

Design and Evaluation of a Hierarchical On-Chip Interconnect for Next-Generation CMPs

Reetuparna Das, Soumya Eachempati, Asit K. Mishra, Vijaykrishnan Narayanan, Chita R. Das

Department of Computer Science and Engineering,
The Pennsylvania State University, University Park, PA 16801
{rdas,eachempa,amishra,vijay,das}@cse.psu.edu

Abstract

Performance and power consumption of an on-chip interconnect that forms the backbone of Chip Multiprocessors (CMPs), are directly influenced by the underlying network topology. Both these parameters can also be optimized by application induced communication locality since applications mapped on a large CMP system will benefit from clustered communication, where data is placed in cache banks closer to the cores accessing it. Thus, in this paper, we design a hierarchical network topology that takes advantage of such communication locality. The two-tier hierarchical topology consists of local networks that are connected via a global network. The local network is a simple, high-bandwidth, low-power shared bus fabric, and the global network is a low-radix mesh. The key insight that enables the hybrid topology is that most communication in CMP applications can be limited to the local network, and thus, using a fast, low-power bus to handle local communication will improve both packet latency and power-efficiency. The proposed hierarchical topology provides up to 63% reduction in energy-delay-product over mesh, 47% over flattened butterfly, and 33% with respect to concentrated mesh across network sizes with uniform and non-uniform synthetic traffic. For real parallel workloads, the hybrid topology provides up to 14% improvement in system performance (IPC) and in terms of energy-delay-product, improvements of 70%, 22%, 30% over the mesh, flattened butterfly, and concentrated mesh, respectively, for a 32-way CMP.

Although the hybrid topology scales in a power- and bandwidth-efficient manner with network size, while keeping the average packet latency low in comparison to high radix topologies, it has lower throughput due to high concentration. To improve the throughput of the hybrid topology, we propose a novel router micro-architecture, called XShare, which exploits data value locality and bimodal traffic characteristics of CMP applications to transfer multiple small flits over a single channel. This helps in enhancing the network throughput by 35%, providing a latency reduction of 14% with synthetic traffic, and improving IPC on an average 4% with application workloads.

1. Introduction

Performance of future many core/CMP systems with 10's to 100's of cores will heavily depend on the underlying on-chip interconnects, also known as Network-on-Chip (NoC) architectures, for facilitating scalable communication between cores, caches and memories. To minimize the communication cost, these systems should be designed such that data accessed by threads will be allocated to cache banks or memories closer to the processing core a thread is running on. Researchers have proposed several hardware [1, 2, 3] and software mechanisms [4] to enforce such *communication locality* in CMPs. An oblivious design that does not optimize data placement to improve communication locality (and hence reduce com-

munication cost) will most likely have poor performance. Hence, an on-chip interconnect design that is cognizant of communication locality is likely to provide better and scalable performance than a traditional orthogonal design style.

In addition, power is no longer an after-thought, secondary metric in designing computer architectures. Along with high performance, low-power/energy-efficiency has become another critical parameter in guiding architecture design and operation. It is projected that an on-chip network power can become a substantial part of an entire chip power by consuming up to 40-60 watts [5] with technology scaling, if we do not design the interconnect carefully. Hence, a low power enabling solution is imperative for medium to large scale on-chip interconnects. Thus, this paper presents an exercise of such a design trade-off study for NoCs, specifically focusing on two key principles - communication locality and power efficiency.

Network topology is a vital aspect of on-chip network design since it determines several power-performance metrics. It determines zero-load latency, bisection bandwidth, router micro-architecture, routing complexity, channel lengths, overall network power consumption, etc. Prior research has used a variety of interconnects such as shared bus [6], ring [7] and mesh/tori. The 2D mesh topologies [8, 9, 10] have been popular for tiled CMPs because of their low complexity and planar 2D-layout properties. Recently, [11, 12] have proposed high radix topologies to minimize hop count, and thus, increase performance.

While mesh on one extreme, scales poorly in terms of performance and energy as a result of increased number of routers and average hop count, high radix topologies on the other hand, reduce the number of hops and routers at the cost of increased per router complexity and possibly energy consumption. The shared bus or ring networks, on the contrary, are simple and power efficient, but do not scale for larger configurations. While most of the prior NoC research have primarily focused on performance and/or energy efficiency with little emphasis on workload characteristics, this paper attempts to optimize these parameters by leveraging the communication locality. Thus, our design is centered around a two-tier hierarchical¹ topology to exploit the communication locality present in applications.

The hierarchical network uses a high bandwidth shared bus for the first level local communication and a 2D mesh network for the global communication. The motivation for using these two well known topologies is also driven by practicality in addition to performance and energy optimizations. Since bus-based designs have been well architected for small (4-8 node) CMPs, we believe a bus-based local network is an ideal candidate to support the local communication. Similarly, we believe the planar, low complexity mesh topology seems a practical choice for the global network.

¹We use the terms hierarchical and hybrid interchangeably in this paper.

We further enhance this topology through a novel router micro-architecture, called *XShare*, which exploits bi-modal traffic and data value locality of applications to transfer multiple smaller flits over a channel. We comprehensively evaluate four topologies (mesh, concentrated mesh, flattened butterfly and our hierarchical design) across different network sizes (16 nodes to 256 nodes based on technology scaling) using both synthetic and application workloads. For the synthetic workload, we use both uniform and nonuniform traffic patterns to examine how the performance (average network latency), power, and energy-delay product changes as a network experiences communication locality.

Overall, the major contributions of this paper are:

- We design a low complexity, power-efficient, hierarchical topology, which is a hybrid of two simple networks, and is optimized for locality in communication. We quantitatively show that such a hierarchical topology can provide on an average an energy delay product gain of 63% over the mesh, 47% over the flattened butterfly, and 33% over the concentrated mesh for uniform and non-uniform traffic patterns and all network sizes. With application workloads and a simple locality aware data mapping policy for a 32-way CMP (64 nodes), the hybrid topology improves system performance (IPC) on an average by 14% over the mesh.
- We propose an optimization, called *XShare*, to increase the concurrency and throughput of the hybrid topology. *XShare* allows multiple small flits (control packets and value data flits) to be transferred over a single channel. *XShare* can further improve the latency of hybrid topology by 14% for uniform random traffic and shifts the saturation throughput by 35%. The IPC gain obtained using this technique is 4%.

The remainder of this paper is organized as follows: In Section 2, we summarize different topologies and their initial power characteristics. This is followed by detailed design of the proposed the hybrid architecture in Section 3. Section 4 describes the simulation infrastructure, followed by results in Section 5, related work in Section 6, and conclusions in Section 8.

2. Preliminaries and Motivation

2.1 Network Topologies

In a medium to large CMP system, an on-chip network needs to minimize the communication overhead for performance and power scalability. This calls for a clustered communication approach wherein most of the communication can be limited to small set of nodes. With this motivation, in this paper we analyze various design options using state-of-art network topologies and propose a hybrid topology that embraces the hierarchical communication paradigm, while minimizing power consumption.

We analyze three topologies namely mesh, concentrated mesh and flattened butterfly, to understand their impact on power and communication locality. Mesh has been the most popular topology for on-chip network so far [8, 9, 10]. This simple and regular topology has a compact 2D layout. Mesh, however, is crippled by rapidly increasing diameter that degrades its performance and power, yielding it unsuitable for larger networks. Researchers have proposed two topologies that address this limitation - flattened butterfly [11] and concentrated meshes [12].

A concentrated mesh (*Cmesh*) preserves the advantages of a mesh and works around the scalability problem by sharing the router between multiple injecting nodes. The number of nodes sharing a router is called the concentration degree of the network. Figure 1(a) shows the layout for *Cmesh* for 64 nodes. In [12], this basic design was enhanced by adding express channels between routers on the perimeter or using two networks. *Cmesh* reduces the

number of routers resulting in reduced hop count and thus yielding excellent latency savings over mesh. *Cmesh* has a radix (number of ports) of 8. It can also afford to have very wide channels (512+ bits) due to a slowly growing bisection.

A flattened butterfly (*fbfly*) reduces the hop count by employing both concentration as well as rich connectivity by using longer links to non-adjacent neighbors. The higher connectivity increases the bisection bandwidth and requires a larger number of ports in the router. This increased bisection bandwidth results in narrower channels. Figure 1(b) shows a possible layout for a flattened butterfly with 64 nodes. The rich connectivity trades off serialization latency for reducing the hop count. Innovative solutions to adapt the layout of flattened butterfly to a 2D-substrate were proposed in [11]. *fbfly* has a radix between 7-13 depending on the network size and small channel widths (128+ bits).

2.2 Initial Power Analysis

We conduct an initial power analysis of routers as a function of radix and channel width. Figure 2(a) shows the design spectrum of topologies with respect to per router power consumption at 2GHz and 35nm technology. Each point in this chart is a possible router configuration dependant on its radix and channel width. Note that, the total network power also depends on the number of routers in the network.

