristic would have time-complexity $O(BV^5)$, where V is the number of vertices in G.

A less complex approach to shortcut selection is demonstrated in Figure 3b. This heuristic reduces the overall diameter of G by simply selecting maximum-cost edges until the budget B is exhausted. The time-complexity of this approach is dominated by the steps required to calculate the cost of each edge-pair, leading to $O(BV^3)$ total comparisons. The architecture-specific edges selected by this algorithm correspond to the RF-I shortcuts shown in Figure 2(b).

We have tried both heuristics and found the resulting set of shortcuts to perform comparably well. Therefore, for the

```
while $B > 0$ do
    Calculate $TotalCost(G) = \sum_{all(x,y)} W_{x,y}$ (for $x \in G, y \in G$)
    maxImprovement = 0
    for each candidate pair of routers i, j s.t. $i \notin UsedSrcs$ and $j \notin UsedDests$ do
        Create new graph $G^{'} = G + edge(i,j)$
        Calculate $TotalCost(G^{'}) = \sum_{all(x,y)} W_{x,y}$ (for $x \in G^{'}, y \in G^{'}$)
        if $(TotalCost(G^{'}) - TotalCost(G)) > maxImprovement$ then
            $maxImprovement = (TotalCost(G^{'}) - TotalCost(G))$
            $max_i$ = i
            $max_j$ = j
        end if
    end for
    $UsedSrcs = UsedSrcs \cup max_i$
    $UsedDests = UsedDests \cup max_j$
    $B = B - 1$
end while
```

(a)

```
while $B > 0$ do
    Calculate $W_{x,y}$ for all $x \in G, y \in G$
    Select edge (i,j) where $W_{i,j}$ = max($W_{x,y}$ s.t. $x \notin UsedSrcs$ and $y \notin UsedDests$)
    Add edge (i,j) to G
    $UsedSrcs = UsedSrcs \cup i$
    $UsedDests = UsedDests \cup j$
    $B = B - 1$
end while
```

(b)

Figure 3. Heuristics For Shortcut Selection

remainder of this paper we shall use the latter, less complex approach.

3.2.2. Selecting Application-Specific RF-I Shortcuts. To reconfigure our set of RF-I shortcuts dynamically for each application (or per workload), we introduce application communication statistics into our cost equation. Intuitively, we wish to accelerate communication on paths that are most frequently used by the application, operating under the assumption that these paths are most critical to application performance. To identify these paths, we need to rely on information that can be readily collected by event counters in our network. The metric we use to guide our selection is inter-router communication frequency. From a given router X to another router Y, communication frequency is measured as the number of messages sent from X to Y.

To determine the maximum benefit of this approach, we assume that this profile is available for the applications we wish to run. Furthermore we can employ the same shortcut-selection algorithm that we used to select architecture-specific shortcuts (in Section 3.2.1), except with a new optimization function $\sum_{all(x,y)} F_{x,y} \cdot W_{x,y}$ where $F_{x,y}$ is the total number of messages sent from router x to router y.

One limitation of the algorithm presented in Section 3.2.1 is that once a shortcut is selected, its source and destination are removed from further consideration. However, if a communication hotspot exists, this restriction prevents more than one shortcut from being placed at this hotspot. Although we restrict ourselves to adding only one additional port per router (and thus only one shortcut to/from a hotspot router), there is no practical reason we cannot place additional shortcuts at nearby routers. Thus, rather than merely optimizing source-to-destination communication, we optimize *region-to-region* communication.

For application-specific RF-I shortcuts, we modify our shortcut-selection algorithm as follows. We alternate between placing shortcuts between source/destination router pairs (as described earlier) and placing edges between source/destination region pairs, where these regions are non-overlapping 3x3 sub-meshes of frequently-communicating and/or distant routers.

We define an inter-region communication metric $CRegion_{A,B}$ between non-overlapping regions A and B as

$$CRegion_{A,B} = \sum_{all(x \in A, y \in B)} F_{x,y} \cdot W_{x,y}$$

To select an inter-region shortcut, we first select a pair of regions I and J such that $CRegion_{I,J} = max(CRegion_{A,B})$ over all non-overlapping regions A, B. Using these regions, we select a shortcut edge (i,j) to add to the graph, where $i \notin UsedSrcs$ and $j \notin UsedDests$, and $i \in I$ and $j \in J$.

An example of application-specific shortcut selection is shown for the **1Hotspot** trace in Figure 2(c). Here we chose number of RF-enabled routers to be 50, and give them a darker shade for clarity. For this network trace, the hot-spot of communication is the cache bank at (7,0) colored black. The effect of region-based selection is apparent, as several routers near the the hotspot are either sources or destinations of selected shortcuts.

