Hierarchical organization is widely used in high-radix routers to enable efficient scaling to higher switch port count. A general-purpose hierarchical router must be symmetrically designed with the same input buffer depth, resulting in a large amount of unused input buffers due to the different link lengths. Sharing input buffers between different input ports can improve buffer utilization, but the implementation overhead also increases with the number of shared ports. Previous work allowed input buffers to be shared among all router ports, which maximizes the buffer utilization but also introduces higher implementation complexity. Moreover, such design can impair performance when faced with long packets, due to the head-of-line blocking in intermediate buffers.

In this work, we explain that sharing unused buffers between a subset of router ports is a more efficient design. Based on this observation, we propose Centralized Input Buffer Design in Hierarchical High-radix Routers (CIB-HIER), a novel centralized input buffer design for hierarchical high-radix routers. CIB-HIER integrates multiple input ports onto a single tile and organizes all unused input buffers in the tile as a centralized input buffer. CIB-HIER only allows the centralized input buffer to be shared between ports on the same tile, without introducing additional intermediate virtual channels or global scheduling circuits. Going beyond the basic design of CIB-HIER, the centralized input buffer can be used to relieve the head-of-line blocking caused by shallow intermediate buffers, by stashing long packets in the centralized input buffer. Experimental results show that CIB-HIER is highly effective and can significantly increase the throughput of high-radix routers.

CCS Concepts: • Computer systems organization → Interconnection architectures; • Hardware → Networking hardware;

Additional Key Words and Phrases: High-radix router, hierarchical organization, centralized input buffer

ACM Reference format:
Cunlu Li, Dezun Dong, Shazhou Yang, Xiangke Liao, Guangyu Sun, and Yongheng Liu. 2021. CIB-HIER: Centralized Input Buffer Design in Hierarchical High-radix Routers. ACM Trans. Arch. Code Optim. 18, 4, Article 50 (July 2021), 21 pages. https://doi.org/10.1145/3468062

This is a new article, not an extension of a conference paper.

The work was supported by National Key Research and Development Program of China under Grant No. 2018YFB0204300, NSFC under Grant No. 62002368, National Postdoctoral Program for Innovative Talents under Grant No. BX20190091, and Excellent Youth Foundation of Hunan Province (Dezun Dong).

Authors’ addresses: C. Li, D. Dong (corresponding author), S. Yang, and X. Liao, College of Computer, National University of Defense Technology, 109 Deya Road, Changsha, Hunan, 410073, China; emails: {cunluli, dong, shazhouyang, xkliaog}@nudt.edu.cn; G. Sun, Peking University, Beijing, 100871, China; email: gsun@pku.edu.cn; Y. Liu, Peng Cheng Laboratory, Shenzhen, 154100, China; email: yhlulu@gmail.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2021 Association for Computing Machinery.
1544-3566/2021/07-ART50
https://doi.org/10.1145/3468062

ACM Transactions on Architecture and Code Optimization, Vol. 18, No. 4, Article 50. Publication date: July 2021.
1 INTRODUCTION

Today, the large-scale supercomputers, such as Tianhe-2 system [25] and Sunway TaihuLight [14], have tens of thousands of computing nodes. The emerging exascale system would require hundreds of thousands of interconnected processors. Thanks to the increase of router bandwidth, these systems can be built from many thin links rather than fewer fat links [22], consisting low-diameter networks. Some excellent networks, such as Dragonfly [21], HyperX [2], Skywalk [15], SlimFly [7], and Bundlefly [23], have been proposed to design efficient low-diameter networks with high-radix routers. High-radix routers can provide high bandwidth and are the foundation of low-diameter networks. Hierarchical organization [22] that partitions a large switch into smaller subswitches has been widely used in high-radix routers, because it enables scalable microarchitecture.

A challenge for hierarchical routers is that a large amount of input buffers are suffering from low utilization. On the one hand, to meet the needs of credit-based flow control, the depth of input buffer should be proportional to the length of the connected link. On the other hand, the network links are usually asymmetrically designed in supercomputers. This design leads to asymmetric requirements of input buffers in different ports of hierarchical routers. However, traditional High-performance Computing (HPC) systems are usually designed with general-purpose routers whose ports are symmetric to meet the needs of different topologies. With such design, input buffers are often designed very deep to fit the longest external links. This design, unfortunately, results in a large amount of unused buffers, particularly in edge routers where the connected links are relatively short. YARC [29], as a typical hierarchical high-radix router, has been widely used in HPC systems. Figure 1 presents the utilization of input buffers in a YARC router [29]. The evaluation is performed using 1-meter endpoint links and 50-meter inter-router links [1] with 128-flit depth input buffers in a 1024-node Fat Tree network [11]. As shown in Figure 1, 86.1% input buffers have a utilization lower than 10%. Therefore, it is necessary to make full use of these unused input buffers in hierarchical routers.

Sharing input buffers between different input ports can improve the buffer utilization of hierarchical routers, and more shared input ports can achieve a more balanced buffer utilization. Work in Reference [1] proposes Stash, a new hierarchical router architecture that shares input buffer between all the input ports in the router. Stash makes full use of unused input buffer by stashing packets in the shared buffer of any port of the router. This method is efficient in balancing buffer utilization but can be expensive in implementation. This is because the design complexity also increases with the number of input ports that participate in sharing. Moreover, with this design, Head-of-Line (HoL) blocking caused by shallow intermediate buffers (row buffers and column buffers) can hurt the performance when faced with long packets. While input buffers are often very deep to cover the round-trip latency of the longest external channels, intermediate buffers are relatively shallow from the short round-trip latency within the switch. In Stash, packets need to be sent to the shared buffer of other ports via intermediate buffers, where the HoL blocking can occur with long packets, if the lengths of these long packets exceed the depth of intermediate buffers.

In this article, we strive to make full use of unused input buffers in hierarchical routers without compromising the performance. We prove that it is better to share unused input buffers between several ports instead of all the ports. Therefore, we propose Centralized Input Buffer Design in Hierarchical High-radix Routers (CIB-HIER), a novel hierarchical router architecture to

\footnotesize
1For example, links to endpoints can be shorter than 1 m, while the longest link can be more than 50 m [12].
2Long packets account for a large percentage of high-performance computing systems.
3It should be noted that CIB-HIER is designed for a large HPC system that needs to utilize long optical links.

ACM Transactions on Architecture and Code Optimization, Vol. 18, No. 4, Article 50. Publication date: July 2021.
Fig. 1. Cumulative probability of input buffer utilization in the hierarchical router (YARC) of a Fat Tree network.

Fig. 2. Design purpose of CIB-HIER. For a 64-port hierarchical router, as the number of ports sharing the input buffer increases, the buffer can be used more evenly (STD_BU curve) but the hardware complexity will also increase (HO curve). YARC suffers from low buffer utilization while Stash suffers from high hardware overhead. CIB-HIER attempts to make the best trade-off between the buffer utilization and the overhead by sharing unused buffers between a small number of ports.

reorganize unused buffers of multiple input ports in a tile into a shared centralized input buffer.  

