2018 IEEE International Conference on Cluster Computing

The Tofu Interconnect D

Yuichiro Ajima, Takahiro Kawashima, Takayuki Okamoto, Naoyuki Shida, Kouichi Hirai, Toshiyuki Shimizu
Next Generation Technical Computing Unit
Fujitsu Limited
Kawasaki, Japan
{aji, t-kawashima, tokamoto, shidax, k-hirai, t.shimizu}@jp.fujitsu.com

Shinya Hiramoto, Yoshiro Ikeda, Takahide Yoshikawa, Kenji Uchida, Tomohiro Inoue
AI Platform Business Unit
Fujitsu Limited
Kawasaki, Japan
{hiramoto.shinya, ikeda.yoshir-02, yoshikawa.takah, k_uchida, inoue.tomohiro}@jp.fujitsu.com

***Abstract*—In this paper, we introduce a new and highly scalable interconnect called Tofu interconnect D that will be used in the post-K machine. This machine will officially be operational around 2021. The letter D represents high "density" node and "dynamic" packet slicing for "dual-rail" transfer. Herein we describe the design and the evaluation results of TofuD. Due to the high-density packaging, the optical link ratio of TofuD has decreased to 25% from the 66% optical link ratio of Tofu2. TofuD applies a new technique called dynamic packet slicing to reduce latency and to improve fault resilience. The evaluation results show that the one-way 8-byte Put latency is 0.49 µs. This is 31% lower than the latency of Tofu2. The injection rate per node is 38.1 GB/s which is approximately 83% of the injection rate of Tofu2. The link efficiency is as high as approximately 93%.**

Keywords*— *high-performance computing, interconnect, high-density packaging, fault resilience

I. Introduction

The Tofu interconnect family is a group of system interconnects for highly scalable HPC systems developed by Fujitsu. The Tofu Interconnect D (TofuD) is a new member to this family and designed for used in the post-K machine [1] that will be operational around 2021. Tofu stands for "torus fusion" that represents the designed combination of dimensions with an independent configuration and a routing algorithm. The letter D represents high "density" node and "dynamic" packet slicing for "dual-rail" transfer. In this paper, we describe the design overview, specification, and evaluation results of TofuD. The design overview includes the new node configuration that incorporates the high-density memory packaging technology, the optimizations for the increasing number of non-uniform memory access (NUMA) domains, and a new packet transfer technique that reduces latency and improves resilience.

Section II explains the background of this work. Section III presents related work. Section IV introduces the design of TofuD, and Section V presents the results of performance evaluation. Section VI concludes this paper.

II. Background

A. Tofu Interconnect

The Tofu interconnect [2][3] was developed for the K computer [4] that became operational in 2012. The 6D mesh/torus network of Tofu achieved high scalability of 82,944 compute nodes, and the virtual 3D torus rank mapping scheme provided both high availability and topology-aware programmability. Tofu was also used in the PRIMEHPC FX10 system which doubled the number of processor cores per node to sixteen from eight of the K computer.

A node address in the physical 6D network is represented by six-dimensional coordinates X, Y, Z, A, B, and C. The A and C coordinates can be 0 or 1, and the B coordinate can be 0, 1, or 2. The range of the X, Y, and Z coordinates depends on the system size. Two nodes whose coordinates are different by 1 in one axis and identical in the other five axes are "adjacent" and are connected to each other. When a certain axis is configured as a torus, the node with coordinate 0 in the axis and the node with the maximum coordinate value are connected to each other. The A- and C-axes are fixed to the mesh configuration and the B-axis is fixed to the torus configuration. Each node has 10 ports for the 6D mesh/torus network. Each of the X-, Y-, Z-, and B-axes uses two ports, and each of the A- and C-axes use one port.

Each link provided 5.0 GB/s peak throughput. Each link had 8 lanes of high-speed differential I/O signals at a 6.25-Gbps data rate. Tofu was implemented as an interconnect controller (ICC) chip with 80 lanes of signals for the network. All links were electric, and there was no optical link in the original Tofu interconnect.

Each node had four Tofu network interfaces (TNIs) so that four data were simultaneously transmitted to four independent directions and four data were received from four independent directions. The injection bandwidth per node was 20 GB/s. The total injection bandwidth (which yields the theoretical peak performance of the nearest neighbor data exchange) of the K computer was 1.66 PB/s. The bisection bandwidth (which yields the theoretical peak performance of global data exchange) of the K computer was 46.1 TB/s for the physical 18×2×2 mesh and the 24×16×3 torus network, or 34.6 TB/s for the virtual 48×36×48 torus network. In a large torus network, there are performance differences of one to two orders of magnitude depending on the communication pattern; therefore topology-aware tuning of applications is important.