3.3. RF-I Enabled Multicast

One clear advantage of RF-I is the ability to perform broadcast on the shared RF-I transmission-lines. Multicast/broadcast is an important operation for many coherence schemes, particularly when designing scalable many-core interconnection networks. Recent work has demonstrated gains by accelerating multicast/broadcast in a conventional NoC using Virtual Circuit Trees (or VCTs) [15]. They proposed a multicast router design, using conventional interconnect, by constructing a tree from multicast source-to-destination pair, and demonstrate dynamic power savings by reusing the trees and avoiding retransmission of the same flits on the common path of a tree.

With many receivers connected to a common and scalable set of transmission lines, RF-I provides a natural means of multicasting. One frequency band can act as a multicast channel, with multiple receivers tuned to that frequency band to receive the multicast. Moreover, RF-I shortcuts and multicast combine naturally together – the multicast channel is effectively another shortcut with multiple destinations. In our adaptive shortcut architecture, we will have N RF-enabled routers and we have the ability to insert up to K shortcuts (depending on our aggregate RF-I bandwidth). Therefore if $N > K$ (which is desirable so we have flexibility in shortcut placement), we will have $N - K$ RF-enabled routers left to tune as receivers on the multicast channel.

While it is not difficult to tune all available receivers in our topology to listen to the same frequency band, thus enabling multicast reception, the challenge is how to determine what component should be allowed to transmit on that frequency band. This requires some form of arbitration to avoid collisions between multicasting components.

In this paper, we limit the senders of multicast messages to be cache banks. We use a directory protocol for cache coherence, and the two multicast messages involved in this protocol would be *invalidates* sent from a cache bank to a number of cores due to a request for write permission on a cache block and *fills* sent from a cache bank to a number of requesting cores. Furthermore, we consider a coarse-grain arbitration approach where only one of our four cache bank clusters is selected as the sender of multicasts for some fixed amount of time - this allows us to amortize the cost of arbitration over many execution cycles.

In a given cache cluster, we designate one cache bank as the multicast transmitter for the entire cluster. This is currently set to be the central cache bank in the cluster. Any cache bank in the cluster that wants to send a multicast should first implicitly send its multicast message via conventional mesh links to the central bank to transmit it on the RF-I waveguide. Special multicast messages (used exclusively for invalidations and fills) are used to distinguish multicast transmissions from other network communication. Multicast messages are enhanced with a *destination bit vector (DBV)* field of 64 bits. Each bit in the DBV represents a core in our 64-core architecture – and a 1 in the position of that core indicates that the core is an intended recipient of the multicast. A multicast message is implicitly sent to the central cache bank in a given cluster for multicast via the conventional mesh links, where it can then be sent over the RF-I.

All multicast receivers in the mesh (attached to the routers of processor cores) are tuned to the same frequency band and receive the message from the single multicast transmitter. For example, if every other core has a receiver (assuming a topology where every other router is RF-enabled) – for the purposes of multicast, every receiver will handle multicast messages for two cores: the core at the RF-enabled router and a neighboring core. Our multicast transmitter will first transmit a flit that contains the DBV and the total number of cycles that the multicast will take, which equals to the total number of flits in that multicast message. Each RF-I receiver has logic to check the DBV of incoming multicast messages – in particular, it is only concerned with the bits representing the cores that are handled by that particular RF-I receiver (two in this 50 RF-enabled router example because each RF-enabled router serves two cores). If any of these bits are set, the receiver will continue to receive the following flits from the multicast transmitter, instantiate copies of these flits to the particular core(s) for which the bit is set. If the DBV in the first flit indicates that a receiver is not an intended destination of a particular broadcast, the receiver will power gate itself for the number of cycles indicated by that first flit to save energy on multicast reception. In this manner, each multicast message received at an RF-I enabled router will reach 0 or more cores depending on the bit values in the DBV. Figure 4 demonstrates an example multicast on a mesh with 50 receivers (one for every two components): (a) the message is first routed from the cache marked X to the designated multicast transmitter, (b) the DBV and cycle count in a flit are broadcasted to all Rx's (shaded), (c) receivers which match the DBV remain active and receive the multicast message, and (d) the message is then locally distributed (shaded nodes get a copy) – note that a message flit is duplicated and delivered as soon as it is received by the Rx, it does not wait for the remaining flits of that message to arrive.

4. Methodology

Large-scale many-core simulation is complicated by long run times and by the fact that current generation applications are not representative of the kinds of workloads such large-scale systems will be running in the years to come [12]. To address these concerns, we leveraged the Garnet [1] network simulation infrastructure and executed both synthetic probabilistic traces and actual application traces.

Details of our network simulation parameters can be found in Figure 5(a). Deadlock situations [5] are handled by eight reserved virtual channels that only use conventional mesh links.