Figure 2 presents the design purpose of CIB-HIER. We take the number of ports that can share the input buffer as the abscissa, and the ordinate gives the Standard Deviation (STD) of buffer utilization (STD_BU) and the simplified hardware overhead (HO) of router under the corresponding configuration. STD_BU is calculated by counting the occupancy rate of all input buffers with an evaluation performed in a 64-port hierarchical router under a uniform random traffic pattern at 30% load. HO is obtained by the simplified formula \((n - 1)/2\), where \(n\) is the number of ports that can share the input buffer. A high value of STD_BU indicates an unbalanced utilization but not necessarily a low value of utilization. However, for fixed-length buffers, unbalanced buffer utilization means that there are a lot of idle buffers, which ultimately results in low buffer utilization. Although STD_BU and HO use different metrics, the trade-off of the buffer design can be simply expressed as STD_BU+HO, which can intuitively reflect our design purpose. As shown in Figure 2, as the number of ports sharing the input buffer increases, STD_BU decreases sharply while HO increases linearly. Traditional hierarchical router (YARC) has the highest value in the STD of the buffer utilization, because it does not share any unused input buffers. Stash shares unused buffers between all the ports in a router, so it has the smallest STD but the largest hardware overhead. CIB-HIER attempts to explore the trade-off between the buffer utilization and the implementation overhead, by only allowing unused input buffers to be shared among a small number of ports. In

\[\text{This design is actually a partially shared input buffer architecture where the unused input buffer can only be shared in the tile. But within the tile, the unused input buffers are organized as a centralized input buffer.}\]
this way, packets can be transmitted to the unused buffer without going through the intermediate buffer to cause the HoL blocking. Previous work has proven that it is feasible to integrate multiple ports on the same tile [10]. CIB-HIER thus allows the unused buffers from all input ports in the same tile to be utilized collectively as a common storage resource, accessible from any port in the tile. In addition to developing new router architecture, we show that the additional storage provided by the centralized input buffer can be used to relieve the HoL blocking caused by shallow intermediate buffers. We achieve this goal by temporarily storing the blocking-caused long packets in the centralized input buffer. Evaluation result shows that our design can significantly improve the throughput of hierarchical routers with acceptable hardware overheads.

In particular, the contributions of this work include the following.

- We explain why existing unused input buffer organization method with extra VCs in intermediate buffers of the hierarchical router can degrade the performance and limit overall network throughput.
- We propose CIB-HIER, a new centralized input buffer organization for hierarchical routers that allows unused input buffer to be fully utilized without introducing extra VCs in intermediate buffers to hurt the overall performance.
- We explore to relieve the impact of HoL blocking caused by shallow intermediate buffers through the utilization of the storage provided by the centralized input buffers in CIB-HIER.

The rest of this article is organized as follows. Related work is presented in Section 2. In Section 3, the motivation is presented to explain why shallow intermediate buffers can hurt the performance of hierarchical routers. In Section 4, the detailed architecture of the new buffer organization for hierarchical high-radix routers is introduced. Evaluation result is presented in Section 5. Finally, we conclude our article in Section 6.

2 RELATED WORK AND BACKGROUND

High-radix routers have attracted renewed attention for the benefits of many slim channels per chip over few fat ones [22]. Many high-radix router microarchitecture have been proposed based on the hierarchical organization. For example, Ahn et al. proposed a buffered folded-Clos network for high-radix switches and the router is actually an on-chip network [3]. Work in Reference [22] proposed a hierarchical high-radix router architecture, which partitions the crossbar into smaller crossbar tiles and places queue memories at their inputs and outputs. Based on the architecture in Reference [22], Scott et al. proposed YARC, which is designed for the BlackWidow high-radix Clos network by Cray [29]. YARC has the same number of crosspoints but provides higher performance than flat crossbars in Reference [22], which makes it a typical high-radix router for off-chip networks.

Figure 3 shows the basic architecture of a hierarchical router with radix $P$. It consists of a two-dimensional array of identical tiles, with $R$ rows and $C$ columns, forming a large crossbar. The detailed tile architecture has also been presented in Figure 3. Each tile (represented by a grey box) contains input buffers, row buffers, column buffers, and a smaller $I \times O$ subswitch that functions as a portion of the switch crossbar. Tile-based microarchitecture of hierarchical routers makes replication and physical implementation in silicon convenient, because each tile is an identical and very regular structure. With the hierarchical architecture in Figure 3, when a packet reaches the head of the input buffer, routing computing will be performed to decide which output port this packet should be sent to. Then, this packet will be transited to the row buffer at the input port of a subswitch, which is the junction of the packet’s input row and output column. Then the packet will transit through the subswitch to the output port and buffered in the associated column.
buffer. Finally, after output arbitration, the packet will be transmitted to the next-hop router via the output port.

Various tile sizes are possible depending on engineering trade-offs such as clock frequency, switch radix, the availability of wiring tracks, and tile complexity. For example, the hierarchical router in Figure 3 can be organized into $3 \times 4$ tiles instead of $3 \times 2$ tiles. Different tile size means different number of input/output ports aggregated in a tile. Previous work designed $8 \times 8$ tiles for a 64-port hierarchical router [29], with each tile aggregated with one input port. Work in Reference [10] proposed a Multiport-Binding Tile-based Router (MBTR) microarchitecture. MBTR aggregates multiple physical ports into a single tile, to reduce the number of tiles and the number of intermediate buffers required in a hierarchical router. The architecture in Figure 3 is an example of MBTR, where two input ports and two output ports are aggregated onto the same tile.

While hierarchical router enables higher performance, it introduces many input buffers and intermediate buffers, which is costly in area and power consumption [33]. Input buffers are usually very deep, because they must be sized for the longest links. Considering modern asymmetric network topologies have a wide range of link lengths, there can be a large amount of unused input buffers. Therefore, current work strives to make full use of these unused input buffers in hierarchical routers. Typically, work in Reference [1] proposed Stash, which stashes packets in unused buffers on other ports, accessible via excess internal bandwidth in the hierarchical router. As shown in Figure 4, Stash shares unused buffer memory in stashing partition through two additional VCs in intermediate buffers. Because of the large number of intermediate buffers, the cost of increasing intermediate buffer capacity is forbidden, thus results in shallower VCs in intermediate buffers. Shallow intermediate buffers can in turn cause HoL blocking to damage the performance of hierarchical routers [24]. Some NoC router designs [16, 28] attempt to make full use of the buffer by adopting distributed shared buffer architecture or centralized buffer architecture, but they are not suitable for off-chip routers. Therefore, it is needed to design a more efficient hierarchical router architecture to make better use of the unused input buffers.

