Aggressive Fine-Grained Power Gating of NoC Buffers

Yibo Wu, Leibo Liu, Liang Wang, Xiaohang Wang, Jie Han, Chenchen Deng, Shaojun Wei

***Abstract*—Power gating is effective for NoCs to reduce the excessive leakage power dissipated by idle network components. Most existing NoC power gating approaches rely on the routing algorithms to mitigate the power gating blocking latency problem. When the network becomes faulty and fault tolerant routing algorithms are applied, these approaches are no longer applicable or can seriously degrade the performance. Other approaches propose fine-grained buffer power gating, but they are too conservative in power saving due to the buffer backpressure flow control. To address these problems, we propose an aggressive fine-grained power gating of flit-sized buffer entries by adopting backpressureless flow control in an input-buffered network. The power gating decisions are made based on the flit deflection rate. However, directly applying the backpressureless flow control leads to the difficulties of multi-flit packet truncation and protocol deadlocks. Therefore, we modify the packet injection architecture to avoid packet truncation. This is done by chaining the local input port with a randomly chosen input port. Finally, we design a progressive recovery framework to handle both livelocks and protocol deadlocks. It does not need to truncate packets or strictly separate different message classes when the network is free of livelocks or protocol deadlocks. Experimental results show that with a hardware overhead of 9.6%, our design can save up to 59% network power consumption in both a fault-free and a faulty NoC with little zero-load latency penalty. Our design also approaches an ideal energy-proportional NoC because it can constantly reduce power consumption over a wide range of injection rates.**

***Index Terms*—NoC; buffer; fine-grained; power gating; backpressureless**

I. Introduction

As a scalable interconnection scheme, Network-on-Chip (NoC) consumes a large portion of total chip power. Due to the low average traffic load of real applications [4], many network components are idle, but they still consume high leakage power [9]. This fact necessitates power gating these components to save leakage power. Previous NoC power gating approaches power gate either idle routers or router components. The main objective of these designs is to save more power at a smaller expense of throughput and latency.

With the transistor technology scaling, NoCs are more prone to suffer from permanent transistor failures [2]. It is usually assumed that these permanent failures can render some bidirectional links faulty and make a regular network (e.g. mesh or torus) become irregular [2], [28], [31], [32]. For example, the authors of [2], [8] claim that many transistors are expected to fail during the manufacturing or over the lifetime due to wear-out. However, even for just 30 gate faults, 5∼50 links are expected to fail [2]. Besides, many heterogeneous networks with different sizes of cores also behave like regular networks with faulty links [32]. Therefore, it is necessary for an NoC power gating approach to function correctly and efficiently in both fault-free and faulty networks.

However, most existing power gating approaches are not applicable to or can incur tremendous performance loss in these faulty NoCs. The reason is that they have to rely on the routing algorithms (mostly deterministic routing algorithms) to mitigate the latency penalty caused by power gating. For example, many approaches, such as Power Punch [9] and Lookahead [26], generally assume using dimension-ordered routing (DOR) algorithm, which can easily predict the routing path for each packet such that the powered-off routers can be woken up in advance and the wakeup latency can be mitigated. However, the predictability becomes very limited for fault tolerant adaptive routing algorithms, making these power gating approaches that rely on DOR no longer applicable to faulty NoCs. Panthre [29] is based on the reconfiguration of up*/down* adaptive routing algorithm [2] and is therefore applicable to faulty NoCs. But in a faulty network, Panthre places too many turn restrictions that would waste path diversity and cause tremendous throughput loss [32].

Other approaches [23], [38] propose to use fine-grained power gating of flit-sized buffer entries[1]. They are not dependent on the routing algorithm and are therefore applicable to faulty topologies. However, these approaches are too conservative in power saving. The main reason is that they are based on a network with credit-based buffer backpressure [10]. To avoid the zero-load latency penalty, there is an upper

This work is supported in part by the National Natural Science Foundation of China (Grant No.61834002), in part by the National Key R&D Program of China (Grant No. 2018YFB2202101), in part by the National Science and Technology Major Project of the Ministry of Science and Technology of China (Grant No. 2018ZX01027101-002), and in part by the National Natural Science Foundation of China under Grant 61804088. (*Corresponding author: Leibo Liu.*)

Yibo Wu, Leibo Liu, Chenchen Deng and Shaojun Wei are with the Institute of Microelectronics, Tsinghua University, Beijing, China. Leibo Liu and Shaojun Wei are also with Beijing National Research Center for Information Science and Technology, Beijing, China. Email: wyb18@mails.tsinghua.edu.cn, {liulb, chenchendeng, wsj}@tsinghua.edu.cn

Liang Wang is with the School of Computer Science and Engineering, Beihang University, Beijing, China. Email: lwang20@buaa.edu.cn

Xiaohang Wang is with the School of Software Engineering, South China University of Technology, Guangzhou, China. Email: xiaohangwang@scut.edu.cn

Jie Han is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. Email: jhan8@ualberta.ca

This article was presented in the International Conference on Hardware/Software Codesign and System Synthesis 2020 and appears as part of the ESWEEK-TCAD special issue.

[1] In the remaining part of this paper, flit-sized buffer entries are always denoted as buffers for brevity.

limit on the number of powered-off buffers. For example, in Flexibuffer [23], the virtual channels (VCs) used by multi-flit packets must maintain at least 3 powered-on buffers to cover the credit round trip delay (typically 3 cycles) and the buffer wakeup latency (assumed 2 cycles).

To resolve these challenges, we propose BleG, which applies **B**ackpressureless flow control [19] in an input-buffered network for aggressive fine-grained buffer power **G**ating. The buffer power gating and wakeup decisions are made independently on every input port based on the flit deflection rate. BleG is applicable to faulty NoCs and can aggressively power gate all buffers at low network loads. However, to combine the backpressureless flow control with an input-buffered network is not trivial for Chip-multiprocessors (CMPs), where the NoC is used for transmitting cache-coherent messages. There mainly exist two difficulties with backpressureless flow control, i.e., multi-flit packet truncation and protocol deadlocks. Existing solutions [12], [19], [27] incur a significant latency and power overhead, and would completely offset the power saving if they are directly applied to BleG. Therefore, we also design new architectures to address these two issues. This paper makes the following contributions:

- We develop an aggressive fine-grained power gating mechanism for NoC buffers. The design is based on an input-buffered network that uses backpressureless flow control. The backpressureless flow control never stalls packets if downstream routers do not have free buffers, so that all buffers can be aggressively power gated. The power gating mechanism does not rely on the routing algorithm, and is therefore applicable to faulty NoCs.
- We modify the multi-flit packet injection architecture to avoid packet truncation and reassembly. This is done by chaining a randomly chosen input port with the local input port. Because body and tail flits do not need to carry routing information, the associated dynamic power is saved.
- We propose a progressive recovery framework to achieve livelock freedom and protocol deadlock freedom on a backpressureless network. It does not truncate packets or strictly separate different message classes when the network is free of livelocks or protocol deadlocks, thereby reducing power consumption and the latency penalty.

Compared with several state-of-the-art approaches in a fault-free mesh network, BleG can save up to 59% network power consumption at an expense of throughput of less than 12%. In a faulty network, BleG can increase the saturation point by up to 91% and reduce the power consumption by up to 59%. BleG can constantly reduce the power consumption over a wide range of injection rates, and is more energy proportional.

The rest of the paper is organized as follows. Section II presents the related work and motivation. Section III details the design of BleG. Section IV presents the experimental results. Section V concludes the paper.

II. Related work and motivation

A. NoC Power Gating

To save more power at a smaller expense of performance, NoC power gating approaches need to address the problems such as break-even time limit, network disconnection, and in particular, the accumulated blocking latency when a packet is to be transmitted through multiple powered-off routers[2] (or router components) [3].