The TNI provided the communication function of remote direct memory access (RDMA) Put/Get, system packet, and Tofu barrier. The system packet was used for system control and IP communication. The Tofu barrier handles multiple stages of communication for barrier synchronization with hardware that is unaffected by OS jitter that severely deteriorates the latency when software handles the communication. Barrier gate (BG) is a hard-wired module that synchronously communicates with other BGs. Specifically, each BG waits for signals from up to two preset BGs, and then transmits signals to up to two other preset BGs. There are two types of BG, start-and-end point and relay point. Each start-and-end point BG is fixedly associated with an interface called a barrier channel (BCH). The MPI library allocates these communication resources at the creation of each communicator. The reduce-broadcast tree algorithm consumes one BCH and five BGs, or the recursive-doubling algorithm consumes one BCH and log2(n) BGs. A BG can perform the reduction operation so that the Tofu barrier can perform all-reduce collective communication that is limited to one element. In Tofu, the Tofu barrier was available only on TNI number 0 and there were 8 BCHs and 64 BGs; 8 BGs were for start-and-end points and 56 BGs were for relay points. Therefore, up to eight communicators per node could simultaneously use the Tofu barrier. When there were multiple processes on a node, the intra-node processes were synchronized by software and the representative process used a BCH for the inter-node synchronization.

B. *Tofu Interconnect 2*

The next version Tofu interconnect 2 (Tofu2) [5][6] was designed for the PRIMEHPC FX100 system launched in 2015. Each node of FX100 had eight packages of hybrid memory cube (HMC) that contained a stack of memory dice. In contrast, each node of the K computer and FX10 had eight inline memory modules that had been used over 30 years. This transition from a wide memory module to a small memory package reduced the node footprint of FX100.

To reduce the node footprint further, the Tofu2 implementation also shifted to processor chip integration from the independent ICC chip of Tofu. Considering the balance with 128 collocated signal lanes for memory on the processor chip, Tofu2 halved the number of signal lanes to 40 from the 80 signal lanes of Tofu. To compensate for halving the number of signal lanes, Tofu2 significantly improved the data rate of the signals from 6.25-Gbps to 25.78125-Gbps by introducing optical links. The link bandwidth and the injection bandwidth per node were increased to 12.5 GB/s and 50 GB/s, respectively.

In the communication function of Tofu2, the following features were extended; RDMA atomic read modify write, triggered communication (called session mode for non-blocking collective communication), and RDMA for system use.

In FX100, not only the number of compute cores were increased to 32, but the recommended number of user processes in a node was also increased from 1 to 2 because two NUMA domains called core-memory groups (CMGs) were introduced on a chip. Therefore, the number of RDMA communication resources called control queues (CQs) was required to be increased to allocate dedicated CQ to each user process. In Tofu, each TNI had three CQs and one out of the three CQs was fixed for system use. For one or two user processes per node, each process was assigned one dedicated CQ per TNI and the MPI communication library internally used four CQs simultaneously. When the number of processes per node exceeded two, the total number of assigned CQs for each process decreased. When the number of processes per node exceeded eight, CQs were shared by multiple processes. In Tofu2, the number of CQs per TNI increased from 3 to 12 to avoid shared CQ even if the number of processes per node was 32.

C. *The Post-K Computer*

The post-K computer is a system developed to replace the K computer and will start operating around 2021. The post-K computer is designed to take full advantage of the assets of the K computer such as applications, users, tools, system operational knowledge, and the facility. The post-K is required not only to expand application domains, but also to significantly improve application performance, specifically up to 100 times or more than that on the K. Fujitsu cooperates with the asset holder RIKEN and develops leading edge technologies of FX100 to construct the post-K machine.

III. Related Work

This section describes the system interconnects used in the recent world-class systems other than the Tofu interconnect family. All systems have the same level of bisection bandwidth which represents the theoretical peak performance of global data exchange. On the other hand, the total injection bandwidth significantly differs depending on the type of network topology. Some systems have a total injection bandwidth close or equal to their own bisection bandwidth and the other systems have a total injection bandwidth much higher than their own bisection bandwidth.

A. *InfiniBand™*

InfiniBand™ (IB) [7] is a standard specification of interconnect defined by the InfiniBand® Trade Association. IB products have been widely used to build HPC clusters. The network interface is called host channel adapter (HCA) and an ordinary HCA is implemented as a discrete chip and mounted on an adaptor card. An ordinary IB network is constructed by using switch boxes. Constructing an interconnection network with independent components such as adapter cards and switch boxes is disadvantageous in terms of packaging density and power consumption. However, there is the advantage in the flexibility of configuration. For example, a node configuration that has an increased number of HCAs enhances injection bandwidth and accelerates communication intensive applications. In the other example, the network configuration called a full-bisection bandwidth fat-tree, of which the bisection bandwidth is equivalent to the total injection bandwidth, suppresses variation in the execution time of applications not optimized for the network topology.