## 3 MOTIVATIONS

In this section, we show the reason why the performance of Stash can be damaged by shallow intermediate buffers. The performance of Stash can be degraded due to the HoL blocking caused by shallow intermediate buffers. Figure 5 presents a comparison of packet latency for a hierarchical router with and without Stash. The evaluation adopts two 8-flit-depth VCs in row buffers for the baseline router and four 4-flit-depth VCs in row buffers for Stash (Stash introduces two additional

---

ACM Transactions on Architecture and Code Optimization, Vol. 18, No. 4, Article 50. Publication date: July 2021.
Fig. 4. Datapath comparison. Datapath of a tile in the hierarchical router is presented in panel (a), while the datapath of a tile enabled with Stash is presented in panel (b). Stash introduces two additional VCs in intermediate buffers and results in shallower VCs.

Fig. 5. Performance comparison of a hierarchical router (HIER) with and without Stash. Transmitting long packets in the intermediate buffer can degrade the performance of Stash due to the HoL blocking. Stash needs two extra VCs in intermediate buffers, resulting in shallower intermediate buffers in case of fixed memory capacity, which further reduces the performance.

VCs, resulting in shallower VCs). 16-flit packets are adopted to evaluate the impact of long packets. Due to the HoL blocking caused by shallow intermediate buffers, Stash reduces the saturation throughput of the hierarchical router by 26.8%. Increasing intermediate buffer size can avoid the performance degradation in Stash. However, such method is too costly as the amount of intermediate buffers is proportional to $O(vp^2)$ where $p$ is the switch radix and $v$ is the number of VCs. Previous work has shown that the on-chip buffers can represent a significant portion of overall area and power consumption of the router [19, 32], which limits the design space of high-radix routers.

Shallow intermediate buffers affect the performance of Stash in two ways. On the one hand, long packets can be blocked from the input buffer to the intermediate buffer and affect the transmission of other short packets. In Stash, if a long packet has been decided to store in an unused input buffer, this packet needs to pass through the intermediate buffer. In this case, if the intermediate buffer is too shallow to buffer the whole packet, this long packet will be congested. As a result, the congested packet can block the intermediate buffer and even the input buffer to damage the router performance. On the other hand, Stash adds two VCs in the intermediate buffer to generate
shallower intermediate VCs with a certain buffer capacity, which further compromises the router performance.

4 BUFFER MICROARCHITECTURE IN CIB-HIER

In this section, we first demonstrate that it is feasible to share unused input buffers between a small number of ports. Then, the detailed architecture of CIB-HIER is presented, including the implementation for different kinds of CIB-HIER. Finally, the utilization of CIB-HIER is presented, where the HoL blocking caused by shallow intermediate buffers can be relieved.

4.1 Share Unused Input Buffers in a Tile

Previous work [9, 10] has proposed MBTR and it has been proved that it is feasible to combine multiple input ports into a single tile in hierarchical routers. MBTR reduces the required number of intermediate buffers while achieving a comparable performance compared with the single-port-binding architecture [29]. For hierarchical routers, the number of ports integrated in a tile determines the radix of the subswitch. Higher radix in the subswitch means more input ports integrated in a tile, which, however, also means more complex circuit logic in the tile. Considering the trade-off between the performance and the implementation overhead, actual router implementation usually only integrates no more than four ports in a tile [10]. Therefore, to clearly introduce our design, a two-port-binding MBTR as shown in Figure 3 is adopted as the baseline of our design in the following sections.

In MBTR, there still exists large amount of unused input buffers in the tile. Recent HPC systems have been designed with sufficient resources to satisfy the demand of various configurations. For example, Intel’s Omni-Path router [8] has 48 ports to support links up to 100 m at 100 Gbps. When such a router is used to implement a large-scale dragonfly network, 25% ports (endpoint-linked ports) will experience a 99% of underutilized input buffers and 50% ports (intra-group-linked ports) will experience a 95% underutilized input buffers. Similarly, when Omni-Path router is adopted to implement leaf routers in a multi-level fat-tree, 50% ports (endpoint-linked ports) will experience a 99% underutilized input buffers. Because more than half of the ports in these hierarchical routers have a large amount of unused buffers, we can assign each tile with one or more input ports that contain unused input buffers. With this design, there will be a large number of unused buffers in each tile.

4.2 Centralized Input Buffer Architecture

CIB-HIER introduces some modifications to the baseline hierarchical router whose datapath has been presented in Figure 4(a), and these changes only appear inside the tile. Figure 6 details the architecture of a tile in CIB-HIER. As shown in Figure 6(a), two input ports and the corresponding input buffers have been integrated in the tile. The input buffer is divided into two banks: the private buffer bank and the shared buffer bank. The private buffer is used in the same way as the traditional input buffers except that its capacity is reduced, while all shared buffers (namely, the Odd bank and Even bank in this case) in the tile are formed as a Centralized Input Buffer (CIB). The capacity of the private buffer is set to the same on each port, and this capacity only needs to meet the requirement of the shortest link (the link that is connected with the endpoint node). Obviously, this design can lead to insufficient input buffer for the input port that is connected with longer link. In this case, we can re-partition a portion of the CIB to this port as a private buffer so that there is enough input buffer for this input port. Based on this design, the tile architecture in CIB-HIER is logically organized as shown in Figure 6(b), where each port is assigned a reduced private input buffer with the same capacity, while the rest of the input buffer is organized as the CIB, which can be shared by all the input ports in the tile.
A design challenge of CIB-HIER is how to access the CIB. As shown in Figure 6(a), a multiplexer is added at the outport of the private input buffer, to decide whether to send a flit to the CIB or directly to the row bus. For an input port, if part of the private input buffer is in the CIB, then flits sent to this input port can be first stored in the private input buffer at the input port and then be transferred to the corresponding private input buffer in the CIB if there is enough buffer space. Note that the transfer process is proactive, that is, once the out port of the private input buffer at the input port is idle and the corresponding private input buffer in the CIB is not full, the flits in the private input buffer in the input port can be transferred to the private input buffer in the CIB automatically. The granularity of this transfer process is packets, thus no out-of-order flits can be introduced inside the packet. We also add a data-path from the private input buffer at the input port directly to the row bus, to allow uncongested packets to be directly sent to the row bus. With such design, packets at the head of VCs in the private input buffer at the input port can be sent to the row buffer directly if the corresponding row buffer has empty space, or be buffered in the CIB if the corresponding row buffer is congested. CIB-HIER can use the capacity provided by CIB to buffer congested packets, thus alleviating the impact of the HoL blocking on router performance.

Another design challenge is how to meet the needs of multiple ports to access CIB at the same time. Fortunately, the odd/even interleaved banks architecture in Stash can solve this problem [1], so we adopted this structure in our design. In our design, any input port in the tile can write data to the CIB at any time. Therefore, write conflicts and read conflicts can emerge in the CIB.
Fig. 7. Storage management in a tile of the CIB-HIER. A portion of the CIB can be used as part of the input buffer for an input port (Input 0 in this case), while the rest is organized as DAMQ and can be used as a shared buffer for all ports in the tile. Each VC in input ports will be associated with a VC in the shared part of CIB to cache packets.