The router power grows with radix and channel width. Further, analyzing the breakup of the router power, we find that for a channel width of 512 bits and for 5-ports, the fraction of crossbar power is 83% of the total router power. We realize that crossbar is one the major power hungry components in the router and the fraction of crossbar power increases as the radix increases.

Cmesh has both high radix as well as high channel width resulting in high per router power. A *Cmesh* router, thus, consumes 2.5 times more power than a mesh router. However, due to the lesser number of routers, *cmesh* consumes lesser power than mesh. Yet, the *high power consumption in Cmesh routers will limit how much we can increase the concentration* and thus, how much we can reduce the hop count via concentration. A flattened butterfly router is energy-efficient. This is because the increase in radix of the router is balanced by smaller channel width.

In summary, router power increases with both the channel width and radix of the router. Switching power is a major portion of the total router power, thus we use a bus for local network. Both the hierarchical topologies and flattened butterfly, have better power efficiency than a concentrated mesh and mesh.

2.3 Communication Locality

We define locality as the percentage of packets injected by a node that are satisfied by its immediate neighbors in the network. Applications with high locality tend to have nearest neighbor communication pattern with high local traffic. Hence, a network topology which can support *high local bandwidth*, and *low latency local communication* is cognizant of communication locality.

A *mesh* has narrow channels and hence low local bandwidth. Also, it has to pay at least one hop latency cost to reach out to 4 neighbors. A concentrated mesh design is better for locality because it has wide channels and hence high local bandwidth. Due to concentration, *Cmesh* can provide low latency local communication with c neighbors, with at least half the latency cost of a mesh, where c is the degree of concentration.

In contrast, flattened butterfly is not conducive for communication locality. It has low local bandwidth due to narrow channels. Thus *fbfly* trades-off local bandwidth for reducing global hop count. A hierarchical design, however, takes complete advantage of communication locality. The local network is a wide, low latency bus

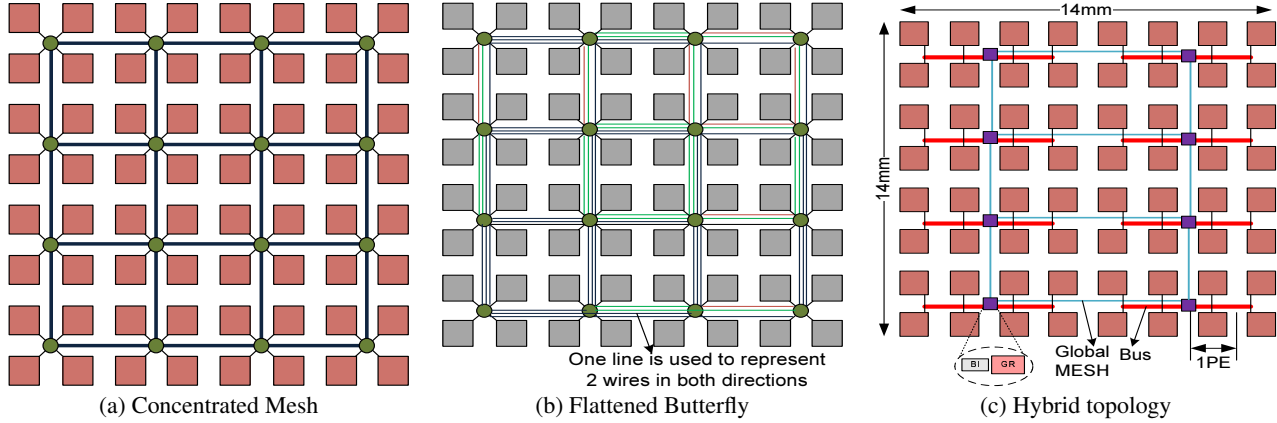


Figure 1. Network topologies for 64 nodes

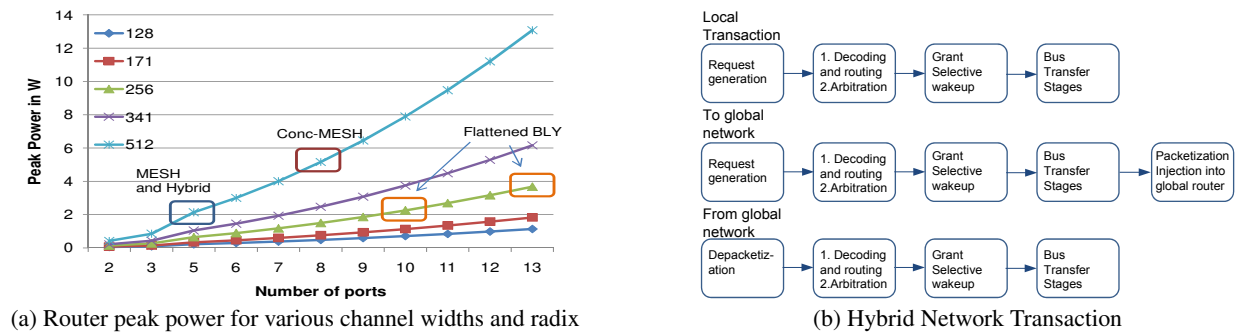


Figure 2.

which provides high local bandwidth as well as low latency communication to at least 8 neighbors.

2.4 Summary

From the above discussion, we observe that the concentrated networks begin to aid the hierarchical communication and C_{mesh} is a good choice for locality, but has lower power efficiency. Flattened butterfly on the other hand does not take advantage of communication locality. A C_{mesh} topology with a mux [11] instead of an internal crossbar is also possible. However, this design will incur significant internal wiring overhead when the concentration degree is scaled beyond four. Thus we move towards a hierarchical design which is energy efficient and takes advantage of communication locality.

3. Hybrid Design

3.1 Architecture

We design a network based on our findings from the previous sections. The proposed hybrid topology is inspired by two influential factors: hierarchical communication and power consumption. Ideally, we want a local network to support at least 6-12 nodes and provide, high bandwidth, low latency. A crossbar would be a good choice, but for its huge power penalty. Thus, we choose a wide bus which would provide us with high bandwidth, low latency (single hop), and low power as the switching energy will be greatly reduced. For the global network, we have large range of topologies which fit in. Our choice was guided by simplicity, low power, and growth rate of bisection bandwidth. We considered three low radix networks, like ring, mesh and torus. We found that torus in the global network performed worse than simple mesh due to the larger bisection bandwidth resulting in narrow channel width. Ring

global network saturated earlier than mesh because of its low path diversity. Thus, we choose an optimal low radix mesh for the global network.

Figure 1(c) shows the layout of the hybrid network. Hybrid topology utilizes a low-power bus architecture (local network) as a medium for concentration and then connects it through a low radix scalable interconnect (global network). Each bus has a local bus interface (BI) that consists of an arbiter and an outgoing FIFO buffer to connect to the global router (GR) as shown in Figure 1(c). The concentration degree of hybrid topology i.e. number of nodes sharing the bus, is fixed to 8 for limiting the saturation and for higher performance. Unlike ARM's AMBA, IBM CoreConnect, Cell EIB [7], and Intel's Front Side Bus, the proposed hybrid bus is a simple protocol free bus and does not model any coherence semantics. Coherence is maintained by directory structure similar to conventional NoC connected CMP.

The global router does not require any change. However, the global and local network channel bandwidth can be different incurring additional overhead for packetization and depacketization for every communication between local and global networks. The packetization and depacketization involves minimal overhead since it requires re-computation of the header field. Depacketization involves combining flits together. Since flit delivery is in-order depacketization is also similar. The BI handles packetization/depacketization.

3.2 Routing

A network transaction in the hybrid topology can be either entirely contained in the local network or will incur global transactions and an additional local transaction in order to reach the destination. Thus, the worst case hop count is proportional to the diameter of the global network. The average hop count of hybrid topology will

be half that of a concentrated mesh, because of higher concentration degree (8 for hybrid and 4 for *cmesh*).

Figure 2(b) shows how transactions are handled in the hybrid network. As shown, a bus transaction has two stages in addition to the actual bus transfer latency and each stage takes one clock cycle. For the global network, we use dimension order routing.

3.3 XShare Channel Sharing

The on chip network is used to transfer various classes of traffic, including data packets and control packets. Data packets are typically large in size (512-1024 bits) and require a large channel width for low network transfer latency. On the other hand, control packets are usually 36-64 bits and do not need such a large channel width. Recent research [13, 14] has demonstrated that even many of the data flits (up to 60% of CMP Cache Data from real workloads) have frequent patterns such as all zeros or all ones. Thus, these data flits can also be encoded in small number of bits (16-32 bits). We define any flit which after encoding or originally uses at most half the channel width as a *short flit*. Existing NoCs implement channels with relatively large width (128-256 bits) to reduce the serialization latency for larger packets. This results in internal fragmentation and reduced link utilization when a *short flit* is transferred since a large portion of the channel width is not used.