4.1. Probabilistic Traces

As a means of exploring the interconnect demand of future applications, we constructed *probabilistic traces* to represent a variety of communication patterns for cooperative multi-threaded applications. Each probabilistic trace is executed on Garnet for 1 million network cycles. Our probabilistic traces are based on the communication patterns in Figure 1 and the actual component placement on our 10x10 mesh design. Request messages and data messages (sent between

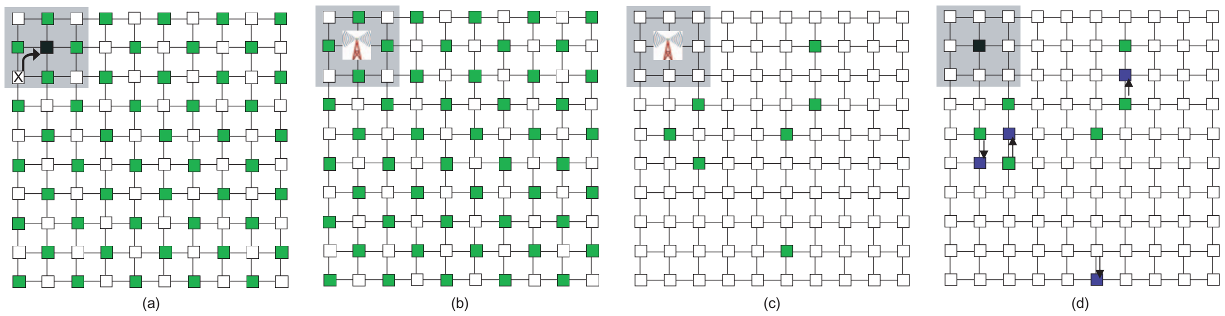


Figure 4. Example Multicast Scenario

Switching Technique	Wormhole
Baseline Mesh Routing	XY Routing
RF-I Shortcut Routing	Shortest Path
Link Bandwidth	16/8/4 bytes
Number of Virtual Networks	2
Virtual Channels (per Virtual Network)	8
Buffer Size	8 flits
Ports for baseline/RF-enabled mesh router	5/6

(a)

Core Parameters	
Number of Cores	64
Fetch/Issue Width	6/3
ROB/Issue-Queue Size	128/12
L1 Instruction/Data Cache	each 8 KB, 4 way, 32-byte block size
L2 Cache	32 banks, each bank 256 KB, 8 way, 128-byte block size

Application	Start At	End At
bodytrack	start of 1st frame	1 billion instructions executed
fluidanimate	start of 1st frame	1 billion instructions executed
jbb	start of transaction execution	100 transactions executed
streamcluster	threads spawned to compute kmedian	1 billion instructions executed
x264	start of 65th frame (to fill pipeline)	end of 65th frame

(b)

Figure 5. (a) Network Simulation Parameters, (b) Application Trace Generation

cores and cache banks or between cores) are 7 bytes and 39 bytes (including the payload) respectively. And messages sent between cache banks and the memory controllers are 132 bytes.

We constructed seven total traces: *uniform*, *uniDF* and *biDF*, *hotbiDF*, *1Hotspot*, *2Hotspot*, and *4Hotspot* respectively. These traces are detailed in Table 1.

4.2. Real Application Traces

We collected network message injection traces from real applications executed upon a 64 core SPARC processor using Simics [19], and then executed these traces on our Garnet model. This allows us to evaluate a number of interconnect design choices for a real application without the recurring overhead of full-system simulation.

The benchmarks used include one commercial workload (SPECjbb2005) and four applications from the PARSEC benchmark suite [3] (bodytrack, fluidanimate, streamcluster, and x264). All PARSEC applications are executed on their simlarge configuration. The network traces of each application are simulated on Garnet for 500 million network cycles (or to completion). The setup of each application is detailed in Figure 5(b).

4.3. Physical Design Modeling

To accurately gauge the impact of RF-I on future many-core architectures, we leveraged a number of physical design tools for our latency, power, and area data. The power modeling of the NoC consists of the modeling of routers, links and RF-I transmitters and receivers. Figure 6(a) shows the technology parameters used in our estimation and their symbols used in the following equations. The Orion [23] power model is used to get the data of router dynamic energy per flit, leakage and area with various router configurations. For links, we take the estimation method used in CosiNoC [22] as a reference. The link dynamic energy per unit length E_{link} is given by $E_{link} = 0.25V_{DD}^2(k_{opt}(c_o+c_p)/h_{opt}+c_{wire})$ where k_{opt} denotes the optimal repeater size and h_{opt} denotes the optimal inter-repeater distance. The k_{opt} can be calculated by the first equation in Figure 6(b) (where V_{DD}, c_o, c_p, and c_{wire} are defined in Figure 6(a)), and h_{opt} can be obtained using IPEM [9] , which is developed to estimate on-chip interconnection latency under a set of interconnect optimization methods for deep submicron technology. We used the buffer-insertion and optimal wire-sizing in IPEM.