If only one write/read port is set for each buffer bank in the CIB. A four-ported buffer bank with two write ports and two read ports, or dual-ported bank operating at twice the normal speed can solve this problem, but the design overheads make it infeasible to be implemented in a tile. A better solution is illustrated in Figure 6(a), where the two banks in the CIB act as odd/even banks. Each bank can store flits, one per cycle, with all even numbered offsets in the Even bank and odd in the Odd bank. Therefore, the write operation and read operation for a multi-flit packet can be interleaved between the two banks. The multiple interleaved bank method allows the CIB to be divided into two partitions at a two-flit granularity. Two Xbars have been added to control write/read operations to the CIB. For each packet to be written to or read from the CIB, there will be a 50% probability of a one-cycle write/read latency, because two input ports may write to or read from the same bank in the CIB in a cycle. For traditional HPC systems that adopt high-radix routers, there will be a latency overhead of a few cycles for a packet transmitting from the source node to the destination node. However, this latency is negligible, because the shortest packet latency in HPC systems is generally greater than 100 cycles.

CIB-HIER, like other general-purpose hierarchical routers, also faces the challenge of providing many VCs for deadlock avoidance and multiple traffic classes. To support this target, the storage management strategy in the CIB has been carefully designed and the detailed architecture has been presented in Figure 7. As shown in the figure, Dynamically Allocated Multi-queue (DAMQ) [31] is adopted in the shared part of the CIB, so that buffers can be dynamically allocated to each VC. Each VC in input buffers is associated with a VC in the shared part of the CIB. If a packet in the input buffer has been decided to be buffered in the CIB, then it will be transited to and buffered in the corresponding VC in the CIB. It should be noted that the private part of the CIB is organized as fixed-length VCs. Such design allows congestion information to be transmitted to the source node as soon as possible, reducing the impact of endpoint congestion on the overall network performance [17]. To maintain the order of flits in the packet, data is written to the CIB at the granularity of the packet. It should be noted that if the number of VCs contained in a router port is large, the scalability of CIB-HIER will be affected. However, more VCs can well alleviate the congestion inside the router caused by the HoL blocking, which is consistent with our design goals. In other words, if the port has a large number of VCs, we do not need to adopt the CIB-HIER storage architecture.

The credit management of CIB-HIER has been carefully designed to avoid the buffer overflow. In CIB-HIER, the number of credits that can be used by upstream routers is related to the capacity.
of the input buffer in the current router. However, the capacity of the input buffer varies among router ports, which makes the number of credits that each port can provide to the upstream router is also different. As shown in Figure 7, some ports (e.g., port 1) store all the input buffers in the private bank, while other ports (e.g., port 0) have only part of the input buffer in the private bank and the rest in the CIB. For ports where the input buffer is in both the private bank and the CIB, the credit received by the upstream router includes the credit in the private buffer and the CIB. Because packets can be actively moved from the private bank to the input buffer in the CIB, buffer overflow can be avoided in the input buffer. For ports whose input buffers are all in the private bank, the available credit is equal to the buffer capacity of the private bank. Note that the shared part of the CIB is not available for upstream routers but packets can be buffered in the shared part of the CIB when the corresponding VC in the shared part of the CIB is not full. Such design can avoid long packets to be congested in the shallow private bank. However, the transfer of packets from the input buffer to the shared buffer in CIB is based on the packet granularity, which may affect the utilization of the shared buffer. Fortunately, this situation only occurs when the corresponding VC in the shared buffer is not empty, and because of the utilization of DAMQ, there are actually not many free buffers. In addition, this design can make the backpressure caused by congestion be delivered to the source node faster and alleviate the impact of congestion on network performance.

4.3 Scalability of the Centralized Input Buffer

Our design can be adapted to other hierarchical high-radix routers, since it is not dependent on a particular hierarchical router microarchitecture. CIB-HIER only changes the architecture inside the tile without significantly affecting the overall structure of the hierarchical router. Therefore, when extending our structure to other hierarchical routers, CIB-HIER only needs to modify the structure inside the tile.

A typical example is to implement CIB in a hierarchical router with single-port-binding tiles. Figure 8 shows the detailed tile architecture of the CIB in this kind of hierarchical routers. As shown in Figure 8, the input buffer is divided into two banks, one for the CIB and the other for the private input buffer. With this design, CIB provides an opportunity to enhance other network features, and there are a wide variety of network features that could exploit the additional storage resources provided by the CIB.

Another design challenge that needs to be addressed is how to extend the CIB into a tile binded with more input ports. In this case, the structure of the CIB is similar to that in Figure 6. We only need to increase the number of banks in the CIB, and it should be equal to the number of input ports on a tile. In addition, each input port also needs to be assigned with a private input buffer. For example, for a tile with four input ports, there will be four banks in the CIB, and these banks can be accessed by different input ports in an interleaved fashion through a $4 \times 4$ Xbar. It is possible to extend CIB-HIER to tiles integrated with more ports, but the design overheads will be higher. One obvious overhead is that a larger Xbar should be added to handle the conflict between different ports when accessing the CIB at the same time. Besides, integrating too many input ports on a tile also increases the waiting time for each port to access the CIB, thereby increasing the transmission latency of the packet. Fortunately, practical hierarchical routers do not integrate too many ports in a tile [9]. This is because the more ports integrated in a tile, the higher radix in the subswitch in hierarchical routers. The high-radix subswitch increases the design complexity and degrades the scalability of the hierarchical router. For example, for a router with 64 input/output ports, if there are 1, 4, and 16 input ports integrated in a tile, the subswitch should be designed as $8 \times 8$, $16 \times 16$, and $32 \times 32$ Xbars, respectively [10]. Therefore, as it is not practical to integrate too many input ports in a tile, there is no need to adopt our design in the hierarchical router that is integrated
with too many ports in a tile. For this reason, although the area of the Xbar will increase in the quadratic trend of the number of input ports in the tile, the area overhead of CIB-HIER is very small, because a tile will not integrate too many input ports.

4.4 Utilization of the Centralized Input Buffer

The design of CIB provides a possibility to enhance a wide variety of network features, such as end-to-end retransmission and improving existing congestion control protocols. However, CIB-HIER can make full use of the CIB by providing benefits relative to router designs for lower-radix topology including other hierarchical topologies [4, 13, 18] and other high-radix design [6]. In this work, we exploit use cases of our design to demonstrate how CIB can be integrated into network designs and the effect it has on performance. In particular, we focus on a typical use case where the HoL blocking caused by shallow intermediate buffers can be relieved by utilizing the buffer capacity provided by CIB.

As has been presented in Section 3, traditional hierarchical high-radix routers require plenty of intermediate buffers, but these intermediate buffers are always designed shallow [10, 29] to cause performance bottleneck. Increasing intermediate buffer size overcomes this problem but is infeasible due to hardware overheads. CIB-HIER can be used to relieve this kind of congestion by storing congested packets in the CIB. It should be noted here that Stash cannot alleviate congestion caused by shallow intermediate buffers in this way, because congested packets will cause new congestion when they pass through the intermediate buffer. Moreover, intermediate buffers in Stash are even more shallow to generate more serious performance degradation.

