

Pseudo-Circuit: Accelerating Communication for On-Chip Interconnection Networks

Minseon Ahn and Eun Jung Kim

Department of Computer Science and Engineering
Texas A&M University
College Station, TX 77843 USA
{msahn, ejkim}@cse.tamu.edu

Abstract—As the number of cores on a single chip increases with more recent technologies, a packet-switched on-chip interconnection network has become a de facto communication paradigm for chip multiprocessors (CMPs). However, it is inevitable to suffer from high communication latency due to the increasing number of hops. In this paper, we attempt to accelerate network communication by exploiting communication temporal locality with minimal additional hardware cost in the existing state-of-the-art router architecture. We observe that packets frequently traverse through the same path chosen by previous packets due to repeated communication patterns, such as frequent pair-wise communication. Motivated by our observation, we propose a pseudo-circuit scheme. With previous communication patterns, the scheme reserves crossbar connections creating pseudo-circuits, sharable partial circuits within a single router. It reuses the previous arbitration information to bypass switch arbitration if the next flit traverses through the same pseudo-circuit. To accelerate communication performance further, we also propose two aggressive schemes; pseudo-circuit speculation and buffer bypassing. Pseudo-circuit speculation creates more pseudo-circuits using unallocated crossbar connections while buffer bypassing skips buffer writes to eliminate one pipeline stage.

Our evaluation results using a cycle-accurate network simulator with traces from SPECComp2001, PARSEC, NAS Parallel Benchmarks, SPECjbb2000, and Splash-2 show 16% improvement in overall network performance and up to 5% reduction in average energy consumption in routers, compared to the state-of-the-art router architecture. Evaluated with synthetic workload traffic, this scheme shows performance improvement by up to 11%.

I. INTRODUCTION

As the number of cores on a single chip increases to achieve higher performance, on-chip interconnection networks have been widely accepted as communication architecture for chip multiprocessors (CMPs), such as Intel 80-core Teraflop [11], Tiler 64-core [31], TRIPS [23], and RAW [27]. Meanwhile, it is projected that a future CMP will have more than hundreds of processor cores in a single chip due to ever-shrinking feature size in CMOS process technology [1]. Along with this trend, the size of the on-chip network is also increasing to provide connectivity to all processor cores. Thus, it is unavoidable to suffer from high communication latency as the number of hops is increasing in the on-chip networks due to the nature of packet switching with per-hop processing.

Communication latency is mainly dominated by the number of hops in the future CMPs due to the size of the on-chip networks. To provide high performance in communication, it is required to have ultra-low per-hop router delay within limited area budget and power constraints. There have been several studies on the low-latency router design in packet-switched networks to reduce per-hop router delay using speculation, pre-computation, or aggressive flow control [17, 18, 19, 21, 22]. However, the recent state-of-the-art router architecture has a complicated pipeline design to support various kinds of communications in on-chip networks. As a result, communication latency and power consumption are subject to the router overhead, such as additional hardware and link overhead.

We observe that every application has a certain amount of communication temporal locality, frequent pair-wise communication in the network. The communication locality enables performance improvement with circuit reuse [12]. Even more in each router, this communication temporal locality can be seen such that the recently used communication path from an input port to an output port can be reused. Fig. 1 shows communication path reusability as communication temporal locality in end-to-end communication and in crossbar connections, which are links within a single router connecting input ports to output ports. It shows about 22% communication temporal locality in end-to-end communication because the locality can be reused only if both the source and the destination are the same as previous. However, communication temporal locality of crossbar connections is increased up to 31%. A crossbar connection can be more possibly reused by future flits even when the future flits traverse the same crossbar connection within a router.

This observation motivates communication acceleration using the communication temporal locality of crossbar connections. If a flit traversal reuses the arbitration information created by previous flits, it does not need switch arbitration in the router pipeline thus reducing per-hop router delay. Our main goal is bypassing router pipeline stages in packet-switched on-chip interconnection networks with minimal hardware addition by exploiting the reusability of the crossbar connections in each router to minimize per-hop router delay.

In this paper, we propose a novel pseudo-circuit scheme for on-chip interconnection networks, which reuses the previous arbitration information if the next flit needs to traverse through the same crossbar connection within a single

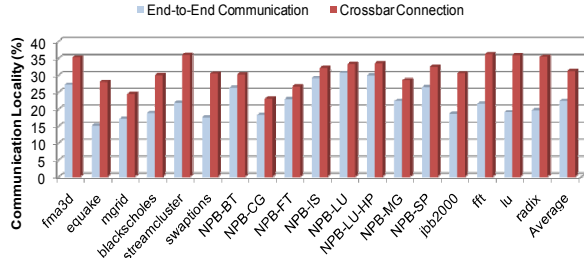


Figure 1. Communication Temporal Locality Comparison

router. It enables the flit to be sent directly to the downstream router bypassing the switch arbitration stage, thus reducing one pipelining stage in the router. A crossbar connection created by a flit traversal within a single router, simply a *pseudo-circuit*, remains connected for future uses. The pseudo-circuit is terminated when another recent flit claims either the input port or the output port. To avoid buffer being full at the downstream router, a pseudo-circuit is also terminated when there is no more credit at the downstream router. This scheme does not need any performance overhead to terminate a pseudo-circuit because pseudo-circuits are managed by crossbar switches within each router.

We also propose two aggressive pseudo-circuit schemes to accelerate communication performance further. First, pseudo-circuit speculation generates more pseudo-circuits with currently unallocated crossbar connections for the future communication based on history information. Second, buffer bypassing allows flits to skip buffer writes at input virtual channels (VCs), thus removing one more pipeline stage. These aggressive schemes increase pseudo-circuit reusability and decrease average per-hop router delay.

To show the effectiveness of the proposed pseudo-circuit scheme, we evaluate performance and energy consumption using our cycle-accurate flit-based wormhole switching on-chip network simulator with credit-based VC flow control. We use both traces and synthetic workloads for traffic models. Traces are extracted from SPECComp2001, PARSEC, NAS Parallel Benchmarks, SPECjbb2000, and Splash-2 on a self-throttling CMP network with 4 MSHRs (Miss Holding Status Register [16]) per processing core in Simics [20]. The pseudo-circuit scheme improves overall network performance up to 16% with traces when combined with both aggressive schemes. It also saves about 5% of energy consumed in routers. Evaluated with synthetic workload traffic, it has performance improvement by up to 11%.

The remainder of this paper is organized as follows. We briefly discuss related work in Section 2. After presenting our pseudo-circuit scheme in Section 3, we describe two aggressive schemes in Section 4. In Sections 5 and 6, we describe evaluation methodology and present simulation results, followed by discussion in Section 7. In Section 8, we conclude our work.

