This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2013.2295523,

Leaving One Slot Empty: Flit Bubble Flow Control for Torus Cache-coherent NoCs

Sheng Ma, Zhiying Wang, *Member, IEEE*, Zonglin Liu
and Natalie Enright Jerger, *Senior Member, IEEE*

Abstract—Short and long packets co-exist in cache-coherent NoCs. Existing designs for torus networks do not efficiently handle variable-size packets. For deadlock free operations, a design uses two VCs, which negatively affects the router frequency. Some optimizations use one VC. Yet, they regard all packets as maximum-length packets, inefficiently utilizing the precious buffers. We propose flit bubble flow control (FBFC), which maintains one free flit-size buffer slot to avoid deadlock. FBFC uses one VC, and does not treat short packets as long ones. It achieves both high frequency and efficient buffer utilization. FBFC performs 92.8% and 34.2% better than LBS and CBS for synthetic traffic in a 4×4 torus. The gains increase in larger networks; they are 107.2% and 40.1% in an 8×8 torus. FBFC achieves an average 13.0% speedup over LBS for PARSEC workloads. Our results also show that FBFC is more power efficient than LBS and CBS, and a torus with FBFC is more power efficient than a mesh.

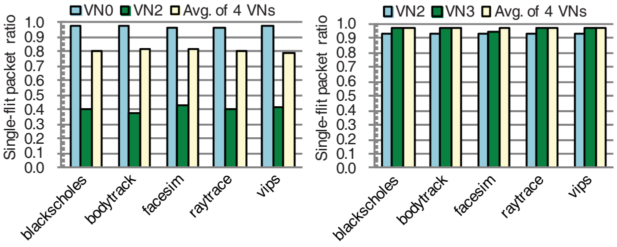
Index Terms—Cache Coherence, Torus Networks-on-Chip, Deadlock Avoidance Theory, Buffer Utilization.

1 Introduction

Optimizing NoCs [19] based on coherence traffic is necessary to improve the efficiency of many-core coherence protocols [41]. The torus is a good NoC topology candidate [52], [53]. The wraparound links convert plentiful on-chip wires into bandwidth [19], and reduce hop counts and latencies [52]. Its node-symmetry helps to balance network utilization [52], [53]. Several products [21], [28], [32] use a ring or 1D torus NoC. Also, the 2D or high dimensional torus is widely used in off-chip networks [4], [18], [44], [51].

Despite the many desirable properties of a torus, additional effort is needed to handle deadlock due to cyclic dependencies introduced by wraparound links. A deadlock avoidance scheme should support high performance with low overhead. Requiring minimum VCs [16] is preferable, because more VCs increase the router complexity. Buffers are a precious resource [24], [46]; an efficient design should maximize performance with limited buffers. There is a gap between existing proposals and these requirements.

A conventional design [20] uses two VCs to remove cyclic dependencies; this introduces large allocators and hurts the router frequency. Optimizations [10], [11] for virtual cut-through (VCT) networks [33] avoid deadlock by preventing the use of the last free packet-size buffer inside rings; only one VC is needed. However, with variable-size packets, each packet must be regarded as a maximum-length packet [4]. This restriction prevents deadlock, but results in poor buffer utilization and performance, especially for short packet dominating coherence traffic.



(a) MOESI directory. (b) AMD's Hammer.

Fig. 1: Single-flit packet ratios. (MOESI directory: VN0 has read request (1-flit), clean write-back (1-flit), dirty write-back (5-flit). VN2 has ACK (1-flit), read response (5-flit). AMD's Hammer: VN2 has ACK (1-flit), read response (5-flit). VN3 has unblock (1-flit), clean write-back (1-flit), dirty write-back (5-flit).)

In addition to the majority short packets, cache-coherent NoCs also deliver long packets. Even though multiple virtual networks (VNs) [20] may be configured to avoid protocol-level deadlock, these two types of packets still co-exist in a single VN. For example, short read requests and long write-back requests are sent in VN0 of AlphaServer GS320, while long read responses and short write-back acknowledgements are sent in VN1; both VNs carry variable-size packets [25]. Similarly, all VNs in DASH [37], Origin 2000 [36], and Piranha [6] deliver variable-size packets.

With a typical 128-bit NoC flit width [24], [39], [46], the majority control packets have one flit; the remaining data packets contain a 64B cache line and have 5 flits. Fig. 1 shows the packet length distribution of some PARSEC workloads [8] with two coherence protocols[1]. Both protocols use four VNs. For each protocol, two VNs carry variable-size packets, and the other two have only short packets. The single-

- *Sheng Ma, Zhiying Wang and Zonglin Liu are with the State Key Laboratory of High Performance Computing, National University of Defense Technology, China. Sheng Ma is also with the National Key Laboratory for Paralleling and Distributed Processing, NUDT.*
E-mail: {masheng, zywang, liuzonglin}@nudt.edu.cn
This research was mostly done when Sheng Ma was a visiting international student at the University of Toronto.
- *Natalie Enright Jerger is with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ontario, Canada.*
E-mail: enright@eecg.toronto.edu

[1] See Sec. 5 for experimental configuration and description.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2013.2295523,

flit packet (SFP) ratios of VN0 in the MOESI directory [45], and VN2 and VN3 in the AMD's Hammer [15] are all higher than 90%. With such high SFP ratios, regarding all packets as maximum-length packets strongly limits buffer utilization. As shown in Sec. 6.2, existing designs' buffer utilization in saturation is less than 40%. This brings large performance loss.

To address existing designs' limitations, we propose a novel deadlock avoidance theory, flit bubble flow control (FBFC), for torus NoCs. FBFC leverages wormhole flow control [18]. It avoids deadlock by maintaining one free *flit-size* buffer slot inside a ring. Only one VC is needed, reducing the allocator size and improving the frequency. Furthermore, short packets are not regarded as long packets in FBFC, leading to high buffer utilization. Based on this theory, we provide two implementations: FBFC-L and FBFC-C.

Experimental results show that FBFC outperforms dateline [20], LBS [10] and CBS [11]. FBFC achieves a ∼30% higher router frequency than dateline. For synthetic traffic, FBFC performs 92.8% and 34.2% better than LBS and CBS in a 4×4 torus. FBFC's advantage is more significant in larger networks; these gains are 107.2% and 40.1% in an 8×8 torus. FBFC achieves an average 13.0% and maximal 22.7% speedup over LBS for PARSEC workloads. FBFC's gains increase with fewer buffers. The power-delay product (PDP) results show that FBFC is more power efficient than LBS and CBS, and a torus with FBFC is more power efficient than a mesh. We make the following contributions:

- Analyze the limitations of existing torus deadlock avoidance schemes, and show that they perform poorly in cache-coherent NoCs.
- Demonstrate that in wormhole torus networks, maintaining one *flit-size* free buffer slot can avoid deadlock, and propose the FBFC theory.
- Present two implementations of FBFC; both show substantial performance and power efficiency gains over previous proposals.

2 Limitations of Existing Designs

Here, we analyze existing designs. Avoiding deadlock inside a ring combined with dimensional order routing (DOR) is the general way to avoid deadlock in tori. We use the ring for discussion.

2.1 Dateline

As shown in Fig. 2, dateline [20] avoids deadlock by leveraging two VCs: VC_{0i} and VC_{1i}. It forces packets to use VC_{1i} after crossing the dateline to form acyclic channel dependency graphs [17], [20]. Dateline can be used in both packet-based VCT and flit-based wormhole networks. It uses two VCs, which results in larger allocators and lower router frequency.

2.2 Localized Bubble Scheme (LBS)

Bubble flow control [10], [49] is a deadlock avoidance theory for VCT torus networks. It forbids the use of the last free packet-size amount of buffers (a packet-size bubble); only one VC is needed. Theoretically, any free packet-size bubble in a ring can avoid deadlock [10], [49]. However, due to difficulties of gathering global information and coordinating resource allocation for all nodes, previous designs apply a localized scheme; a packet is allowed to inject only when the receiving VC has two free packet-size bubbles [10], [49]. Fig. 3 gives an example. Here, three packets, P_0, P_1 and P_2, are waiting. Theoretically, they all can be injected. Yet, with a localized scheme, only P_0 can be injected since only VC_1 has two free packet-size bubbles. LBS requires each VC to be deep enough for two maximum length packets.

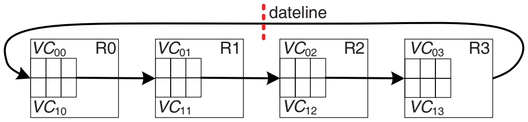


Fig. 2: Dateline uses two VCs.

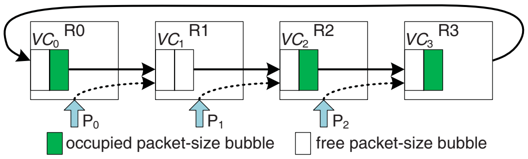


Fig. 3: LBS uses one VC with two packet-size bubbles.

2.3 Critical Bubble Scheme (CBS)

Critical Bubble Scheme (CBS) [11] marks at least one packet-size bubble in a ring as critical. A packet can be injected only if its injection will not occupy a critical bubble. Control signals between routers track the movement of critical bubbles. CBS reduces the minimum buffer requirement to one packet-size bubble. In the example of Fig. 4, the bubble at VC_2 is marked as critical; P_2 can be injected. P_1 cannot be injected since its injection would occupy the critical bubble. Requiring that critical bubbles can be occupied only by packets in a ring guarantees that there is at least one free bubble to avoid deadlock. When P_3 advances into VC_2, the critical bubble moves into VC_1. Now, VC_1 maintains one free bubble.

2.4 Inefficiency with Variable-size Packets

LBS and CBS are proposed for VCT networks; they are efficient for constant-size packets. Yet, as observed by the BlueGene/L team, LBS deadlocks with variable-size packets due to bubble fragmentation [4]. Fig. 5 shows an example with 1-flit packets and 2-flit packets. A free full-size (two-slot) bubble exists in VC_2 at cycle 0. When P_0 moves into VC_2, the bubble is fragmented across VC_1 and VC_2. VCT re-allocates a VC only if it has enough space for an entire packet. Since VC_2's free buffer size is less than P_1's length, P_1 cannot advance and deadlock results. CBS has a similar problem. To handle this issue, BlueGene/L regards each packet as a maximum-length packet [4]. Now, there is no bubble fragmentation. However, this reduces buffer utilization, especially in coherence traffic, whose majority is short packets.