To relieve the impact of the HoL blocking caused by shallow intermediate buffers, CIB-HIER stores congested packets in the CIB, and allows other packets to be transited to other congestion-free subswitches rather than waiting in the input buffer. We choose packet-level transmission rather than flit-level transmission for this process. Flit-level transmission can minimize the impact of the HoL blocking in input buffers. With this strategy, a packet can be buffered to the CIB once it is blocked in the input buffer, even if part of this packet has been transferred to the row.
buffer. However, such strategy can introduce out-of-order flits in CIB and row buffers. Conservative VC allocation strategy [27] can be used to avoid the problem of out-of-order flits but will bring about performance degradation due to low buffer utilization. A better solution is adopting packet-level transmission. With this strategy, a packet must be transmitted to the CIB as a whole. This strategy does not introduce out-of-order flits in a packet while relieving the impact of HoL blocking.

A disadvantage of packet-level transmission in CIB-HIER is that transferring data in the packet granularity instead of flit can lead to underutilization buffer in the CIB. In theory, if flit-level transmission is adopted, CIB can be fully utilized. This is because whenever there is free space in the CIB, we can use the free buffer space by scheduling the flits in the reduced input buffer to the CIB. However, packet-level transmission can degrade the buffer utilization in the CIB. If the head flit of a packet in the reduced input buffer has been transmitted, then the subsequent flits of this packet cannot be stored in the corresponding VC of the CIB. For this reason, if a packet is blocked in the reduced input buffer, then the buffer utilization rate of the corresponding VC in the CIB can be reduced, and in the worst case, the utilization rate of this VC is 0. Fortunately, as we adopt DAMQ in the CIB, only the private part of a VC can be influenced by the low utilization, and the shared part can be utilized by other VCs. Therefore, in the worst case, the buffer utilization of the CIB is $1 - (N \times p)/(S + N \times v \times p)$, where $N$ is the number of input ports in a tile, $v$ is the number of VCs in the input buffer, $p$ is the length of the private buffer in a VC in the CIB, and $S$ is the total space of the shared buffer in the CIB. In this way, if $N = 2$, $v = 2$, $p = 16$, and $S = 336$ (we adopt 4/5 input buffers as the CIB), then the utilization rate of CIB is 92% in the worst case. This shows that the packet-level transmission will not have a great impact on the buffer utilization of the CIB.

Another disadvantage of packet-level transmission in CIB-HIER is that a packet cannot be immediately transferred to the CIB once it is blocked. Therefore, CIB-HIER+ is proposed to alleviate the impact of this disadvantage and fully utilize the buffer capacity in CIB. CIB-HIER+ is designed based on CIB-HIER but adopting an initiative packet transmission strategy that transfers packets to the CIB predictively. In other words, once a packet arrives at the head of VCs in the input buffer, CIB-HIER+ will decide to store this packet in the CIB or send it directly to the row buffer, regardless of whether it is blocked or not. The design challenge of CIB-HIER+ is to determine which packets should be stored in the CIB when the packets in the input buffer are not congested. In hierarchical routers, most HoL blocking caused by shallow row buffers is related with long packets. Moreover, once shallow intermediate buffers cause HoL blocking in input buffers, long packets can stress this effect by passing the congestion state to the intermediate buffer. Therefore, CIB-HIER+ adopts a simpler and more practical strategy, where long packets are initiatively stored in the CIB to reduce the impact of the congestion. With this strategy, packets arriving at the head of a VC in the input buffer will be judged to see if it is a long packet, if yes and the outport of input buffer is busy (namely, there are other packets in the input buffer transmitting to the row buffer), this long packet will be decided to be sent to the CIB. It should be noted that packets sent to the row buffer have higher priority than the packets sent to the CIB. Therefore, the outport of input buffer can only use idle cycles to send packets to the CIB, which relieves the performance degradation caused by packet transmission to the CIB. In addition, the principle to decide which packet is a long packet is related to the actual systems and will be different from different systems.

It should be pointed out that if there are too many long packets, the performance of CIB-HIER+ can be decreased. The performance advantage of CIB-HIER+ comes from storing congested long packets in the CIB to avoid the impact of HoL blocking on other short packets. But the performance of CIB-HIER+ may be degraded if all arriving packets are long packets. In this case, CIB-HIER+ can
only randomly select packets and dispatch them to the CIB, because the probability of congestion for each packet is similar. In this way, the performance of CIB-HIER+ will be similar to CIB-HIER. Because long packets will also cause congestion inside the router, it can reduce the performance of CIB-HIER to a certain extent as well. Compared with CIB-HIER, long packets traffic reduces the performance of Stash more significantly. This is because, on the one hand, long packets need to pass through the intermediate buffer when transferring between the idle buffers of different ports, which cause congestion inside the router. On the other hand, the additional intermediate VCs of Stash will reduce the depth of the intermediate buffer and further aggravate the congestion inside the router.

5 EVALUATION

5.1 Methodology

We use a modified version of Booksim [11], a cycle-accurate interconnection network simulator, to evaluate the performance of CIB-HIER. Dsent [30] is adopted to evaluate area and power consumption. Evaluations are performed in Flattened Butterfly (FBFly) [20] and Fat Tree (FT) [11] networks. We use iSLIP allocators [11] for both switch allocation and VC allocation and all separable allocators perform input arbitration before output arbitration. Each data of our evaluations is obtained after more than 30,000 cycles of simulation.

A MBTR-like hierarchical router (HIER) is adopted as the baseline of evaluations. Baseline HIER has 64 input ports, and each input port is associated with an input channel and an input buffer. In HIER, each row buffer contains 2 VCs with 16-flit capacity while the column buffer contains 2 VCs with 10-flit capacity. The input buffer has 2 VCs with 128-flit capacity. In HIER, multiport binding tile-based router microarchitecture [10] is adopted to aggregate four input/output ports in the same tile. Therefore, the subswitch in each tile is actually a 16 × 16 Xbar.

A two-dimension FBFly with 256 routers and 8,704 nodes is used in our evaluations. Every router in FBFly has 64 input/output ports, among them 34 ports are connected with terminals and 30 ports connected with other routers. In FBFly, we use minimum random routing for routing computation [11]. We also use a two-level FT network with 1,024 nodes, and each level in FT has 32 routers. In FT, router in the top-level has 32 ports with each port connected with a router in the second level. Second-level router has 64 ports, among which 32 ports are connected with top-level routers and the other 32 ports are connected with terminal nodes. To model long global channels in FT and FBFly, we assume 50-cycle channel latency for links between routers and 10-cycle latency for links connected with terminal nodes. We assume 50-cycle latency for long links, because the HPC system usually uses optical fibers for long links. Each fiber needs two SerDeses (serializer/deserializer) to complete the conversion of photoelectric signals and the delay of SerDes is relatively large.