The values for r_0, r_{wire}, I_{off}, and w_{min} are also defined in Figure 6(a). The leakage and area of one link are given by the lower two equations in Figure 6(b) where D is the distance between two adjacent routers. In 32 nm technology, the RF-I

Uniform: A random traffic distribution – components are equally likely to communicate with all other components.
Dataflow: Components clustered into groups which are functionally laid out in a dataflow-like fashion on our mesh. Components are biased to communicate with components within their group and with components in groups that neighbor them on either one side (unidirectional dataflow) or both sides (bidirectional dataflow). This pattern would be seen in a data decomposition like medical imaging or a functional decomposition into a pipelined pattern (like an imaging pipeline or a cryptographic algorithm).
Hotspot: One or more components in the mesh are sending/receiving a disproportionate amount of traffic – a hotspot in the mesh. This can be exhibited by caches holding frequently used synchronization vars or a master/worker paradigm.
Hot Bidirectional Dataflow: The **Dataflow** pattern but with one group in the quadrant sending/receiving a disproportionate amount of traffic. This differs from **Hotspot** as communication is still biased in the dataflow pattern direction. This pattern could be seen in a pipelined parallel application where communication load is not evenly balanced across all parts of the pipeline.

Table 1. Probabilistic Trace Patterns

Technology	32nm
Clock frequency	2 GHz
Supply voltage (V_{DD})	0.9 V
Minimum transistor size (w_{min})	70 nm
Transistor off current (I_{off})	0.00034 mA/μm
Minimum Transistor output resistance (r_0)	5 KOhm
Minimum Transistor output capacitance (c_p)	0.0165 fF
Minimum Transistor input capacitance (c_0)	0.105fF
Metal wire resistance per unit length (r_{wire})	1.2 Ohm/μm
Metal wire capacitance per unit length (c_{wire})	0.15 fF/μm
2x minimal spacing metal wire capacitance per unit length	0.0918 fF/μm
Distant metal wire capacitance per unit length	0.0833 fF/μm

(a)

$$k_{opt} = \sqrt{\frac{r_0 c_{wire}}{r_{wire} c_0}}$$

$$Lkg_{link} = Lkg_{repeaters} = 1.5 V_{DD} I_{off} w_{min} k_{opt} / h_{opt}$$

$$Area_{link} = Area_{repeaters} = k_{opt} w_{min}^2 D / h_{opt}$$

(b)

Figure 6. (a) Technology Parameters Used in Power Estimation of NoC, (b) Power Equations

energy consumed per bit transmitted has been projected to be 0.75 pJ, and the silicon area per Gbps is 124 μm^2 [5], [7]. We employ these values in this work to estimate the energy and active-layer area of RF-I components. Using the router, link and RF-I power models in conjunction with transmission flow statistics gathered from our microarchitecture simulator, we can obtain the power, total energy and area of the NoC. In this work, we report power-consumption as the average instantaneous power (in Watts) over the execution of an application.

5. Results

5.1. RF-I Shortcuts

We present results for three different architectures in this section. The **baseline** topology does not have any RF-I shortcuts. The **static** shortcut topology has a set of RF-I shortcuts that are always fixed selected by the algorithm in Section 3.2.1. The **adaptive** shortcut topology has a set of RF-enabled routers that can be dynamically tuned to match the communication patterns of the application. Here, reconfiguration is done once for the entire application. For both **static** and **adaptive** configurations, we only select 16 shortcuts as mentioned in Section 4.

5.1.1. Required Number of RF-I Enabled Routers. Even though we have a fixed number of shortcuts available in our RF-I transmission lines, the more RF-enabled routers we have, the more freedom we have with which to allocate these shortcuts. However, each RF-enabled router comes with a cost in terms of power and area – we want to retain reasonable flexibility while achieving our main goal of power-efficient NoC design. To this end, we present results for two points in this design space – a topology with 50 RF-enabled routers (also illustrated in Figure 2(a)) and 25 RF-enabled routers. The RF enabled routers are placed in a staggered fashion to minimize the distance any given component would need to travel to reach the RF-I.

Figure 7 demonstrates the tradeoff between flexibility and overhead when varying the number of RF-enabled routers for a fixed amount of RF-I bandwidth. Bars represent the average network latency/flit (primary y-axis) and diamonds represent power (secondary y-axis). Results for our seven probabilistic traces are normalized to the **baseline** architecture (no RF-I). The first bar shows the latency of the **static** shortcuts (where RF-I shortcuts are fixed at design time and do not adapt to the application): we see a 20% reduction in latency on average with an 11% increase in power (both with respect to baseline) The second bar represents our **adaptive** architecture with 50 RF-enabled routers: we see a latency reduction of 32% on average at the cost of an average 24% gain in power. (We also tried the maximal case of 100 RF-enabled routers, but this case performed quite comparably to 50 RF-enabled, and thus we do not address it further in