II. RELATED WORK

To provide low latency in on-chip interconnection networks, several techniques have been proposed to reduce per-hop router delay. A speculative pipelined router design

[22] reduces 1 pipeline stage by parallelizing switch arbitration (SA) with virtual-channel allocation (VA) using speculation. A low-latency router [21] reduces per-hop router delay to fit in one cycle in very low traffic by removing control overhead from the critical path using pre-computation. SPAROFLO [17] switch allocation increases efficiency of crossbar connections by separating allocation mechanisms and prioritizing old requests. Token flow control [19] uses tokens to disseminate information about availability of network resources and lookahead link traversal to send a setup flit one cycle prior to flit traversal, thus resulting in bypass router pipelines with additional wiring overheads for tokens and lookahead link traversal. In Express Virtual Channel (EVC) [18], packets are virtually bypassing intermediate routers by reserving some VCs with higher priority than the other normal VCs in order to form an express channel within a single dimension using latches in the intermediate routers, thus minimizing per-hop router delay when packets keep traversing in the same dimension.

Several topologies for on-chip interconnection networks are proposed in [5] to reduce the average number of hops using high-radix routers [15]. Multiple independent parallel networks are also introduced in several topologies such as concentrated meshes to improve wire and area utilization [5]. By accommodating multiple injection channels, traffic can be evenly distributed into all parallel networks. Flattened butterfly [14] minimizes communication latency by providing express channels between nodes in the same dimension. Recently proposed express cube topology [10] uses multidrop express channels to send packets to the nodes in the same dimension in a bandwidth-efficient manner.

III. PSEUDO-CIRCUIT: REUSING CROSSBAR CONNECTIONS

The key design goal of the proposed pseudo-circuit scheme is to reduce the communication latency by reusing crossbar connections established by previous flits. In this section, we describe an existing general router architecture widely used in on-chip interconnection networks and present our pseudo-circuit scheme which reuses arbitration information created by previous communication.

A. General Router Architecture

We use a state-of-the-art router architecture [22] as shown in Fig. 2. As shown in Fig. 6, it has 2 pipelined stages performing virtual-channel allocation (VA) and switch arbitration (SA) at the first stage, and switch traversal (ST) at the second stage. Routing calculation (RC) is removed from the critical path by adopting lookahead routing [9] that generates routing information of the downstream router. SA is speculatively performed in parallel with VA. The VC allocator logic allocates one available VC at the input port of the downstream router. The switch arbiter logic arbitrates input and output ports of the crossbar. When a flit enters a router, it should be written into the buffer in one of VCs of the input port during buffer write (BW). After the flit traverses through the crossbar, we assume link traversal (LT) takes one cycle.

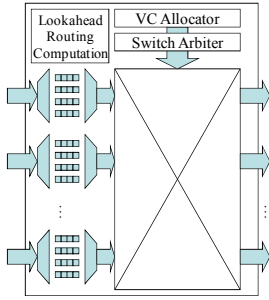


Figure 2. Baseline Router Architecture

Each router has multiple VCs per input port for low Head-of-Line blocking. It uses flit-based wormhole switching [8] and credit-based VC flow control [7] for a small buffer cost to minimize the area cost in on-chip interconnection networks. This flow control provides back-pressure from downstream routers to upstream routers to avoid buffer overflow.

Communication messages are transmitted as packets. A sender network interface (NI) splits a packet into multiple flits to fit in the communication bandwidth for flow control and injects them serially. At the receiver NI, flits are restored to a packet after receiving all flits. The first flit of a packet is called a header flit, where routing information is stored, and the last flit is a tail flit. The other flits are called body flits. Once the header flit is routed to its destination according to its routing information, the remaining flits follow the header flit from source to destination.

B. Pseudo-Circuit Creation and Reuse

A pseudo-circuit is defined as a currently available crossbar connection from an input port to an output port within a single router made by the switch arbiter using previous communication. Every flit traversal in a router creates a crossbar connection from an input port to an output port after arbitrated by the switch arbiter, which is written to the pseudo-circuit register in the input port. After the traversal, the pseudo-circuit remains connected for future uses until it is terminated.

Fig. 4 (a) shows pseudo-circuit creation by a flit traversal. The flit traversal from the input port N to the output port E creates a pseudo-circuit, and its information is written to the pseudo-circuit registry in the input port. If the next flit arriving at the same input VC in the same input port needs to traverse the same crossbar connection in the router as shown in Fig. 4 (b), it simply goes directly to the crossbar bypassing SA because the crossbar connection is already set up and remaining connected. To bypass SA completely, each pseudo-circuit needs to hold the recent arbitration history of the switch arbiter. Since the first part of the switch arbiter arbitrates the input VCs, each pseudo-circuit should hold the recently used input VC number, so the next flit is expected to come to the same input VC in order to reuse the pseudo-circuit.

A pseudo-circuit is reused if a flit coming from the same VC at the input port has the same routing information as the previous flit to the same output port. In order to check if the flit can traverse the pseudo-circuit, the router needs to

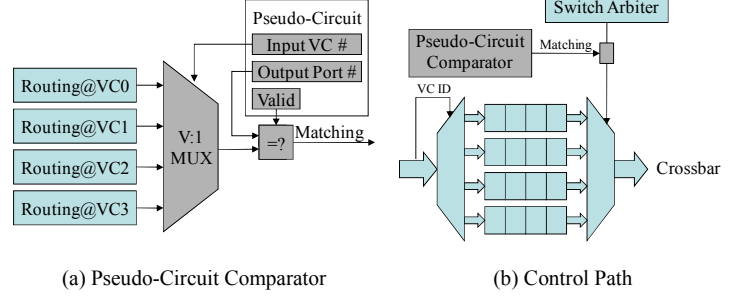


Figure 3. Pseudo-Circuit Architecture

compare the routing information of the flit with the pseudo-circuit information created by previous communication. It performs one simple comparison of routing information stored in input VCs. If the comparison logic asserts a matching signal, the flit can traverse the pseudo-circuit. If the routing information of the flit is different from the pseudo-circuit, the pseudo-circuit is terminated by the switch arbiter (See Section 3.C). In this case, there is no performance overhead because the flit experiences the baseline pipeline stages. The routing information is always carried by header flits. Once the header flit has matching routing information with the pseudo-circuit, the following flits coming to the same VC can bypass SA without the routing information until the pseudo-circuit is terminated.

The pseudo-circuit remains connected unless there is another recent pseudo-circuit conflicting with the pseudo-circuit. If the pseudo-circuit is connected, it is ready to send flits directly to the downstream router through the crossbar without SA. Therefore, if the comparator generates a matching signal with the current pseudo-circuit, the flit can traverse the crossbar bypassing SA. When the flit goes to the crossbar without SA, we assume that VA is performed independently from SA and ST as shown in Fig. 6. If SA is speculative, VA information is not needed until the flit is arriving at the downstream router.

Fig. 3 (a) shows the pseudo-circuit comparator logic that determines whether the flit can use the pseudo-circuit. This logic contains two registers, a one-bit flag, one multiplexer, and one comparator; a register for the input VC of the pseudo-circuit, a register for the output port of the pseudo-circuit, a one-bit flag indicating whether or not the pseudo-circuit information stored in the registers is valid, a multiplexer to select one input VC, and a comparator to compare the routing information of the selected flit. Every input port has pseudo-circuit comparator logic because every input port can be connected with a pseudo-circuit. Since the area overhead of the logic is very small compared to the other router control logic, we assume the hardware overhead of the pseudo-circuit scheme is negligible.