Mellanox's dual-rail EDR IB HCA will be used in the Summit system [8] which will start full operation in 2019. The injection bandwidth per node is 25 GB/s. The total injection or bisection bandwidth will be approximately 115 TB/s. The TaihuLight system, which started operation in 2016, also used Mellanox's IB HCAs and switch chips [9]. The Sunway network of TaihuLight was constructed as a four-stage tapered fat-tree. The total injection bandwidth was 512 TB/s and the bisection bandwidth was approximately 70 TB/s. There was a rare example of IB HCA integration. Oracle's Sonoma processor [10] was designed for high-density scale-out servers and there were two built-in HCAs on a chip. The injection bandwidth per node was 13.6 GB/s.

B. *Omni-Path*

Omni-Path [11] is Intel's HPC interconnect family. In the first generation, the host fabric interface (HFI) is implemented as a discrete chip and mounted on an adaptor card or integrated into a CPU package. Omni-Path is considered likely to be used in the future Aurora system [12]. The first-generation Omni-Path was used in the Oakforest-PACS system that became operational in 2016. The injection bandwidth per node was 12.5 GB/s. The total injection or bisection bandwidth was 102.6 TB/s.

C. *Aries Interconnect*

The Aries interconnect [13] developed by Cray is a highly scalable system interconnect that employs a Dragonfly-based topology. The network interface and the router were implemented together in a discrete chip. Each Aries chip had four network interfaces and connected four nodes. Each network interface had two ports to connect the internal router port. Each router port operated at a link throughput of 4.7 GB/s for global links or 5.25 GB/s in a group of 384 nodes. Therefore, the injection bandwidth per node was 10.5 GB/s. The upgraded Piz Daint system that started operation in 2016 used Aries. The total injection bandwidth and the bisection bandwidth were 71 TB/s and 36 TB/s respectively.

D. *Blue Gene/Q Five-dimensional Torus*

IBM Blue Gene/Q (BG/Q) was a highly scalable supercomputer that had a five-dimensional torus network [14][15]. Each node has 10 links for the torus network and each link provides 2.0 GB/s peak throughput. The injection bandwidth per node was 20 GB/s. The Sequoia system that started classified operations in 2013 was a BG/Q system with 98,304 nodes. The total injection bandwidth was 1.97 PB/s and the bisection bandwidth was 49.2 TB/s. The characteristics and performance of the BG/Q five-dimensional torus network were similar to those of the 6D mesh/torus network of the Tofu interconnect.

IV. Design of TofuD

This section describes the design of TofuD focusing on the difference compared to Tofu2.

A. *Node Configuration*

Figure 1 shows a block diagram of the post-K computer node. The number of CMGs increased to four from two of Tofu2, and the number of TNIs also increased from four to six. The CMGs and the TNIs are connected by the network on chip (NOC). As the number of CMGs increases, there is a difference in the distance between TNIs and each CMG. Two CMGs are far from TNIs, and the other two CMGs are near TNIs.

Figure 2 shows a prototype CMU. Two processor packages and three cable cages are cooled by water. One compute node consists of one package in which one processor chip and four stacks of high bandwidth memory (HBM) are integrated. As a trade-off with the use of the high-density memory packaging technology, the number of memory stacks per node has halved from FX100 that used eight packages of HMC. In order to balance with the halved number of memory stacks, the TofuD again halved the number of signal lanes to 20 from 40 of Tofu2.

To reduce the hardware cost, the TofuD uses mainstream quad-lane active optical cables. Half of the CMUs in a shelf connect two optical cables of the X- and Y-axes, and the other half connect three optical cables of the X-, Y-, and Z-axes. Each active optical cable is shared by two links in the same direction of two compute nodes on the same CMU. Although the number of signals for each active optical cable is one-third of that of the board-mount optical assembly used in Tofu2, the number of optical modules on the board reduces to 2.5 from 8 of FX100 owing to the reductions in the optical link ratio, number of high-speed signals per node, and number of nodes per board.