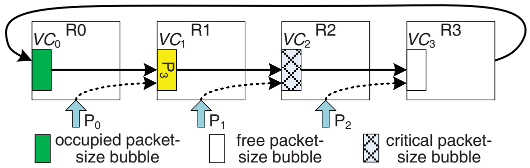


Fig. 4: CBS uses one VC with one packet-size bubble.

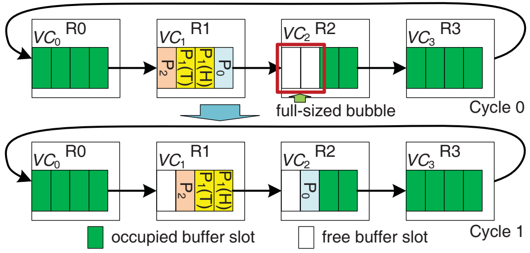


Fig. 5: Deadlock with variable-size packets. (P_i(H) and P_i(T): head and tail flits of P_i. Starting with this figure, each box represents one buffer slot, while it represents a packet-size amount of buffers in Figs. 3 and 4.)

3 Flit Bubble Flow Control

We first propose the FBFC theory. Then, we give two implementations. Finally, we discuss starvation.

3.1 Theoretical Description

We notice that maintaining one free *flit-size* buffer slot can avoid deadlock in wormhole networks. This insight leverages a property of wormhole flow control: advancing a packet with wormhole does not require the downstream VC to have enough space for the entire packet [18]. To show this in Fig. 6, a free buffer slot exists in VC_2 at cycle 0; P_0 advances at cycle 1. A free slot is created in VC_1 due to P_0's movement. Similarly, P_3's head flit moves to VC_1 at cycle 2, creating a free slot in VC_0. This free buffer slot cycles inside the ring, allowing all flits to move.

The packet movement in a ring does not reduce free buffer amounts since forwarding one flit leaves its previously occupied slot free; only injection reduces free buffer amounts. The theory is declared as follows.

***Theorem* 1**: If packet injection maintains one free buffer slot inside a ring, there is no deadlock with wormhole flow control.

Proof Sketch: A deadlock configuration in wormhole networks involves a set of cyclically dependent flits where no flit can move [17]. In a ring, a cyclic dependency needs the participation of all VCs. Thus, we only need to prove that a flit in any VC can advance.

Proof: Assume there is only one free buffer slot at VC_{i+1} and all other VCs are full. We label VC_{i+1}'s upstream VC in the ring as VC_i. There are two possible situations for the flit f at VC_i's head.

1) f is a head flit. If f arrives at the destination, it can be ejected. If f needs to advance into VC_{i+1}, we consider the packet P_k which most recently utilized VC_{i+1}. Again, there are two possible situations.

1.1) P_k was forwarded from VC_i into VC_{i+1}. Since now the head flit f of another packet is at the head of VC_i, P_k's tail flit has already advanced into VC_{i+1}. f can advance with wormhole flow control.

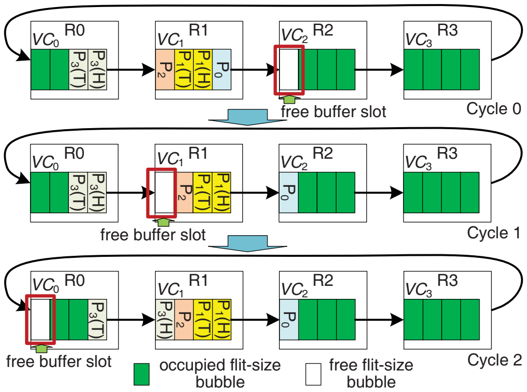


Fig. 6: A wormhole routing example.

1.2) P_k was injected into VC_{i+1}. Its tail flit must have already advanced into VC_{i+1}. Otherwise, the tail flit will occupy the free buffer slot, which violates the premise that the injection procedure maintains one free buffer slot. f can advance.

2) f is a body or tail flit. It can be ejected or forwarded.

In all cases, a flit can move. □

Since one free buffer slot (flit bubble)[2] avoids deadlock, we call this theory flit bubble flow control (FBFC). DOR removes the cyclic dependency across dimensions; combining DOR with FBFC avoids deadlock in tori. FBFC has no bubble fragmentation; its bubble is flit-size. Thus, FBFC does not regard each packet as a maximum-length packet. Only one VC is needed; this improves the frequency. FBFC uses wormhole to move packets inside a ring. It requires the injection procedure to leave one slot empty. Later, we show two schemes to satisfy this requirement.

3.2 FBFC-Localized (FBFC-L)

The key point in implementing FBFC is to maintain a free buffer slot inside each ring. We first give a localized scheme: FBFC-Localized (FBFC-L). When combined with DOR, a dimension-changing packet is treated the same as an injecting packet. The rules of FBFC-L are as follows: (i) Forwarding of a packet within a dimension is allowed if the receiving VC has one free buffer slot. This is the same as wormhole. (ii) Injecting a packet (or changing its dimension) is allowed only if the receiving VC has one more free buffer slot than the packet length. This requirement ensures that after injection, one free buffer slot is left in the receiving VC to avoid deadlock.

Fig. 7 shows an example. Three packets are waiting. The number of free slots in VC_2 and VC_3 are 2 and 4; they are one more than the lengths of P_3 and P_4, respectively. P_3 and P_4 can be injected. After injection, at least two free slots are left in the ring. P_2 cannot be injected since VC_1 only has 1 free slot, which is equal to P_2's length. However, according to wormhole flow control, the free slot in VC_1 allows P_1's head flit to advance. In FBFC-L, each VC must have one more buffer slot than the size of the longest packet.

[2] We use flit bubble and buffer slot interchangeably.

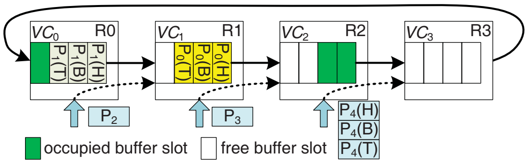


Fig. 7: FBFC-L example. (P_i(H), P_i(B) and P_i(T): Head, body and tail flits of P_i. P_2 and P_3 have one flit.)

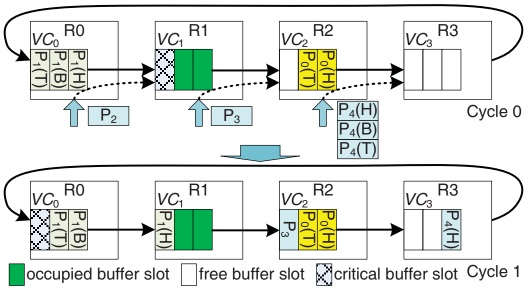


Fig. 8: FBFC-C example. (P_2 and P_3 have one flit.)

3.3 FBFC-Critical (FBFC-C)

To reduce the minimum buffer requirement, we propose a critical design: FBFC-Critical (FBFC-C). FBFC-C marks at least one free buffer slot as a critical slot, and restricts this slot to only be occupied by packets traveling inside the ring. The rules of FBFC-C are as follows: (i) Forwarding of a packet within a dimension is allowed if the receiving VC has one free buffer slot, no matter if it is a *normal* or *critical* slot. (ii) Injecting a packet is only allowed if the receiving VC has enough free *normal* buffer slots for the entire packet. After injection, the critical slot must not be occupied. This requirement maintains one free buffer slot.

Fig. 8 shows an example. At cycle 0, one critical buffer slot is in VC_1. VC_2 and VC_3 have enough free normal slots to hold P_3 and P_4 respectively; P_3 and P_4 can be injected. They do not occupy the critical slot, indicating that the existence of a free slot (the critical slot) elsewhere in the ring. Since the only free slot in VC_1 is a critical one, P_2 cannot be injected. Yet, this critical slot allows P_1's head flit to move. At cycle 1, P_1's head flit advances into VC_1, moving the critical slot backward into VC_0. This is done by R0 asserting a signal to indicate to R3 that it should mark the newly freed slot in VC_0 as a critical one. More details are provided in Sec. 4. The minimum buffer requirement of FBFC-C is the same as CBS; a VC must can hold a largest packet. This is one slot less than FBFC-L.

The injection of FBFC-L and FBFC-C is similar to VCT; they require enough buffers for packets before injection. After injection, a minimum of one slot is left free for wormhole. They can be regarded as applying VCT for injection (or dimension-changing) in wormhole networks. These hybrid schemes are straightforward ways to address existing designs' limitations.

3.4 Starvation

FBFC-L and FBFC-C must deal with starvation. The starvation in FBFC-L is intrinsically the same as in LBS [10]: Injecting packets need more buffers than inside-ring traveling packets. Fig. 9 shows a starvation example for FBFC-L. Here, if node R0 continually injects packets, such as P_0, destined for R3, then P_1 cannot be injected. We design a starvation prevention mechanism; if a node detects starvation, it will notify all other nodes in a ring to stop injecting. A sideband network conveys the control signal ('*starve*'). Once blocked cycles of P_1 exceed a threshold value, R1 asserts the '*starve*' signal. R0 stops injecting after receiving '*starve*' and forwards it to R3. All nodes except R1 stop injecting. Finally, P_1 can be injected. Then, R1 deasserts '*starve*' to resume other nodes' injection. To handle the corner case of multiple nodes simultaneously detecting starvation, the '*starve*' carries a '*ID*' field to differentiate the nodes of a ring. Since the sideband network is unblocking, a router can identify the sending time slot of '*starve*' based on the '*ID*' field. The '*ID*' field and the sending time slot order '*starve*' signals. If the incoming '*starve*' has a higher order than the currently serving '*starve*', the router forwards the incoming signal to its neighbor.

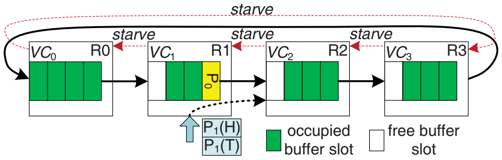


Fig. 9: A starvation example for FBFC-L.

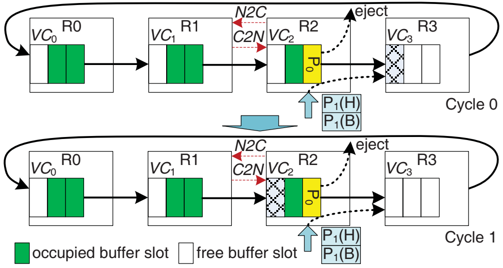


Fig. 10: A starvation example for FBFC-C.