This scheme needs additional delay to compare routing information with the pseudo-circuit before sending flits to the crossbar in ST. According to our HSPICE analysis at 45nm, the pseudo-circuit comparator takes 37ps, which can be fit into ST because ST takes only 215ps in the baseline microarchitecture and the pipeline period is 299ps which is determined by VA. SA has the same delay as it does in the

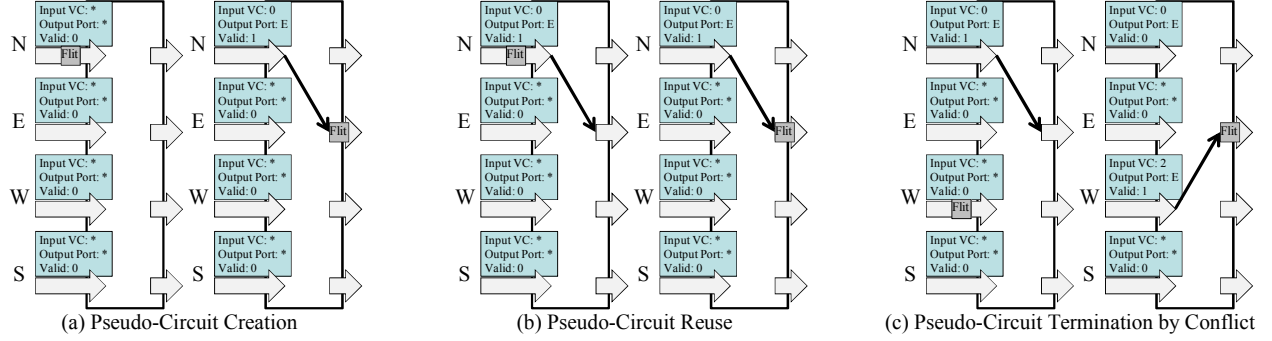


Figure 4. Pseudo-Circuit Creation, Reuse, and Termination

baseline (290ps) because switch allocation in SA is performed independently from pseudo-circuits. Since pseudo-circuit information is updated in parallel with crossbar switch setup after switch allocation and termination is performed while switch allocation is done, no additional delay is required in SA. Therefore, this scheme has no additional overhead in delay analysis and does not affect the router clock frequency.

C. Pseudo-Circuit Termination

There are two conditions for pseudo-circuit termination; (1) a conflict with another recent pseudo-circuit and (2) congestion at the downstream router on the output port. If either the input port or the output port is assigned to another pseudo-circuit, the previous pseudo-circuit must be terminated because one input or output port cannot have more than one pseudo-circuit. If, for instance, another flit at a different input port claims the same output port as shown in Fig. 4 (c), a new pseudo-circuit is created, and the previous pseudo-circuit is terminated and disconnected while clearing the valid bit in the pseudo-circuit registry without changing the registers. After termination, no flit is accepted for the terminated pseudo-circuit without SA because there is no pseudo-circuit. Thus, router latency is improved only when there is a proper pseudo-circuit connected from an input port to an output port in the router. Since pseudo-circuit traversal is made only when no other flit in SA claims any part of the pseudo-circuit, this scheme is free from starvation.

A pseudo-circuit is also terminated when the downstream router connected to the output port is congested. In order that the pseudo-circuit existence guarantees credit availability of the output port, the router needs to check the credit of the output port when performing SA. If congestion is detected, the pseudo-circuit at the output port with no credit should be immediately terminated to avoid buffer overflow. After the pseudo-circuit is terminated, no flit can go directly to the crossbar without SA. Since the switch arbiter performs arbitration based on the credit availability in the downstream routers, flits cannot traverse to the congested output port until the congestion is relieved. In this case, they need to stay in the buffer at the input VC. If the router also experiences congestion because of the congestion at the output port, flits coming from the upstream routers occupy buffers without traversing. This congestion produces Head-of-Line blocking and the following flits are also stored in the buffer. Once the buffers in the input port are full, credit back-pressure results

in pseudo-circuit termination in the upstream router. If network congestion is maintained for a long time, it is finally propagated to the source node.

Pseudo-circuit termination does not need any performance overhead. Pseudo-circuits are managed by the switch arbiter in each router and each pseudo-circuit is connected by a crossbar switch within a crossbar. Terminating the pseudo-circuit simply disconnects the crossbar switch while clearing the valid bit at the pseudo-circuit register in the input port. If the switch arbiter needs to connect another crossbar connection using either the input port or the output port of the pseudo-circuit, it turns on the crossbar switch while terminating the previous pseudo-circuit. Thus, there is no performance overhead when a pseudo-circuit is terminated.

IV. AGGRESSIVE PSEUDO-CIRCUIT SCHEMES

In this section, we present two aggressive pseudo-circuit schemes for further performance improvement. First, *pseudo-circuit speculation* creates more pseudo-circuits from currently unallocated crossbar connections for future flits using history information to increase pseudo-circuit reusability. Second, *buffer bypassing* allows flits to skip buffer writes at input VCs to reduce per-hop router delay. These two aggressive schemes can be combined with the basic pseudo-circuit scheme for further performance improvement.

A. Pseudo-Circuit Speculation

Generally, not all of the possible crossbar connections are allocated to flits or pseudo-circuits. However, those unallocated crossbar connections may be claimed by near-future flits. If near-future flits traverse these crossbar connections as pseudo-circuits, the flits can bypass SA, thus reducing per-hop router delay. Speculatively creating more pseudo-circuits by connecting the unallocated crossbar connections improves performance because it increases pseudo-circuit reusability.

The prediction is performed based on the history information at each input port. If the output port most recently connected to an input port becomes available, pseudo-circuit speculation restores the pseudo-circuit when performing SA, connecting the input port and the output port again. There are two conditions for speculative pseudo-circuit restoration. First, if the output port becomes available, pseudo-circuit speculation revives the pseudo-circuit. Second, if there is

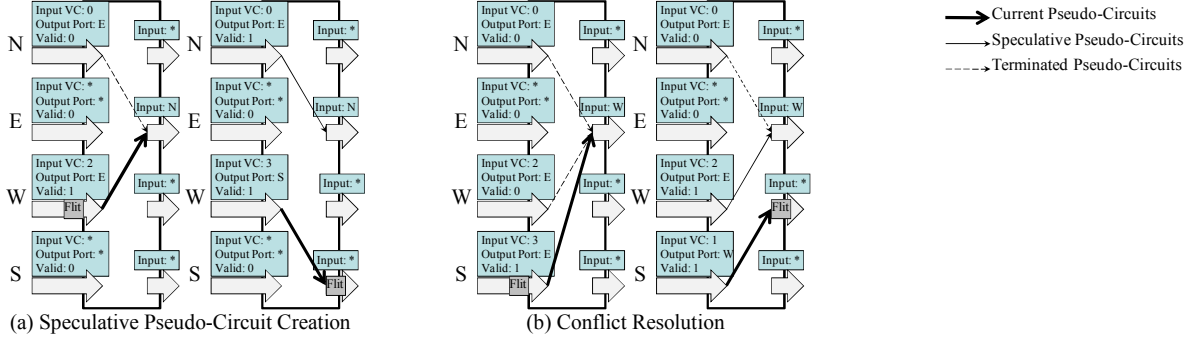


Figure 5. Pseudo-Circuit Speculation

congestion relief at the downstream router of the output port, the pseudo-circuit can be speculatively reestablished.