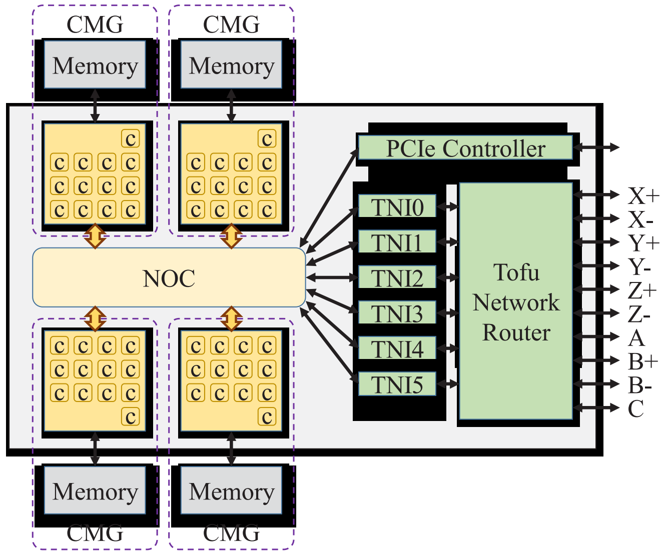

Fig. 1. Block diagram of the post-K computer node

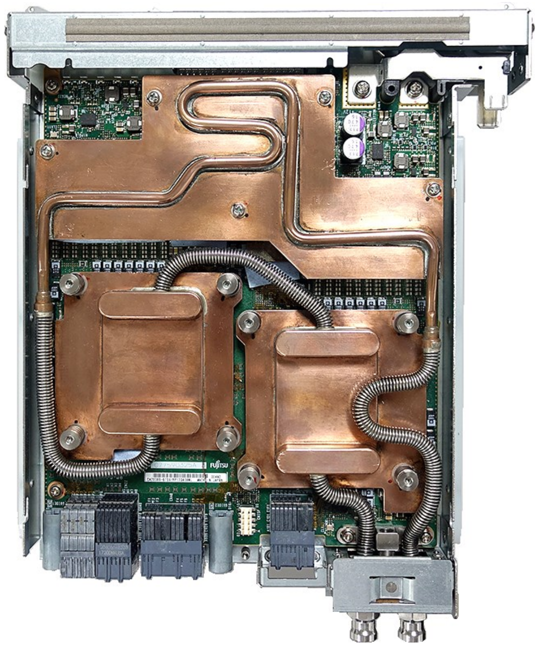

Fig. 2. Prototype CPU memory unit

B. Package Structure and Link Configuration

In a rack of the post-K computer, each of the upper and lower halves of the rack houses 192 nodes with the geometry (X, Y, Z, A, B, C) = (2, 2, 4, 2, 3, 2). Each half rack accommodates four building blocks called "shelves," two in the front-side and two in the rear-side. The geometry of a shelf is (X, Y, Z, A, B, C) = (1, 1, 4, 2, 3, 2). Figure 3 shows a prototype rack of the post-K computer. Each side of the rack stores four shelves vertically. Each shelf houses 24 CPU memory units (CMUs) that loads two nodes connected in C-axis.

All connections in a half rack use electric links and the connections out of a half rack use optical links. Therefore, half of the connections in the X- and Y-axes and one fourth of the connections in the Z-axis use optical links. Because of the high-density packaging and large structure of the half rack, the optical link ratio of the TofuD is as low as 25%, which has substantially decreased from 66% for Tofu2 that used optical links for connection out of a 2U chassis with the geometry (X, Y, Z, A, B, C) = (1, 1, 3, 2, 1, 2).

Fig. 3. Prototype rack of the post-K computer

C. Injection Rate per Node

Table I shows the comparison of node and link configurations within the Tofu family. TofuD uses a high-speed signal of 28-Gbps data rate that is approximately 9% faster than that of Tofu2. However, due to the reduction of the number of signals, TofuD reduces the link bandwidth to 6.8 GB/s, which is approximately 54% for Tofu2. To compensate the reduction in the link bandwidth, TofuD increases the number of simultaneous communications from 4 of Tofu2 to 6. The injection rate of TofuD is enhanced to approximately 80% of that of Tofu2. There are six adjacent nodes in the virtual 3D torus therefore topology-aware algorithms can use six simultaneous communications effectively.

The logic circuits of TofuD operate at a 425-MHz clock frequency, which is about 9% faster than the clock frequency of Tofu2. The width of the datapath decreases from 256 to 128 bits as the number of signal lanes decreased.