FBFC-C has another starvation scenario in addition to the previous one; it is due to the critical bubble stall. CBS has a similar issue [13]. Fig. 10 shows that the critical bubble is in VC_3 at cycle 0. The bubble movement depends on the packet advancement. If all packets in VC_2, such as P_0, are destined for R2, they will be ejected. Since no packet moves to VC_3, the critical bubble stalls at VC_3. P_1 cannot be injected. This can be prevented by proactively transferring the critical bubble backward if the upstream VC has a free normal bubble. As shown in Fig. 10, a pair of '*N2C*' ('NormalToCritical') and '*C2N*' ('CriticalToNormal') signals are used. If R2 detects that the critical bubble stall prohibits P_1's injection, it asserts '*N2C*' to R1. If VC_2 has a normal free bubble, R1 will change it into a critical one in cycle 1. The '*C2N*' notifies R2 that the critical bubble in VC_3 can now be changed into a normal one. P_1 can be injected. Note that the bubble status is maintained at upstream routers.

4 Router Microarchitecture

In this section, we discuss wormhole routers for FBFC. We also discuss VCT routers for LBS and CBS.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TC.2013.2295523,

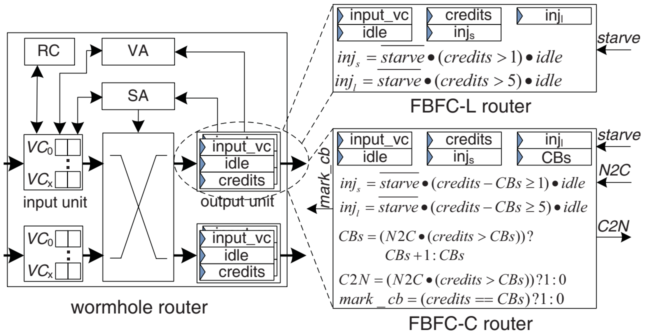

RC
VA
SA
VC0
VCx
input unit
input_vc
idle
credits
output unit
wormhole router
input_vc
credits
injl
idle
injs
starve
$inj_s = \overline{starve} \bullet (credits > 1) \bullet idle$
$inj_l = \overline{starve} \bullet (credits > 5) \bullet idle$
FBFC-L router
input_vc
credits
injl
idle
injs
CBs
starve
N2C
mark_cb
$inj_s = \overline{starve} \bullet (credits - CBs \geq 1) \bullet idle$
$inj_l = \overline{starve} \bullet (credits - CBs \geq 5) \bullet idle$
$CBs = (N2C \bullet (credits > CBs))? \; CBs + 1 : CBs$
C2N
$C2N = (N2C \bullet (credits > CBs))?1:0$
$mark\_cb = (credits == CBs)?1:0$
FBFC-C router

Fig. 11: FBFC routers.

4.1 FBFC routers

The left side of Fig. 11 shows a canonical wormhole router, which is composed of the input units, routing computation (RC) logic, VC allocator (VA), switch allocator (SA), crossbar and output units [20], [23]. Its pipeline includes: RC, VA, SA and switch traversal [20], [23]. The output unit tracks downstream VC status. The '*input_vc*' register records the allocated input VC of a downstream VC. The one-bit '*idle*' register indicates whether the downstream VC receives the tail flit of last packet. '*Credits*' records credit amounts. Lookahead routing [20] performs RC in parallel with VA. To be fair with VCT routers, wormhole routers try to hold SA grants for entire packets; it prioritizes VCs that got switch access previously [35].

FBFC mainly modifies output units. As shown in the upper-right side of Fig. 11, two one-bit registers, 'inj_s' and 'inj_l', are needed for the bi-modal length coherence traffic. They record whether a downstream VC is available for injecting (or dimension-changing) short and long packets. When packets will be injected (or change dimensions), VA checks the appropriate register according to packet lengths. Single-flit packets require at least 2 credits in a downstream VC. A 5-flit packet needs at least 6 credits. If the incoming '*starve*' signal is asserted to prevent starvation for some other router, both registers are reset to forbid injection. Fig. 11 shows the logic. This logic can be pre-calculated and is off the critical path.

The lower-right side of Fig. 11 shows the output unit of FBFC-C router. Another register, '*CBs*', records critical flit bubble counts. The logic of 'inj_s' and 'inj_l' is modified; packet injection is only allowed if the downstream VC has enough free normal slots. Specifically, '$credits - CBs$' is not less than the packet length. FBFC-C routers proactively transfer critical slots to prevent starvation. When there is an incoming '*N2C*' signal, the output unit checks whether there are free normal slots. If there are, '*CBs*' is increased by 1, and '*C2N*' is asserted to inform the neighboring router to change its critical slot into a normal one. The '*mark_cb*' signal is asserted when a flit will occupy a downstream critical slot; it informs the upstream router to mark the newly freed slot as critical. Similarly, this logic is off the critical path.

4.2 VCT routers

We discuss VCT routers for LBS and CBS. A typical VCT router [20], [22] is similar to the wormhole one shown in Fig. 11. The main difference is VC allocation: VCT re-allocates a VC only if it guarantees enough space for an entire packet. The advance of a packet returns one credit, which represents the release of a packet-size amount of buffers. We apply some optimizations to favor LBS and CBS. The SA grant holds for an entire packet. Since VA guarantees enough space for an entire packet, once a head flit moves out, that packet's remaining flits can advance without interruption; the packet's all occupied buffers will be freed in limited time. Thus, a credit is returned once a head flit moves out. The lookahead credit return allows the next packet to use this VC even if there is only one free slot, overlapping the transmission of an incoming packet and an outgoing packet. This optimization brings an injection benefit for CBS which we discuss in Sec. 7.2. LBS router's output unit is similar to that of FBFC-L router. The difference is that the LBS router only needs one 'inj' register since all packets are regarded as long packets. The '*credits*' register records free buffer slots in the unit of packets instead of flits. CBS router's output unit also has these differences. Since CBS only starves due to the critical bubble stall, there is no incoming '*starve*' signal.

5 METHODOLOGY

We modify Booksim [30] to implement FBFC-L and FBFC-C to compare with dateline, LBS and CBS. We use both synthetic traffic and real applications. Synthetic traffic uses one VN since each VN is independent. The traffic has randomly injected 1-flit packets and 5-flit packets. The baseline single-flit packet (SFP) ratio is 80%, which is similar to the overall SFP ratio of a MOSEI directory protocol. The warmup and measurement periods are 10,000 and 100,000 cycles.

Although FBFC works for high dimensional tori, we focus on 1D and 2D tori as they best match the physical layouts. The routing is DOR. Buffers are precious; most evaluation uses 10 slots at each port per VN. Bubble designs have one VC per VN. Dateline divides 10 slots into two VCs; 5 slots/VC covers credit round-trip delays [20]. Instead of injecting packets to VC_{0i} first (Fig. 2), then switching to VC_{1i} after the dateline [20], a balancing optimization is applied to favor dateline; injecting packets choose VCs according to whether they will cross dateline [51]. Packets use VC_{1i}s if they will cross dateline. Otherwise, they use VC_{0i}s. CBS and FBFC-C set one critical bubble for each ring; CBS marks 5 slots as a packet-size critical bubble, and FBFC-C marks 1 slot as a flit-size critical bubble. The starvation threshold values (STVs) in FBFC-L and LBS are 30 cycles. The STVs due to critical bubble stall in CBS and FBFC-C are 3 cycles.

VA and SA delays determine router frequencies [7], [48]. Dateline uses 2 VCs per VN, resulting in large

TABLE 1: The delay (in FO4) results. (bubble: 1 VC/VN, dateline: 2 VCs/VN.)

		Ring (#port=3)			Torus (#port=5)		
#VN		bubble	dateline	Inc.	bubble	dateline	Inc.
1	VA	8.4	12.2	45%	10.0	13.8	38%
	SA	6.9	11.7	69%	8.5	13.3	57%
2	SA	11.7	16.5	41%	13.3	18.1	36%
3	SA	14.5	19.3	33%	16.1	20.9	30%
4	SA	16.5	21.3	29%	18.1	22.9	27%

TABLE 2: Full system simulation configuration.

Topology	16 cores, 4×4 torus
L1 cache (D & I)	private, 4-way, 32KB each
L2 cache	private, 8-way, 512KB each

allocators and long critical paths. A technology-independent model [48] is used to calculate the delays, as shown in TABLE 1. Separable input-first allocators [20] with matrix arbiters [20] are used. VA is independent for each VN [7], making SA the critical path with multiple VNs. Dateline's SA delay with 4 VNs is ∼30% higher than bubble designs.

To measure full-system performance, we leverage FeS2 [45] for x86 simulation and BookSim for NoC simulation. FeS2 is a module for Simics [40]. We run PARSEC workloads [8] with 16 threads on a 16-core CMP. Since dateline's frequency can be different with bubble designs, we do not evaluate dateline for real workloads. The frequency of simple CMP core can be 2∼4 GHZ, while the frequency of NoC is limited by allocator speeds [27], [42]. We assume cores run 2× faster than the NoC. Each core is connected to private, inclusive L1 and L2 caches. Cache lines are 64 bytes; long packets are 5 flits with a 16-byte flit width. We use a MOESI directory protocol to maintain the coherence among L2 caches; it uses 4 VNs to avoid protocol-level deadlock. Each VN has 10 slots. Workloads use *simsmall* input sets. The task runtime is the performance metric. TABLE 2 gives the configuration.

6 Evaluation on 1D Tori (Rings)

6.1 Performance

Our evaluation for synthetic patterns [20] starts with an 8-node ring. As shown in Fig. 12, FBFC-L is similar to FBFC-C; although FBFC-C needs 1 less slot for injection, this benefit is minor since 10 slots are used. FBFC obviously outperforms LBS and CBS. Across all patterns, the average saturation throughput[3] gains of FBFC-C over LBS and CBS are 73.5% and 33.9%. LBS and CBS are limited by regarding short packets as long ones. The advance of a short packet in FBFC-C uses 1 slot, while it uses 5 slots in LBS and CBS. LBS is also limited by its high injection buffer requirement; CBS shows an average 29.6% gain over LBS.