However, when dealing with the blocking latency, most approaches are dependent on the routing algorithm. This induces a serious problem that when the network becomes faulty and needs to use a different routing algorithm, these approaches are not applicable anymore or can seriously degrade the performance. For example, NoRD [3] relies on routing in a ring bypass network to bypass the powered-off routers. But the ring bypass network is statically decided in the design time and can become broken due to faulty links. TooT [14] and SPONGE [16] observe that in a network using DOR, packets are more likely to go straight instead of making turns. They use bypass paths between straight directions to avoid waking up powered-off routers. But their observation can lose efficacy in fault tolerant routing algorithms. SMART [15] designs an XYX routing to reduce the probability that a packet enroute encounters a powered-off router. But the XYX routing can lead to network disconnection in faulty NoCs. Both Power Punch [9] and Lookahead [26] rely on DOR to predict the routers that are going to be encountered by the packet several hops away and utilize the hop count slack to wake them up in advance. But the predictability is usually very limited for fault tolerant adaptive routing algorithms because they are merely capable of predicting the routers one hop away. Most existing NoC power gating approaches, such as TooT, SPONGE, SMART, Power Punch and Lookahead, require the routing algorithm to be deterministic. Deterministic routing is however hard to implement in faulty NoCs, unless extremely complex routing tables are used. Panthre [29] observes that in a network using up*/down* routing [2], packets can be steered away from powered-off links to avoid wakeup. Panthre is applicable to faulty NoCs due to the routing table reconfiguration and deadlock-free routes provided by up*/down* routing. But it would waste path diversity and incur a tremendous throughput loss [32]. Besides, even if the network is fault-free and regular, the up*/down* routing that Panthre depends on is inferior to DOR in terms of saturation point.

In contrast, some approaches [23], [38] depends on the buffer usage for fine-grained power gating of buffers. They are not dependent on the routing algorithm and are applicable to faulty NoCs. They mitigate the problem of wakeup latency because the latency to wake up a buffer is trivial compared with the latency to wake up a whole router (assumed 2 cycles [23] versus 8 cycles [9]) and can be easily mitigated. However, these approaches are conservative in saving leakage power because they are based on a network using flow controls with buffer backpressure [20]. For example, Flexibuffer [23] is based on a network using credit-based buffer backpressure. The virtual networks (VNets) for single-flit packets cannot be power gated, so that at least 1 buffer remains powered-on to avoid packet dropping. The VNets for multi-flit packets should maintain at least 3 powered-on buffers to cover both the buffer wakeup latency (assumed 2 cycles) and the credit round trip

[2] It is generally assumed 6∼12 cycles to wake up a powered-off router.

latency (3 cycles), so that there is no zero-load latency penalty. Apparently, Flexibuffer [23] cannot achieve the ideal goal for a buffer power gating mechanism that all buffers can be safely power gated without impacting the zero-load latency.

To tackle the aforementioned problems, we design a fine-grained buffer power gating mechanism that is based on a network using the backpressureless flow control. The fine-grained buffer power gating allows the approach to be applicable to faulty NoCs. The backpressureless flow control allows all buffers to be aggressively power gated. However, there are additional difficulties associated with applying the backpressureless flow control, which are detailed as follows.

B. Backpressureless Flow Control

Input-buffered NoCs widely adopt flow controls with buffer backpressure that stalls packets when there is no available buffer in downstream routers. But backpressureless flow control does not stall packets in this circumstance. When two flits contend for the same output port, one of them is deflected to another port. As long as the number of input ports is equal to the number of output ports on every router, packet dropping can be avoided [10], [27]. The backpressureless flow control is usually used in bufferless NoCs [12], [27], so that all the hardware overhead and power consumption of the buffers can be eliminated. But in existing backpressureless networks, two difficulties remain and hinder their effectiveness in CMPs, where the NoC is used for transmitting cache-coherent messages. The first one is multi-flit packet truncation. The second one is protocol deadlocks.

Multi-flit packet truncation occurs due to the packet injection and the livelock freedom mechanism [27]. For example, a 5-flit packet routing is possibly interrupted by another packet and truncated into 2 new packets. If the 2 new packets reach the destination out of order, packet reassembly is required to maintain the correct flit order of the original packet. To enable reassembly, every flit needs to carry the routing information, which incurs a higher dynamic power. Moreover, previous approaches [12], [19] use existing Miss-Status Handling Registers (MSHR) as reassembly buffers and claim that there is no additional hardware overhead for reassembly buffers. But the reassembly process can still lead to latency penalty, which is not taken into consideration before. In one case study that we conduct under uniform random traffic in an 8×8 mesh NoC, we observe that the MSHR reassembly process increases the average flit latency by nearly 13.5% when the flit injection rate is merely 0.1 flits/node/cycle and the network is far from being saturated.

The protocol deadlock problem is also not well resolved in previous backpressureless approaches [12], [27], [36]. Protocol deadlocks occur due to the circular resource dependence among different message classes [39]. In networks with buffer backpressure, protocol deadlocks are usually avoided by using multiple virtual networks (VNet) [20] to separate different message classes. mDISHA [39] proposes to use progressive recovery to handle protocol deadlocks, but is not applicable to a faulty network. SurfNoC [37] is proposed for non-interfering NoCs and can be extended to support protocol deadlock freedom. However, the ideas of using multi-VNet, mDISHA and SurfNoC are not applicable to backpressureless NoCs. To the best of our knowledge, in backpressureless NoCs, the only method that can handle protocol deadlocks without using multiple physical networks is Surf-Bless [36]. Surf-Bless is a confined-interference routing mechanism proposed for bufferless networks. It can be also applied to avoid protocol deadlocks in a bufferless network. By placing different turn restrictions for different time domains, Surf-Bless extends SurfNoC and reduces the latency penalty of time-multiplexing a bufferless network. However, the turn restrictions force some packets to be deflected on certain nodes, making Surf-Bless livelock-prone and not applicable to faulty NoCs. To conclude, it is necessary to design a mechanism that can handle protocol deadlocks in a backpressureless network.

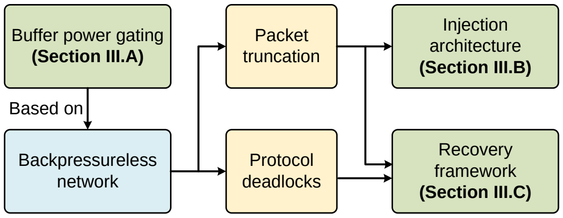


Fig. 1. BleG design overview

The above difficulties render the prior entirely bufferless designs not effective in cache-coherent systems. Since we are proposing BleG for power gating in backpressureless networks, we must design addtional mechanisms to handle the above difficulties that result from the backpressureless flow control. Therefore, we design a new injection architecture and a progressive recovery framework to handle the difficulties of packet truncation and protocol deadlocks.

III. BleG design

In this section, we detail the design of BleG. Figure 1 shows the design overview. Section III-A proposes the fine-grained buffer power gating of BleG based on a backpressureless network. However, directly applying the backpressureless flow control has two difficulties in CMPs, which are packet truncation and protocol deadlocks. To handle them, Section III-B slightly modifies the multi-flit packet injection architecture to avoid injection-induced packet truncation, and Section III-C proposes the progressive recovery framework to resolve both livelock freedom mechanism induced packet truncation and protocol deadlocks.

A. Power Gating and Wakeup Decision Making

1) Integrating backpressureless flow control into input-buffered NoCs:
BleG combines the backpressureless flow control with an input-buffered network, which is similar to the idea proposed in [27]. When a flit enters a router, if the buffer queue is empty, this flit directly attends output port allocation without being stored in buffers. Otherwise, the incoming flit is buffered while the flit at the head of the buffer queue attends output port allocation.

Algorithm 1 Output port allocation rules

```
1: function ALLOCATION($flit$)
2:     if $flit$ is a head flit then
3:         if $flit$ loses in output port contention then
4:             if there is a free buffer in this input port then
5:                 $flit$ is stored in this input port
6:             else
7:                 $flit$ is deflected to another output port
8:             end if
9:         else
10:            $flit$ moves forward
11:        end if
12:    else
13:        $flit$ follows its head flit
14:    end if
15: end function
```

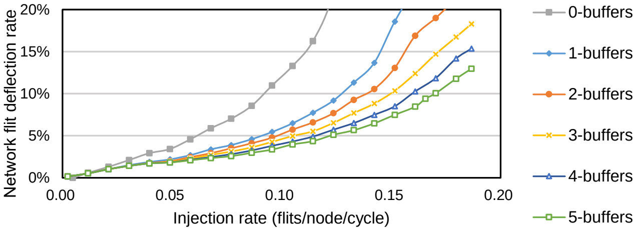


Fig. 2. Relationship between the number of powered-on buffers and the network flit deflection rate. The assigned synthetic traffic pattern is bit complement. The network flit deflection rate is the probability for a flit to be deflected from its productive output port.