To reduce internal fragmentation in channels and to improve link utilization, we propose the *XShare* mechanism. *XShare* allows multiple *short flits* to be transferred over a single, wider channel. Figure 3(a) shows a generic crossbar for a mesh topology, where the input ports from the four cardinal directions and from the local PE (North: N_{in} , South: S_{in} , East: E_{in} , West: W_{in} , Injection: PE_{in}) are mapped to the output ports (North: N_{out} , South: S_{out} , East: E_{out} , West: W_{out} , Ejection: PE_{out}). However, one output port may be mapped to only one of the input ports at any time (E_{in} mapped to S_{out} in Figure 3 (a)). This allows a single flit to traverse the crossbar out to a particular output port(S_{out}), blocking all other input ports. Figure 3 (b) shows the *XShare* crossbar, where the crossbar is broken into two logical data sets, thus two input ports (N_{in} and E_{in} in Figure 3 (b)) can be mapped to a single output port. Therefore, if two *short flits* want to go to the same output port they can both simultaneously traverse the crossbar and subsequently the link to reach the next router. A low overhead zero-one detector can be used to identify *short flits* at injection.

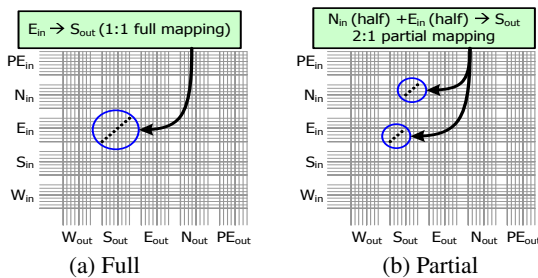


Figure 3. Baseline and XShare Crossbar Mapping

3.4 XShare Input Buffer Logic

To support the combination of shorter flits, few modules of the router needs to be re-engineered. Figure 4 shows the input buffer logic modifications (W is the width of the channel). Essentially, the data path of the input DEMUX and the switch MUX is split into two separable halves. The incoming combined flit has two virtual channel ids (VCIDs) in its header and it is demuxed based on the VCIDs into two separate virtual channels. Similarly a switch allocation (SA) control logic now sends two demux signals that

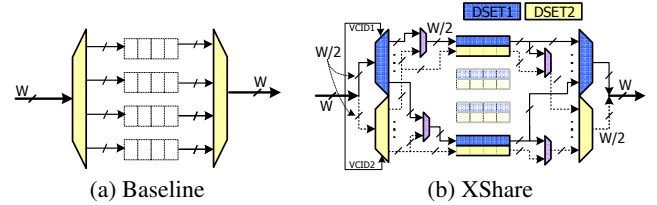


Figure 4. Baseline and XShare input buffers

determine which half of the crossbar a flit needs to be routed (Upper Half is Data Set1 (DSET1) and Lower Half is Data Set2 (DSET2)). The buffer can still be maintained as a FIFO buffer with no changes. They are treated as two separate logical halves (DSET's) by the MUX/DEMUX logic. The main overhead comes from including the second layer of smaller muxes (color coded purple in the figure). The buffer read / buffer write stages are the shortest stages in a generic router pipeline, and have sufficient slack for 2:1 muxes.

3.5 XShare Switch Arbitration Logic

Figure 5 (a) shows the switch arbitration (SA) stage of a typical on-chip router. The SA stage can be broken into two stages. In the first stage, (SA stage 1) a $v:l$ arbiter for each input port chooses which of its virtual channels (v_i) can bid for the output port. In the second stage, (SA stage 2) a $p:l$ arbiter for each output port (p_o) chooses one of the input ports (p_i). Thus, at the end of the SA stage, a (p_i, v_i) pair is chosen for each output port p_o and a crossbar is mapping is enabled between p_i and p_o .

Short flits can be combined by *XShare* in two cases.

- If any two input virtual channels (VCs) within a **single** input port request the same output port, and each has a short flit, then they can be combined. A combined request for both these VCs can be sent to SA stage 1. If the combined request wins SA stage 2, both the short flits can be sent together.
- If there are two input VCs, each with a short flit, but from **different** input ports requesting for the same output port, they can be combined in SA stage 2.

The complexity of *XShare* can be attributed to selecting the **second** flit in SA stage. This can be done in three ways as shown in Figure 5. We chose the design shown in Figure 5(c) for evaluations. In this design there are two simultaneous $p:l$ arbiters. Only short flit requests are sent to the second $p:l$ arbiter. In case the top arbiter picks a short flit, the second arbiter will supply a matching short flit. The probability that both arbiters will pick the same flit is very low. The area overhead of *XShare* is primarily due to extra arbiters and is around 3% of router area obtained from Synopsys synthesis at 90nm.

Although *XShare* can be applied as a generic technique to any topology, it is better suited for the hybrid topology because: (1) In hybrid topology, any two short flits requesting for the shared bus can always be combined; (2) The complexity of the additional *XShare* arbitration logic is $O(p)$, where p is radix of the router. Thus, a low radix router like a hybrid global interconnect ($p=5$) is more suitable than routers with high radix routers; (3) Also, lower the number of ports, higher is the probability of finding flits that can be combined, because most flits would need to go to the same output port. (4) Finally, *XShare* is primarily a throughput improvement mechanism, and hence more suitable for hybrid topology which saturates earlier due to high concentration.

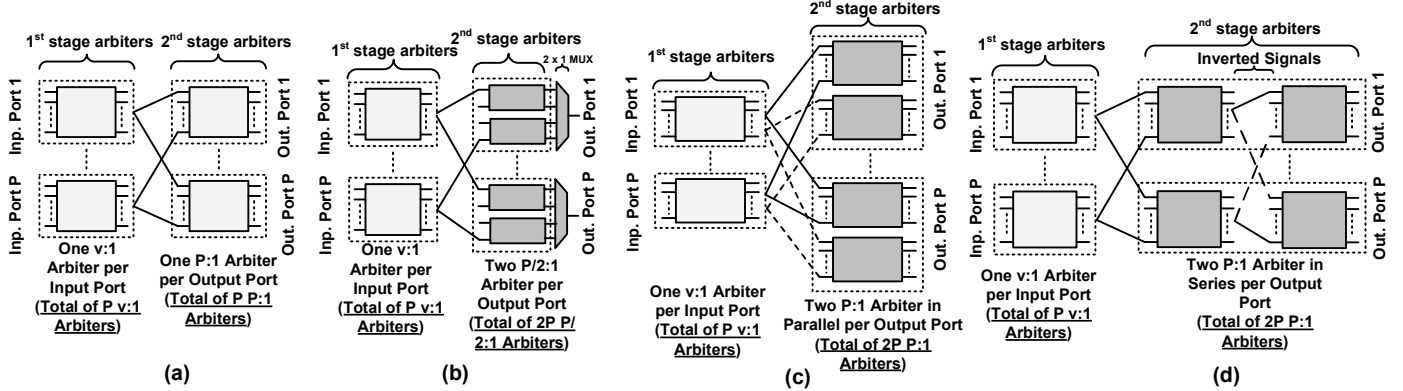


Figure 5. Design options for *XShare* SA Stage (a) Baseline arbiters (b) Each p:1 arbiter broken into smaller p/2:1 arbiters (c) Two simultaneous p:1 arbiters (d) Two back to back p:1 arbiters

4. Experimental Platform

4.1 Technology Assumptions

We explore the design space over a period of the 10 years starting from 70nm (2007) technology (corresponds to 16-node NoC) down to 18nm (2018) technology node (corresponds to 256-node NoC). We assume a 14mmx14mm die size, which remains constant across technologies and network node size of 3.5mm (CPU or cache) at 70nm that scales down by a factor of 0.7 for each technology generation. The wire parameters for global layers are obtained from PTM [15] and we assume 1 metal layer to be dedicated for the network. (Wire width at 70nm is 450nm). Assuming cores occupy half the chip area [5] and the cores use all the available metal layers, we use a conservative estimate of 4096 horizontal tracks per die at 70nm. The number of wires increases by 30% due to wire-width scaling with each technology generation. For a given topology, channel width is calculated from the bisection bandwidth and total number of wires that is fixed across all four topologies. These are presented in Figure 6 (a) for different technologies/networks.

The delay and power model of CACTI 6.0 [16] is used for estimating the values for link and the bus. Table 3 gives our assumptions of lengths and the values of dynamic energy and leakage power for bus and the links. The delay and power model of bus include the higher loading due to senders and receivers. The additional overhead of dedicated point-to-point links required for request and grant lines for the bus have also been modeled. Both the network and cores run at a frequency of 2GHz. We used Orion power model for estimating the router power[17].