In Fig. 12, dateline is superior to LBS and CBS. The results are reported in *cycles* and *flits/node/cycle*. These metrics do not consider router frequencies. According to TABLE 1, if all routers are optimized to maximum frequencies, dateline is ∼30% slower than bubble designs. To make a fair comparison, we leverage frequency independent metrics. The *seconds* is used for latency comparison. Due to its lower frequency, dateline's *cycle* in *seconds* is 30% longer than bubble design's *cycle* in *seconds*. Thus, dateline's zero-load latency in *seconds* is 30% higher. The *flits/node/second* is used for throughput comparison. Since dateline's *cycle* in *seconds* is longer than bubble design's *cycle* in *seconds*, dateline's throughput in *flits/node/second* drops. For example, its throughput in *flits/node/second* for uniform random is 7.5% lower than CBS.

Dateline divides buffers into two VCs. Shallow VCs make packets span more nodes, which increases chained blocking effect [55]; this brings a packet forwarding limitation for dateline. Yet, dateline is superior to FBFC for injection and dimension-changing: long packets can inject or change dimensions with 1 free slot. We introduce a metric: injection/dimension-changing (IDC) count. The IDC count of a packet includes the number of times a packet is injected plus the number of times it changes dimensions.

The trends between dateline and FBFC depend on hop counts and IDC counts of the traffic. FBFC-C outperforms dateline for all patterns. A ring has no dimension changing; all patterns' IDC counts are no more than 1, hiding dateline's merit. The largest gains are 29.2% and 18.8% for transpose and tornado. Transpose's IDC and hop counts are 0.75 and 2.25. Tornado's IDC and hop counts are 1 and 4. They reveal dateline's limitation on packet forwarding.

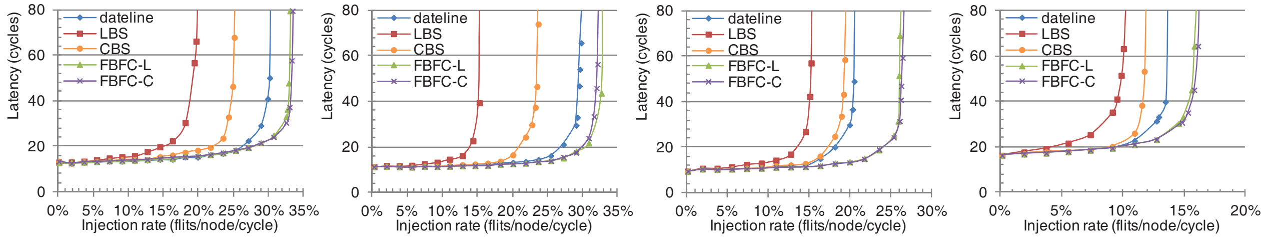
6.2 Buffer Utilization

To delve into performance trends, the average buffer utilization of all VCs is shown in Fig. 13. The maximum and minimum rates are given by error bars. Buffers can support high throughput; higher utilization generally means better performance. For LBS and CBS, the average rates are 13.0% and 19.2% in saturation. They inefficiently use buffers. LBS requires more free buffers for injection; its utilization is lower than CBS. Dateline's minimum rate is always 0; one VC is never used. For example, VC_{00} in Fig. 2 is not used. This scenario combined with chained blocking limits its buffer utilization. Dateline's average and maximum rates at saturation are 23.2% and 71.8%, while these rates are 39.5% and 89.8% for FBFC-C.

6.3 Latency of Short and Long Packets

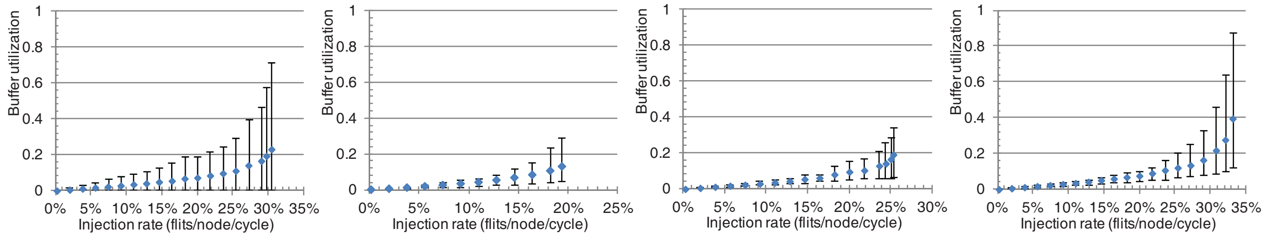
FBFC's injection of long packets requires more buffers than short ones. Fig. 14 shows latency compositions. 'InjVC' and 'NI' are delays in injection VCs and network interfaces. 'Network' is all other delays. Long and short packets are treated the same in LBS and CBS; they show similar delays at injection VCs and inside the ring. Long packets have 4 more flits; they spend ∼4 more cycles in network interfaces. With low-to-medium injection rates (≤20%) in FBFC-C, the delays at injection VCs for long and short packets are

[3] The saturation point is measured as the injection rate at which the average latency is 3 times the zero load latency.



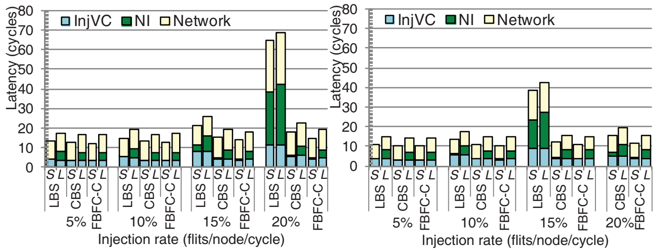
(a) Uniform random. (b) Bit rotation. (c) Transpose. (d) Tornado.

Fig. 12: The performance for an 8-node ring.



(a) Dateline. (b) LBS. (c) CBS. (d) FBFC-C.

Fig. 13: The buffer utilization with uniform random traffic until network saturation.



(a) Uniform random. (b) Bit rotation.

Fig. 14: Latencies of short ('*S*') and long ('*L*') packets.

similar. With higher loads, the difference increases. Even when saturated, they are only 3.4 and 3.2 cycles for the two patterns. FBFC-C does not sacrifice long packets. FBFC-C's acceleration of short packets helps long packets since short and long packets are randomly injected. Indeed, compared with FBFC-C, LBS and CBS sacrifice short packets. For example, with a 20% injection rate for uniform random, short packets spend 11.7 and 5.4 cycles in injection VCs for LBS and CBS, and 4.3 cycles for FBFC-C.

7 Evaluation on 2D Tori

Sec. 6 analyzes the performance for 1D tori with buffer utilization and latency composition. This section thoroughly analyzes the performance for 2D tori with several configurations for further insights.

7.1 Performance for a 4×4 Torus

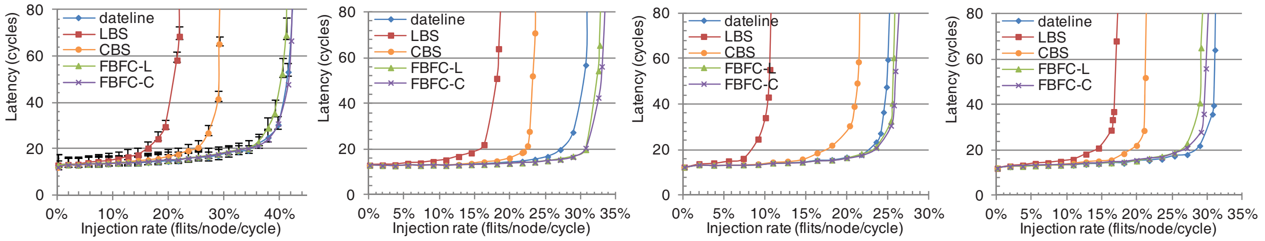
Fig. 15 shows the performance for a 4×4 torus with the baseline configuration. The error bars in Fig. 15a are average latencies of long and short packets. The average gains of FBFC-C over LBS and CBS are 92.8% and 34.2%. FBFC's high buffer utilization brings these gains. CBS shows an average 45.7% gain over LBS, and the highest one is 100% for transpose. Most transpose traffic is between the same row and column; many packets change dimensions at the same router requiring the same port. CBS's low dimension-changing buffer requirement yields high performance.

Compared with the ring (Fig. 12), the trends between FBFC-C and dateline for a 2D torus change. Dateline performs similarly to FBFC-C for uniform random and transpose. A 2D torus has a dimension changing step. The IDC counts of these patterns are both 1.5, and their hop counts are 3. As a result, injection and dimension-changing factor significantly into performance. Dateline's low injection and dimension-changing buffer requirement brings gains. Dateline outperforms FBFC-C by 5.7% for hotspot. This pattern sends packets from different rows to the same column of 4 hotspot nodes, exacerbating FBFC-C's injection limitation. FBFC-C outperforms dateline by 6.4% for bit rotation. Packets of bit rotation change dimensions by requiring different ports; the light congestion mitigates FBFC-C's limitation. These results do not consider frequencies. If routers are optimized to maximum frequencies, dateline's performance will drop.

As shown in Fig. 15a, the delay difference between long and short packets is almost constant for LBS and CBS; long packets have ∼4 cycles more delay. Dateline's difference is 6.2 cycles in saturation. FBFC-L's difference is 9.8 cycles. It is ∼3 cycles more than the ring; a 2D torus has one dimension-changing step. We measure the behavior after saturation by increasing the load to 1.0 flits/node/cycle. All designs maintain performance after saturation. DOR smoothly delivers injected packets. Adaptive routing may drop performance after saturation because escape paths drain packets at lower rates than injection rates [11].

7.2 Sensitivity to SFP Ratios

As discussed in Sec. 4.2, VCT routers' lookahead credit return overlaps packet incoming and outgoing; a packet can move to a VC with 1 free slot. It brings an injection or dimension-changing benefit for CBS over FBFC. CBS's packet injection begins with 1 free slot. FBFC-C's long packet injection needs 5 normal slots.



(a) Uniform random. (b) Bit rotation. (c) Transpose. (d) Hotspot.

Fig. 15: The performance for a 4×4 torus.

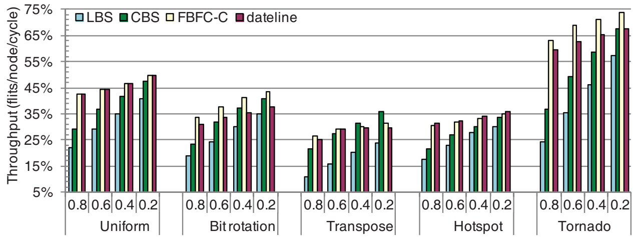


Fig. 16: The performance of several SFP ratios.