During the output port allocation, a flit obeys the rules given in Algorithm 1. If a flit attending allocation is a body or tail flit, the flit is directly granted with the output port which its head flit has been sent to (line 13). When two or more flits are contending for the same output port, only one is granted with the output port and moves forward (line 10), whereas the others are deflected away from this productive output port. The deflections can be performed by using the allocator of [12], [34] or by adding a wavefront allocator after the normal allocator to give a random matching between unused ports. If a head flit loses in contention (line 3) and is going to be deflected, the head flit can wait in a free buffer (line 5) and attend allocation again in the next cycle. However, to avoid buffer overflow, when the buffer queue is fully occupied, the head flit cannot be stored in buffers but must be deflected (line 7). Due to the lack of buffer backpressure, the head flit can be deflected to any output port that has not been taken by another flit, regardless of whether the downstream routers have free buffers or not. In the extreme case where all buffers are powered-off, all flits constantly move forward without being buffered. The backpressureless network of BleG uses only 1 VC that is shared across multiple message classes.

2) Power gating mechanism:

The power gating mechanism of BleG is based on the fact that powered-on buffers can be used to avoid deflections. When a flit is deflected away from its destination, the latency increases and more dynamic power is consumed [13]. But with powered-on buffers, a flit is allowed to attend allocation for several times until it is granted with the productive output port or the buffer queue becomes full, thereby avoiding deflections. Figure 2 is a case study to illustrate the strong relationship between the number of powered-on buffers and the network deflection rate. The deflection rate is defined as the percentage of flits that are deflected during a round of allocation, namely, the probability for a flit to be deflected. Different curves represent networks with different fixed numbers of powered-on buffers on every input port. As the injection rate increases, the network deflection rate increases due to the contentions among packets. At the same injection rate, the network deflection rate decreases as the number of powered-on buffers increases.

Each input port makes the power gating decisions independently. The flit deflection rate of each input port is compared with a predefined threshold to decide if a buffer should be power gated or woken up. If the injection rate is low, the contention during allocations is low and few packets are deflected. Therefore, the buffer usage should be low and most buffers can be power gated to save buffer leakage power. As the injection rate increases, the flit deflection rate of an input port increases and negatively impacts the latency and dynamic power. Therefore, powered-off buffers of this input port are required to be woken up to reduce the deflection rate.

To compare the flit deflection rate with the predefined threshold, the input port of each router holds two counters to calculate the deflection rate. The two counters are named grant counter C_g and deflection counter C_d. C_g denotes the number of all flits sent from this input port. C_d denotes the number of flits that are deflected from this input port. The deflection rate of an input port is defined as $\frac{C_d}{C_g}$. Therefore, when C_g reaches a fixed value (100 in this paper, other values can also be used), C_d is compared with the value of $threshold \times C_g$ to make the power gating decision, and then both the counters are reset to zero. If C_d is greater, which indicates that the deflection rate is above the threshold, then one of the powered-off buffers is woken up to reduce the deflection rate. If C_d is 0, then one of the powered-on buffers is scheduled to be power gated. This buffer cannot be power gated instantly until the buffer becomes idle so as to avoid unnecessarily dropping flits.

The predefined threshold is a tradeoff between the power and packet latency. If the threshold is too low, more buffers should be woken up to reduce the deflection rate, and the leakage power saving would be trivial. If the threshold is too high, more buffers can be power gated to save more leakage power. But this comes at the expense of increased deflections and latency. The link dynamic power also increases because more packets are deflected and need to take longer routes to reach their destinations. To achieve a better tradeoff between performance and power, the predefined threshold is configured as 5% in this paper. Thus C_d is compared with 5 when C_g reaches 100 to make power gating decisions. Section IV-C2 provides a sensitivity analysis on the threshold configuration.

3) Advantages analysis:

Applicability to faulty NoCs: Compared with existing power gating approaches, BleG has a better applicability to faulty NoCs due to three reasons. First, BleG does not rely on the routing algorithm for power gating and eliminates the routing dependence problem. BleG is plug-and-play with different topologies and routing algorithms. Second, the backpressureless flow control is applicable to faulty NoCs without dropping packets if the number of input ports equals to that of output ports on every router. This requirement is satisfied when the link faults are bidirectional on commonly used topologies (e.g. mesh or torus) [17], [27]. Third, because BleG is based on a backpressureless network and there is no buffer dependence, routing deadlocks are avoided in BleG like other backpressureless approaches [12], [27]. BleG does not require using multiple VCs or routing restrictions. In faulty networks, there are many complex routing deadlock freedom mechanisms [28], [32]. For example, SWAP [28] resolves routing deadlocks by swapping packets of adjacent routers. However, when BleG is applied in a faulty network, there is no need to pay attention to routing deadlocks anymore.

Aggressive power saving but little blocking latency: Unlike Flexibuffer [23], the backpressureless flow control of BleG allows all buffers (except for the buffers of the local input port) to be aggressively power gated without incurring any blocking latency in the extreme cases. This is because BleG only power gates buffers without power gating crossbars. With the backpressureless flow control applied, packets can be sent to downstream routers even if downstream routers have no powered-on buffers. In contrast, in conventional power gating approaches, packets have to wait for the downstream routers or buffers to be woken up before moving forward. Therefore, BleG can maximize the potential buffer leakage power saving while eliminating the blocking latency. Moreover, although there are deflections that may incur longer routing paths and increase the latency of individual flits, under lower traffic loads, the deflection rate is low and deflections have minimal impact on the latency as shown in Figure 2.

Energy proportional NoCs: Coarse-grained power gating approaches are very sensitive to the traffic load, because every incoming flit would wake up the whole powered-off routers for service. The coarse granularity hinders them to achieve ideal energy-proportional NoCs [5], [11], [22]. An energy-proportional NoC should consume proportionally lower power when the traffic load is lower. But existing coarse-grained power gating approaches typically show significant power saving only when the injection rate is lower than 0.1 flit/node/cycle [3], [9], [16], while the network is still far from saturation. BleG outperforms these approaches in terms of energy proportional NoCs due to both the fine-grained power gating and aggressive power saving. In Section IV, we will show that BleG can save more leakage power over a significantly wider range of injection rates.

Break-even time limit: In existing NoC power gating approaches, a powered-off router (or router component) should be woken up whenever a packet encounters this router. If the time that a router has been power gated is shorter than the break-even time (generally assumed to be 10 cycles) [3], instead of saving leakage power, the power gating mechanism would consume even more power due to the wakeup charging. In contrast, BleG mitigates the impact of break-even time limitation. This is because the power gating decisions are made interval by interval, and each interval can last more than 100 cycles (when C_g reaches 100), which is much longer than the break-even time.

B. Multi-Flit Packet Injection Architecture

The first difficulty of applying backpressureless flow control in cache coherent systems is the multi-flit packet truncation and reassembly. To avoid them, this section modifies the multi-flit packet injection architecture.

Injection-induced multi-flit packet truncation occurs when an injecting packet is interrupted by another packet to avoid buffer overflow or packet dropping. To allow a multi-flit packet to be injected without truncation in a backpressureless network, there should exist one input link that is consecutively free of incoming flits until the whole packet finishes injection [27]. However, the lack of buffer backpressure makes it difficult to prevent flits of the upstream router from using this link. Therefore, when a multi-flit packet is partially injected, a possible scenario is that suddenly all connected links have incoming flits, but none of these flits is to be ejected. Under this circumstance, we need to allocate 4 output ports to 5 input ports. To avoid buffer overflow, the injection of the multi-flit packet has to be interrupted, causing packet truncation. To maintain in-order flit delivery in a packet, the truncated packet should be reassembled at the destination. The reassembly incurs both a higher dynamic power for header information transmission and a longer latency as analyzed in Section II-B.

To avoid injection-induced packet truncation, a possible solution is to add more buffers in each input port to temporarily store the incoming flits. However, assuming that a packet consists of 5 flits, this solution would require adding 5 buffers in each input port and induce much hardware overhead.

We modify the injection architecture for multi-flit packet to avoid packet truncation. Neither packet reassembly nor additional buffers are required. We observe that these additional buffers are no longer required if incoming flits can be stored in the buffers of the local input port. This is because when a flit is injected, a buffer in the local input port is released. Therefore, we address this problem by chaining one randomly chosen input port (denoted as a direction input port) with the local input port. An incoming flit from the link connected with the direction input port is allowed to use buffers of both input ports. When a flit is injected, this flit will release a buffer in the local input port. Then this released buffer of the local input port can be used to temporarily store the incoming flit of the direction input port. This method can be viewed as virtually increasing the number of available buffers in the direction input port to store incoming flits. With such an injection architecture, we only need to allocate 4 output ports for 4 input ports, such that packet truncation can be avoided.