4.2 Simulation Setup

The detailed configuration of our baseline simulation set-up is given in Table 1. Each terminal node in the network consists of either a core or a L2 cache bank. We implemented a detailed full-system cycle-accurate hybrid NoC/ cache simulator for CMP architectures with Simics as a front end. The memory hierarchy implemented is governed by a two-level directory cache coherence protocol. Each core has a private write-back L1 cache. The L2 cache is shared among all cores and split into banks. Our coherence model includes a MESI-based protocol with distributed directories, with each L2 bank maintaining its own local directory. The network connects the cores to L2 cache banks and to the on-chip memory controllers(MC) (Figure 6 (b)).

For the interconnects, we implemented a state of art low-latency packet-based NoC router architecture. The NoC router adopts the deterministic X-Y routing algorithm, finite input buffering, and wormhole switching and virtual channel flow control. We also

SPLASH 2: Is a suite of parallel scientific workloads. Each benchmark executed one threads per processor.
SPEComp: We use SPEomp2001 as another representative workload. The results of applu, mgrid and swim are presented.
Commercial Applications. (1) TPC-C, a database benchmark for online transaction processing (OLTP), (2) SAP, a sales and distribution benchmark, and (3) SJBB and (4) SJAS, two Java-based server benchmarks. The traces were collected from multiprocessor server configurations at Intel Corporation.

Table 2. Application workloads

model the detail artifacts of specific topologies that can lead to increased pipeline stages in link traversal. The parameter we use across different topologies are given in Figure 6 (a).

For the performance analysis, we use synthetic and a diverse set of application workloads comprising of scientific and commercial applications. We run each application for at least one billion instructions. The commercial applications are run for at least 10000 transactions. The workload details are summarized in Table 2.

Parameters	Bus				
	70nm	50nm	35nm	25nm	18nm
Length (mm)	7	4.9	3.43	2.4	1.68
Delay (ps)	498.9	442.9	353.9	247.7	173.4
Energy (pJ)	1.4	0.67	0.28	0.20	0.14
Leakage (nW)	23.5	13.3	3.5	2.4	1.7
	Link				
	70nm	50nm	35nm	25nm	18nm
Length (mm)	3.5	2.45	1.7	1.2	0.84
Delay (ps)	233	208.8	167.5	117.3	82.1
Energy (pJ)	0.6	0.29	0.12	0.08	0.06
Leakage (nW)	10.2	5.49	1.4	0.98	0.69

Table 3. Energy and delay of bus and inter-router links

5. Performance Evaluation

We measure average network latency, power, and energy-delay product for comparing the four networks (mesh, concentrated mesh, flattened butterfly, and our hierarchical topology) by varying the network size and workload. For the application workloads, we compare the IPC for a 32-way CMP (Network size is 64).

5.1 Synthetic Traffic

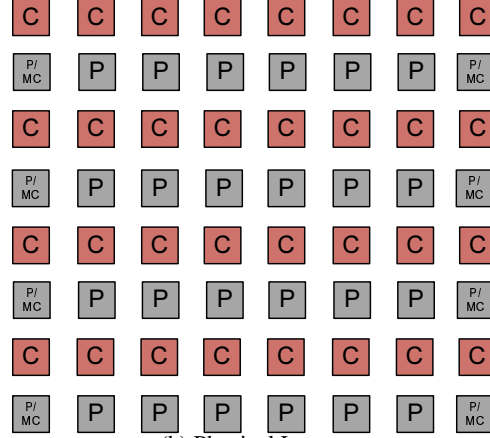
In this section, we present the results for two synthetic traffic patterns, namely uniform random (UR) and non-uniform/localized

Processor Pipeline	SPARC 2 GHz processor, two-way out of order, 64-entry instruction window
L1 Caches	64 KB per-core(private), 4-way set associative, 128B block size, 2-cycle latency, split I/D caches
L2 Caches	1MB banks,shared, 16-way set associative, 128B block size, 6-cycles latency, 32 MSHRs
Main Memory	4GB DRAM,up to 16 outstanding requests for each processor, 400 cycle access
Network Router	2-stage wormhole switched, virtual channel flow control, 1024 maximum packet size

Table 1. Baseline Processor, Cache, Memory and Router Configuration

Topology	No. of nodes	Channel Width	Conc. Degree	Radix	VCs	Buffer Depth	No. of Routers	Total Wires
Mesh	16	512	1	5	4	4	16	4096
	64	512	1	4	4	4	64	8192
	256	512	1	4	4	4	256	16384
CMesh	16	512	4	8	4	2	4	4096
	64	512	4	8	4	2	16	8192
	256	512	4	8	4	2	64	16384
Fbfly	16	512	4	7	2	8	4	4096
	64	256	4	10	2	8	16	8192
	256	128	4	13	2	16	64	16384
Hyb	16	512	8	3	4	8	2	2048
	64	512	8	5	4	4	8	8192
	256	512	8	5	4	4	32	16384

(a) Network parameters



(b) Physical Layout

Figure 6. Network assumptions

traffic (NU), in Figure 7 . We expect applications to lie in between these two synthetic traffic patterns. Although we have experimented with several non-uniform traffic distributions, here we only discuss results for traffic that consists of 75% local traffic, where the destination is one hop away from the source, and the rest 25% traffic is uniformly distributed to the non-local nodes. Figure 7 (a-f) shows the latency throughput curves versus per node injection rate, for the two traffic patterns and various network sizes. From the figure, it is clear that mesh provides the best throughput but the highest end-to-end latency. As the network sizes increases, all the topologies saturate earlier because of higher number of injecting nodes, leading to higher injection rate. Mesh always provides very high throughput due to higher path diversity. All hierarchical topologies saturate earlier than Mesh because of higher pressure on the smaller network.

Latency Analysis of Uniform Random:

For the 16 node configuration, *Cmesh* and flattened butterfly have the same channel bandwidth. As a result, flattened butterfly has better latency than *Cmesh*, because of its rich connectivity and destinations being only one hop away. Hybrid topology has the best latency but the lowest throughput. The low latency of hybrid topology at 16 node is due to the high percentage of traffic being confined to a single bus (up to 50%), which makes it highly latency efficient.

At larger network sizes, flattened butterfly *fbfly* latency is dominated by the serialization cost and consequently has higher latency. The reason being that the bisection bandwidth of the flattened butterfly grows rapidly due to the rich connectivity resulting in narrower channels. Hybrid topology and *Cmesh* perform in a similar fashion and better than the others for UR traffic.

Latency Analysis of Non-Uniform (75% local traffic): As depicted in Figures 7(b), 7(d) and 7(f), the hierarchical topology benefits maximum from the NU traffic by exploiting the communication locality. *Cmesh* is a close competitor for NU traffic. As the network size increases, *fbfly* performs worse than simple mesh in throughput

and latency. The reason being that the one hop latency of mesh is smaller than *fbfly*. The hierarchical element in *fbfly* offers no benefit because of the low channel bandwidth leading to higher one-hop latency.

The saturation point for all topologies with NU traffic shifts towards the higher-end compared to UR traffic by as much as 50% for large network sizes. This reveals two interesting points. First, this shows that with high local traffic, hierarchical topologies like *cmesh* and hybrid can offer higher throughput. Second, for uniform traffic, the saturation bottleneck is the global network, not the bus/concentration since the local network could support an injection load rate of up to 10% with NU traffic. Theoretically, the bus can take load up to 12.5% (1 packet every cycle for 8 injecting nodes). The main observations from these results are:

- The Hybrid topology provides a high-bandwidth, low-latency medium which can be better exploited by local traffic. Although it saturates earlier than other topologies, it still can handle most real world workloads that typically have more locality than UR and have less than 10% injection rate.
- *Fbfly* scalability is constrained by its narrow channel width.
- In the hierarchical topologies, the achieved throughput is primarily limited by the global network.
- Mesh provides the highest throughput, but at increased latency. However, it performs well with local traffic.