TABLE I. DATA RATES OF SIGNAL AND INJECTION RATES

	Tofu	Tofu2	TofuD
Number of signal lanes per node	80	40	20
Data rate (Gbps)	6.25	25.78125	28.05
Link bandwidth (GB/s)	5.0	12.5	6.8
Number of TNIs per node	4	4	6
Injection bandwidth per node (GB/s)	20	50	40.8

D. Communication Resources

TABLE II shows a comparison of the number of communication resources within the Tofu family. Both the number of compute cores and the number of TNIs per node increased by 1.5 times from Tofu2, and the number of CQs per TNI remained constant at 12. In Tofu2, there was no change in the Tofu barrier. In TofuD, the amount of communication resources for the Tofu barrier has increased as the number of CMGs has increased. To allocate a BCH from a different TNI to each CMG, the Tofu barrier becomes available on all TNIs in TofuD, and the number of resources per node increased significantly for both BCH and BG. The ratio of the BCH to BG increased from 1:8 to 1:3 because the reduce-broadcast tree algorithm for the intra-node part of synchronization is assumed to reduce the number of BGs to be used. The buffer size of each BG is also expanded so that the Tofu barrier can perform all-reduce of eight integer or three floating point elements with one synchronization.

TABLE II. NUMA DOMAIN AND COMMUNICATION RESOURCES

	Tofu	**Tofu2**	**TofuD**
Number of compute cores per node	8, 16	32	48
Number of CMGs per node	1	2	4
Number of TNIs per node	4	4	6
Number of CQs per node	12	48	72
Number of BCHs per node	8	8	96
Number of BGs per node	64	64	288

E. Dynamic Packet Slicing for Dual-rail Transfer

The physical coding sublayer (PCS) of Tofu2 was developed based on the 100Gb Ethernet technology. The packet transfer latency of Tofu2 was increased to approximately 0.3 μs from approximately 0.1 μs for Tofu because of the complex transmission technology including encoding, symbol detection, multi-lane distribution, and lane-to-lane deskew. In Tofu2, there was another issue in the fault-tolerance feature as follows. Tofu2 introduced the link degradation feature that reduced the number of active lanes without losing a packet. However, once the link degraded, the number of lanes never recovered; therefore, there is no fault resilience.

To address these issues, TofuD applies a new technique called dynamic packet slicing for dual-rail transfer. To address the latency issue, TofuD implements independent PCS for each signal lane and splits a packet in the data-link layer. To address the fault-resilience issue, TofuD duplicates a packet and redundantly transfers it in both lanes as opposed to reducing the number of active lanes. The data link layer adds information to the packet, indicating that the packet has been split or duplicated. The data link layer monitors the receiver-side PCS's detection frequencies of CRC and other transmission errors and adds the transmission quality status information to the packet as well. The data link layer determines the split mode of the packet, depending on the received transmission quality status information.

Figure 4 shows the frame format that includes a routing header, a transport layer packet (TLP), and padding space for a data link layer packet (DLLP). First, the data link layer stores a DLLP to the frame. Next, the data link layer simultaneously generates two slices from the frame. The routing header is duplicated to the two slices, TLP and DLLP are split or duplicated, and the padding is removed. Finally, the two slices are distributed to two PCSs and each PCS adds a preamble, a CRC code called FCS, and inter-frame gap to the slice.

Figure 5 shows the undivided slice format that includes a routing header, full TLP, full DLLP, and control codes to envelop the payload. Figure 6 shows the divided slice formats that includes a routing header, a split TLP, a split DLLP, and control codes to envelop the payload. The PAT field in a slice indicates the pattern of packet splitting, and the STAT field indicates the status of the observed transmission quality. The PAT field is defined as a 3-bit width field for future expansion to quad-lane.