Figure 3 shows the injection architecture as an example of avoiding packet truncation. The upper pale blue box shows the direction input port (in Figure 3 none of the 5 buffers are power gated). The lower pale blue box shows the local input port. These two input ports are connected by the chaining path. The large capital letters 'A', 'C' and 'D' denote flits belonging to different packets and their subscripts denote the flit type ('H': a head flit, 'B': a body flit, 'T': a tail flit). Only one flit of these two input ports can use the crossbar at a time.

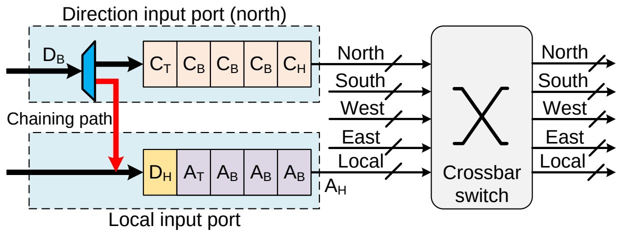


Fig. 3. BleG injection architecture. The direction input port is one randomly chosen input port (which is 'North' in this figure), other input ports are omitted for brevity.

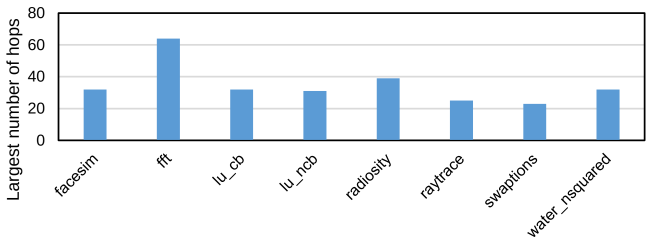


Fig. 4. The largest number of hops that packets have traversed under benchmark traces of Synfull. The network configuration is given in Section IV. Synfull models the cache coherence protocol. Therefore, with a finite network interface queue length, theoretically livelocks and protocol deadlocks could form.

When flit A_H is being injected, all flits of packet C wait in the direction input port. The incoming flit D_H passes through the chaining path and is buffered in the injection buffer released by flit A_H. In the subsequent cycles, flits of packet A are injected consecutively without truncation, and the incoming flits of packet D are buffered in the injection buffers released by packet A.

One possible concern is that the chaining path itself can become faulty. To improve fault tolerance, there are chaining paths between the local input port and all other input ports. But we statically select one input port as the direction input port and disable other chaining paths.

In the injection process, if there is a flit in the local input port, this flit is allowed to be injected when any of the conditions below is met. First, a complete packet has just been transmitted from the direction input port and currently there is no outstanding flit in the direction input port buffers. Second, after the head flit of the injecting packet is injected, the rest flits of this packet should be consecutively injected after this head flit. When there is an incoming flit on the link connected with the direction input port, this flit should be buffered in the local input port via the chaining path if another flit is being injected and the direction input port buffer queue is full.

To make the injection architecture able to avoid packet truncation regardless of the number of powered-on buffers in the direction input port, the buffers of the local input port should never be power gated. The local input port uses on/off-based buffer backpressure [20] to manage the transmission of packets from the network interface.

C. Progressive recovery for livelock freedom and protocol deadlock freedom

Existing backpressureless approaches would bring about multi-flit packet truncation when dealing with the deflections induced livelocks. The protocol deadlock problem is also not well addressed. To address these problems, we propose a progressive recovery framework that can achieve both livelock freedom and protocol deadlock freedom.

1) Necessity to use progressive recovery: Instead of proactively avoiding livelocks and separating different message classes to avoid protocol deadlocks, BleG relaxes the restrictions on the network until potential livelocks or protocol deadlocks are detected. Then BleG uses a progressive recovery framework to recover from both the livelocks and protocol deadlocks.

Relaxing the restrictions when the network is free of livelocks or protocol deadlocks can gain more power saving and performance improvement. Existing backpressureless approaches achieve livelock freedom by giving a packet the highest priority and ensuring that this packet will reach its destination. The prioritized packet should never be deflected and can often truncate other packets. For example, BLESS [27] uses an age-based priority. CHIPPER [12] designates the source router and MSHR id of one packet that is prioritized over other packets. A different one is AFC [19] that uses flit-by-flit routing to achieve probabilistic livelock freedom [10]. All these avoidance-based mechanisms have to truncate multi-flit packets to achieve livelock freedom. However, with a recovery-based mechanism, there is no need for packets to be truncated when the network is free of livelocks. Consequently, the additional dynamic power and reassembly latency can be avoided. The idea of recovery can also be used to address protocol deadlocks. If the network is free of protocol deadlocks, the network does not need to strictly separate different message classes.

However, the efficacy of a recovery-based mechanism is highly dependent on the probability of livelocks and protocol deadlocks. A lower probability can minimize the negative impact due to the power and performance overheads of the recovery procedures.

To illustrate the low probability of livelocks and protocol deadlocks, we conduct a case study in Figure 4 and plot the largest number of hops that packets have traversed in BleG. The largest number of hops can be used to track a potential livelock, because a packet trapped in livelocks would be constantly routed without ever reaching its destination and the largest number of hops would be very high. The largest number of hops can also be used to track a potential protocol deadlock in a backpressureless network. If a protocol deadlock occurs in a backpressureless network (e.g. with only request and reply packets), the request packets in the network cannot be ejected because the network interface input queue is full. These request packets are also constantly routed even if they can reach their destinations. Therefore, a protocol deadlock can also significantly increase the largest number of hops.

We run BleG in an 8×8 mesh network without any livelock or protocol deadlock freedom mechanisms under real workloads adopted from Synfull [4]. In the results given in

Figure 4, the largest number of hops is 64[3], which implies that few livelocks or protocol deadlocks occur and is in accordance with the observations of [39]. The low probability of livelocks and protocol deadlocks in BleG can be explained by two reasons. First, the average injection rate of Synfull is not high. Second, both the fine-grained buffer power gating and injection architecture of BleG can physically or virtually change the number of powered-on buffers, thereby creating randomness in the formation of livelocks and protocol deadlocks and helping avoid them.

Based on the above analyses, it is reasonable to turn to a recovery framework to handle the problems of livelocks and protocol deadlocks.

2) Detailed detection and recovery procedures design: The progressive recovery mechanism of BleG includes how to detect and how to recover from both livelocks and protocol deadlocks.

Detection: Two criteria are used to detect livelocks and protocol deadlocks. The first criterion is the number of hops that a packet has traversed. Every head flit spares 8 bits to record the number of hops. If a router detects that a packet has traversed 255 hops, a potential livelock or protocol deadlock is detected. The second criterion is the input queue length and is for expediting the detection of protocol deadlocks. When the network interface input queues are consecutively full for over a threshold time, e.g. 30 cycles, a potential protocol deadlock is detected.

Recovery: During the recovery, the network is free of livelocks or protocol deadlocks, so that all packets in the network will be ejected and the network is fully recovered after the recovery. This is the fundamental idea of the recovery mechanism. To achieve livelock freedom, all multi-flit packets are forced to be truncated and sent flit-by-flit, thereby enabling probabilistic livelock freedom [19]. This requires a small subnetwork that is only powered-on during the recovery for header information transmission. To achieve protocol deadlock freedom, the whole network is time-multiplexed by different time domains and each time domain is assigned to a message class. For example, with a cache coherence protocol of only request and reply packets, BleG uses two time domains to separate them. The time domains of the whole network are the same at any one time and change every cycle. In the first time domain, only reply flits are routed and reply flits from network interface can be injected. In the second time domain, both request and reply flits can be routed but the injection is forbidden. In this way, reply flits are never blocked by request flits and can always reach their destinations, thereby resolving protocol deadlocks.

To further detail the recovery framework, Figure 5 plots the flow chart of BleG recovery procedures. After a livelock or a protocol deadlock is detected, the network starts recovery procedures which include 4 stages ($tag2 \sim tag5$ in Figure 5). The recovery procedures require a 1-bit wire grid, 2 additional buffers in every input port (except for the local input port) to avoid buffer overflow and a small subnetwork for header information transmission in flit-by-flit routing.