Fig. 5 (a) shows an example when a speculative pseudo-circuit is created in a router. In this figure, the input port N is speculatively connected to the output port E because the previous pseudo-circuit from the input port N was recently connected to the output port E and this output port is now available and unallocated. To resolve conflict when more than one input port have the same history information to the same output port, pseudo-circuit speculation has a register to store the input port number of the last pseudo-circuit at each output port and connects the output port only to the input port the register indicates. This history register at each output port indicates the input port of the most recently terminated pseudo-circuit. For instance, the previous pseudo-circuit at the input port W is speculatively connected to the output port E instead of the input port N because the input port W is more recently connected to the output port E as shown in Fig. 5 (b). To avoid buffer overflow in the downstream router, pseudo-circuit speculation does not create any pseudo-circuit to the output port of the congested downstream router which has no available credit.

Since every input port has a pseudo-circuit comparator, the pseudo-circuit scheme performs comparison of the pseudo-circuit with the routing information of an incoming flit. If the speculation is correct, the flit can traverse directly to the crossbar because the comparator generates a matching signal. If not, the comparator does not assert a match. In the mismatch case, flits go through SA resulting in no additional penalty after terminating the speculative pseudo-circuit. Since the termination of a speculative pseudo-circuit is performed in parallel with switch allocation, pseudo-circuit speculation has no negative performance impact.

B. Buffer Bypassing

We observe that flits traversing pseudo-circuits go directly to the crossbar at the next cycle after buffer write (BW) in most cases. Since the router anticipates flits after creating pseudo-circuits, there is no need to spend one cycle to write the flits to buffers at input VCs if the flits can traverse the pseudo-circuits. Instead, the flits can bypass buffers through the bypass latch at the input VC and go directly to the crossbar only if the pseudo-circuit is already available and connected from the input port to the designated output port. Only if a flit arriving at the input port has the same routing

information to the same output port of the pseudo-circuit, it can bypass the input buffer at the input VC. In this case, the flit goes directly to the crossbar, saving 2 cycles in per-hop router delay as shown in Fig. 6. Otherwise, the flit is stored in the input.

Buffer bypassing can be implemented with write-through input buffers [29] with a bypass latch per each input VC. When a pseudo-circuit is created, the bypass latch is turned on to enable incoming flits to bypass the buffer at the corresponding input VC unless the buffer is occupied. A flit can go to the bypass latch when the VC and the routing information of the flit are matched with the pseudo-circuit. If, for example, an incoming flit has the same routing information when the bypass latch is turned on, it goes directly to the crossbar through the bypass latch. If, however, the incoming flit has different routing information, the flit is stored in the buffer without bypassing buffer. Due to this conflict, the pseudo-circuit is immediately disconnected and invalidated while the bypass latch is turned off. To avoid buffer overflow, the pseudo-circuit is also terminated and the bypass latch is turned off if the output port is out of credit before a flit arrives. Thus, the pseudo-circuit can guarantee credit availability of the output port. Since buffer bypassing condition is controlled by a pseudo-circuit and its comparator logic, buffer bypassing has small hardware overhead.

When buffer bypassing is turned on, there is no need to store the whole flit. VA is performed only for header flits and it needs the output port numbers only. However, an incoming flit is always written to buffer just in case when the speculation is failed. If the speculation is correct, there is no pointer increment and the buffer will be overwritten by the next flit.

V. EXPERIMENTAL METHODOLOGY

We evaluate performance and energy consumption using our cycle-accurate on-chip network simulator implementing pipelined routers with buffers, VCs, arbiters, and crossbars. The network simulator is configured with 4-flit buffer per each VC and 4 VCs per each input port. We assume that the bandwidth of a link is 128 bits with additional error correction bits. We use both traces and synthetic workloads for traffic models. We extract traces from several multi-threaded benchmarks; fma3d, quake, and mgrid from SPECComp2001 [4]; blackscholes, streamcluster, and swaptions from PARSEC [6]; NAS parallel benchmarks [2]; SPECjbb2000

Baseline System No Pseudo-Circuit	Head Flit	BW	VA (299) SA (290)	ST (215)	LT	
	Body/Tail Flit		BW	SA	ST	LT
Pseudo-Circuit	Head Flit	BW	VA (299) PC+ST (252)	LT		
	Body/Tail Flit		BW	PC+ST	LT	
Pseudo-Circuit with Buffer Bypassing	Head Flit	VA (299) PC+ST (252)	LT			
	Body/Tail Flit		PC+ST	LT		

Figure 6. Pipeline Stages: Numbers are indicating critical path delay (ps).

[3]; FFT, LU, and radix from Splash-2 [32]. To extract the trace information, we use Simics [20], a full system simulator, configured as a SunFire multiprocessor system with UltraSPARCIII+ processors running with Solaris 9 operating system. We develop a customized timing-model interface modeling out-of-order cores with 4 MSHRs per each processing core to implement a self-throttling CMP network [16]. Our CMP configuration has 32 out-of-order processors and 32 L2 cache banks in a single chip, modeling static non-uniform cache architecture (S-NUCA) [13]. Fig. 7 shows the layout of the CMP configuration in this experiment. Processing cores and L2 cache banks are connected through the on-chip interconnection network. We use the concentrated mesh topology [5] for the on-chip interconnection network where each router connects 2 processing cores and 2 L2 cache banks to the interconnection network and routers are connected as a 2D mesh topology. Each core has 32KB L1 caches for data and instructions. L2 caches are unified and shared by all processing cores in an address-interleaved manner. We use CACTI model [28] to estimate the latency and the area constraint of the caches. Table I shows the detailed parameters used in the experiment.