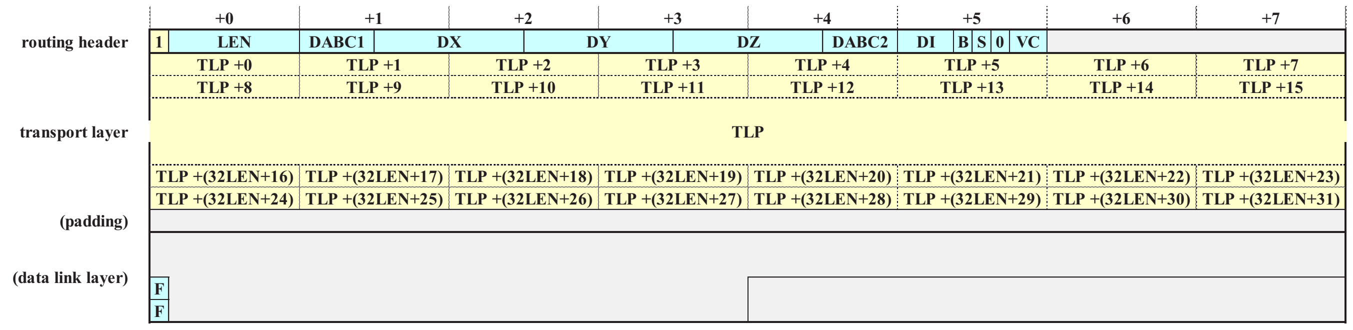

Fig. 4. Frame format

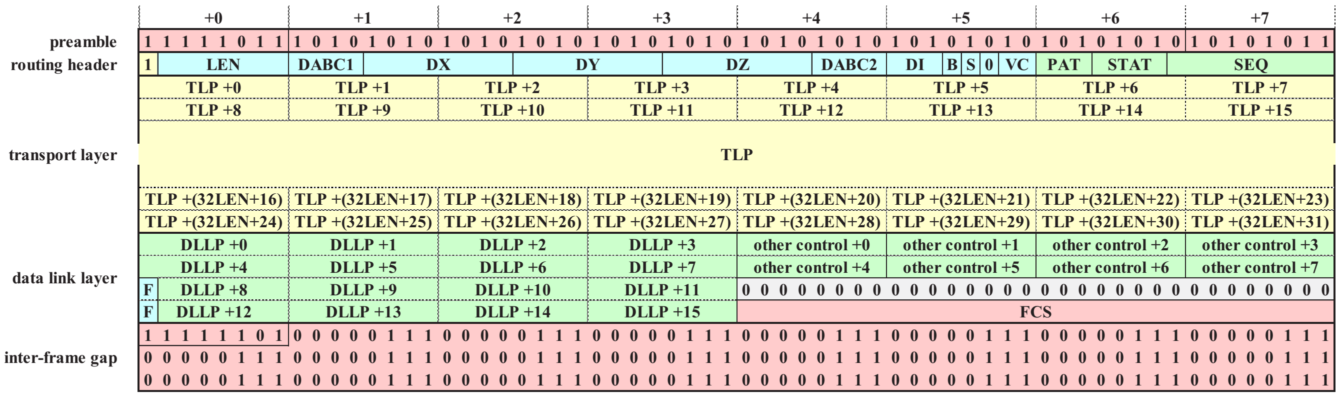

Fig. 5. Undivided slice format for the duplicate-mode

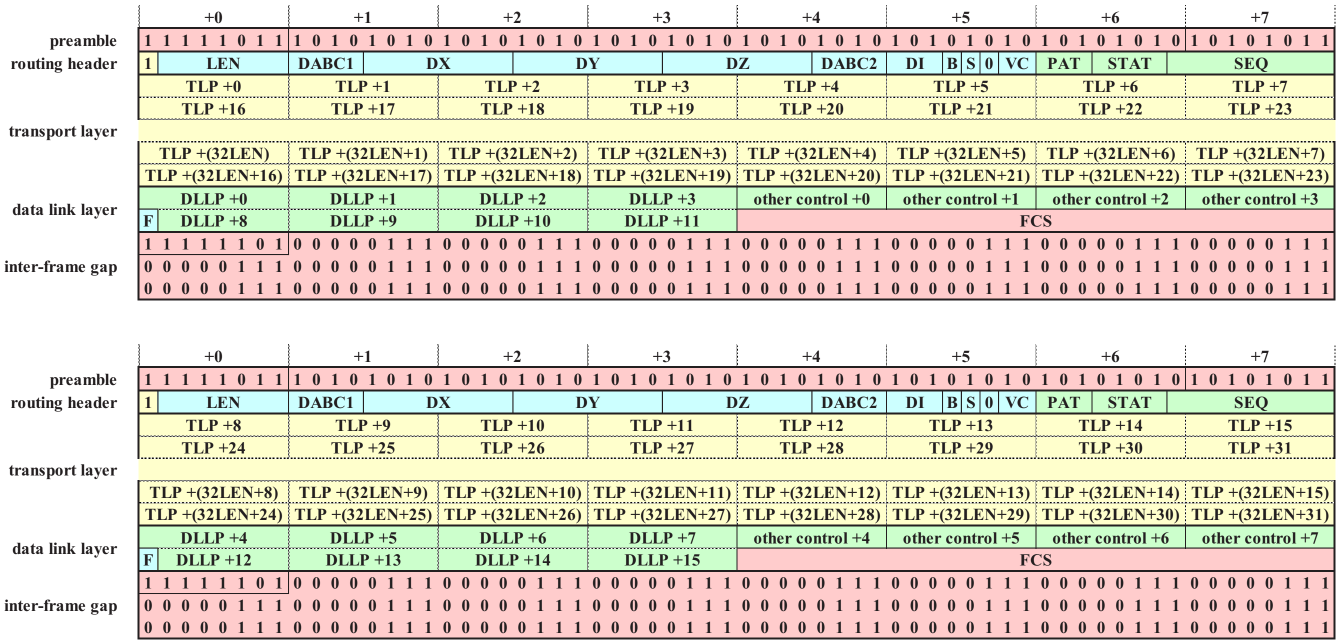

Fig. 6. Divided slice format for the split-mode

V. Performance Evaluation

This section gives early evaluation results of the fundamental performance of TofuD.