5.1.1 Power Analysis

Figure 9(a-f) provide the power plots for all network sizes and traffic patterns. Mesh is obviously very power-inefficient at all network sizes. The power consumption of mesh at 64 and 256 nodes reaches as high as 100W, implying that it may not be a viable option for designing low-power, scalable NoCs. Both *Cmesh* and *fbfly* have higher power consumption than hybrid due to the high switching energy (high radix networks) at 16 nodes. At larger

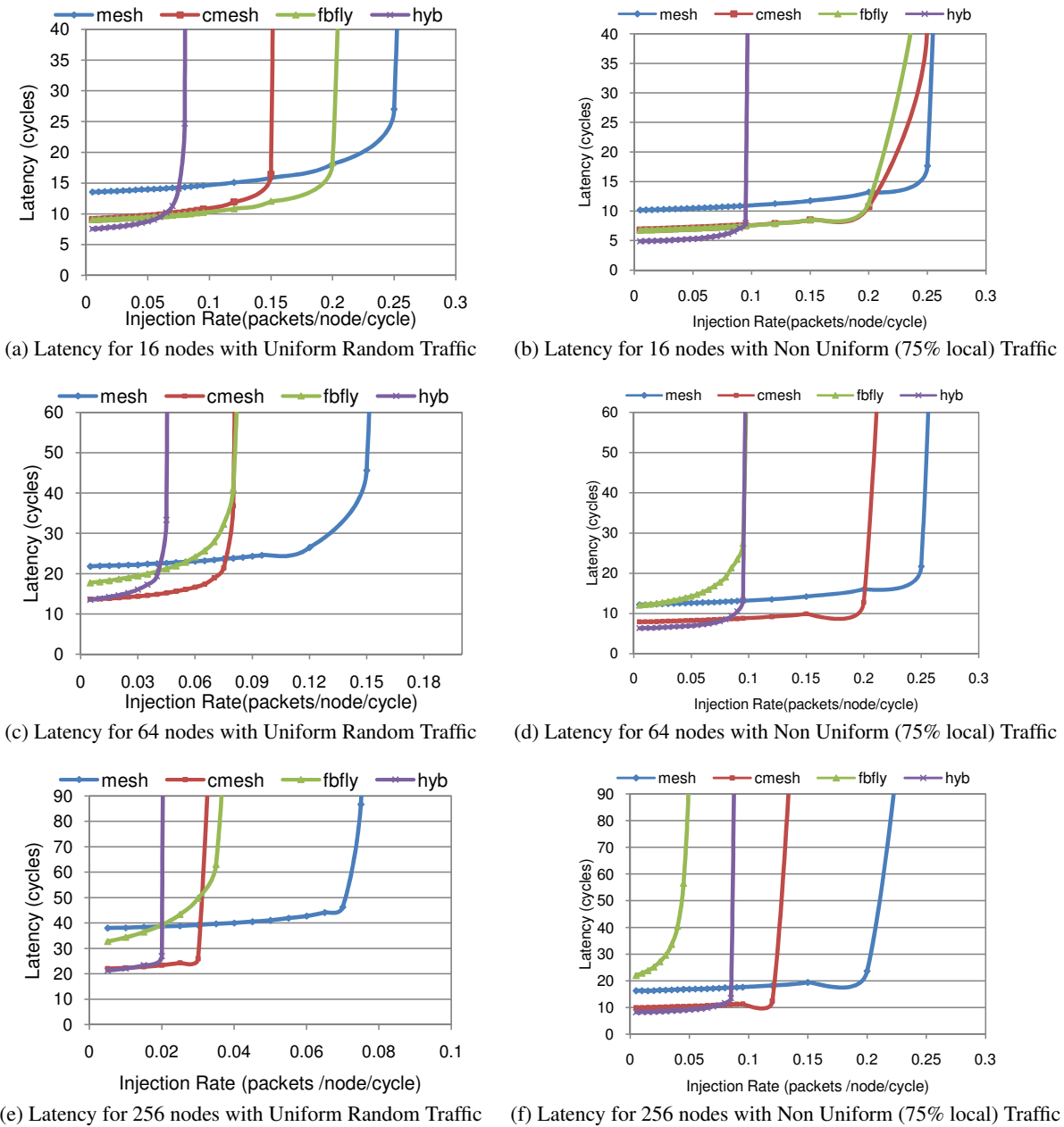


Figure 7. Load-Latency curves for Synthetic Traffic

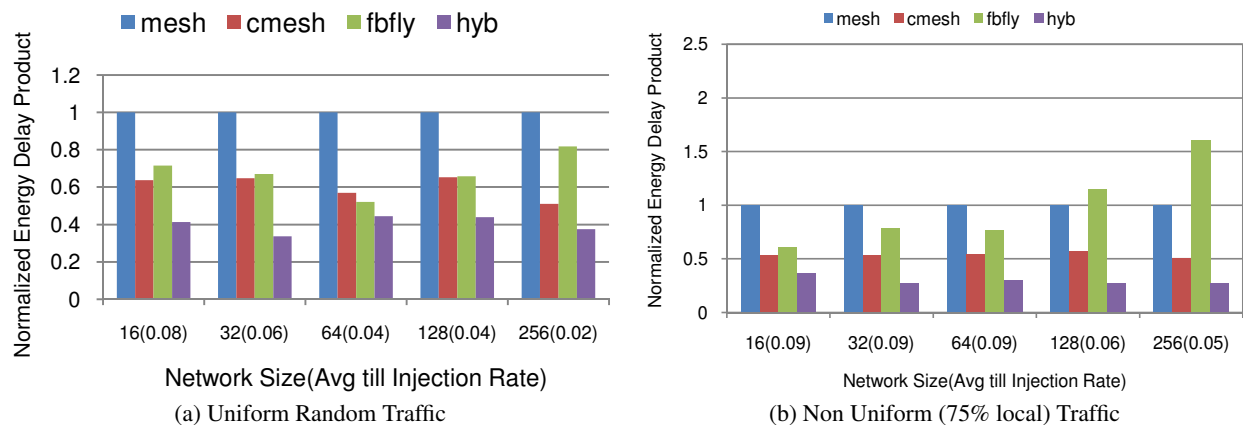


Figure 8. Energy delay product normalized to mesh topology for synthetic traffic

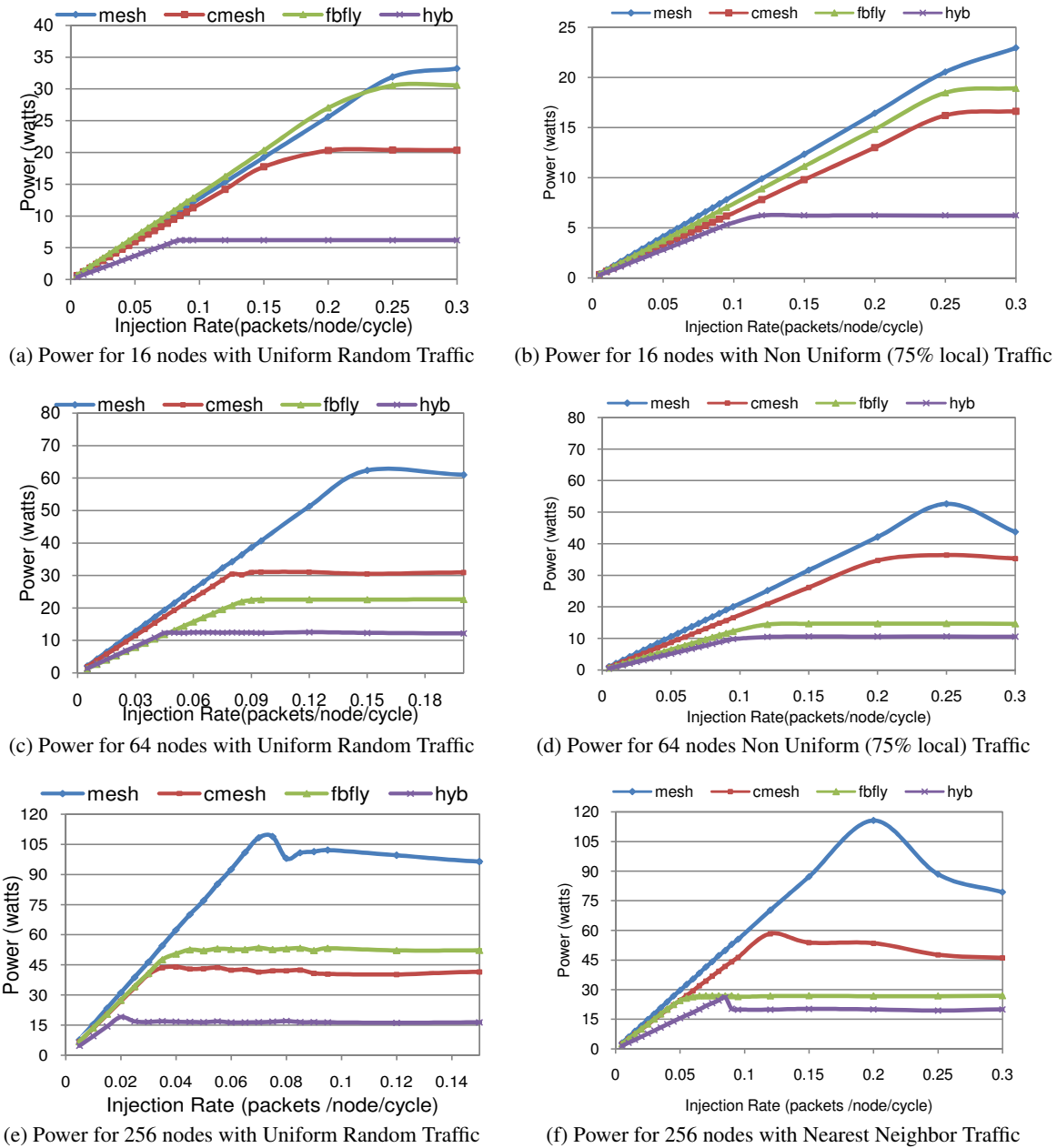


Figure 9. Load-Power curves for Synthetic Traffic

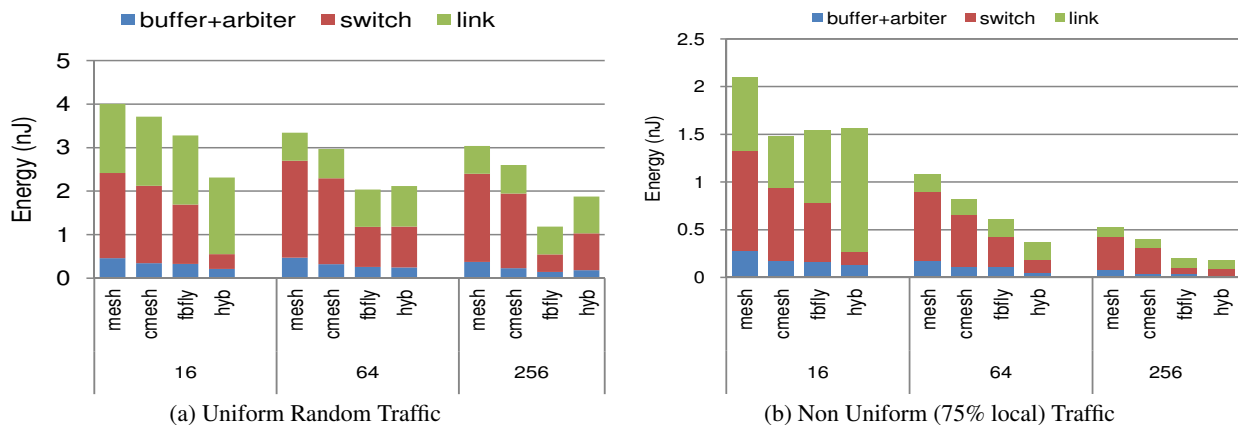


Figure 10. Energy per message for Synthetic Traffic

network sizes, we observe that *fbfly* becomes more power-efficient as the channel width decreases, whereas *Cmesh* always has higher energy as the channel width is fairly constant across network sizes. Hybrid topology has the lowest power consumption since it greatly reduces the switching energy almost eliminating it at the local level. Thus, across all traffic patterns and network sizes, the hierarchical design seems a clear winner.

The energy per message results, depicted in Figure 10, give the energy that would be required for a message to reach its destination. It can be found that the hybrid topology has very low switching energy for small network sizes, but has higher link energy. At small network size, the hybrid link and switching energies (link:Switch LSE) are in the ratio 3:1. As the network size increases, the fraction of switching energy increases and is about 1:1 at 256 nodes for UR traffic and for NU traffic the LSE goes from 10:1 to 1:1. In case of hybrid topology, the switching energy corresponds to the global network. This shows that as the network size increases the fraction of local traffic decreases. For *Cmesh*, the switching energy increases from 50% at 16 nodes to 70% at 256 nodes for both UR and NU traffic. The NU traffic energy-per-message decreases (Fig. 10 (b)) with technology scaling (for larger networks), while the UR traffic is unable to take full advantage of technology scaling.

Energy delay product (EDP) in Figure 8 shows the trend across 5 network sizes and the values are averaged for all injection rates just below saturation point. All the values are normalized with respect to mesh. Hybrid topology has the least EDP for both UR and NR traffic and *fbfly* degrades for both UR and NR traffic. Under UR traffic, Hybrid improves EDP over mesh by 57%, *fbfly* by 34%, CMESH by 25%. For NR traffic, it does 69% better than mesh, 61% for *fbfly*, and 41% better than *Cmesh* EDP averaged over all the network sizes.

In summary, hybrid topology provides low-power, fast, high-bandwidth communication that can be efficiently used in the presence of high locality. Yet, it saturates faster than the state-of-art topologies.

5.2 System Performance

This section demonstrates the impact of network topology on system performance, and also quantifies the gains in network across different topologies for real applications.

We evaluate two data placement policies. The first policy spreads consecutive cache blocks across the banks. This is the static non-uniform cache mapping (SNUCA) policy from [1]. This policy is oblivious to locality and makes no attempt to place the data closer to the processing nodes that need it. Our second policy for data mapping is an operating system managed page allocation policy, and we adopt a similar approach as in [4]. We map the addresses to cache bank in the granularity of a physical page instead of cache blocks. Every physical page in main memory maps to a unique bank. With a page fault, when a new physical page is allocated to a virtual page, the operating system chooses a page that would map to a cache bank closest to the processor which requested the page. Thus, the scheme preserves locality and we expect future CMP systems to require such locality schemes. This policy is a very naive data mapping policy. Researchers have proposed very sophisticated hardware [3] as well as software solutions [4] to improve data mapping and communication locality even further.