We use a directory-based MSI cache coherence protocol. If an L1 cache has a miss, it always generates a request to the corresponding L2 cache bank. After retrieving the data blocks, the L2 cache bank sends a response to the requesting L1 cache. To simplify the cache coherence states, we use write-through and write-invalidation. The cache coherence protocol has 3 different types of transactions. A read transaction is initiated by a read miss in L1 caches; a write

TABLE I. CMP CONFIGURATION PARAMETERS

L1I Cache	1-way 32KB	# Cores	32 out-of-order
L1D Cache	4-way 32KB	# L2 Banks	32 512KB bank
L1 Latency	1 cycle	Cache Block Size	64B
Unified L2 Cache	16-way 16MB	Memory Latency	300 cycles
L2 Bank Latency	20 cycles	Clock Frequency	5GHz

TABLE II. ENERGY CONSUMPTION CHARACTERISTICS OF ROUTER COMPONENTS

Buffer	Crossbar	Arbiter
20.19pJ	65.38pJ	0.20pJ
23.54%	76.22%	0.24%

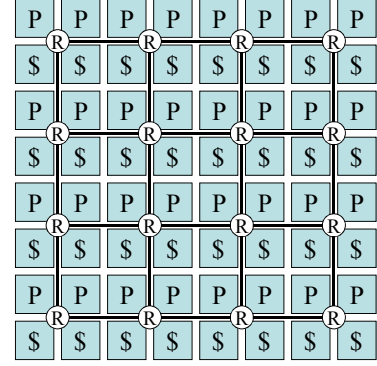


Figure 7. Layout of On-Chip Networks

transaction is initiated by a write miss in L1 caches; a coherence management transaction is initiated to keep shared copies coherent. The size of a network packet depends on whether it contains a data block. If the packet has an address only, it is 1 flit long because the size of an address is fit into a flit. If it has both an address and a data block, it is 5 flit long because the last 4 flits contain the data.

We use Orion [30] to estimate router energy consumption. Table II shows characteristics of energy consumption and the percentage of energy consumed in each router component at 45nm. In this experiment, we assume that the amount of energy consumed in pseudo-circuit comparators can be negligible because it can be implemented with simple logics compared to the other router control logics.

In this experiment, we use dimension order routing (DOR) [26] algorithms (XY, YX) and O1TURN [24]. O1TURN randomly chooses the first dimension to traverse between XY and YX. We also use two VC allocation policies. First, dynamic VA policy chooses an output VC based on buffer availability in a downstream router. This allocation policy is generally used in interconnection networks. Second, static VA policy chooses an output VC based on the destination of the communication path. If, for example, two different communications have the same destination ID, these are on the same VC at all input ports. If they share the same communication path from a certain point in the network, they use the same pseudo-circuit in each router in this shared communication path. This static VA is similar with [25] because VC is statically allocated per flow, but we use the destination ID only in order to increase reusability.

VI. PERFORMANCE EVALUATION

In this section, we evaluate the proposed pseudo-circuit scheme to examine how it affects performance and power consumption in on-chip interconnection networks with traces from several benchmarks. We also evaluate its performance with synthetic workload traffic.

A. Performance and Energy Consumption with Traces

Fig. 8 (a) shows overall performance of our proposed scheme when the application traces are used. We use network latency reduction normalized to latency of the baseline system without any pseudo-circuit scheme. To make a fair

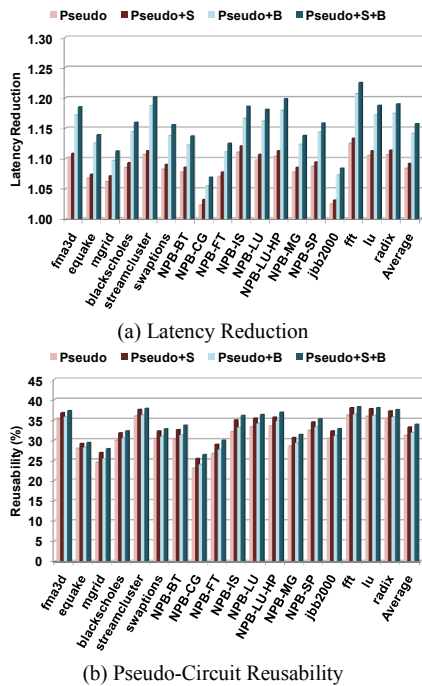


Figure 8. Overall Performance

comparison, we choose the baseline system with O1TURN and dynamic VA scheme, which provides the best performance in the baseline system. Throughout this paper, the pseudo-circuit scheme without any aggressive scheme is indicated by **Pseudo**. The pseudo-circuit scheme with pseudo-circuit speculation is indicated by **Pseudo+S**. The pseudo-circuit scheme with buffer bypassing is indicated by **Pseudo+B**. The pseudo-circuit scheme with both aggressive schemes is indicated by **Pseudo+S+B**. Pseudo-circuit speculation has small contribution in latency reduction due to limited prediction capability as shown in Fig. 8 (a). On average, we achieve 16% of latency reduction when the pseudo-circuit scheme with both aggressive schemes is applied.

Fig. 8 (b) shows overall pseudo-circuit reusability in 7 benchmark applications. Pseudo-circuit reusability, simply reusability, is defined by the percentage of flits reusing pseudo-circuits. The higher reusability is, the more performance improvement is expected. Note that buffer bypassing does not actually increase reusability, but reduces one more cycle in per-hop router delay when the incoming flit bypasses buffer writing. If pseudo-circuit speculation and buffer bypassing are combined together, more performance enhancement is achieved because average per-hop router delay can be reduced more than when the two aggressive schemes are working separately.

Fig. 9 shows the details of latency reduction with 3 routing algorithms (**XY**, **YX**, **O1TURN**) and 2 VC allocation policies (**Static VA**, **Dynamic VA**). Among all possible combinations, dimension order routing (DOR) algorithms with static VA has the best latency reduction in all pseudo-circuit schemes although there is subtle differences in performance between XY and YX. This performance

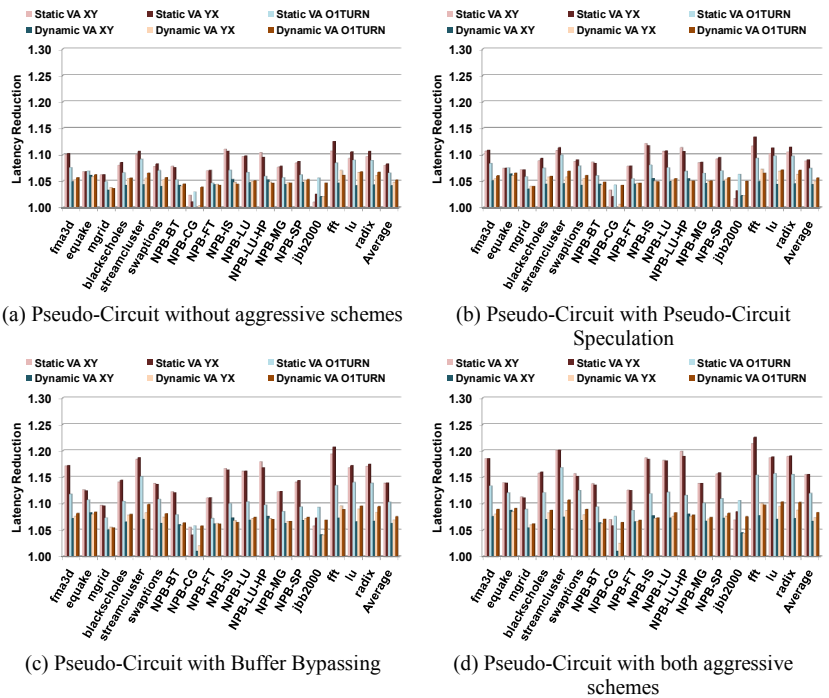


Figure 9. Network Latency Reduction

difference between two DOR algorithms comes from asymmetry in application traces. Generally, higher pseudo-circuit reusability results in higher latency reduction. As shown in Fig. 10, DOR with static VA maximizes pseudo-circuit reusability compared to the other combinations because it always chooses the same output port and the same VC for flows to the same destination. We observe that routing algorithms and VA policies have higher impact on reusability than application locality does. Leveraged by this, our pseudo-circuit scheme with DOR and static VA has the best performance improvement in most multi-threaded benchmarks in general. Note that YX with static VA has traffic concentration causing slightly better pseudo-circuit reusability but less latency reduction than XY with static VA due to contention.