A. Evaluation Environment

The communication performance of TofuD was evaluated by system-level logic simulations. The simulation models were built using the Verilog RTL codes for the production, and included multiple nodes. The simulations were performed on Cadence's hardware emulators. The simulated processor cores executed the test programs that used the TofuD hardware directly. The latency results were measured directly from the simulation waveforms; thus we obtained one-way latencies without halving average round-trip latencies. The throughput results were derived from the measured latency values.

For Tofu and Tofu2, the evaluation results of latency breakdown were obtained from the simulation waveforms as well as TofuD. The other results of Tofu and Tofu2 were evaluated with actual machines using the low-level communication library.

In these preliminary evaluations, the test programs included no communication software stack such as an MPI library; therefore, the evaluation results included no software overhead, and all test programs performed nearest-neighbor communication.

B. Latency

TABLE III shows the evaluated results of the latencies of Tofu, Tofu2, and TofuD. In each evaluation, it is assumed that a Put transfer is executed between the nearest neighbor nodes on the same board, and the time from when the initiator process started the Put transfer to when the target process read the data was measured.

In Tofu, the direct descriptor feature reduced the latency by more than 0.2 µs. In Tofu2, the cache injection feature reduced the latency by nearly 0.2 µs. Both these reductions in Tofu and Tofu2 are the result of bypassing the main memory with the newly introduced features of the network interface.

In TofuD, the latency is reduced by approximately 0.2 µs again. Overall, the latency has been reduced by 46% from Tofu and 31% from Tofu2. The reduction is mainly due to the over-hauling of the transmission technology such as the compensation for signal skew, and reconsideration of the pipeline design of data-paths. There is an additional penalty of approximately 0.05 µs if the initiator process runs on a far CMG in the initiator node and the target process also runs on a far CMG in the target node. Although the difference is small in TofuD, the increasing density and locality on the chip may impact the communication latency in future systems.

Figure 7 presents the breakdowns of latency of one-way and one-hop Put transfer. A latency value for each component was obtained from the simulation waveforms. In Tofu2, the packet transfer latency through one link and two switches was increased by approximately 0.2 µs from Tofu due to the complex PCS derived from 100 Gb Ethernet. The packet transfer latency of TofuD achieved nearly the same latency as Tofu owing to the new dynamic packet slicing technique. In TofuD, the part of the one-way Put latency other than the packet transfer was almost the same as Tofu2. In total, approximately 0.2 µs of one-way Put latency has been reduced in TofuD compared with Tofu2.

C. Injection Rate

TABLE IV lists the evaluation results of injection rates and efficiencies of Tofu, Tofu2, and TofuD. In Tofu and Tofu2, four Put transfers in different directions were simultaneously executed and total throughputs were evaluated. In TofuD, six Put transfers in different directions were executed.

The injection rate of TofuD is more than two times higher than that of Tofu and 17% lower than that of Tofu2. The efficiencies of Tofu are lower than that of a single Put transfer, because Tofu was not integrated in the processor chip, leading to a bottleneck in the bus that connects the processor chip and the interconnect controller chip. The relatively low efficiencies are mainly because of the packet size of the bus, which includes only one cache line of data.

TABLE III. ONE-WAY 8-BYTE PUT LATENCIES BETWEEN NEAREST NEIGHBOR NODES OF TOFU FAMILY

	Communication settings	Latency [µs]
Tofu	Descriptor on main memory	1.15
	Direct Descriptor	0.91
Tofu2	Cache injection OFF	0.87
	Cache injection ON	0.71
TofuD	To/From far CMGs	0.54
	To/From near CMGs	0.49