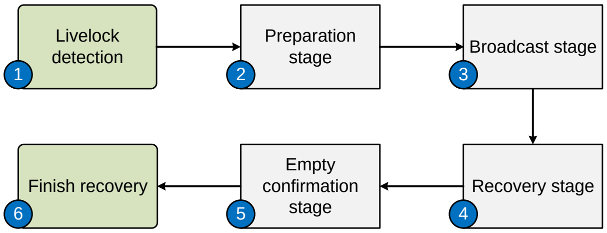


Fig. 5. Flow chart of BleG livelock recovery. After the livelock (or protocol) detection, the recovery procedures include 4 stages as the grey boxes show.

- Preparation stage ($tag2$): When a router detects a livelock, this router places a voltage pulse on the 1-bit wire grid to inform all other routers of the livelock. After receiving the pulse, all routers wake up the additional buffers and the subnetwork. Packet injection and routing stay as normal.
- Broadcast stage ($tag3$): This stage synchronizes the time domains of all routers. When the router that detects the livelock has woken up the buffers, this router generates and broadcasts a triggering message via normal links. The message carries a countdown so that all routers will enter the next stage simultaneously with the same time domain. The paths of broadcast is recorded for the empty confirmation state.
- Recovery stage ($tag4$): This stage uses flit-by-flit routing and time-multiplexing to eject all packets as described above.
- Empty confirmation stage ($tag5$): When a router finds that the input queues of its network interface are free, and it has not received any flits for 3 consecutive cycles, it enters the empty confirmation stage. It starts sending an empty message. The empty message is backpropagated to the router that detects the livelock via the recorded paths. When the router that detects the livelock has gathered empty messages from all routers, this router places a voltage pulse on the wire grid to finish the recovery procedures ($tag6$).
- Subnetwork for header information transmission: The subnetwork has a similar configuration with the main network, except that it does not have route computation units and allocators, and is only wide enough to carry the header information. Because BleG avoids packet truncation when there is not livelock, the sunbetwork is usually power gated. During the preparation stage, the subnetwork is woken up. During the broadcast stage, the header information of head flits are copied to the buffers of the subnetwork so that body and tail flits can carry the header information. The subnetwork is strictly synchronized with the main network in terms of allocation and routing.

The whole detection and recovery procedures do not cause any packet dropping or retransmission. The main performance overhead is the increased latency. This is because time-multiplexing increases the latency per hop [37] and packet reassembly is required. However, because the probability of livelocks and protocol deadlocks is low unless the network becomes deeply saturated, this performance overhead is negligible to the overall network performance. The recovery procedures also incur power overheads in terms of the additional message transmission, the subnetwork and recovery

[3] The value ’64’ corresponds to the unluckiest packet that undergoes more deflections than other packets. The average number of hops is still close to a network without deflections.

controllers. However, these power overheads are mitigated in a large degree because of the idea of using recovery. In the evaluation, we have considered all these performance and power overheads and show that BleG still achieves a satisfying tradeoff between performance and power saving.

IV. Evaluation

In this section, we present experimental results of BleG compared with other approaches. All the experiments are performed using gem5 [7] simulator with Garnet2.0 [1] in an 8x8 mesh network. The network of BleG is backpressureless whereas the network of other approaches uses credit-based buffer backpressure [10]. To avoid protocol deadlocks, the network of other approaches with buffer backpressure has 3 VNets and each VNet has 1 packet-sized VC. The size of data packets is 5 flits and the size of control packets is 1 flit. The flit size is 128 bits. The network of BleG has only 1 VC with 7 buffers, 2 of which are only powered-on during the recovery process. The local input port has 5 buffers. The direction input port of every router is randomly selected. The router model we used has 1 cycle router latency [30] and 1 cycle link traversal latency. All approaches use non-atomic VC allocation [6] and buffer bypassing [18], so that the performance and power consumption are optimized for the 1-VC network. When a router or a buffer is power gated, the power supply is completely cut off. We assume that the wakeup time of a router is 8 cycles [9] and the wakeup time of a buffer is 2 cycles [23]. The break-even time for power gating routers (or router components) is 10 cycles [9], [23].

After gathering the runtime statistics, we use DSENT [35] to estimate network power consumption under 22 nm technology. The frequency is 1 GHz and the router delay is 0.87 ns. For routers with different numbers of ports, the unused buffers and crossbars are power gated. We also consider the power overheads due to additional custom components. The power overheads of the counters and the recovery controller in each router is 7.44e-5 W, which is calculated by summing up the leakage power of all individual gates [21], [35]. Other power overheads are derived from DSENT.

In the experiments conducted in a fault-free network, BleG uses DOR for route computation and is compared with the following approaches:

No-PG: A baseline network without power gating. DOR is used for route computation.

Power Punch: A state-of-the-art power gating approach that power gates whole routers and uses DOR for route computation.

Flexibuffer: A fine-grained buffer power gating approach that uses DOR for route computation.

Panthre: A power gating approach that uses the same routing tables from ARIADNE [2] for route computation. ARIADNE is a fault-tolerant routing algorithm that uses up*/down* routing restrictions for deadlock freedom.

In the experiments conducted in faulty networks, Power Punch is not compared with because it is not applicable to a faulty network. Because Flexibuffer needs to be combined with both a routing algorithm and a deadlock freedom mechanism for correct function, we compare with two different implementations of Flexibuffer. The first one combines Flexibuffer and ARIADNE (denoted as 'A+Flexibuffer'). The second one combines Flexibuffer with the deadlock recovery framework of SPIN [33] and the routing tables of ARIADNE (but the up*/down* routing restrictions are removed, denoted as 'S+Flexibuffer'). BleG is combined with the routing tables from ARIADNE for route computation. However, because BleG is backpressureless and avoids routing deadlocks, the up*/down* routing restrictions are removed. In summary, BleG is compared with the following approaches:

No-PG: A baseline network without power gating. The same routing tables from ARIADNE are used for route computation.

A+Flexibuffer: A fine-grained buffer power gating approach that uses the same routing tables from ARIADNE for route computation.

S+Flexibuffer: A fine-grained buffer power gating approach that uses the routing tables from ARIADNE for route computation. The up*/down* routing restrictions are removed and SPIN is integrated for deadlock freedom.

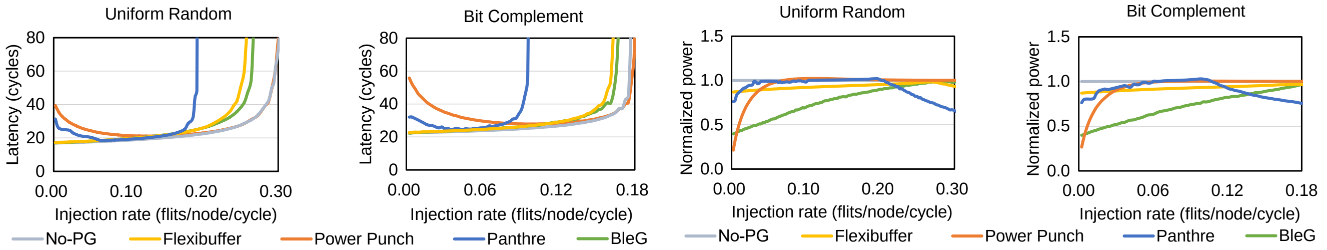
Panthre: A power gating approach that uses the same routing tables from ARIADNE for route computation.

A. Under Synthetic Traffic Patterns

Figure 6∼8 plot the latency and network power comparisons under synthetic traffic patterns. The runtime is 100K cycles and the warmup time is 10K cycles. The network of Figure 6 is free of faulty links. The network of Figure 7 has a fixed clustered distribution of 5 faulty links. The links between the following pairs of routers are faulty: (27, 26), (27, 35), (27, 28), (28, 20), (28, 29). The network of Figure 8 has a fixed distribution of 10 faulty links. The links between the following pairs of routers are faulty: (39, 31), (41, 49), (17, 25), (10, 11), (41, 40), (50, 58), (35, 27), (60, 52), (20, 21), (27, 28). Although the same number of faulty links can create multiple different faulty topologies, we only show two specific topologies here to illustrate how the latency and power consumption generally change as the injection rate increases. Other numbers and distributions of faulty links can generate similar trends.

In Figure 6, BleG can reduce the power consumption of No-PG by up to 59%, at the expense of 12% and 7% of saturation point under uniform random and bit complement traffic respectively. BleG is more energy proportional because it is able to consistently reduce the power consumption even if the network is near saturation. Another remarkable advantage of BleG is that BleG incurs little zero-load latency penalty. Because BleG does not power gate crossbars and is based on a backpressureless network, a packet is not prevented from reaching downstream routers even if the downstream routers have no powered-on buffers, thereby avoiding the blocking latency problem which is universal in other power gating approaches [3], [9]. Besides, as shown in Figure 2, the low overall deflection rate under low traffic loads ensures that deflections do not incur much additional latency.