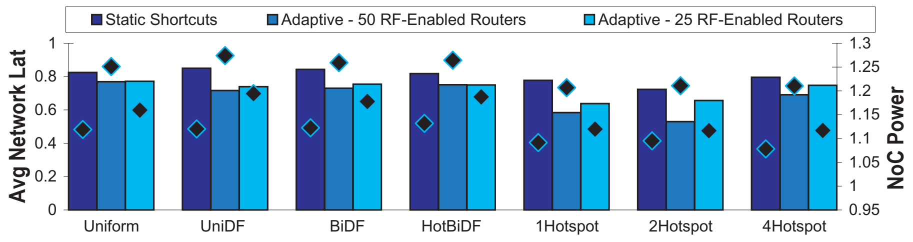


Figure 7. Tradeoff between Number of RF-Enabled routers and Performance

this work.) The third bar represents our **adaptive** architecture with only 25 RF-enabled routers: here, the reduced flexibility drops our latency reduction to 28% on average and the cost in power to 15% on average. By limiting the number of RF-enabled routers available for **adaptive** shortcut selection, we can achieve significant power savings at the cost of the flexibility of adaptation. However, our intent is to reduce the overall bandwidth of the conventional interconnect in this study, which requires a high degree of flexibility to recapture performance, and we therefore choose 50 RF-enabled routers as the **adaptive** design point of interest for rest of this study.

5.1.2. Power Savings from Reduced Bandwidth. If we use RF-I shortcuts to handle the bulk of our communication load, the underlying mesh topology can be simplified to improve power efficiency. Our baseline topology uses 16B links between routers, and we consider the impact on latency and power from reducing this to 8B and 4B. Figure 8 provides results (normalized to the 16B **baseline** configuration) demonstrating the power/performance tradeoff. When reducing the **baseline** mesh bandwidth to 8B, we save 48% power on average, but at the cost of increasing latency by 4%. If we further reduce this bandwidth to 4B, we see a 72% reduction in power and a 27% increase in latency. In contrast, our **static** set of shortcuts helps reduce this latency gap between 16B and 4B mesh links: we are able to reduce power by 67%, but we still see 11% latency increase. Moreover, by adapting to the communication pattern of the application our **adaptive** shortcuts completely close this latency gap at 4B, with an average 1% latency reduction and 62% power savings over the 16B **baseline**. Hotspot traces benefit the most from our **adaptive** shortcuts since they can customize the network to accelerate the traffic directed to and from the hotspots. The **static** shortcuts cannot adapt to the unbalanced traffic directed to and from the hotspots, with many RF-I links left underutilized. The power of this adaptation allows our **adaptive** configuration at a 4B mesh to even outperform a 16B **baseline** by as much as 13% for these hotspot traces. For our real application traces, on average we save 67% power including the overhead incurred for RF-I for our **adaptive** architecture on a 4B mesh; while maintaining network latency on average that is comparable to the **baseline** at a 16B mesh.

In Table 2, we show the total network-on-chip area of the designs discussed thus far in this paper, broken down into router area, link area, and RF-I area on the active (silicon) layer. As access points are added to the network, router area increases due to a need for more 6-port mesh routers, and RF-I area increases due to the placement of additional RF-transmitters and receivers in the network. Wire area is comprised of the signal repeaters which are placed on the active layer, and is halved each time the link bandwidth of a particular topology is halved. In addition to the power and performance gains of using RF-I on a reduced-bandwidth topology, we see that RF-I also enables a reduction in the silicon area cost of the interconnection network. Using 50-access points on a 4B mesh enables an area reduction of 82.3% compared to the baseline 16B mesh.

5.2. RF-I Multicast

To gauge the impact of multicast, we augment our probabilistic traces with special multicast messages that originate at a cache in our topology and are sent to some number of cores. The destination set of cores for a given multicast message is chosen randomly. However, we simulate multicast destination reuse by ensuring that some percentage of these messages are identical source-to-destinations pairs. We examine two levels of locality in destination set selection – assume the total number of multicast messages is M, in the 20% case, we ensure that all multicast messages will only use a number of 20%*M distinct source-to-destinations pairs. Likewise in the 50% case, we ensure that all multicast messages will use a number of 50%*M distinct source-to-destinations pairs.

Figure 9 compares the performance of a baseline mesh, VCT [15], and RF-I multicast with 50 RF-enabled routers. For RF-I multicast, we examine multicast alone (MC) and multicast with shortcuts (MC+SC). Note that the RF-I multicast (MC) case assumes a single RF-I channel is dedicated to multicast and that *no* RF-I shortcuts are enabled – all 50 RF-enabled routers can receive on the multicast channel. In multicast with shortcuts (MC+SC), 15 adaptive shortcuts are added to our 50 RF-enabled router architecture (using 15 Rx's), and the remaining 35 Rx's are tuned to the multicast channel. Results are shown for two different levels of locality