Our proposed pseudo-circuit scheme improves performance better with DOR and static VA in most benchmarks. If, however, an application has highly skewed and over-utilized network hotspots, DOR cannot relieve the hotspot traffic. Instead, O1TURN can achieve higher performance with the pseudo-circuit scheme than DOR because it distributes traffic into every dimension and utilizes every possible link to achieve load balancing. For example, Fig. 9 shows that jbb2000 has better performance with O1TURN than with DOR unlike the other benchmarks due to uneven traffic.

Fig. 11 shows normalized energy consumption in routers. Basically, the pseudo-circuit schemes without buffer bypassing have virtually no energy saving because the amount of energy consumed in arbiters is much smaller than the amount of energy consumed in buffers. However, buffer bypassing scheme reduces energy consumption because the energy portion used in buffer read and write is quite large.

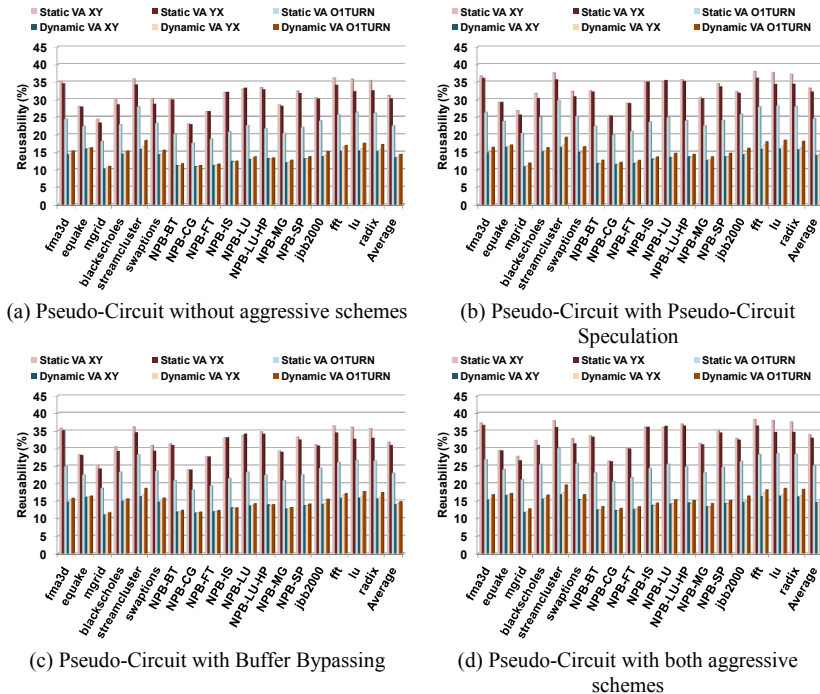


Figure 10. Pseudo-Circuit Reusability

When combined with both buffer bypassing and pseudo-circuit speculation, the pseudo-circuit scheme has more energy saving because it can bypass input buffers more often. Thus, it achieves about 5% energy saving on average.

B. Performance Evaluation with Synthetic Workload Traffic

We also conduct experiments with 3 different kinds of synthetic workload traffic. First, we generate uniform random (**UR**) traffic which randomly sends packets evenly to all nodes. Thus, it has equal chances to use all links. In this traffic, the destination of each communication path may differ from the previous one because it is randomly selected every time. The second traffic is bit complement (**BC**). It generates traffic from every source to one single destination based on bit complement operation. Thus, it has a longer average Manhattan distance than UR. This longer distance results in faster saturation in the network. The third traffic is bit permutation (**BP**) whose communication pattern is generated based on matrix transpose. This traffic has only one destination for each source like BC, but the average Manhattan distance is same as UR. Since every communication needs to cross the diagonal line at the same point with the dimension order routing algorithms, this traffic saturates the network much earlier than BC. In the experiments with synthetic traffic, all packets are 5 flit long.

Fig. 12 shows performance improvement in synthetic workload traffic. We show only the results of XY with static VA policy due to space limit. At any traffic load before saturation, the pseudo-circuit scheme performs better than the baseline system with all synthetic workload. In low-load traffic, UR and BP has nearly 11% performance improvement while BC has only around 6% performance improvement. It

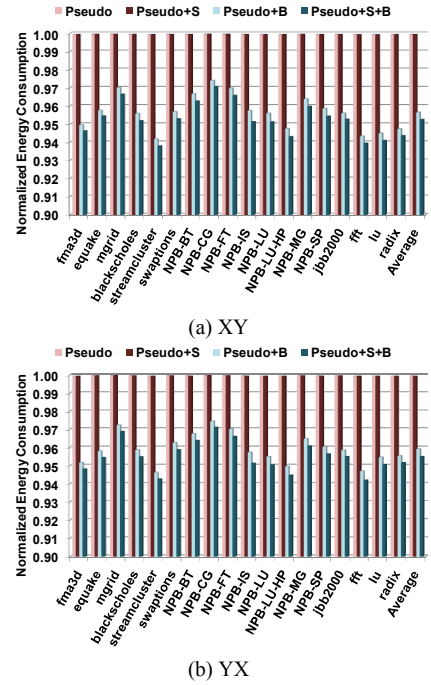


Figure 11. Overall Energy Consumption

is expected that UR has less communication locality because the next packet can be destined to a different destination from a previous packet. However, these two consecutive communications may have a common path in dimension order routing algorithms. Since the pseudo-circuit scheme exploits pseudo-circuit reusability within a single router, it can improve performance within the common path.

VII. DISCUSSION

In this section, we evaluate our pseudo-circuit scheme in various topologies, such as Multidrop Express Cube [10] and Flattened Butterfly [14], to show how it is affected by topologies. We also compare the pseudo-circuit scheme with other on-chip network techniques, such as Express Virtual Channels (EVC) [18].

A. Impact on Various Topologies

Fig. 13 shows communication latencies of several topologies, normalized to the baseline system in a mesh topology. Due to space limit, we show the results of fma3d with DOR and static VA, but we observed that the other benchmarks have the same trend. To show performance improvement of the pseudo-circuit scheme in various topologies, we use a mesh, a concentrated mesh (**CMESH**) [5], Multidrop Express Cube (**MECS**) [10], and Flattened Butterfly (**FBFLY**) [14]. In this experiment, we assume all physical channels have the same bandwidth and each input port has 4 VCs. MECS is configured without any replicated channel, thus resulting in less crossbar complexity than FBFLY.