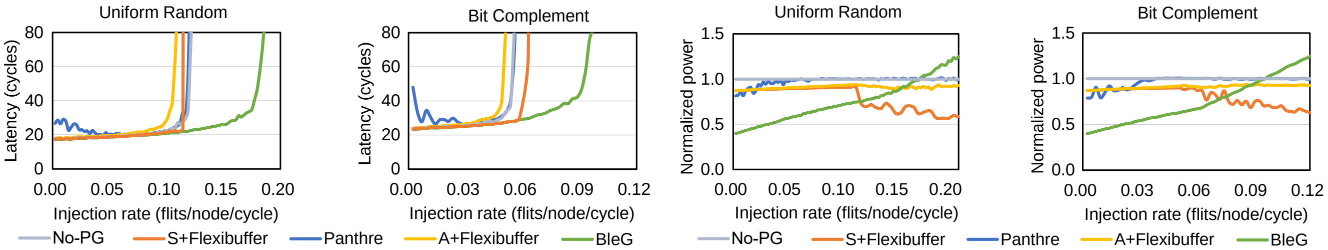
Flexibuffer also incurs little zero-load latency penalty. This is because Flexibuffer maintains at least 3 powered-on buffers



(a) Latency and throughput comparison

(b) Normalized power comparison

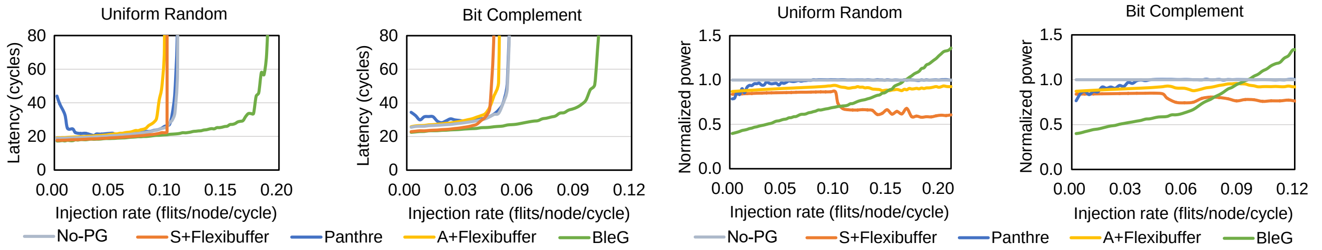
Fig. 6. Comparisons under synthetic traffic patterns in a fault-free network. The results of power are normalized to No-PG.



(a) Latency and throughput comparison

(b) Normalized power comparison

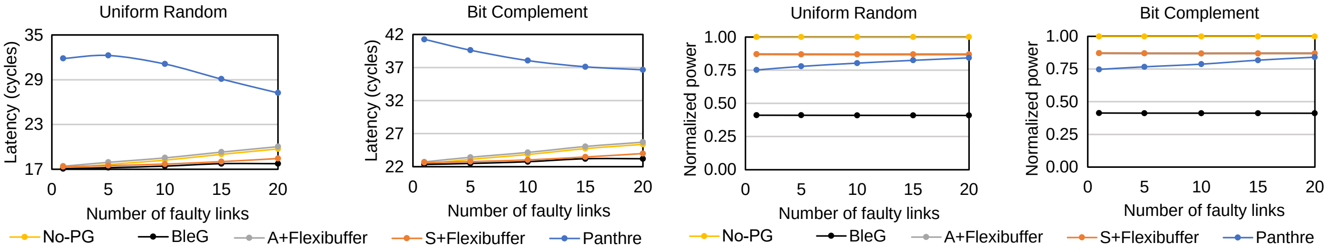
Fig. 7. Comparisons under synthetic traffic patterns in a faulty network with 5 faulty links. The results of power are normalized to No-PG.



(a) Latency and throughput comparison

(b) Normalized power comparison

Fig. 8. Comparisons under synthetic traffic patterns in a faulty network with 10 faulty links. The results of power are normalized to No-PG.



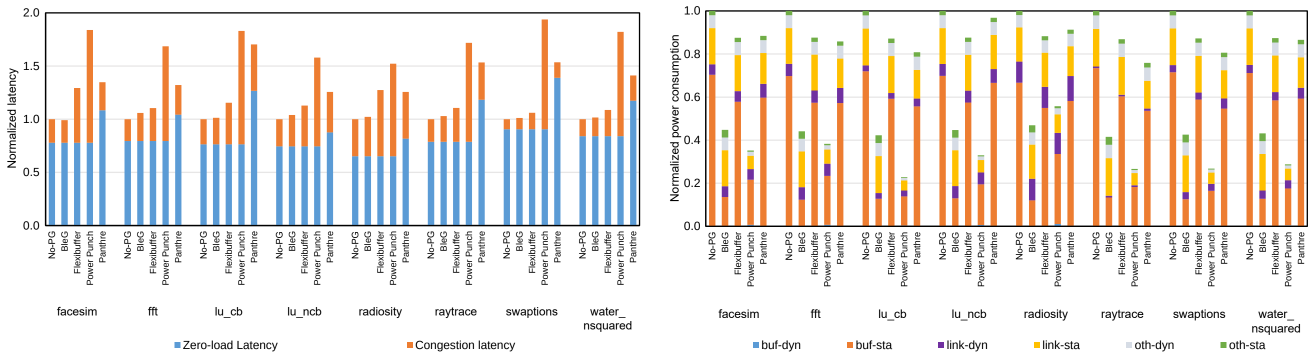
(a) Average zero-load latency comparison

(b) Average zero-load power consumption comparison

Fig. 9. Average zero-load latency and power consumption comparisons in faulty networks. The results of power are nomalized to No-PG.

to cover the buffer wakeup latency. However, compared with BleG, Flexibuffer has a slightly lower saturation point and is only capable of reducing the power consumption of No-PG by up to 13%. Power Punch reduces more power than BleG when the traffic load is near zero. This is because Power Punch can power gate both buffers and crossbars while BleG can only power gate buffers. However, Power Punch incurs a high latency penalty due to packet blocking in a single-cycle router pipeline network. In Power Punch, the authors assume a 3-stage router pipeline [9] so that an 8-cycle router wakeup latency can be completely hidden by sending wakeup signals 3 hops ahead. Therefore, in the network with 1-stage router pipeline of Figure 6, Power Punch fails to completely hide the router wakeup latency. When the injection rate is low, Panthre has neither a large power saving nor a small latency penalty, which can be attributed to the serious packet misroutes. When the network becomes over-saturated, although Panthre consumes less power compared with other approaches, this comes at the expense of a 40∼60% lower over-saturation throughput. The up*/down* routing (ARIADNE) used by Panthre is not comparable to DOR in terms of throughput in a fault-free network.

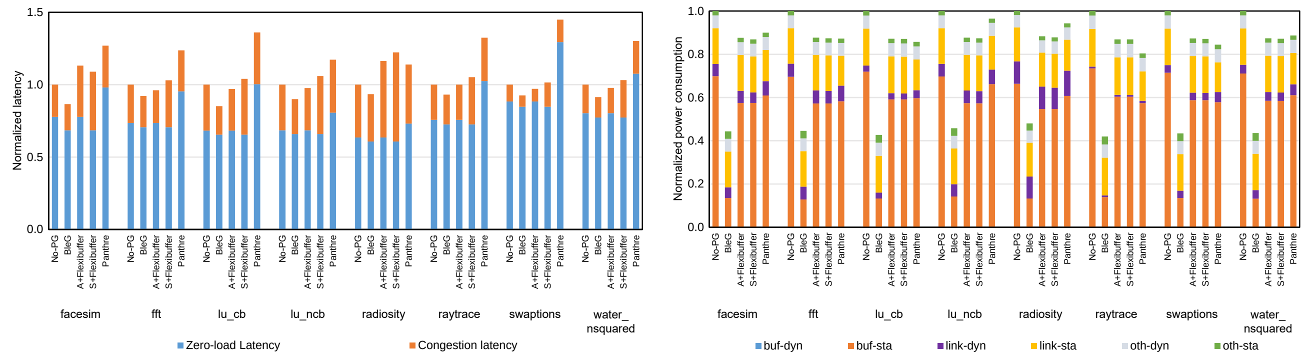
In Figure 7 and 8, BleG and S+Flexibuffer have the lowest zero-load latency due to the removal of up*/down* routing restrictions. Panthre has the longest latency due to both routing restrictions and packet detours. BleG can reduce the power



(a) Normalized latency comparison

(b) Normalized power consumption comparison

Fig. 10. Breakdown of latency and power consumption under real workloads in a fault-free network. The results are normalized to No-PG.