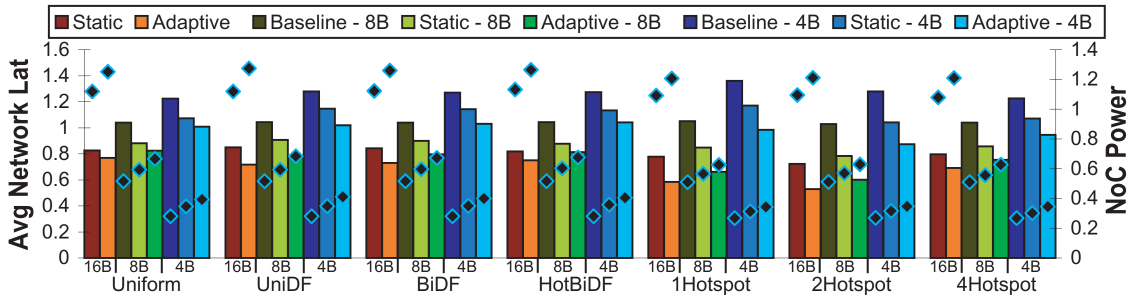


Figure 8. Impact of Mesh Bandwidth Reduction on our Probabilistic Traces

Design	Router Area	Link Area	RF-I Area	Total
Mesh Baseline (16B)	30.21	0.08	0	30.29
Mesh Baseline (8B)	9.34	0.04	0	9.38
Mesh Baseline (4B)	3.23	0.02	0	3.25
Mesh (16B) Arch-Specific	32.06	0.08	0.51	32.65
Mesh (16B) + 50 RF-I APs	35.99	0.08	1.59	37.66
Mesh (8B) Arch-Specific	9.86	0.04	0.51	10.41
Mesh (8B) + 50 RF-I APs	10.97	0.04	1.59	12.60
Mesh (4B) Arch-Specific	3.39	0.02	0.51	3.92
Mesh (4B) + 50 RF-I APs	3.73	0.02	1.59	5.34

Table 2. Area of Network Designs (in mm^2)

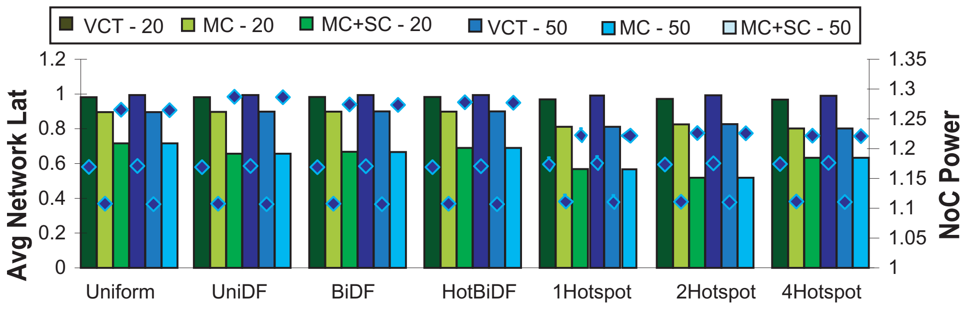


Figure 9. Multicast Power and Performance

– 20, which represents high locality destination sets, and 50, which represents moderate locality destination sets.

For the high locality configuration, VCT is able to provide an almost 3% reduction in latency compared to 16B baseline mesh (at a 5.4% silicon area cost, consumed by table structures required to maintain multicast trees). However VCT performs worse for the moderate locality configuration. VCT is really designed to reduce congestion, and the congestion with a directory coherence protocol is not severe enough to provide gain. RF-I multicast provides an average 14% reduction in latency at an 11% cost in power. RF-I multicast with shortcuts provides an average 37% reduction in latency with a 25% cost to NoC power. However, as we demonstrated in Section 5.1.2, we can maintain high performance when reducing mesh bandwidth, and provide an overall power savings.

5.3. Unified Analysis

Figure 10 summarizes the designs explored in this work, demonstrating the most effective choices from a power and performance perspective. Results are averaged over the execution of our probabilistic traces. Each line in both graphs represents a network architecture whose power and performance are normalized to that of the reference architecture: the **baseline** mesh topology with 16B inter-router links, and with no RF-interconnect. For a given line, the tallest (highest-power) point represents that architecture with 16B inter-router links, the next-tallest point represents a design with 8B links, and the shortest (lowest-power) point represents a design with 4B links.