The communication latency is calculated as $T = H_{avg} * t_{router} + D * t_{link} + T_{serialization}$ where H_{avg} is the average number of hops, t_{router} is per-hop router delay, D is average

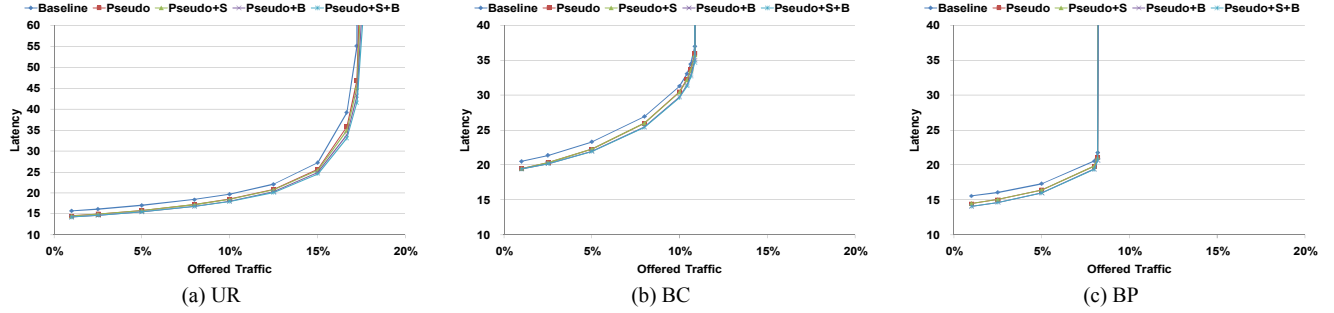


Figure 12. Performance Comparison with Synthetic Workload Traffic

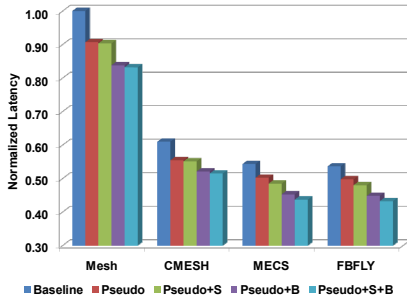


Figure 13. Impact on Various Topologies on *fma3d*

distance from source to destination, and t_{link} is unit length delay. Since link latency and serialization latency are constants, the total communication latency is determined by the product of per-hop router delay and the average number of hops. The pseudo-circuit scheme reduces per-hop router delay regardless of the underlying topology while recently proposed topologies reduce the number of hops. Simulation results show the pseudo-circuit scheme achieves up to 20% performance improvement in any topology. Therefore, combination with recent topologies results in more than 50% latency reduction compared to the baseline system with a mesh topology.

B. Comparison with Express Virtual Channels

Fig. 14 shows performance comparison with EVC [18]. We use 4 VCs per each input port and 4-flit buffer per each VC in both techniques. We use dynamic EVC with the maximum number of hops $l_{max} = 2$ where 2VCs are reserved for express VCs (EVCs) and the other 2 VCs are used for normal VCs (NVCs). In each graph in this figure, latency is normalized to the baseline system in each topology.

EVC reserves some VCs to send packets to routers in multiple hops away in the same dimension. Thus, it improves communication latency by prioritizing these EVCs over the other NVCs because it provides express channels virtually. If, however, a network topology has a small number of routers in a single dimension, our experiment shows that EVCs are not sufficiently used and this unbalanced usage may cause performance degradation due to the reduced number of NVCs. For example, the concentrated mesh topology does not have performance improvement on average with EVC as shown in Fig. 14 (b) because most flits are stored in NVCs, and these NVCs are easily filled with flits. Thus, EVC is heavily

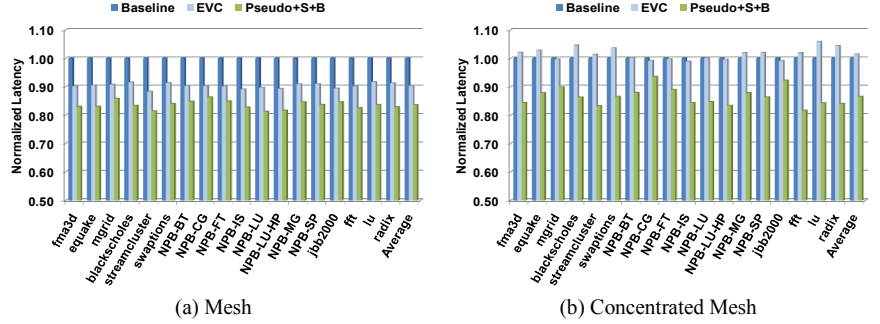


Figure 14. Comparisons with Express Virtual Channels (EVC)

dependent on the topology used in on-chip interconnection networks.

The pseudo-circuit scheme has several advantages. First, there is no topological restriction in creating pseudo-circuits. Pseudo-circuits can be established from any input port to any output port. Thus, it is topologically independent as shown in Section 7.A. Second, the pseudo-circuit scheme does not reserve any resource when creating pseudo-circuits. If there is a pseudo-circuit in the input port and the next flit is destined to another output port, the pseudo-circuit is terminated to enable the flit to traverse through the crossbar. Besides, there is no performance overhead to terminate the pseudo-circuit. Finally, there is no starvation, since the pseudo-circuit is simply disconnected and terminated immediately to avoid starvation when there is a pseudo-circuit conflict with flits in SA.

VIII. CONCLUSIONS

Communication latency has become the performance bottleneck in CMPs as the number of cores in a chip is increased with higher technology. To overcome the bottleneck, designing a low-latency on-chip network is crucial. We introduce a pseudo-circuit scheme to reduce per-hop router delay by reusing pseudo-circuits, crossbar connections within a router with previous arbitration information. This scheme enables flits to bypass SA when they are traversing through the same communication path created by previous communication. For further performance improvement, we also propose two more aggressive schemes; pseudo-circuit speculation and buffer bypassing. Pseudo-circuit speculation generates more pseudo-circuits using currently unallocated crossbar connections for future communication while buffer bypassing allows flits to skip

buffer writes at input VCs and removes one more pipeline stage from per-hop router delay. Combined with both aggressive schemes, the pseudo-circuit scheme enhances overall performance of on-chip interconnection networks by 16% with traces from SPECComp2001, PARSEC, NAS Parallel Benchmarks, SPECjbb2000, and Splash-2. It also improves about 5% of energy consumption in routers. Evaluated with synthetic workload traffic, this scheme shows performance improvement by up to 11%. If applied to the recently proposed topologies, it improves performance by up to 20%, resulting in more than 50% latency reduction.

We plan to integrate our design in a full system simulator to evaluate the overall system performance such as IPC. We would like to enhance the pseudo-circuit scheme to improve performance in any traffic load because the pseudo-circuit hardly reduces communication latency in high-load traffic due to contentions between flits.