(a) Normalized latency comparison

(b) Normalized power consumption comparison

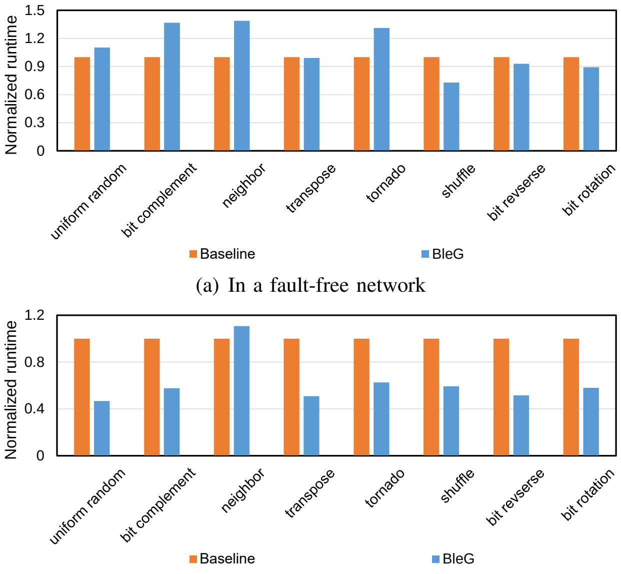
Fig. 11. Breakdown of latency and power consumption under real workloads in a faulty network, with the topology of Figure 8. The results are normalized to No-PG.

consumption of No-PG by up to 59%. A particular difference compared with the fault-free results of Figure 6 is that, instead of reducing the saturation point, BleG increases the saturation point of No-PG by 52% under uniform random traffic and 87% under bit complement traffic in Figure 7, and by 77% under uniform random traffic and 91% under bit complement traffic in Figure 8. This is because the deflections of BleG bring about a better adaptivity and improves the network bandwidth utilization. Therefore, although BleG consumes more power when the network becomes over-saturated, the energy-delay product (EDP) of BleG is in fact over 22∼30% lower than No-PG. Compared with BleG, other approaches such as S+Flexibuffer, A+Flexibuffer and Panthre merely achieve a conservative power reduction. Existing power gating mechanisms that are applicable to faulty NoCs can only achieve conservative power saving and can incur significant performance loss. When the network becomes over-saturated, S+Flexibuffer suffers from over-saturation throughput degradation. Therefore, S+Flexibuffer can reduce the power consumption by approximately 40% while it increases the EDP of No-PG by nearly 100% actually.

Figure 9 shows the average zero-load latency and power consumption of BleG and other approaches in faulty networks. For each number of faulty links, we keep randomly generating different topologies for evaluation until the average results stabilize. BleG consistently has the lowest latency and the greatest power saving, which is in accordance with Figure 7∼8. Due to the removal of up*/down* routing restrictions, S+Flexibuffer has a slightly lower latency than ARIADNE and A+Flexibuffer. The power saving of A+Flexibuffer is almost equal to that of S+Flexibuffer. Although Panthre saves more power than A+Flexibuffer and S+Flexibuffer, it incurs a longer latency due to the excessive detours.

B. Under Real Workloads

Figure 10 and 11 plot the breakdown of latency and power consumption under real workloads. The workloads are adopted from Synfull [4]. The configurations of the workloads and cores in an 8 × 8 network are the same as [4], [25]. The runtime is 5M cycles. All results are normalized to the latency and power consumption of No-PG. The latency is decomposed into zero-load latency and congestion latency. The congestion latency includes the latency due to unnecessary queueing, deflections and buffer stalling. The power consumption is decomposed into the dynamic and static power of links (link-dyn and link-sta), buffers (buf-dyn and buf-sta) and other network components (oth-dyn and oth-sta). The dynamic power of buffers is almost negligible because we have used buffer bypassing technique for all approaches in the experiments. The dynamic power and static power of crossbars are included in link-dyn and link-sta, respectively. The dynamic power and static power of other components, such as clocks, allocators, BleG recovery controllers and the subnetwork, are included in oth-dyn and oth-sta, respectively. All the latency and power overheads incurred by the progressive recovery procedures in BleG are taken into consideration.



(a) In a fault-free network

(b) In a faulty network, with the topology of Figure 8

Fig. 12. Runtime comparisons when the protocol deadlocks occur frequently, the results are normalized to the baseline that uses multi-VNet for protocol deadlock avoidance

In Figure 10, BleG on average increases the latency by merely 3% while reducing the overall power consumption by 56%. The backpressureless flow control and the low deflection rate mitigate the blocking latency problem. Besides, BleG saves power at a very slight expense of saturation point, as shown in Figure 6. Therefore, BleG has a similar performance to No-PG. In terms of power consumption, although the deflections and recovery procedures of BleG increase the power consumption to some extent, the small increase is however completely offset by the drastic reduction in buffer static power. Although Power Punch reduces the power consumption by 67% on average and is more energy-saving than BleG, it incurs a 74% latency penalty and is not suitable to applications that are sensitive to memory latency. Flexibuffer increases the latency by 15% but conservatively reduces the power consumption by only 13%. Panthre increases the latency by 44% and only reduces the power consumption by 14%. It is obvious that BleG achieves the best tradeoff between power consumption and performance on a fault-free network.

In Figure 11, because BleG significantly improves the saturation point in a faulty network due to better adaptivity as shown in Figure 7∼8, BleG reduces the congestion latency of the baseline. The zero-load latency of BleG and S+Flexibuffer is also slightly lower than other approaches because the up*/down* routing restrictions are removed. Due to the aggressive power gating of buffers, BleG again achieves the greatest power saving.

C. Additional Experiments

1) Recovery efficiency:

Due to the low average injection rate of Synfull benchmarks, the frequency of livelocks and protocol deadlocks is low in the experiments of Figure 10 and Figure 11. To better illustrate the recovery efficiency of BleG, we conduct experiments that model the cache coherence under synthetic traffic [24], [39] and stress the network with heavy traffic loads. There are request and reply packets. Every processor needs to generate 10000 request packets in total. When the request packet reaches its destination, the destination sends a reply packet back to the source after 80 cycles to model the memory latency. The runtime is the shortest time for all processors to receive 10000 complete reply packets. When a processor has sent 16 request packets but has not received any reply packets back, it stops sending request packets to model the limited number of MSHRs. Otherwise, the processor generates request packets at the rate of 1 packet/cycle. The network interface has separate input and output queues for both request and reply packets. Each queue is 16 packet-sized. The frequency of livelocks and protocol deadlocks is higher due to the high injection rates. For example, the protocol deadlocks occur once every 970 cycles under uniform random traffic. Figure 12 plots the runtime comparisons in both fault-free and faulty networks. The runtime of BleG is normalized to the baseline that uses multiple VNets (each with 1 VC) to achieve protocol deadlock avoidance. In Figure 12(a), the runtime of BleG is on average 8.9% higher than the baseline. This is in accordance with Figure 6(a) that BleG incurs no zero-load latency overhead and a slight reduction in saturation point. In Figure 12(b), the runtime of BleG is on average 37.8% lower than the baseline. The significant reduction in runtime can be attributed to both the high recovery efficiency and the improved saturation point in faulty networks, as also shown in Figure 7(a) and Figure 8(a). Figure 12 illustrates that the recovery efficiency of BleG is acceptable in a heavily loaded network. In real systems, due to high cache hit rate, the rate that request packets are generated would be much lower and the runtime gap would be much smaller.

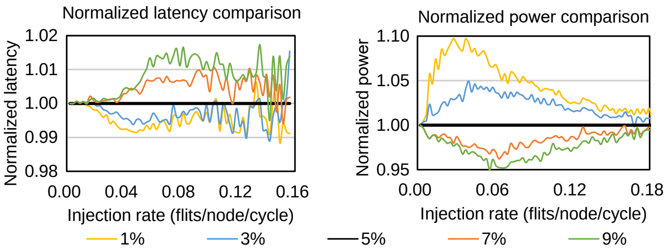


Fig. 13. Impact of the deflection rate threshold. The assigned traffic pattern is bit complement and the network is fault-free. Results are normalized to the 5% case.