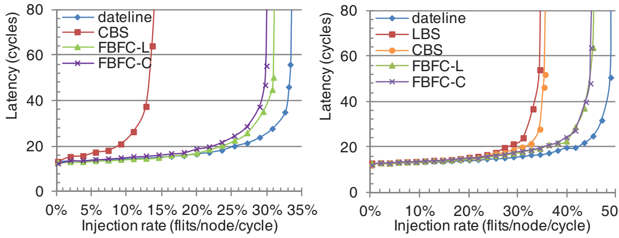
Thus, SFP ratios affect trends between CBS and FBFC. Fig. 16 conducts sensitivity studies on SFP ratios.

The performance of LBS and CBS increases linearly with reduced ratios; more long packets proportionally improve their buffer utilization. FBFC-C and dateline perform slightly better with lower ratios. They try to hold SA grants for entire packets; long packets reduce SA contention. Although FBFC-C's gains over CBS reduce with lower ratios, FBFC-C is always superior for most traffic patterns. FBFC-C performs 10.1% better than CBS for tornado with a 0.2 ratio. Tornado sends traffic from node (i, j) to $((i+1)\%4, (j+1)\%4)$. Each link only delivers traffic for one node pair; there is no congestion. Transpose is different; CBS outperforms FBFC-C with 0.4 and 0.2 ratios. This pattern congests turn ports, emphasizing CBS's injection benefit.

We also experiment with a 64-bit flit width; the short and long packet have 1 and 9 flits. With longer packets, the negative effect of regarding short packets as long ones in LBS and CBS becomes more significant. Meanwhile, FBFC-C needs more slots for long packet injection. These factors result in similar trends for a 64-bit configuration as those in Fig. 16.

7.3 Sensitivity to Buffer Size

We perform sensitivity studies on buffer sizes. In Fig. 17a, CBS, FBFC-C and FBFC-L use minimum buffers that ensure correctness. They use 5, 5 and 6 slots. Dateline uses two 3-slot VCs. In Fig. 17b, bubble designs use a 15-slot VC, and dateline uses two 8-slot VCs. Although FBFC-C's injection limitation has higher impact with fewer buffers, dateline only performs 7.6% better than FBFC-C with 5 slots/VC. Dateline's shallow VCs (3 slots) cannot cover the credit round-trip delay (5 cycles), making link-level flow control the bottleneck [20]. FBFC-C's gain over CBS increases with fewer buffers. The gains are 26.6% with 15 slots/VC, 41.4% with 10 slots/VC, and 121.8% with 5 slots/VC. Comparing Figs. 15a and 17a, CBS with 10 slots/VC is similar to FBFC-C with 5 slots/VC. With half as many buffers, FBFC-C is comparable to CBS. LBS with 15 slots/VC (Fig. 17b) only has a 5.2% gain over FBFC-C with 5 slots/VC.



(a) 5 (6) slots. (b) 15 (16) slots.

Fig. 17: The performance of uniform random with other buffer sizes. (FBFC-L and dateline use 6 slots/VN in Fig. 17a, and dateline uses 16 slots/VN in Fig. 17b.)

LBS almost matches CBS with 15 slots/VC. More buffers mitigate LBS's high injection buffer limitation. Additional results show that with abundant buffers, bubble designs performs similarly; there is little difference among them as many free buffers are available anyway. The convergence points depend on the traffic. For example, due to congested ports in transpose, at least 30 slots/VC are needed for LBS to match CBS. For uniform random, 50 slots/VC are needed for CBS to match FBFC. Many buffers are required for convergence, which makes high buffer utilization designs, such as FBFC, winners in reasonable configurations.

7.4 Scalability for an 8×8 Torus

Fig. 18 explores scalability to an 8×8 torus. Two buffer sizes are evaluated: one assigns 10 slots and the other assigns 5 or 6 slots (the same as Fig. 17a). As the network scales, traffic's hop count increases, exacerbating dateline's limitation for packet forwarding. Meanwhile, their IDC counts are similar to a 4×4 torus. Thus, FBFC-C's injection bottleneck is masked in larger networks. With 10 slots, FBFC-C outperforms dateline for all patterns. The largest gain is 26.5% for tornado, whose average hop count is 7. With 5 slots, FBFC-C outperforms dateline for 4 patterns. Larger networks place greater pressure on buffers, worsening inefficient buffer utilization of LBS and CBS. With 10 slots, FBFC-C has a 82.5% gain over CBS for uniform random, while it is 41.4% in a 4×4 torus. With 10 slots, the average gains of FBFC-C over LBS and CBS are 107.2% and 40.1% for the 8 patterns. With 5 slots, the average gain of FBFC-C over CBS is 78.7%.

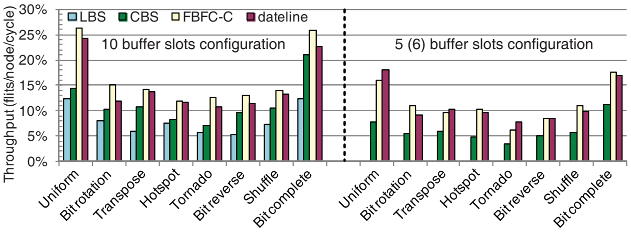
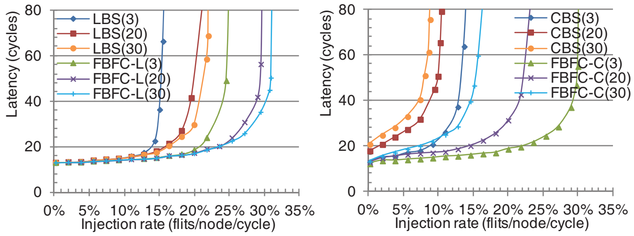


Fig. 18: The performance for an 8×8 torus.



(a) Localized schemes.

(b) Critical schemes.

Fig. 19: The performance with several STVs.

7.5 Effect of Starvation

LBS [10] has limited discussion of starvation; CBS [11] relies on adaptive routing and therefore does not address starvation. We analyze starvation in a 4×4 torus. Reducing buffers makes starvation more likely. We use the same buffer size as in Fig. 17a (LBS uses 10 slots) with uniform random. Larger networks or other patterns are similar.

Starvation in LBS and FBFC-L is essentially the same. Fig. 19a shows their performance with three starvation threshold values (STVs). They perform poorly with the 3-cycle STV. A small STV causes a router to frequently assert the '*starve*' signal (Fig. 9) to prohibit other nodes' injection, which negatively affects overall performance. We also evaluate the saturation throughput with several STVs ranging from 3 to 50 cycles. LBS and FBFC-L perform better with larger STVs until 30 cycles, and then remain almost constant. We set the STVs in LBS and FBFC-L to 30 cycles.

CBS and FBFC-C can starve due to the critical bubble stall. Also, FBFC-C has the same starvation as FBFC-L. FBFC-C uses two STVs; one for each type of starvation. We fix the STV in FBFC-C for the same starvation as FBFC-L to 30 cycles, and analyze the other type of starvation. As shown in Fig. 19b, the smaller the STV, the higher the performance. The proactive transfer of critical bubble does not prohibit injection; even with many false detections, there is no negative effect. In contrast, the performance drops if packets cannot be injected for a long time. With a 20-cycle STV, if one node suffers starvation, it will move the critical bubble after 20 cycles. Then it starts injecting. This lazy reaction not only increases the zero-load latency, but also limits the saturation throughput. Since the proactive transfer of critical bubble needs 2 cycles, we set the critical bubble stall STV to 3 cycles.

7.6 Real Application Performance

Fig. 20 shows the speedups relative to LBS for PARSEC workloads. FBFC supports higher network throughput, but system gains depend on workloads. FBFC benefits applications with heavy loads and bursty traffic. For `blackscholes`, `fluidanimate` and `swaptions`, different designs perform similarly. Their computation phases have few barriers and their working sets fit into caches, creating light network loads. They are unaffected by techniques improving network throughput, such as FBFC.

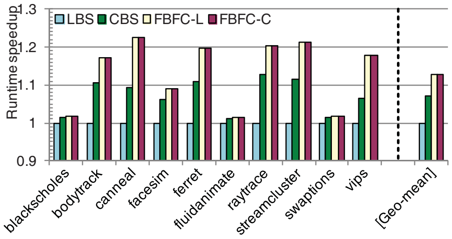


Fig. 20: System speedup for PARSEC benchmarks.

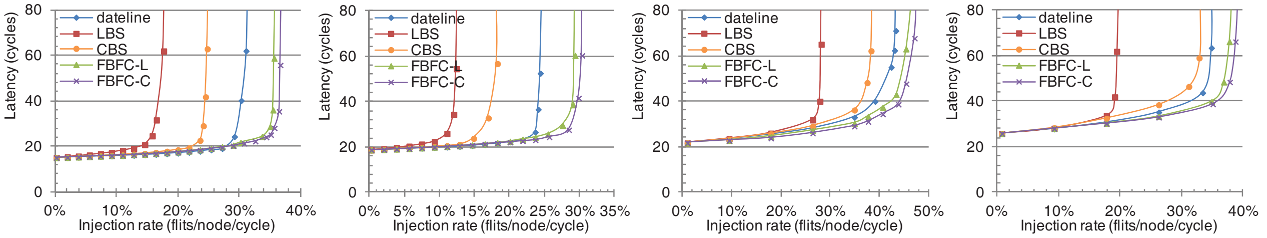
Network optimizations affect the other 7 applications. Both CBS and FBFC see gains. The largest speedup of FBFC over LBS is 22.7% for `canneal`. Two factors bring the gains. First, these applications create bursty traffic and heavy loads. Second, the two VNs with hybrid-size packets have relatively high loads. Across the 7 applications, VN0 and VN2 averagely have 70.8% loads, including read request, write-back request, read response, write-back ACK and invalidation ACK. These relatively congested VNs emphasize FBFC's merit in delivering variable-size packets. Across all workloads, FBFC and CBS have average speedups of 13.0% and 7.5% over LBS.

Compared with synthetic traffic, the real application gains are lower. It is due to the configured CMP places light pressure on network buffers. Other designs, such as concentration [20] or configuring fewer buffers, can increase the pressure. FBFC shows larger gains in these scenarios. For example, we also evaluate performance with 5 slots/VN. FBFC-C achieves an average 9.8% and maximum 20.2% speedup over CBS. Also, the application runtime of FBFC-C with 5 slots/VN is similar to that of LBS with 10 slots/VN.