Specifically, the policy can be improved by letting it dynamically adapt to transient behavior of applications, allowing migration of data, allocating the page to its frequent user, rather than its first user, and also co-scheduling sharing threads nearby. We evaluate our system using the simple scheme since it works sufficiently well to demonstrate the concept of locality aware network design. The choice of workloads for which we present the results, was guided by sensitivity of application to network performance and our effort

in trying our best to cover the spectrum. Applications with higher L1 misses per kilo instructions (11mpki) would inject more packets per cycle into the network, thus would see higher impact of network savings on system performance. However, with applications (e.x. *mgrid*, *swim*) which have high L2 misses per kilo instructions (12mpki), memory system becomes the bottleneck. Thus, ideally applications with high L1 and L2 misses will be most sensitive to the on-chip network (e.x. *applu*, *ocean*, *barnes*).

5.2.1 Communication Locality

Figures 11 (a) and (b) reveal that all the hierarchical topologies have higher locality because of higher concentration. Our simplistic data mapping policy is unaware of the sharers of the data and we do not implement any thread scheduling. The locality aware policy gives 10x higher locality than the uniform bank spreading scheme (lower order bank interleaving) as shown in Figures 11(a) and (b). Even our *simple* page allocation policy shows consistently high locality for all commercial applications. For scientific applications however, the page allocation policy is unable to harness the benefit of the available locality because of imbalance in bank allocation (first user has the data in the closest bank) and locality oblivious thread scheduling. If all the sharers of a data are closely located then the locality would increase. In addition for workloads that have a large L2 cache requirement locality is limited. For *swim* and *mgrid*, locality is very high because these benchmarks are memory intensive and hence most of the traffic goes to the appropriate memory controller.

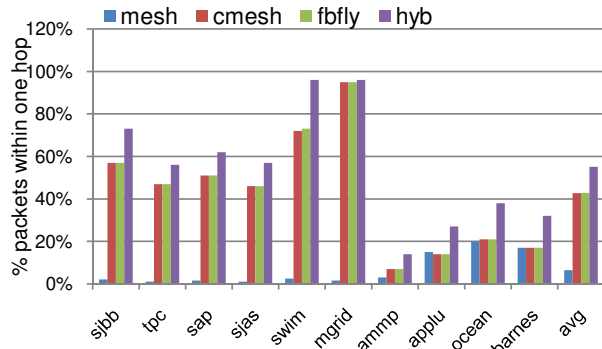
5.2.2 Network Latency

Figures 11 (c) and (d) compare the network latency across applications for both the data mapping schemes. With locality aware bank mapping, applications demonstrate tapered traffic patterns. Locality favoring topologies like *cmesh* and hybrid consistently show lower latency than *fbfly*. This is essentially because flattened butterfly *trades-off channel bandwidth to reduce global hop count*. When the locality is high, global hop count reduction is not fully exploited and the one hop latency is also high due to low channel bandwidth. *Cmesh* and hybrid topologies have equivalent performance. In case of the hybrid topology, in spite of higher locality, it is only marginally better than *cmesh* in network latency because a non-local transaction involves two local transfers(3 cycles) apart from traversing across the global network. Thus, the global transactions offset the benefit obtained by the additional locality. However, since hybrid topologies have even lower global hop counts than concentrated mesh, we expect with larger networks, the overhead of bus transactions to be amortized by reduced hop counts. Moreover, better data mapping and thread scheduling can greatly reduce average distance of global hops for hybrid topologies. Also, note that the absolute latency values are smaller with the locality aware scheme as opposed to the uniform bank mapping, indicating that a locality aware scheme in general is a better design than the oblivious data mapping.

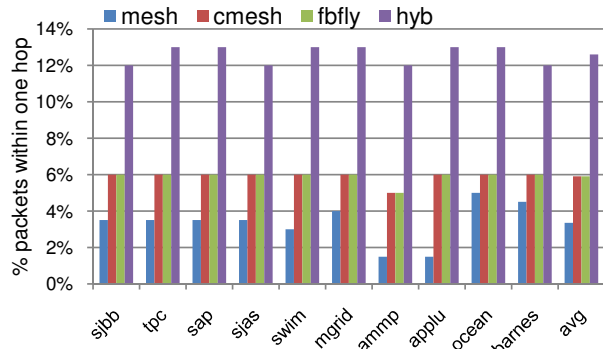
5.2.3 Impact on IPC

On an average, over all applications, hybrid and *cmesh* perform 13% better than mesh in IPC and 10% better than flattened butterfly as depicted in Figures 11 (e) and (f) . In addition note that flattened butterfly may not benefit from a better routing protocol, because the injection rates are sufficiently low, and the bottleneck is the narrow channel bandwidth, not the path diversity. Applications which are sufficiently sensitive to network latency can provide up to 38% IPC improvements.