In the CIB-HIER, a portion of the input buffer is partitioned for CIB while leaving the remaining storage to completely satisfy the normal demand of the input port (that is, to be served as the private input buffer in our design). We conservatively partition the buffers with 4/5 of the storage configured for CIB on all input ports, and the private input buffer only contains 1/5 of the storage. For Stash, we partition 4/5 of buffers for stashing part on endpoint ports, and none on the ports that connected with other routers. We restrict the amount of stashing storage is fully available (namely, 100% Capacity Stash), because such design represent the best performance of Stash. We assume a fixed budget in intermediate buffers both for CIB-HIER and Stash. Therefore, Stash has 4 VCs in intermediate buffers (row buffer and column buffer) while CIB-HIER has only 2 VCs. The depth of intermediate VC for Stash is designed to be only half that of the CIB-Router.\footnote{Evaluations for Stash with the same VC depth of CIB-HIER in intermediate buffers but result trends were similar to the baseline HIER and is not included in this article.}

ACM Transactions on Architecture and Code Optimization, Vol. 18, No. 4, Article 50. Publication date: July 2021.
Fig. 9. Latency evaluations for CIB-HIER in the Fat Tree (FT) network using 16-flit packets.

We use synthesis traffic such as uniform random, transpose, badperm [11] and “mix” to evaluate our design. Badperm traffic pattern sends packets from the same row of the hierarchical router to a specific column. Mix traffic pattern is actually a mix of uniform traffic pattern and hotspot traffic pattern. For hotspot traffic, 10% nodes are randomly selected as hotspot receivers that accept traffic from other nodes while other nodes have 5% probability to send traffic to hotspot nodes and 95% probability to send traffic to a randomly selected node. 4-flit, 16-flit and 32-flit packets are used to simulate actual packets. To test the performance of CIB-HIER+, we perform the evaluation with 1/8 32-flit packets and 7/8 4-flit packets, and specify 32-flit packets as long packets to be allowed to stored in CIB. We perform evaluation of power and area for CIB-HIER at 22 nm technology scaling with 1.0V operating voltage. Operating frequency is set to 1.0 GHz. Data width of the link and numbers of bit per flit are all set to 64 bits.

For real workload evaluation, we integrated the modified Booksim with a commodity-level simulator and use the trace of NAS Parallel Benchmarks (NPB) [5]. The NAS benchmarks are derived from computational fluid dynamics code and have gained wide acceptance as a standard indicator of supercomputer performance [26]. In the evaluation, we use the trace of the following NPB workloads: IS (integer sort), EP (embarrassingly parallel), MG (geometric multigrid v-cycle from production elliptic solver), CG (conjugate gradient), and FT (fast Fourier transform) [26]. We adopt the 1,024-node Fat Tree network mentioned above. To evaluate the impact of router congestion on real workloads, we have half of the network nodes inject uniform traffic with each other at 50% injection rate. The network is warmed up for 30,000 cycles with uniform traffic only, and then workload trace packets were injected.

5.2 Evaluation Result

We first evaluate the performance of CIB-HIER and compared it with existing hierarchical routers in different networks. Results in FT network are presented in Figure 9. As can be seen in the figure, CIB-HIER outperforms both HIER and Stash. Compared with HIER, CIB-HIER can achieve 32.3%,
24.1%, 10.6%, and 11.5% improvement in saturation throughput under uniform, transpose, mix, and badperm traffic patterns, respectively. Compared with HIER, CIB-HIER+ can achieve 40.6%, 29.8%, 14.2%, and 13.8% improvement in saturation throughput under uniform, transpose, mix, and badperm traffic patterns, respectively. The performance improvement of CIB-HIER comes from the buffer space provided by the CIB. With CIB, congested packets can be buffered in the CIB when private input buffer is blocked. Moreover, if the private input buffer is congested, packets buffered in the CIB can be sent to the row bus, which also relieves the impact of the HoL blocking in input buffers to achieve higher performance. In Figure 9, the performance of Stash is much lower than that of HIER and CIB-HIER. Compared with HIER, Stash reduces the saturation throughput by 23.1%, 28.6%, 2.4%, and 12.2% under uniform, transpose, mix, and badperm traffic patterns, respectively. The reason for performance degradation in Stash is that the shallow intermediate buffers stress the impact of HoL blocking.

We further evaluate the performance of our design in FBFly network using 16-flit packets, and the result is presented in Figure 10. As Figure 10 shows, CIB-HIER achieves 30.1%, 17.3%, 8.5%, and 6.2% improvement in saturation throughput under uniform, transpose mix and badperm traffic patterns, respectively. CIB-HIER+ achieves 38.6%, 25.1%, 9.7%, and 17.9% improvement in saturation throughput under uniform, transpose mix and badperm traffic patterns, respectively. Stash reduces the saturation throughput by 17.8%, 9.4%, 6.9%, and 16.3%, respectively. The maximum throughput that the FBFly network can achieve is significantly lower than that of the FT network. This is because the size of the FBFly network is significantly larger than the FT network (×8 in terminal nodes). In the FBFly, performance improvement achieved by CIB-HIER is also smaller than that in the FT. This is because the transmission path of the packet in the FBFly network is shorter than in FT network, which reduces the impact of network congestion, resulting in less performance improvement in CIB-HIER.

CIB-HIER provides a new architecture to utilize the unused input buffer. CIB-HIER+ predictively utilize the space provided by the CIB to relieve the impact of the HoL blocking caused by shallow
intermediate buffers. As CIB-HIER+ can store long packets in the CIB to avoid the blocking from the row buffer to the input buffer, it can achieve better performance compared with CIB-HIER and Stash. Figure 11 present the evaluation result in the FT network using a traffic load consisted of 12.5% long packets and 87.5% short packets. As shown in Figure 11, CIB-HIER+ outperforms CIB-HIER in saturation throughput. Compared with CIB-HIER, CIB-HIER+ can improve the saturation throughput by 7.6%, 6.3%, 5.3%, and 5.6% under uniform, transpose mix and badperm traffic patterns, respectively.

Maximum throughput is measured and the results are presented in Figure 12. As can be seen from the figures, CIB-HIER outperforms Stash and HIER under each of four traffic patterns, and CIB-HIER+ achieves even better performance than CIB-HIER. CIB-HIER transmits the congested packet to the CIB, which reduces the latency of waiting packets in a router and increases the maximum throughput as a result. Under each traffic pattern, CIB-HIER+ achieves much lower maximum throughput; this is because long packets can be speculatively stored to the CIB to avoid the impact of the HoL blocking caused by long packets.
CIB-HIER provides buffer space by designing CIB. CIB can be used to store packets that cause congestion in intermediate buffers and input buffers. The packets buffered in the CIB will not affect the latency distributions while achieving lower average packet latency. We evaluate the latency distribution of CIB-HIER+ and present the result in Figure 13. The evaluations are performed in the FT network with the saturation injection rate using uniform traffic pattern. The traffic load consists of 1/8 32-flit packets and 7/8 4-flit packets. As shown in Figure 13, the proportion of packets with lower latency is much larger in CIB-HIER+ than in HIER and Stash. Moreover, CIB-HIER+ does not introduce significant tail latency although with additional buffer space provided by CIB.