7.7 Large-scale Systems and Message Passing

The advance of CMOS technology will integrate hundreds or thousands of cores in a chip [9]. Some current many-core chips, including 60-core Xeon Phi [28], use the shared memory paradigm. Yet, cache coherence faces scalability challenges with more cores. Message passing is an alternative paradigm. For example, the 64-core TILE64 uses a message passing paradigm [57]. It remains an open problem to design an appropriate paradigm for large-scale systems [41]. We evaluate FBFC for both paradigms. As a case study, we use a 256-core platform organized as a 16×16 torus.

In large-scale systems, assuming uniform communication among all nodes is not reasonable [47]. Workload consolidation [38] and application mapping optimizations [14] increase traffic locality; we leverage an exponential locality traffic model [47], which exponentially distributes packet hop counts. For example, with



(a) Shared Memory, $\lambda = 0.5$. (b) Shared Memory, $\lambda = 0.3$. (c) Message Passing, $\lambda = 0.5$. (d) Message Passing, $\lambda = 0.3$.

Fig. 21: The performance for a 16×16 torus with exponential locality traffic.

distribution parameter $\lambda = 0.5$, the traffic average hop is $1/\lambda = 2$, and 95% traffic is within 6 hops, and 99% traffic is within 10 hops. We evaluate two distribution parameters, $\lambda = 0.5$ and $\lambda = 0.3$.

The packet length distribution of shared memory traffic is the same as the baseline configuration in Sec. 5; 80% packets have one flit, and the others have five flits. All designs use 10 slots per VN. We assume that the packet length distribution of message passing traffic is similar to BlueGene/L; packet sizes ranges from 32 to 256 bytes [4]. With a 16-byte flit width, packet lengths are uniformly distributed between 2 to 16 flits. All designs use 32 slots per VN.

Fig. 21 shows the performance. The overall trends among different designs are similar to an 8×8 torus (Fig. 18). LBS and CBS are limited by inefficient buffer utilization. LBS is further limited by its high injection buffer requirement. Dateline's limitation for packet forwarding restricts its performance. FBFC efficiently utilizes buffers, and yields the best performance.

The performance gaps between FBFC and other bubble designs depend on distribution parameter λ. The smaller the λ, the larger the average hops. Larger average hops emphasize efficient buffer utilization; thus, FBFC gets more performance gains. For shared memory traffic, FBFC-C performs 47.7% better than CBS with $\lambda = 0.5$, and this gain is 66.5% with $\lambda = 0.3$. Message passing traffic shows similar trends. FBFC-C performs 18.7% better than CBS with $\lambda = 0.5$, and the gain increases to 23.8% with $\lambda = 0.3$.

FBFC's gains for message passing traffic is lower than shared memory traffic. With $\lambda = 0.5$, FBFC-C performs 105.2% better than LBS for shared memory traffic, while it is 68.8% for message passing traffic. The packet length distributions are different. For shared memory traffic, the average packet length is 1.8 flits, and LBS regards each packet as a 5-flit packet. The average packet length of message passing traffic is 9 flits, and LBS regards each packet as a 16-flit packet. The maximum packet length of shared memory traffic is ~2.8 times of the average length, while it is 2 for message passing traffic. This brings the drop of FBFC-C's gain for message passing traffic.

8 Overheads: Power and Area

This section conducts power and area analysis of our designs. We also compare tori with meshes.

8.1 Methodology

We modify a NoC power and area model [5], which is integrated in Booksim [30]. We calculate both dynamic and static power. The dynamic power is formulated as $P = \alpha C V_{dd}^2 f$, with α the switching activity, C the capacitance, V_{dd} the supply voltage, and f the frequency. The switching activities of NoC components are obtained from Booksim. The capacitance, including gate and wire capacitances, is estimated based on canonical modeling of component structures [5].

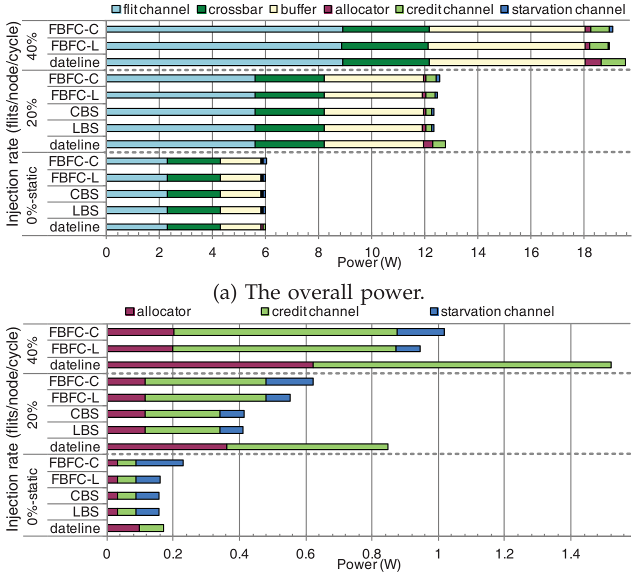
The static power is calculated as $P = I_{leak} V_{dd}$, with I_{leak} the leakage current. The leakage current is estimated by taking account of both component structures and input states [5]. For example, the inserted repeater, composed of a pair of pMOS and nMOS devices, determines the wire leakage current. Since pMOS devices leak with high input and nMOS devices leak with low input, the repeater leaks in both high and low input states. The wire leakage current is estimated as the average leakage current of a pMOS and an nMOS device [5].

The router area is estimated based on detailed floorplans [5]. The wires are routed above other logic; the channel area only includes the repeater and flip-flop areas. The device and wire parameters are obtained from ITRS report [29] for a 32nm process, at 0.9 V and 70°C. All designs are assumed to operate at 1 GHz based on a conservative assumption.

We assume a 128-bit flit width. The channel length is 1.5 mm; an 8×8 torus occupies ~150 mm^2. Repeaters are inserted to make signals traverse a channel in 1 cycle. The number and size of repeaters are chosen to minimize energy. VCs use SRAM buffers. We assume four VNs to avoid protocol-level deadlock. Allocators use the separable input-first structure [20]. We leverage the segmented crossbar [56] to allow a compact layout and reduce power dissipation. Packets are assumed to carry random payloads; two sequential flits cause half of the channel wires to switch.

8.2 Power Efficiency

Fig. 22 shows the power of an 8×8 torus with uniform random traffic. The 0% injection rate bars are static power. All designs use 10 slots per VN. We divide the NoC into the flit channel, crossbar, buffer, allocator, credit channel, and starvation channel. LBS and CBS cannot support more than 35% injection rate. All designs consume similar power with the same load. The



(a) The overall power.

(b) The allocator and sideband channel power.

Fig. 22: The power of an 8×8 torus.

flit channel, crossbar, and buffer together consume more than 93% of overall power. These components are similar for all designs. The allocator consists of combinational logic, and it induces low power.

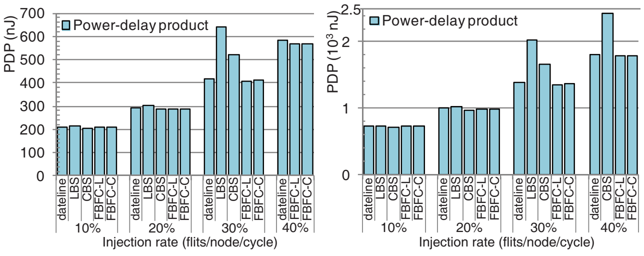
Bubble designs' credit channel has 3 wires; two bits encode 4 VCs and one valid bit. Dateline's credit channel has 4 wires; three bits encode 8 VCs and one valid bit. The starvation channel of LBS and FBFC-L has 6 wires. Three bits identify one node among 8 nodes of a ring. Two bits encode four VNs, and one valid bit. CBS's starvation channel uses 6 wires. '*N2C*' and '*C2N*' signals need two bits, and both signals use two bits to encode 4 VNs. FBFC-C handles two types of starvation; its starvation channel has 12 wires. These credit channels and starvation channels are narrower than flit channels; they induce low power.

To clarify differences, Fig. 22b shows the allocator and sideband channel power. Starvation channels are not needed for dateline. Yet, their activities are low. For example, FBFC-L's starvation channels keep idle until 34% injection rate. Dateline uses large allocators. Also, dateline's credit channel has one more wire than bubble designs. Dateline consumes higher power than bubble designs with 20% and 40% injection rates.

Although bubble designs' credit channels are narrower than starvation channels, two reasons cause credit channels to consume higher dynamic power. First, starvation channels are not needed for injection/ejection ports. An 8×8 torus has 384 credit channels and 256 starvation channels. Second, credit channels' activities are higher. VCT routers return one credit for each packet, and wormhole routers return one credit for each flit. Credit channels of LBS and CBS consume lower dynamic power than FBFC.

Fig. 23a evaluates the power-delay product (PDP). Compared with LBS and CBS, FBFC reduces latencies for heavy loads, improving PDP. With 20% and 30% injection rates, FBFC-L's PDP is 5.9% and 56.9% lower than LBS. FBFC-L's PDP is 27.6% lower than CBS at a 30% injection rate. Dateline's power and latency are similar to FBFC. Its PDP is similar to FBFC. We also evaluate other traffic patterns. The trends are similar to uniform random. All designs consume similar power, and FBFC's latency optimization reduces the PDP. For example, with transpose, FBFC-L's PDP is 34.2% lower than LBS at a 15% injection rate, and its PDP is 28.9% lower than CBS at a 20% injection rate.

To show the impact of network scaling on power efficiency, Fig. 23b gives the PDP on a 16×16 torus. The exponential locality traffic with $\lambda = 0.3$ in Sec. 7.7 is used; $\lambda = 0.5$ has a similar trend. Since FBFC optimizes latencies, it still offers power efficiency gains in larger NoCs. FBFC-L's PDP is 32.7% and 18.2% lower than LBS and CBS for a 30% injection rate, and its PDP is 26.6% lower than CBS for a 40% injection rate.



(a) 8×8 torus, uniform random.

(b) 16×16 torus, $\lambda = 0.3$.

Fig. 23: The PDP results.

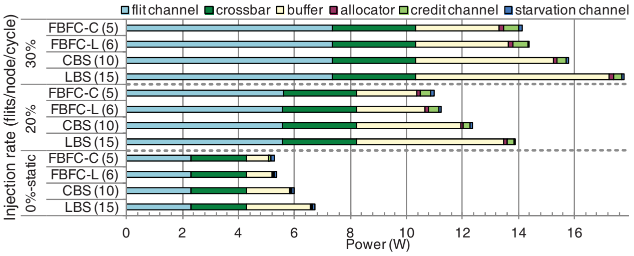


Fig. 24: The power when bubble designs support similar saturation throughput in an 8×8 torus.