Figure 10a compares the power and performance of mesh designs on unicast architectures: those which have no notion of multi-receiver messages. The relative heights of each line in the figure are relatively close, indicating that for a topology with 4B inter-router links, all design choices exhibit similar power consumption. In this figure we include a comparison between ”Mesh Static Shortcuts” (the **static** RF-I configuration) and ”Mesh Wire Shortcuts” (the same **static** shortcuts, implemented in conventional, buffered wire). RF-I enables faster, single-cycle shortcuts, whereas conventional

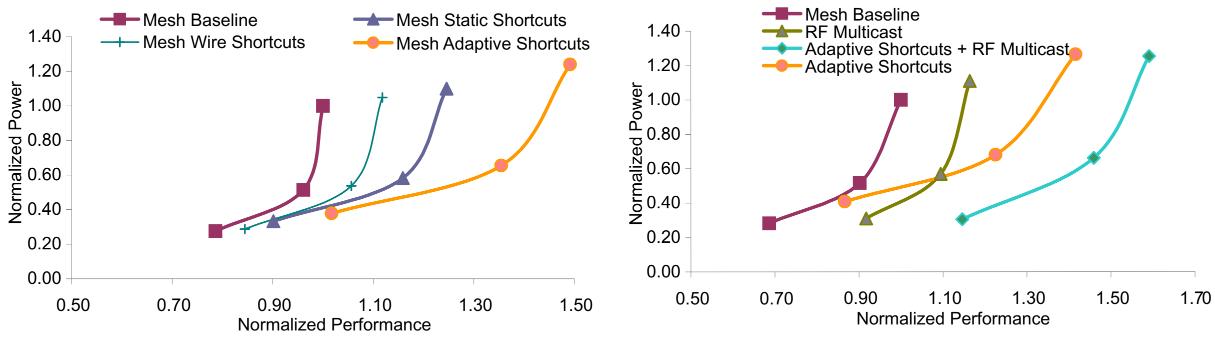


Figure 10. Overall Results Comparison for a) Unicast Architectures, and b) Multicast Architectures

wire requires multiple clock cycles to transmit long distances on chip [13]. Furthermore, RF-I has the advantage of enabling adaptive shortcut reconfiguration, which can be exploited for even further gain.

These results demonstrate that the most cost-effective design for such an architecture (in terms of power and latency) is a mesh with **adaptive** RF-I shortcuts, reconfigured on a per-application basis. With the baseline mesh links reduced to a width of 4B, **adaptive** RF-I performs comparably to the baseline mesh, while saving 65% NoC power and 82% silicon area.

Figure 10b compares power and performance of multicast architectures: those with special multicast messages sent from cache-banks to multiple cores. For reference, we include the 16B **baseline** mesh as well as a mesh overlaid with adaptive RF-I shortcuts, where each multicast message is transmitted as a set of unicast messages. The most cost-effective design for a multicast architecture combines all the techniques addressed in this paper: a reduced-bandwidth 4B mesh with 15 adaptive RF-I shortcuts as well as RF-I multicast. This architecture realizes a 15% performance improvement over the baseline unicast mesh architecture, while saving 69% NoC power and 82% silicon area on average.

6. Related Work

Beckmann and Wood [2] introduced the use of transmission lines to reduce communication latency between L2 cache banks and the cache controllers. In future CMP's with a large number of cores and cache banks on the die, it is essential to extend such schemes for improving the latency of both core-to-core and core-to-cache communication. And while transmission lines provide low-latency shortcuts in a mesh topology, they do not take advantage of frequency divided communication.

Kirman et al. [17] employed optical technology to design a shared bus for a 64-core CMP. While their design does take advantage of the low-latency and high bandwidth characteristics of optical technology via simultaneous transmission on different wavelengths, they examine optical interconnect to augment a bus topology instead of a more scalable mesh.

Ogras and Marculescu [20] explored the enhancement of a standard mesh network via the addition of application-specific long-range links between pairs of frequently-communicating routers. Unlike the single-cycle shortcuts enabled by transmission-line technology, Ogras and Marculescu implement their long-range links using a higher-latency point-to-point pipelined bus. Hu et. al [14] observe that in addition to power and latency co-optimization, topology selection and wire style assignment are important aspects of NoC design. As with the work proposed in this paper, these shortcuts are based on application-specific communication patterns. However, the selection algorithms proposed by previous researchers were intended for use at design-time on application-specific processors, and are too complex to employ in a general-purpose architecture.

None of the aforementioned studies have considered dynamic modification of transmission line allocation nor dynamic shortcut selection to match the communication demands of an application workload. Neither have they explored the power savings possible when reducing the conventional interconnect bandwidth while matching performance using the alternative interconnect.

Kim et al. [16] observed that there is no single network design that can provide optimal performance across a range of communication patterns, and proposed to customize interconnect to a particular application. However, they do not consider power consumption when evaluating the cost of their customizable NoC design, although this is an increasingly critical metric.

7. Conclusion

In this work, we reduce power consumption in an aggressive many-core NoC via bandwidth reduction of the baseline inter-router network links. We demonstrate that the use of RF-I shortcuts can compensate for the loss of bandwidth, by maintaining or even improving network latency on a variety of traffic patterns and applications. We leverage dynamically adaptive RF-I shortcuts in our network topology, providing communication bandwidth only where required. We show that adaptive RF-I can enable a 65% NoC power savings as well as 82.3% area savings via mesh bandwidth reduction from 16B to 4B, while maintaining comparable performance. Furthermore, on architectures that distinguish multicast from unicast communication, we demonstrate that RF-I can be utilized to provide both multicast acceleration and adaptive shortcuts, providing an overall 15% performance improvement while saving 69% NoC power and 82.3% silicon area.