2) Deflection rate threshold:

We conduct experiments for the sensitivity study of the deflection rate threshold configuration. Figure 13 plots the normalized latency and power consumption comparisons as the injection rate increases. Different curves represent BleG configured with different deflection rate thresholds. In the figure of latency comparison, BleG with a smaller threshold generally has a lower latency. This is because a smaller threshold tends to wake up more buffers to reduce the deflection rate and the latency, making the power gating mechanism more conservative. But the latency difference is quite small. In the figure of power comparison, BleG with the threshold of 9% usually consumes the least power. When the threshold is greater than 5%, more buffers can stay powered-off and more buffer leakage power is saved. Additionally, we observe that as the injection rate increases and the network becomes more congested, many input ports require all buffers to be woken up regardless of the deflection rate threshold. Therefore, the power consumption gap among different choices of the thresholds becomes smaller.

3) Hardware overhead:

The additional hardware required by BleG mainly includes:

- Buffer power gating: BleG uses the modified linked-list buffer management proposed in Flexibuffer [23] to manage the aggressive fine-grained buffer power gating. BleG also requires 2 counters at every input port for deflection rate comparison.
- Injection architecture: BleG requires the chaining paths to avoid injection-induced packet truncation.
- BleG requires a one-bit wire grid to start and finish the recovery procedures. During the recovery, 2 additional buffers at every input port are used to avoid buffer overflow and a 16-bit subnetwork is used for header information transmission. There are some other necessary components (e.g. the finite state machine) at every router for the recovery procedures described in Section III-C2.

By synthesizing the modified router using the Design Compiler under 45nm TSMC library, the area of a baseline router is 68033 μm^2. The area of BleG router is 74553 μm^2. BleG incurs a hardware overhead of approximately 9.6%. Although not trivial, the hardware overhead is worthwhile considering the significant power saving and throughput improvement of BleG in faulty NoCs.

V. Conclusion

Existing NoC power gating approaches are either not applicable to faulty topologies or conservative in power saving. We propose to apply backpressureless flow control for fine-grained buffer power gating. The packet injection architecture is modified and a progressive recovery framework is designed to tackle packet truncation and protocold deadlocks. A better tradeoff between power and performance can be achieved. In future work, increasing the saturation point and reducing the deflection rate will be considered.

References

[1] N. Agarwal et al. Garnet: A detailed on-chip network model inside a full-system simulator. In *ISPASS*, pages 33–42, 2009.

[2] K. Aisopos et al. Ariadne: Agnostic reconfiguration in a disconnected network environment. In *PACT*, pages 298–309, 2011.

[3] L. Chen andothers. Nord: Node-router decoupling for effective power-gating of on-chip routers. In *MICRO*, pages 270–281, 2012.

[4] M. Badr et al. Synfull: Synthetic traffic models capturing cache coherent behaviour. In *ISCA*, pages 109–120, 2014.

[5] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 2007.

[6] Daniel Becker. Efficient microarchitecture for network-on-chip routers. 10 2019.

[7] Nathan Binkert et al. The gem5 simulator. *SIGARCH Comput. Archit. News*, 39(2):1–7, 2011.

[8] S. Borkar. Designing reliable systems from unreliable components: the challenges of transistor variability and degradation. *IEEE Micro*, 25(6):10–16, 2005.

[9] L. Chen et al. Power punch: Towards non-blocking power-gating of noc routers. In *HPCA*, pages 378–389, 2015.

[10] William Dally et al. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[11] Reetuparna Das et al. Catnap: Energy proportional multiple network-on-chip. In *ISCA*, pages 320–331, 2013.

[12] C. Fallin et al. Chipper: A low-complexity bufferless deflection router. In *HPCA*, pages 144–155, 2011.

[13] C. Fallin et al. Minbd: Minimally-buffered deflection routing for energy-efficient interconnect. In *NOCS*, pages 1–10, 2012.

[14] H. Farrokhbakht et al. Toot: an efficient and scalable power-gating method for noc routers. In *NOCS*, pages 1–8, 2016.

[15] H. Farrokhbakht et al. Smart: A scalable mapping and routing technique for power-gating in noc routers. In *NOCS*, pages 1–8, 2017.

[16] Hossein Farrokhbakht et al. Sponge: A scalable pivot-based on/off gating engine for reducing static power in noc routers. In *ISLPED*, pages 1–6, 2018.

[17] Mohammad Fattah et al. A low-overhead, fully-distributed, guaranteed-delivery routing algorithm for faulty network-on-chips. In *NOCS*, pages 18:1–18:8, 2015.

[18] Hangsheng Wang et al. Power-driven design of router microarchitectures in on-chip networks. In *Proceedings. 36th Annual IEEE/ACM International Symposium on Microarchitecture, 2003. MICRO-36.*, pages 105–116, 2003.

[19] Syed Ali Raza Jafri et al. Adaptive flow control for robust performance and energy. In *MICRO*, pages 433–444, 2010.

[20] N. E. Jerger et al. *On-Chip Networks: Second Edition*. Morgan & Claypool, 2017.

[21] A. B. Kahng, Bin Li, L. Peh, and K. Samadi. Orion 2.0: A fast and accurate noc power and area model for early-stage design space exploration. In *2009 Design, Automation Test in Europe Conference Exhibition*, pages 423–428, 2009.

[22] G. Kim, H. Choi, and J. Kim. Tcep: Traffic consolidation for energy-proportional high-radix networks. In *ISCA*, pages 712–725, 2018.

[23] G. Kim et al. Flexibuffer: Reducing leakage power in on-chip network routers. In *DAC*, pages 936–941, 2011.

[24] Hanjoon Kim, Seulki Heo, Junghoon Lee, Jaehyuk Huh, and John Kim. On-chip network evaluation framework. In *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, page 10, 2010.

[25] Z. Li et al. The runahead network-on-chip. In *HPCA*, pages 333–344, 2016.

[26] H. Matsutani et al. Performance, area, and power evaluations of ultrafine-grained run-time power-gating routers for cmps. *TCAD*, pages 520–533, 2011.

[27] Thomas Moscibroda et al. A case for bufferless routing in on-chip networks. In *ISCA*, pages 196–207, 2009.

[28] Mayank Parasar, Natalie Enright Jerger, Paul V. Gratz, Joshua San Miguel, and Tushar Krishna. Swap: Synchronized weaving of adjacent packets for network deadlock resolution. In *Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture*, page 873–885, 2019.

[29] R. Parikh et al. Power-aware nocs through routing and topology reconfiguration. In *DAC*, pages 1–6, 2014.

[30] S. Park et al. Approaching the theoretical limits of a mesh noc with a 16-node chip prototype in 45nm soi. In *DAC*, pages 398–405, 2012.

[31] V. Puente, J. A. Gregorio, F. Vallejo, and R. Beivide. Immunet: A cheap and robust fault-tolerant packet routing mechanism. In *ISCA*, page 198, 2004.

[32] A. Ramrakhyani et al. Static bubble: A framework for deadlock-free irregular on-chip topologies. In *HPCA*, pages 253–264, 2017.

[33] A. Ramrakhyani et al. Synchronized progress in interconnection networks (spin): A new theory for deadlock freedom. In *ISCA*, pages 699–711, 2018.

[34] A. Runge and R. Kolla. Using benes networks at fault-tolerant and deflection routing based network-on-chips. In *2016 Tenth IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, pages 1–8, 2016.

[35] C. Sun et al. Dsent - a tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling. In *NOCS*, pages 201–210, 2012.

[36] P. Wang et al. Surf-bless: A confined-interference routing for energy-efficient communication in nocs. In *DAC*, pages 1–6, 2019.

[37] Hassan M. G. Wassel, Ying Gao, Jason K. Oberg, Ted Huffmire, Ryan Kastner, Frederic T. Chong, and Timothy Sherwood. Surfnoc: A low latency and provably non-interfering approach to secure networks-on-chip. In *ISCA*, page 583–594, 2013.

[38] Xuning Chen et al. Leakage power modeling and optimization in interconnection networks. In *ISLPED*, pages 90–95, 2003.

[39] Yong Ho Song et al. A progressive approach to handling message-dependent deadlock in parallel computer systems. *IEEE Transactions on Parallel and Distributed Systems*, 14(3):259–275, 2003.