CIB-HIER+ speculatively store packets that can cause blocking to the CIB to relieve the impact of HoL blocking. CIB-HIER+ can achieve more significant performance improvement when the network experiences a severe congestion, especially when faced with a hotspot traffic pattern. Figure 14 compares the transient behavior of CIB-HIER+ to Stash in average packet latency. In this evaluation, background load is a uniform random traffic pattern with an injection rate of 40%, and the hotspot traffic continues to inject 20 cycles from the 2,000th cycle with an injection rate of 100%. The transient hotspot traffic not only causes congestion at hotspot end-nodes but also makes packets blocked at intermediate routers. For these congested packets, both CIB-HIER+ and Stash provide space to temporarily buffer these packets. However, the shallow intermediate buffers introduced by Stash can cause more packets congested in the intermediate buffer, and attenuates the mitigation effect of stashing process on network congestion. Moreover, as packets in Stash can be buffered in the stashing portion in any tile, the waiting latency of these packets has also been increased. As shown in Figure 14, CIB-HIER+ can react to the transient traffic faster than Stash. Stash experiences a high packet latency for a long period of time after the end of transient traffic.
In addition to synthetic traffic patterns, we also evaluate our design with NPB workloads. Execution time of NPB is presented in Figure 15 and the result has been normalized to that of HIER. As shown in Figure 15, while Stash can damage the performance of the baseline HIER, CIB-HIER and CIB-HIER+ achieve better performance for all benchmarks. For CIB-HIER+, the average reduction in execution time is 12.6%, while the maximum reduction is 16.7% with CQ benchmark. The performance of Stash, CIB-HIER and CIB-HIER+ under different benchmarks has a relationship with the traffic pattern and the length of the packet. CG has a large number of collective communications and contains many long packets, while EP has many embarrassingly parallel operations, which can cause congestion inside the hierarchical router. Stash introduces additional VCs in the intermediate buffer to cause shallow VCs in the intermediate buffer, which intensifies the impact of congestion on router performance. However, CIB-HIER and CIB-HIER+ does not introduce additional VCs, which avoids the impact of shallow intermediate buffers on router performance. Moreover, CIB-HIER+ can temporarily store congested packets in the CIB to further improve router performance. As a result, CIB-HIER+ achieves more performance improvements under all benchmarks, especially EP and CG. Compared with HIER, the performance of Stash is reduced in all benchmarks, and the performance degradation is more obvious under MG and CG. This is because Stash adopts additional intermediate VCs, which reduces the depth of the intermediate buffer to hurt the router performance as a result.

We evaluate the power consumption of CIB-HIER, CIB-HIER+ and the results are shown in Figure 16. In our evaluation, we set the injection rate to be the same as average injection rate of NPB benchmarks. Figure 16 shows the power consumption of a router for CIB-HIER, CIB-HIER+, Stash, and HIER, respectively. The evaluation is conducted in three different configurations of hierarchical routers, each of which integrated 1, 2, and 4 ports in a tile, respectively. In total, binding more ports in a tile means less power consumption. CIB-HIER+, CIB-HIER, and Stash all introduce power overhead but CIB-HIER+ and CIB-HIER experience less increase in power consumption than Stash. For four-port-binding tile-based architecture, CIB-HIER, CIB-HIER+, and Stash can increase the power consumption by 1.22%, 1.45%, and 3.68%, respectively.
As a resource-constrained fabric, router design often focuses on communication performance and area cost. CIB-HIER only adds some Xbars and the corresponding logic circuits to input ports, so that the resource addition is marginal. Figure 17 shows the area usage results. Since CIB-HIER only changes the architecture inside a tile, the area usage is smaller than Stash. For four-port-binding tile-based architecture, CIB-HIER, CIB-HIER+, and Stash can increase the area by 1.25%, 1.63%, and 3.75%, respectively.

6 CONCLUSION
High-radix hierarchical router is usually designed with symmetric ports to result in a significant amount of unused input buffer. Previous work shared unused buffers between all ports by adding two internal virtual channels. However, such design can introduce and suffer from shallow intermediate buffers because of the HoL blocking. This article proposes CIB-HIER, a new architecture to share the unused input buffer between a few ports that can be integrated in the same tile of hierarchical routers. CIB-HIER reorganizes unused input buffer in the same tile as a centralized input buffer, which can be utilized by all ports in this tile. The space provided by the centralized input buffer can be further explored to relieve the performance degradation caused by shallow intermediate buffers. CIB-HIER is efficient in sharing unused input buffers without introducing additional internal VCs. Evaluation result shows that CIB-HIER can achieve much better performance than existing method and the hardware overhead is moderate.

ACKNOWLEDGMENTS
The authors thank the anonymous reviewers for their valuable feedback. We gratefully acknowledge members of Tianhe interconnect group at NUDT for many inspiring conversations.

REFERENCES
[1] Matthias A. Blumrich, Nan Jiang, and Larry R. Dennison. 2018. Exploiting idle resources in a high-radix switch for supplemental storage. In Proceedings of the IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC’18). 1–13.
[2] Jung Ho Ahn, Nathan Binkert, Al Davis, Moray McLaren, and Robert S. Schreiber. 2009. HyperX: Topology, routing, and packaging of efficient large-scale networks. In Proceedings of The ACM International Conference for High Performance Computing, Networking, Storage and Analysis (SC’09). 41:1–41:11.
[3] Jung Ho Ahn, Sungwoo Choo, and John Kim. 2012. Network within a network approach to create a scalable high-radix router microarchitecture. In Proceedings of the 18th IEEE International Symposium on High Performance Computer Architecture (HPCA’12). 1–12.
[4] Rachata Ausavarungnirun, Chris Fallin, Xiangyao Yu, Kevin Kai-Wei Chang, Greg Nazario, Reetuparna Das, Gabriel H. Loh, and Onur Mutlu. 2016. A case for hierarchical rings with deflection routing: An energy-efficient on-chip communication substrate. Parallel Comput. 54 (2016), 29–45.
[5] D. H. Bailey, E. Barszcz, H. D. Simon, V. Venkatkrishnan, S. K. Weeratunga, J. T. Barton, D. S. Browning, R. L. Carter, L. Dagum, and R. A. Fatoohi. 1991. The NAS parallel benchmarks. In Proceedings of the ACM/IEEE International Conference on Supercomputing. 158–165.
[6] Maciej Besta, Syed Minhaj Hassan, Sudhakar Yalamanchili, Rachata Ausavarungnirun, Onur Mutlu, and Torsten Hoefler. 2018. Slim noc: A low-diameter on-chip network topology for high energy efficiency and scalability. ACM SIGPLAN Notices 53, 2 (2018), 43–55.

[7] Maciej Besta and Torsten Hoefler. 2014. Slim Fly: A cost effective low-diameter network topology. In Proceedings of the IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC’14). 348–359.