ACKNOWLEDGEMENT

We sincerely thank Ki Hwan Yum for his comments on earlier drafts on this paper. We also thank the anonymous reviewers for their useful comments.

REFERENCES

- [1] The International Technology Roadmap for Semiconductors, <http://www.itrs.net/>
- [2] Nas Parallel Benchmarks, <http://www.nas.nasa.gov/Resources/Software/npb.html>
- [3] Specjbb 2000 Benchmark, <http://www.spec.org/jbb2000/>
- [4] Speccomp 2001 Benchmark Suite, <http://www.spec.org/omp/>
- [5] J. Balfour and W. J. Dally, "Design Tradeoffs for Tiled Cmp on-Chip Networks," in Proceedings of the 20th annual International Conference on Supercomputing, Cairns, Queensland, Australia, 2006.
- [6] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The Parsec Benchmark Suite: Characterization and Architectural Implications," in Proceedings of the 17th international conference on Parallel Architectures and Compilation Techniques Toronto, Ontario, Canada, 2008.
- [7] W. J. Dally, "Virtual-Channel Flow Control," IEEE Transactions on Parallel and Distributed Systems, vol. 3, pp. 194-205, 1992.
- [8] W. J. Dally and C. L. Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," IEEE Transactions on Computers, vol. C-36, pp. 547-553, 1987.
- [9] M. Galles, "Scalable Pipelined Interconnect for Distributed Endpoint Routing: The Sgi Spider Chip," in Proc. Hot Interconnects 4, 1996, pp. 141-146.
- [10] B. Grot, J. Hestness, S. W. Keckler, and O. Mutlu, "Express Cube Topologies for on-Chip Interconnects," in IEEE 15th International Symposium on High Performance Computer Architecture, HPCA, 2009, pp. 163-174.
- [11] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar, "A 5-GHz Mesh Interconnect for a Teraflops Processor," IEEE Micro, vol. 27, pp. 51-61, 2007.
- [12] N. E. Jerger, L.-S. Peh, and M. Lipasti, "Circuit-Switched Coherence," in International Symposium on Networks-on-Chip, NOCS, 2008.
- [13] C. Kim, D. Burger, and S. W. Keckler, "An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated on-Chip Caches," in Proceedings of the 10th international conference on Architectural support for programming languages and operating systems, San Jose, California, 2002.
- [14] J. Kim, J. Balfour, and W. Dally, "Flattened Butterfly Topology for on-Chip Networks," in Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture, 2007.
- [15] J. Kim, W. J. Dally, B. Towles, and A. K. Gupta, "Microarchitecture of a High Radix Router," in 32nd annual International Symposium on Computer Architecture, ISCA, 2005, pp. 420-431.
- [16] D. Kroft, "Lockup-Free Instruction Fetch/Prefetch Cache Organization," in Proceedings of the 8th Annual International Symposium on Computer Architecture, ISCA, Minneapolis, Minnesota, United States, 1981.
- [17] A. Kumar, P. Kundu, A. P. Singh, L. S. Peh, and N. K. Jha, "A 4.6tbits/S 3.6ghz Single-Cycle Noc Router with a Novel Switch Allocator in 65nm Cmos," in 25th International Conference on Computer Design, ICCD, 2007, pp. 63-70.
- [18] A. Kumar, L.-S. Peh, P. Kundu, and N. K. Jha, "Express Virtual Channels: Towards the Ideal Interconnection Fabric," in Proceedings of the 34th annual International Symposium on Computer Architecture, ISCA, San Diego, California, USA, 2007.
- [19] A. Kumar, L. S. Peh, and N. K. Jha, "Token Flow Control," in 41st IEEE/ACM International Symposium on Microarchitecture, MICRO-41, 2008, pp. 342-353.
- [20] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, and B. Werner, "Simics: A Full System Simulation Platform," Computer, vol. 35, pp. 50-58, 2002.
- [21] R. Mullins, A. West, and S. Moore, "Low-Latency Virtual-Channel Routers for on-Chip Networks," in Proceedings of the 31st Annual International Symposium on Computer Architecture, ISCA, Munchen, Germany, 2004.
- [22] L.-S. Peh and W. J. Dally, "A Delay Model and Speculative Architecture for Pipelined Routers," in Proceedings of the 7th International Symposium on High-Performance Computer Architecture, HPCA, 2001.
- [23] K. Sankaralingam, R. Nagarajan, L. Haiming, K. Changkyu, H. Jaehyuk, D. Burger, S. W. Keckler, and C. R. Moore, "Exploiting Iip, Tlp, and Dlp with the Polymorphous Trips Architecture," in 30th annual International Symposium on Computer Architecture, ISCA, 2003, pp. 422-433.
- [24] D. Seo, A. Ali, W.-T. Lim, N. Rafique, and M. Thottethodi, "Near-Optimal Worst-Case Throughput Routing for Two-Dimensional Mesh Networks," in Proceedings of the 32nd annual International Symposium on Computer Architecture, ISCA, 2005.
- [25] K. S. Shim, M. H. Cho, M. Kinsy, T. Wen, M. Lis, G. E. Suh, and S. Devadas, "Static Virtual Channel Allocation in Oblivious Routing," in Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip: IEEE Computer Society, 2009.
- [26] H. Sullivan and T. R. Bashkow, "A Large Scale, Homogeneous, Fully Distributed Parallel Machine, I," SIGARCH Comput. Archit. News, vol. 5, pp. 105-117, 1977.
- [27] M. B. Taylor, J. Psota, A. Saraf, N. Shnidman, V. Strumpfen, M. Frank, S. Amarasinghe, A. Agarwal, W. Lee, J. Miller, D. Wentzloff, I. Bratt, B. Greenwald, H. Hoffmann, P. Johnson, and J. Kim, "Evaluation of the Raw Microprocessor: An Exposed-Wire-Delay Architecture for Iip and Streams," in 31st annual International Symposium on Computer Architecture, ISCA, 2004, pp. 2-13.
- [28] S. Thoziyoor, N. Muralimanohar, and N. P. Jouppi, "Cacti 5.0," HP Technical Reports, 2007.
- [29] H.-S. Wang, L.-S. Peh, and S. Malik, "Power-Driven Design of Router Microarchitectures in on-Chip Networks," in Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture, MICRO-36, 2003.
- [30] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A Power-Performance Simulator for Interconnection Networks," in Proceedings. 35th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO-35, 2002, pp. 294-305.
- [31] D. Wentzloff, P. Griffin, H. Hoffmann, B. Liewei, B. Edwards, C. Ramey, M. Mattina, M. Chyi-Chang, J. F. Brown, and A. Agarwal, "On-Chip Interconnection Architecture of the Tile Processor," IEEE Micro, vol. 27, pp. 15-31, 2007.
- [32] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The Splash-2 Programs: Characterization and Methodological Considerations," in Proceedings of the 22nd annual International Symposium on Computer Architecture, ISCA, S. Margherita Ligure, Italy, 1995.