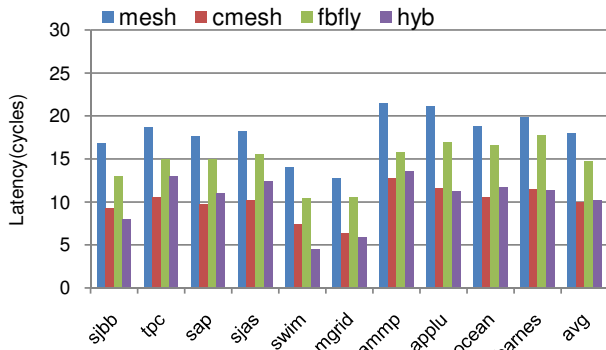
We also compare the average IPC of applications with locality aware page mapping to naive low order bank interleaving at the granularity of cache blocks for the hybrid topology. From Figure



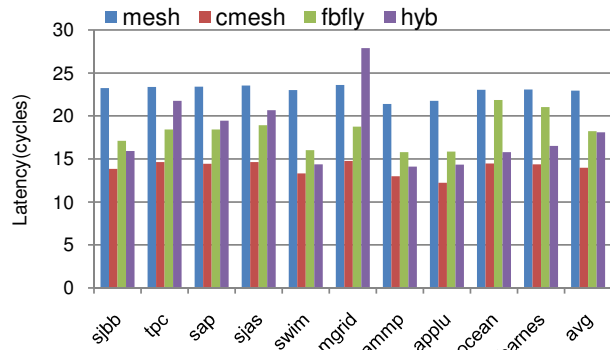
(a) Communication locality with locality aware mapping



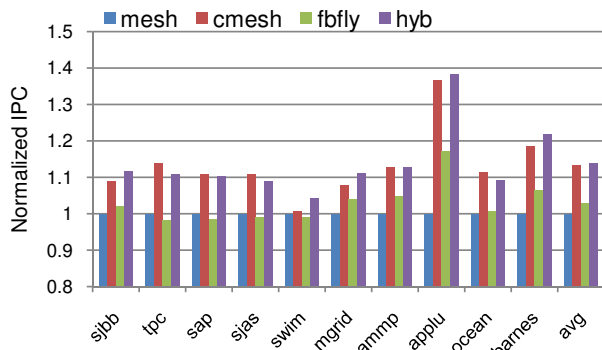
(b) Communication locality with uniform bank spreading



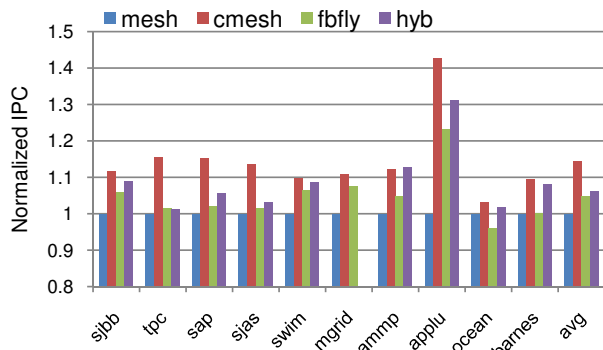
(c) Network latency with locality aware mapping



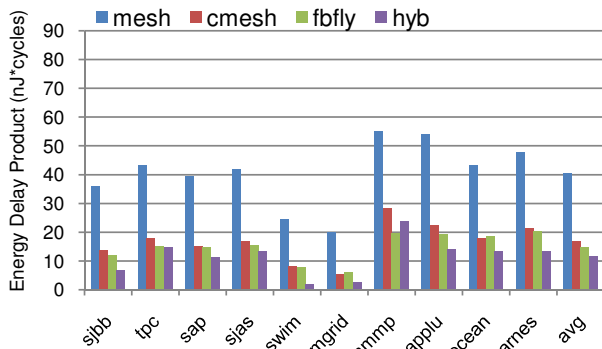
(d) Network latency with uniform bank spreading



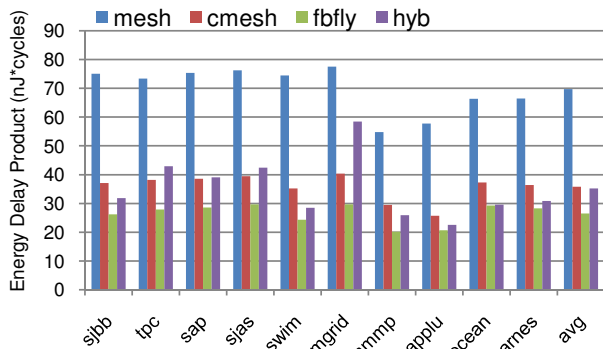
(e) System IPC with locality aware mapping



(f) System IPC with uniform bank spreading



(g) Network edp with locality aware mapping



(h) Network edp with uniform bank spreading

Figure 11. Application Results for 64 node CMP

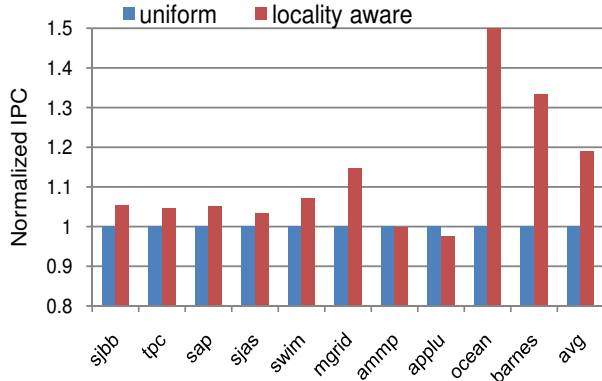


Figure 12. IPC comparison of locality aware mapping with uniform bank spreading mapping

12, it is clear that even a naive locality aware policy gives at least 19% improvement over the lower order interleaving. This demonstrates that data mapping is an important factor and hardware-software co-designs can offer lucrative benefits.

5.2.4 Network energy delay product

Hybrid topologies consistently show lower energy-delay-product (edp) over all the other topologies for the locality aware mapping as shown in 11 (g). Overall, hybrid topology performs 30% better in edp over *cmesh* and 22% better over *fbfly*. Flattened butterfly is more energy efficient than *cmesh* because of narrower crossbars as seen earlier, and thus, has lower edp than *cmesh* inspite of having higher latency. Swim and mgrid have low edp than the others because they are memory intensive and only those set of network resources to reach the memory controllers are being used. In case of ammp benchmark, the naive scheme results in imbalanced load (a few banks getting all the data and lot of sharers) and thus, negatively impacts its performance. Thus, a data-mapping scheme should not only try to preserve locality, but also, should balance load for enhanced performance. Also, note that the energy-delay-product is at least 2x smaller with the locality aware scheme than the uniform lower-order bank interleaving mapping, indicating that a locality aware scheme in general is a better design than oblivious data mapping.

5.3 XShare Performance

In this section, we illustrate the benefits of using XShare architecture. XShare can benefit from three traffic characteristics: (a) High percentile of short flits in CMP traffic; (b) Traffic patterns which share common paths between source and destinations; The results are depicted in Figure 13 for a 64 node network Figure 13 (a)

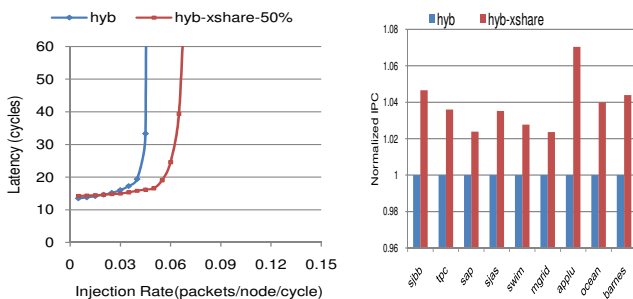


Figure 13. XShare (a) latency curve for UR traffic (b) IPC for applications

shows the gain due to XShare with hybrid topology for uniform random traffic pattern with 50% short flits in the traffic. Initially, at low load, we see a marginal penalty due to XShare because of increased blocking. However, XShare shifts the saturation and improves throughput by an order of magnitude. This is because XShare technique is able to multiplex the links *simultaneously* when ever short flits occur in the traffic stream and hence, deliver more packets per cycle. Figure 13 (b) shows that we gain on an average 4.4% in IPC across applications with XShare over the baseline hybrid topology.