[8] Mark S. Birrittella, Mark Debbage, Ram Huggahalli, James Kunz, Tom Lovett, Todd Rimmer, Keith D. Underwood, and Robert C. Zak. 2015. Intel omni-path architecture: Enabling scalable, high performance fabrics. In Proceedings of the 23rd IEEE Annual Symposium on High Performance Interconnects (HOTI’15). 1–9.

[9] Yi Dai, Kai Lu, Liquan Xiao, and Jinshu Su. 2019. A cost-efficient router architecture for HPC interconnection networks: Design and implementation. IEEE Trans. Parallel Distrib. Syst. 30, 4 (2019), 738–753.

[10] Yi Dai, Kefei Wang, Qu Gang, Liquan Xiao, Dezun Dong, and Xingyun Qi. 2017. A scalable and resilient microarchitecture based on multiprot binding for high-radix router design. In Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS’17). 8–17.

[11] William James Dally and Brian Patrick Towles. 2004. Principles and Practices of Interconnection Networks. Elsevier.

[12] Jack Dongarra. 2016. Report on the sunway taihuulift system. University of Tennessee, Department of Electrical and Computer Science, Tech. Rep. UT-EECS-16-742 (Jun 2016), 1–24.

[13] Chris Fallin, Greg Nazario, Xiangyao Yu, Kevin Chang, Rachata Ausavarungnirun, and Onur Mutlu. 2012. MinBD: Minimally-buffered deflection routing for energy-efficient interconnect. In Proceedings of the IEEE/ACM 6th International Symposium on Networks-on-Chip. 1–10.

[14] Haohuan Fu, Junfeng Liao, Jizhe Yang, Lanning Wang, Zhenya Song, Xiaomeng Huang, Chao Yang, Wei Xue, Fangfang Liu, and Fangli Qiao. 2016. The sunway taihuulift supercomputer: System and applications. Sci. China Info. Sci. 59, 7 (2016), 1–16.

[15] Ikki Fujiwara, Michihiro Koibuchi, Hiroki Matsutani, and Henri Casanova. 2014. Skywalk: A topology for HPC networks with low-delay switches. In Proceedings of the 28th IEEE International Symposium on Parallel and Distributed Processing. 263–272.

[16] Syed Minhaj Hassan and Sudhakar Yalamanchili. 2013. Centralized buffer router: A low latency, low power router for high radix noc. In Proceedings of the 7th IEEE/ACM International Symposium on Networks-on-Chip (NoCS’13). 1–8.

[17] Gwangsun Kim, Changhyun Kim, Jiyoung Jeong, Mike Parker, and John Kim. 2016. Contention-based congestion management in large-scale networks. In Proceedings of the 49th IEEE/ACM International Symposium on Microarchitecture (MICRO’16). 1–13.

[18] Hanjoon Kim, Gwangsun Kim, Seungryoul Maeng, Hwasoo Yeo, and John Kim. 2014. Transportation-network-inspired network-on-chip. In Proceedings of the IEEE 20th International Symposium on High Performance Computer Architecture (HPCA’14). 332–343.

[19] John Kim, William J. Dally, and Dennis Abts. 2006. Adaptive routing in high-radix clos network. In Proceedings of the IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC’06). 7–17.

[20] John Kim, William J. Dally, and Dennis Abts. 2007. Flattened butterfly: A cost-efficient topology for high-radix networks. In Proceedings of the ACM SIGARCH Computer Architecture News, Vol. 35. 126–137.

[21] John Kim, William J. Dally, Steve Scott, and Dennis Abts. 2008. Technology-driven, highly-scalable dragonfly topology. In Proceedings of the ACM SIGARCH Computer Architecture News, Vol. 36. IEEE Computer Society, 77–88.

[22] John Kim, William J. Dally, Brian Towles, and Amit K. Gupta. 2005. Microarchitecture of a high radix router. In Proceedings of the 32th ACM/IEEE Annual International Symposium on Computer Architecture (ISCA’05). 420–431.

[23] Fei Lei, Dezun Dong, Xiangke Liao, and Jose Duato. 2020. Bundlefly: A low-diameter topology for multicore fiber. In Proceedings of the ACM/IEEE International Conference on Supercomputing. 1–11.

[24] Cunlu Li, Dezun Dong, Xiangke Liao, John Kim, and Changhyun Kim. 2019. DeepHiR : Improving high-radix router throughput with deep hybrid memory buffer microarchitecture. In Proceedings of the ACM/IEEE International Conference on Supercomputing. 403–413.

[25] Xiang-Ke Liao, Zheng-Bin Pang, Ke-Fei Wang, Yu-Tong Lu, Min Xie, Jun Xia, De-Zun Dong, and Guang Suo. 2015. High performance interconnect network for Tianhe system. J. Comput. Sci. Technol. 30, 2 (2015), 259–272.

[26] Qingda Lu, Jiesheng Wu, Dhabaleswar Panda, and P. Sadayappan. 2004. Applying MPI derived datatypes to the NAS benchmarks: A case study. In Proceedings of the International Conference on Parallel Processing Workshops. 1–9.

[27] Sheng Ma, Natalie Enright Jerger, and Zhiying Wang. 2012. Whole packet forwarding: Efficient design of fully adaptive routing algorithms for networks-on-chip. In Proceedings of the 18th IEEE International Symposium on High Performance Computer Architecture (HPCA’12). 1–12.

[28] Rohit Sunkam Ramanujam, Vassos Soteriou, Bill Lin, and Li-Shiuan Peh. 2010. Design of a high-throughput distributed shared-buffer NoC router. In Proceedings of the 4th ACM/IEEE International Symposium on Networks-on-Chip. 69–78.

[29] Steve Scott, Dennis Abts, John Kim, and William J. Dally. 2006. The blackwidow high-radix clos network. ACM SIGARCH Comput. Archit. News 34, 2 (2006), 16–28.
[30] Chen Sun, Chia-Hsin Owen Chen, George Kurian, Lan Wei, Jason Miller, Anant Agarwal, Li-Shiuan Peh, and Vladimir Stojanovic. 2012. DSENT—A tool connecting emerging photonics with electronics for opto-electronic networks-on-chip modeling. In Proceedings of the 6th IEEE/ACM International Symposium on Networks on Chip. 201–210.

[31] Yuval Tamir and Gregory L. Frazier. 1992. Dynamically-allocated multi-queue buffers for VLSI communication switches. IEEE Trans. Comput. 41, 6 (1992), 725–737.

[32] David Zats, Tathagata Das, Prashanth Mohan, Dhruba Borthakur, and Randy Katz. 2012. DeTail: Reducing the flow completion time tail in datacenter networks. ACM SIGCOMM Comput. Commun. Rev. 42, 4 (2012), 139–150.

[33] Jia Zhan, Jin Ouyang, Fen Ge, Jishen Zhao, and Yuan Xie. 2016. Hybrid drowsy SRAM and STT-RAM buffer designs for dark-silicon-aware NoC. IEEE Trans. Very Large Scale Integr. Syst. 24, 10 (2016), 3041–3054.

Received October 2020; revised February 2021; accepted May 2021