As shown in Sec. 7.3, with half as many buffers, FBFC performs the same as CBS in an 8×8 torus. With one third of buffers, FBFC performs similarly to LBS. Fig. 24 gives the power of bubble designs when they performs similarly. FBFC-C and FBFC-L use 5 and 6 slots per VN. CBS and LBS use 10 and 15 slots per VN. FBFC-C's buffer static power is 49.8% lower than CBS, which results in FBFC-C's network static power to be 11.5% lower than CBS. FBFC-C's overall network static power is 21.3% lower than LBS. High loads increase dynamic power. Yet, even with a 20% injection rate, FBFC-C's power is still 20.4% and 10.4% lower than LBS and CBS. Since now all designs perform similarly, FBFC's PDP is lower as well.

In summary, with the same buffer size, all designs consume similar power. FBFC's starvation channels induce negligible power. Since FBFC significantly outperforms existing bubble designs, it achieves much lower PDP, in both 8×8 and 16×16 tori. When bubble designs perform similarly with different buffer sizes, FBFC consumes lower power and offers PDP gains.

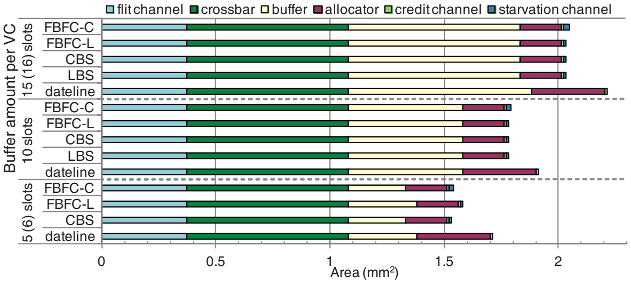


Fig. 25: The area of an 8×8 torus.

8.3 Area

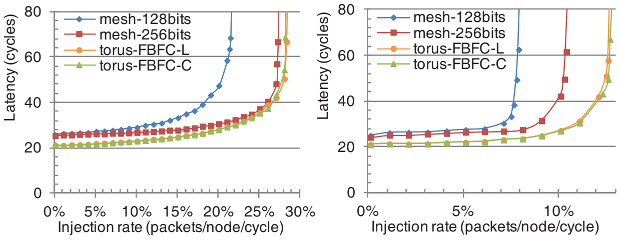
Fig. 25 shows the area results. The areas of flit channel and crossbar are similar for all designs. With the same buffer amount, the area differences among designs are mainly due to the allocator, credit channel and starvation channel. Dateline's allocator is ∼2 times larger than bubble designs; this causes dateline to consume most area. With 10 slots per VN, dateline's overall area is 7.4% higher than FBFC-L. When bubble designs perform similarly, FBFC's area benefit is more significant. CBS's network area with 10 slots per VN is 15.6% higher than FBFC-C with 5 slots per VN. The overall area of LBS with 15 slots per VN is 31.9% higher than FBFC-C with 5 slots per VN.

8.4 Comparison with Mesh

We compare tori with meshes. The routing is DOR. The torus uses FBFC to avoid deadlock. Based on a floorplan model [5], the mesh channel length is 33.3% shorter than the torus one; it is 1.0 mm. Inserted repeaters make the channel delay be 1 cycle. Two meshes are evaluated. One uses a 128-bit channel width, which is the same as the torus. Yet, its bisection bandwidth is half of the torus. The other mesh uses a 256-bit width to achieve the same bisection bandwidth as the torus. All networks have the same packet size distribution. The 128-bit width mesh uses 10 slots per VN. Its traffic has 80% 5-flit packets and 20% 1-flit packets. The 256-bit width mesh uses 5 slots per VN. Its traffic has 80% 3-flit packets and 20% 1-flit packets.

Fig. 26 gives the performance. Since flit sizes of evaluated networks are different, the 'injection rate' is measured in 'packets/node/cycle'. Torus' wraparound channels reduce hop counts. For both patterns, the torus shows ∼20% lower zero-load latencies than the mesh. With half of the bisection bandwidth, *mesh-128bits*' saturation throughput is 24.2% and 37.0% lower than the torus for uniform random and transpose. With the same bisection bandwidth, *mesh-256bits*' saturation throughput is similar to the torus for uniform random. The transpose congests mesh's center portion [43]; *mesh-256bits*' saturation throughput is still 17.3% lower than the torus.

Fig. 27 shows the power and area. With the same channel width, *mesh-128bits*' static power is 30.5% lower than the torus; it is due to the optimization of buffers and flit channels. An 8×8 mesh has 288 ports with buffers, while an 8×8 torus has 320 ports. Two factors brings power reduction for mesh channels. First, an 8×8 mesh has 352 flit channels, while an 8×8 torus has 384 flit channels. Second, mesh channels are 33.3% shorter than torus ones. Thus, even 256-bit mesh channels consume less static power than torus ones. Yet, the 256-bit channel width quadratically increases crossbar power. The *mesh-256bits*' overall static power is 77.7% higher than the torus.



(a) Uniform random. (b) Transpose.

Fig. 26: The performance comparison.

With high loads, mesh's center congestion increases dynamic power; *mesh-128bits*' benefit over FBFC-L decreases. With 10% and 20% injection rates, its power is 15.7% and 10.9% less than FBFC-L. With a 20% injection rate, *mesh-256bits*' channel power is higher than FBFC-L. The PDP reflects power efficiency. With a 10% injection rate, FBFC-L's PDP is 6.4% and 64.9% less than *mesh-128bits* and *mesh-256bits*. Its power efficiency increases with high loads. With a 20% injection rate, FBFC-L's PDP is 33.7% less than *mesh-128bits*.

The network area of *mesh-128bits* is 17.4% less than FBFC-L. The *mesh-128bits* uses fewer buffers and channels. Due to the large crossbar of *mesh-256bits*, its network area is 107.2% higher than FBFC-L.

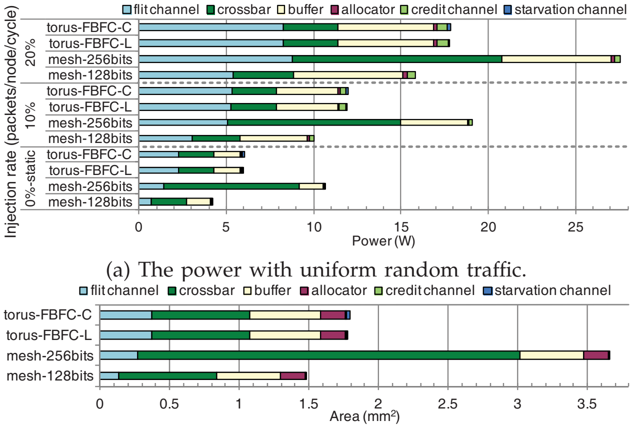
We also compare a 16×16 mesh with a 16×16 torus. The static power and area benefits of *mesh-128bits* decrease with larger networks. The mesh's benefit of using fewer flit channels and ports decreases. In a 64-node network, the torus has 9.1% more flit channels and 11.1% more ports than the mesh. These numbers are 4.3% (1536 vs. 1472) and 5.3% (1280 vs. 1216) in a 256-node network. *mesh-128bits*' static power in a 16×16 mesh is 24.3% less than that of a 16×16 torus, and its network area is 13.4% less. The PDP is more favor to FBFC. FBFC-L's PDP is 24.5% and 57.9% less than *mesh-128bits* and *mesh-256bits* at a 15% injection rate for the exponential locality traffic with $\lambda = 0.3$.

In summary, although with the same channel width, the mesh consumes less power and area than the torus, its performance is poor due to limited bisection bandwidth. With FBFC applied, the torus is more power efficient than the mesh for the same channel width. With the same bisection bandwidth, the mesh consumes much higher power than the torus. Applying FBFC on the torus is well scalable.

9 Discussions and Related Work

9.1 Discussions

FBFC efficiently addresses the limitations of existing designs. It is an important extension to packet-size



(a) The power with uniform random traffic.

(b) The area.

Fig. 27: The power and area comparison.

bubble theory. The insight of '*leaving one slot empty*' enables other design choices. For example, combining with dynamic packet fragmentation [24], a packet can inject with one free normal slot. When one flit's injection will consume the critical slot, this packet stops injecting and changes the waiting flit into a head flit. This design allows VC depths to be shallower than the largest packet. Based on a similar insight, an efficient deadlock avoidance design is proposed for wormhole networks [12]. Its basic idea is coloring buffer slots into white, gray or black to convey global buffer status locally. This is different from our design. FBFC uses local buffer status with hybrid flow control which combines VCT and wormhole. Also, we mainly focus on improving the performance for coherence traffic, which consists of both long and short packets.

Similar to dateline and packet-size bubble designs, FBFC is a general flow control. It can be adopted in various topologies as far as there is a ring in the network. For example, Immunet [50] achieves fault-tolerant by constructing a ring in arbitrary topologies for connectivity. The ring uses LBS to avoid deadlock. Instead, FBFC can be used; it will support higher performance with fewer buffers. MRR [2] leverages the ring of rotary router [1] to support multicast. The ring uses LBS. FBFC can be used as well. By configuring a ring in the network, FBFC can support both the unicast and multicast for streaming applications [3]. Also, FBFC can support fully adaptive routing [11], [49]. Bubble designs use one VC; there is head-of-line blocking. FBFC can combine with dynamically allocated multi-queue [46], [54] to mitigate this blocking and further improve buffer utilization.

9.2 Related Work

The Ivy Bridge [21], Xeon Phi [28] and Cell [32] use ring networks. The ring is much simpler than the 2D or high dimensional torus, and it is easy to avoid deadlock through end-to-end backpressure or centralized control schemes [32], [34]. The ring networks of these chips [21], [28], [32] guarantee injected packets cannot be blocked, which is similar to bufferless networks. Bufferless designs generally do not consider deadlock as packets are always movable [24]. Our research is different. We focus on efficient deadlock avoidance designs for buffered networks, which supports higher throughput than bufferless networks. Except for dateline [20], LBS [10] and CBS [11], there are other designs. Priority arbitration is used for single-flit packets with single-cycle routers [34]. Prevention flow control combines priority arbitration with prevention slot cycling [31]; it has deadlock with variable-size packets. Turn model [26] only allows non-minimal routing in tori. A design allows deadlock formation, and then applies a recovery mechanism [52].