6. Related Work

This section summarizes the prior work on on-chip interconnect topologies. For several decades, bus has been used as the medium for on-chip communication between shared memory and the processors. The main advantages a bus are easy of design, low power consumption and simpler coherence maintenance, since it aids snooping. Kumar et al. [6] presented a comprehensive analysis of interconnection mechanisms for small scale CMPs. They evaluate a shared bus fabric, a cross bar interconnection, and point to point links. Pinkston et al. [7] examined the cell broadband engine's interconnection network which utilizes two rings and one bus to connect 12 core elements. These simple topologies are suitable for small scale systems and are not scalable for larger systems.

Recently, there has been a lot of research for building packet-switched networks using regular topology such as Mesh. Tile [18], a CMP built by Tiler company, uses five Meshes for connecting 64 nodes. The Intel 80-core prototype also used a Mesh and found that the interconnection power can be as high as 50% of the chip power [9]. TRIPS [10] is another example that used Mesh for its interconnection network. Wang et al. [19] did a technology oriented, and energy aware topology exploration. They explored mesh and torus interconnects with different degrees of connectivity for generalized on-chip Networks. It has been shown that mesh is not scalable for large systems, because of its rapid growing diameter resulting in large latency and high power consumption [12]. Balfour and Dally [12] proposed using concentrated mesh topologies and express channels (balanced meshes) for medium sized on chip networks. Kim et al. [11] propose a high radix flattened butterfly topology for on-chip networks to overcome the scalability limitations of a mesh.

Borkar [5] motivated the use of wide buses in conjunction with circuit switched networks for power efficiency. Muralimanohar et al. [20], customized the interconnect for improving the NUCA cache access time using different metal layers for latency and bandwidth optimizations. A combination of bus and mesh was used in the NUCA cache and evaluated for an 8-core CMP and a network size of up to 32 nodes. In [21], the authors have proposed a hybrid topology, where they have broken a larger mesh into smaller sub-meshes and used a hierarchical ring as the global network. Their motivation was to decrease the global hop count and reduce congestion at the center of a large mesh.

Our work differs from all these prior works in that, we design a hybrid topology to suit the communication needs of applications running on CMP systems and evaluate its scalability, performance and power behaviors across the forthcoming technology generations for designing larger systems. We consider access patterns in typical CMPs to design the network and thereby provide latency and power benefits.

7. Conclusions

The ubiquity of CMPs has marked a paradigm shift towards communication-centric design optimizations for performance and power. In this context, NoCs are expected to play a major role in

optimizing the communication overheads, thereby improving performance and power savings of CMPs. One approach to minimize communication overhead is to exploit the communication locality of applications in designing the underlying interconnects. In this work, we attempt to exploit the communication locality by designing a two-tier hierarchical topology for facilitating local and global communication between cores. The hierarchical topology consists of a bus-based local network and a mesh-connected global network. We evaluate and compare the hybrid topology against three other state-of-art NoC topologies for various network sizes and workloads. Our simulation results indicate that the proposed hierarchical design can outperform the other networks in terms of latency, power consumption and energy-delay product specifically with localized communication.

On an average, the hybrid topology can give up to 63% improvement over mesh, 33% over concentrated mesh and 47% over flattened butterfly, in energy delay product for synthetic traffic. Using a locality aware data mapping policy for application workloads, we find that the hybrid topology can take more advantage of the locality than other topologies. The proposed interconnect gives on an average 14% benefit in IPC and 70% reduction in energy-delay-product over a mesh for a 32-way (64-node) CMP. A limitation of the hierarchical NoC is that it saturates faster than the other designs, and in order to enhance the throughput, we propose a novel channel sharing technique, called *XShare*, that can increase concurrency, and can provide up to 14% latency improvement and 35% higher throughput for synthetic traffic. Also, traffic patterns with high locality will push the saturation further. Even with its early saturation, the proposed hybrid topology can handle most of the application workloads because of their low injection rates and communication localities.

In conclusion, this paper makes a case for designing communication locality aware hierarchical networks that can provide better performance and power consumption compared to other state-of-the-art high radix NoCs. We expect that by utilizing more intelligent schemes for preserving locality in large CMP systems, the performance and power envelopes of a hybrid topology can be enhanced further.

8. Acknowledgments

This work is supported in part by National Science Foundation grants CCF-0702617 and CNS-0202007, funds from Intel Corporation. The views expressed herein are not necessarily those of the NSF, Intel. We thank Ravi Iyer for commercial workload traces and Satish Narayanasamy for the initial suggestion of looking at application locality aware topologies. We also wish to thank Onur Mutlu for helpful comments and discussions.

References

- [1] B. M. Beckmann and D. A. Wood, "Managing wire delay in large chip-multiprocessor caches," in *MICRO 37: Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*, 2004, pp. 319–330.
- [2] Z. Chishti, M. D. Powell, and T. N. Vijaykumar, "Optimizing replication, communication, and capacity allocation in cmps," *SIGARCH Comput. Archit. News*, vol. 33, no. 2, pp. 357–368, 2005.
- [3] J. Huh, C. Kim, H. Shafi, L. Zhang, D. Burger, and S. W. Keckler, "A nuca substrate for flexible cmp cache sharing," in *ICS '05: Proceedings of the 19th Annual International Conference on Supercomputing*, 2005, pp. 31–40.
- [4] S. Cho and L. Jin, "Managing distributed, shared l2 caches through os-level page allocation," in *MICRO 39: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 455–468.
- [5] S. Borkar, "Networks for multi-core chips: A contrarian view," in *Special Session at ISLPED 2007*.
- [6] R. Kumar, V. Zyuban, and D. M. Tullsen, "Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling," *SIGARCH Comput. Archit. News*, vol. 33, no. 2, pp. 408–419, 2005.
- [7] T. W. Ainsworth and T. M. Pinkston, "Characterizing the cell eib on-chip network," *IEEE Micro*, vol. 27, no. 5, pp. 6–14, 2007.
- [8] M. B. Taylor, W. Lee, J. Miller *et al.*, "Evaluation of the raw microprocessor: An exposed-wire-delay architecture for ilp and streams," in *ISCA '04: Proceedings of the 31st annual international symposium on Computer architecture*. Washington, DC, USA: IEEE Computer Society, 2004, p. 2.
- [9] S. Vangal, J. Howard, G. Ruhl *et al.*, "An 80-tile 1.28tflops network-on-chip in 65nm cmos," in *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, 11-15 Feb. 2007, pp. 98–589.
- [10] K. Sankaralingam, R. Nagarajan, R. McDonald *et al.*, "Distributed microarchitectural protocols in the trips prototype processor," in *MICRO 39: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 480–491.
- [11] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," *Microarchitecture, 2007. MICRO 2007. 40th Annual IEEE/ACM International Symposium on*, pp. 172–182, Dec. 2007.
- [12] J. Balfour and W. J. Dally, "Design tradeoffs for tiled cmp on-chip networks," in *ICS '06: Proceedings of the 20th annual international conference on Supercomputing*. New York, NY, USA: ACM, 2006, pp. 187–198.
- [13] R. Das, A. K. Mishra, C. Nicopoulos, D. Park, V. Narayanan, R. Iyer, and C. R. Das, "Performance and Power Optimization through Data Compression in Network-on-Chip Architectures," in *HPCA '08: Proceedings of the 14th International Symposium on High Performance Computer Architecture*, 2008.
- [14] D. Park, S. Eachempati, R. Das, A. K. Mishra, V. Narayanan, Y. Xie, and C. R. Das, "MIRA : A Multilayered Interconnect Router Architecture," in *35th International Symposium on Computer Architecture (ISCA)*, 2008.
- [15] W. Zhao and Y. Cao, "New generation of predictive technology model for sub-45nm design exploration," in *ISQED '06: Proceedings of the 7th International Symposium on Quality Electronic Design*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 585–590.
- [16] N. Muralimanohar, R. Balasubramonian, and N. Jouppi, "Optimizing nuca organizations and wiring alternatives for large caches with cacti 6.0," *Microarchitecture, 2007. MICRO 2007. 40th Annual IEEE/ACM International Symposium on*, pp. 3–14, 1-5 Dec. 2007.
- [17] H. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A Power-Performance Simulator for Interconnection Networks," in *ACM/IEEE MICRO*, Nov 2002.
- [18] D. Wentzlaff, P. Griffin, H. Hoffmann *et al.*, "On-chip interconnection architecture of the tile processor," *IEEE Micro*, 2007.
- [19] H. Wang, L.-S. Peh, and S. Malik, "A technology-aware and energy-oriented topology exploration for on-chip networks," in *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 1238–1243.
- [20] N. Muralimanohar and R. Balasubramonian, "Interconnect design considerations for large nuca caches," *ISCA '07: Proceedings of the 34th Annual International Symposium on Computer Architecture*, pp. 369–380, 2007.
- [21] S. Bourduas and Z. Zilic, "A hybrid ring/mesh interconnect for network-on-chip using hierarchical rings for global routing," in *NOCS*, 2007, pp. 195–204.