Acknowledgments

This research was supported in part by Semiconductor Research Corporation grants 1317 and 1796. The authors acknowledge the support of the GSRC Focus Center, one of five research centers funded under the Focus Center Research Program. We wish to thank our reviewers for useful feedback.

References

[1] N. Agarwal, L-S Peh, and N. Jha. Garnet: A detailed interconnection network model inside a full-system simulation framework. Technical Report CE-P08-001, Dept. of Electrical Engineering, Princeton University, 2007.

[2] B.M. Beckmann and D.A. Wood. TLC: Transmission line caches. In *Proceedings of MICRO-36*, December 2003.

[3] C.Bienia, S. Kumar, J. P. Singh, and K. Li. The PARSEC benchmark suite: Characterization and architectural implications. Technical Report TR-811-08, Princeton University, 2008.

[4] M. F. Chang, I. Verbauwhede, C. Chien, Z. Xu, J. Kim, J. Ko, Q. Gu, and B. Lai. Advanced RF/baseband interconnect schemes for inter- and intra-ulsi communications. In *IEEE Transactions on Electron Devices*, July 2005.

[5] M.F. Chang, J. Cong, A. Kaplan, M. Naik, G. Reinman, E. Socher, and S-W. Tam. CMP network-on-chip overlaid with multi-band RF-interconnect. In *Proceedings of HPCA-14*, February 2008.

[6] M.F. Chang, V.P. Roychowdhury, L. Zhang, H. Shin, and Y. Qian. RF/wireless interconnect for inter- and intra-chip communications. *Proceedings of the IEEE*, 89(4), April 2001.

[7] M.F. Chang, E. Socher, R. Tam, J. Cong, and G. Reinman. RF interconnects for communications on-chip. In *International Symposium on Physical Design (ISPD)*, April 2008.

[8] L. Cheng, N. Muralimanohar, K. Ramani, R. Balasubramonian, and J.B. Carter. Interconnect-aware coherence protocols for chip multiprocessors. In *Proceedings of ISCA-33*, June 2006.

[9] J. Cong and D.Z. Pan. Interconnect estimation and planning for deep submicron designs. In *Proceedings of DAC-36*, 1999.

[10] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers, 2004.

[11] W.J. Dally. Express cubes: Improving the performance of k-ary n-cube interconnection networks. *IEEE Transactions on Computers*, 40(9), 1991.

[12] C. Grecu, A. Ivanov, P. Pande, A. Jantsch, E. Salminen, U. Ogras, and R. Marculescu. Towards open network-on-chip benchmarks. In *International Symposium on Networks-on-Chip*, May 2007.

[13] R. Ho, K.W. Mai, and M. Horowitz. The future of wires. *Proceedings of the IEEE*, 89(4), April 2001.

[14] Yuanfang Hu, Yi Zhu, Hongyu Chen, Ronald Graham, and Chung-Kuan Cheng. Communication latency aware low power noc synthesis. In *DAC '06: Proceedings of the 43rd annual conference on Design automation*, 2006.

[15] N. E. Jerger, L-S Peh, and M. Lipasti. Virtual circuit tree multicasting: A case for on-chip hardware multicast support. In *International Symposium on Computer Architecture (ISCA)*, 2008.

[16] M. Mercaldi Kim, J. Davis, M. Oskin, and T. Austin. Polymorphic on-chip networks. In *International Symposium on Computer Architecture(ISCA-35)*, June 2008.

[17] N. Kirman, M. Kirman, R.K. Dokania, J.F. Martinez, A.B. A psel, M.A. Watkins, and D.H. Albonesi. Leveraging optical technology in future bus-based chip multiprocessors˙ In *Proceedings of MICRO-39*, December 2006.

[18] R. Kumar, N. Jouppi, and D. Tullsen. Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling. In *32nd International Symposium on Computer Architecture*, June 2005.

[19] P. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, and B. Werner. Simics: A full system simulation platform. In *IEEE Computer*, Feb 2002.

[20] U.Y. Ogras and R. Marculescu. It's a small world after all: Noc performance optimization via long-range link insertion. *IEEE Transactions on VLSI Systems*, 14(7), July 2006.

[21] 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*, 27(5):96–108, 2007.

[22] A. Pinto, L. Carloni, and A. Sangiovanni-Vincentelli. Constraint-driven communication synthesis. In *Design Automation Conference*, June 2002.

[23] H. Wang, X. Zhu, L-S. Peh, and S. Malik. Orion: A power-performance simulator for interconnection networks. In *Proceedings of MICRO-35*, November 2002.