FBFC observes that most coherence packets are short. Several designs use this observation, including packet chaining [42], the NoX router [27] and whole packet forwarding [39]. Configuring more VNs, such as 7 VNs in Alpha 21364 [44], can eliminate co-existence of variable-size packets. Yet, additional VNs have overheads; using minimum VNs is preferable. DASH [37], Origin 2000 [36], and Piranha [6] all apply protocol-level deadlock recovery to eliminate 1 VN; they utilize 2 VNs to implement 3-hop directory protocols. These VNs all carry variable-size packets.

10 Conclusion

Optimizing NoCs for coherence traffic improves the efficiency of many-core coherence protocols. We observe two properties of cache coherence traffic: short packets dominate the traffic, and short and long packets co-exist in NoC. Then we propose an efficient deadlock avoidance theory, FBFC, for torus networks. It maintains one free *flit-size* buffer slot to avoid deadlock. Only one VC is needed, which achieves high frequency. Also, FBFC does not treat short packets as long ones; this yields high buffer utilization. With the same buffer size, FBFC significantly outperforms LBS and CBS, and is more power efficient as well. When bubble designs perform similarly, FBFC consumes much less power and area. With FBFC applied, the torus is more power efficient than the mesh.

Acknowledgments

We thank the anonymous reviewers for their thorough and constructive suggestions. This work is supported by 863 Program of China (2012AA010905), NSFC (61272144, 61303065, 61070037), the UofT, NSERC of Canada, Research Project of NUDT (JC13-06-02), Ph.D. Programs Foundation of Ministry of Education of China (20134307120028).

References

[1] P. Abad et al. Rotary router: an efficient architecture for CMP interconnection networks. In *ISCA*, 2007.

[2] P. Abad et al. MRR: Enabling fully adaptive multicast routing for CMP interconnection networks. In *HPCA*, 2009.

[3] A. Abdel-Gawad et al. TransCom: Transforming stream communication for load balance and efficiency in networks-on-chip. In *MICRO*, 2011.

[4] N. Adiga *et al.* Blue Gene/L torus interconnection network. *IBM J. Res. Dev.*, 49(2.3):265 –276, March 2005.

[5] J. Balfour and W. Dally. Design tradeoffs for tiled CMP on-chip networks. In *ICS*, 2006.

[6] L. Barroso et al. Piranha: a scalable architecture based on single-chip multiprocessing. In *ISCA*, 2000.

[7] D. Becker and W. Dally. Allocator implementations for network-on-chip routers. In *SC*, 2009.

[8] C. Bienia et al. The PARSEC benchmark suite: characterization and architectural implications. In *PACT*, 2008.

[9] S. Borkar. Thousand core chips: a technology perspective. In *DAC*, 2007.

[10] C. Carrion et al. A flow control mechanism to avoid message deadlock in k-ary n-cube networks. In *HiPC*, 1997.

[11] L. Chen et al. Critical bubble scheme: An efficient implementation of globally aware network flow control. In *IPDPS*, 2011.

[12] L. Chen et al. Worm-bubble flow control. In *HPCA*, 2013.

[13] L. Chen and T. Pinkston. Personal communication, 2012.

[14] C. Chou and R. Marculescu. Run-time task allocation considering user behavior in embedded multiprocessor networks-on-chip. *IEEE TCAD*, 29(1):78 –91, Jan. 2010.

[15] P. Conway and B. Hughes. The AMD Opteron northbridge architecture. *Micro, IEEE*, 27(2):10 –21, Mar.-Apr. 2007.

[16] W. Dally. Virtual-channel flow control. *IEEE Trans. Parallel Distrib. Syst.*, 3(2):194 –205, Mar. 1992.

[17] W. Dally et al. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. Comput.*, May 1987.

[18] W. Dally and C. Seitz. The Torus routing chip. *Distributed Computing*, 1:187–196, 1986.

[19] W. Dally and B. Towles. Route packets, not wires: on-chip interconnection networks. In *DAC*, 2001.

[20] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, San Francisco, USA, 2003.

[21] S. Damaraju et al. A 22nm IA Multi-CPU and GPU system-on-chip. In *ISSCC*, 2012.

[22] J. Duato et al. A comparison of router architectures for virtual cut-through and wormhole switching in a NOW environment. *J. Parallel Distrib. Comput.*, 61(2):224–253, Feb. 2001.

[23] N. Enright Jerger and L. Peh. *On-Chip Networks*. Morgan and Claypool Publishers, 2009.

[24] C. Fallin et al. CHIPPER: A low-complexity bufferless deflection router. In *HPCA*, 2011.

[25] K. Gharachorloo et al. Architecture and design of AlphaServer GS320. In *ASPLOS*, 2000.

[26] C. Glass and L. Ni. The turn model for adaptive routing. In *ISCA*, 1992.

[27] M. Hayenga and M. Lipasti. The NoX router. In *MICRO*, 2011.

[28] Intel. Intel Xeon Phi Coprocessor: Datasheet. https://www-ssl.intel.com/content/www/us/en/processors/xeon/xeon-phi-coprocessor-datasheet.html, 2012.

[29] ITRS. International Technology Roadmap for Semiconductors, 2010 edition. http://www.itrs.net, 2010.

[30] N. Jiang et al. A detailed and flexible cycle-accurate network-on-chip simulator. In *ISPASS*, 2013.

[31] A. Joshi and M. Mutyam. Prevention flow-control for low latency torus networks-on-chip. In *NOCS*, 2011.

[32] J. Kahle et al. Introduction to the Cell multiprocessor. *IBM J. Res. Dev.*, 49(4.5):589 –604, Jul. 2005.

[33] P. Kermani et al. Virtual cut-through: a new computer communication switching technique. *Computer Networks*, 1979.

[34] J. Kim and H. Kim. Router microarchitecture and scalability of ring topology in on-chip networks. In *NoCArc*, 2009.

[35] A. Kumar et al. A 4.6Tbits/s 3.6GHz single-cycle NoC router with a novel switch allocator in 65nm CMOS. In *ICCD*, 2007.

[36] J. Laudon and D. Lenoski. The SGI Origin: a ccNUMA highly scalable server. In *ISCA*, 1997.

[37] D. Lenoski et al. The directory-based cache coherence protocol for the DASH multiprocessor. In *ISCA*, 1990.

[38] S. Ma et al. DBAR: an efficient routing algorithm to support multiple concurrent applications in networks-on-chip. In *ISCA*, 2011.

[39] S. Ma et al. Whole packet forwarding: Efficient design of fully adaptive routing algorithms for NoCs. In *HPCA*, 2012.

[40] P. S. Magnusson et al. Simics: A full system simulation platform. *Computer*, 35:50–58, Feb. 2002.

[41] M. Martin, M. Hill, and D. Sorin. Why on-chip cache coherence is here to stay. *Commun. ACM*, 55(7):78–89, 2012.

[42] G. Michelogiannakis et al. Packet chaining: efficient single-cycle allocation for on-chip networks. In *MICRO*, 2011.

[43] A. K. Mishra et al. A case for heterogeneous on-chip interconnects for CMPs. In *ISCA*, 2011.

[44] S. Mukherjee et al. The Alpha 21364 network architecture. In *Hot Interconnects*, 2001.

[45] N. Neelakantam et al. FeS2: A full-system execution-driven simulator for x86. In *Poster presented at ASPLOS*, 2008.

[46] C. Nicopoulos et al. ViChaR: A dynamic virtual channel regulator for network-on-chip routers. In *MICRO*, 2006.

[47] G. P. Nychis et al. On-chip networks from a networking perspective: Congestion and scalability in many-core interconnects. In *SIGCOMM*, 2012.

[48] L. Peh and W. Dally. A delay model and speculative architecture for pipelined routers. In *HPCA*, 2001.

[49] V. Puente et al. The adaptive bubble router. *J. Parallel Distrib. Comput.*, 61(9):1180–1208, Sep. 2001.

[50] V. Puente et al. Immunet: dependable routing for interconnection networks with arbitrary topology. *IEEE Trans. Comput.*, 57(12):1676–1689, 2008.

[51] S. L. Scott et al. The Cray T3E network: Adaptive routing in a high performance 3D torus. In *Hot Interconnects*, 1996.

[52] M. Shin and J. Kim. Leveraging torus topology with deadlock recovery for cost-efficient on-chip network. In *ICCD*, 2011.

[53] A. Singh et al. GOAL: a load-balanced adaptive routing algorithm for torus networks. In *ISCA*, 2003.

[54] Y. Tamir and G. Frazier. High-performance multiqueue buffers for VLSI communication switches. In *ISCA*, 1988.

[55] A. Vaidya et al. Impact of virtual channels and adaptive routing on application performance. *IEEE Trans. Parallel Distrib. Syst.*, 12(2):223 –237, Feb. 2001.

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

[57] D. Wentzlaff et al. On-chip interconnection architecture of the Tile processor. *Micro, IEEE*, 27(5):15–31, Sept.-Oct. 2007.

Sheng Ma received the B.S. and Ph.D. degrees in computer science and technology from the National University of Defense Technology (NUDT) in 2007 and 2012, respectively. He visited the University of Toronto from Sept. 2010 to Sept. 2012. He is currently an Assistant Professor of the School of Computer, NUDT. His research interests include on-chip networks, SIMD architectures and arithmetic unit designs.

Zhiying Wang received the PhD degree in electrical engineering from the NUDT in 1988. He is currently a Professor with School of Computer, NUDT. He has contributed over 10 invited chapters to book volumes, published 240 papers in archival journals and refereed conference proceedings, and delivered over 30 keynotes. His main research fields include computer architecture, computer security, VLSI design, reliable architecture, multi-core memory system and asynchronous circuit. He is a member of the IEEE and ACM.

Zonglin Liu received the B.S. and Ph.D. degrees in mechanical engineering from the National University of Defense Technology (NUDT) in 1998 and 2004, respectively. He is currently an Associative Professor of the School of Computer, NUDT. His research interests include general purpose processor and DSP architecture and designs, VLSI logic designs.

Natalie Enright Jerger received the B.A.Sc. degree from the Department of Electrical and Computer Engineering at Purdue University, and the M.S.E.E. and Ph.D. degrees from the Department of Electrical and Computer Engineering, University of Wisconsin - Madison. She is currently an Assistant Professor in the Electrical and Computer Engineering Department at the University of Toronto. Her research interests include on-chip networks, many-core architectures and cache coherence protocols. She is a member of the ACM and a senior member of the IEEE.