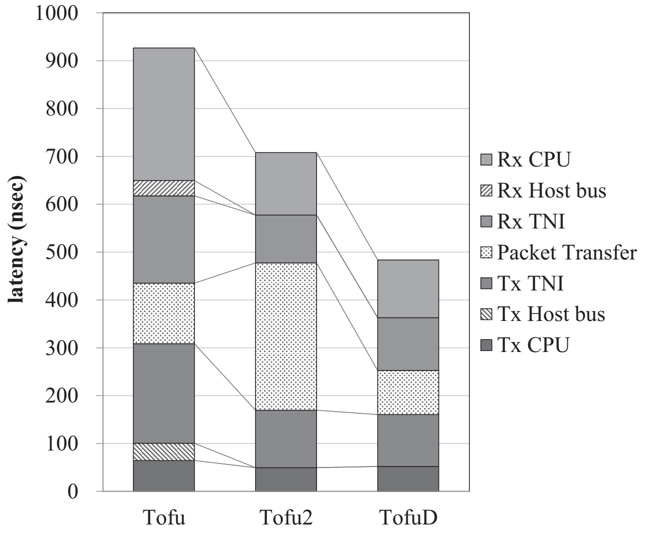

Fig. 7. Comparison of latency breakdowns of one-way Put transfer

TABLE IV. INJECTION RATES AND EFFICIENCIES OF SIMULTANEOUS PUT TRANSFERS OF TOFU FAMILY

	Injection rate [GB/s]	Efficiency [%]
Tofu (K)	15.0	77
Tofu (FX10)	17.6	88
Tofu2	45.8	92
TofuD	38.1	93

Tofu2 and TofuD are integrated into the processor chips and the efficiencies of injection rates are almost the same as that of the single Put transfer presented in the next subsection.

D. Throughput

TABLE V shows the evaluated results of Put throughputs and the efficiencies of Tofu, Tofu2, and TofuD. The throughput of TofuD is 33% faster than that of Tofu and 45% slower than that of Tofu2. The efficiencies exceed 90% for all versions. These high efficiencies are the distinctive characteristics of the Tofu interconnect family, and are due to the rather large packet size for an HPC interconnect. Although a larger packet size is costly in design, it also reduces the software overheads of system-wide communication protocols such as IP over Tofu.

TABLE V. THROUGHPUTS OF PUT TRANSFER AND EFFICIENCIES OF THE TOFU FAMILY

	Throughput [GB/s]	Efficiency [%]
Tofu	4.76	95
Tofu2	11.46	92
TofuD	6.35	93

The efficiency of Tofu2 is slightly lower than that of Tofu and TofuD. This mainly because of the overhead of data alignment. Tofu and TofuD were implemented in 128-bit data-paths and the data alignment was 16 bytes. Tofu2 was implemented in 256-bit width and the alignment was 32 bytes.

E. *Intra-node Latency of the Tofu Barrier*

The Tofu barrier is extended for intra-node use in TofuD. This subsection presents the evaluated latency results of the intra-node Tofu barrier. First, the latency of each component was evaluated from the waveform of a simple test that uses only one BCH and two BGs connected in series. The latency result of a BCH and a start-and-end BG was approximately 0.48 µs, and the latency result of a relay BG was nearly 0.13 µs.

Next, intra-node synchronization latencies using Tofu barrier were evaluated using the test programs. The number of BCHs to be synchronized varied from 4 to 48. If the number of BCHs exceeds the number of TNI, multiple BCHs were used in a TNI. The test programs used the reduce-broadcast tree algorithm for intra-TNI synchronization and the recursive doubling algorithm for inter-TNI synchronization. The total number of used BGs per node and the number of communication stages for each test program was shown in TABLE VI. In these test programs, one process operated all BCHs; therefore, the deviation of the synchronization start time was small as compared with the actual usage condition in which each BCH is operated by a different process.

Figure 8 shows the evaluated results and the estimated latencies. The minimum latencies were estimated so that the latency component of relay BGs increased in proportion to the log2 of the number of BCHs. However, as the number of BCHs per TNI increased beyond 1, the evaluation results became worse than the estimated minimum latencies. The waveform result showed that all BCHs and BGs were serially processed. The latency of the BCH and the BG at the start point were overlapped between BCHs for 0.19 µs out of 0.48 µs and the remaining 0.29 µs were serialized. The estimated latencies of processing the BG and the BCH serially were close to the evaluation results.

The evaluation results showed that there was the latency penalty when allocating multiple BCHs from the same TNI to the same communicator. The MPI library should be implemented using the Tofu barrier avoiding this penalty as follows. If the number of processes in a node does not exceed six, the MPI library should allocate one BCH to each process from different TNI. If the number of processes in a node exceeds six, the MPI library should allocate one BCH to each of six groups of processes. Each group of processes share one BCH and synchronize within the group via memory.

TABLE VI. CONFIGURATIONS OF THE TEST PROGRAMS OF THE TOFU BARRIER

Number of start-and-end points	1	4	8	16	48
Number of TNIs	1	4	6	6	6
Max. number of BCHs per TNI	1	1	2	3	8
Max. number of BGs per TNI	2	2	5	9	24
Number of communication stages	2	2	4	6	9

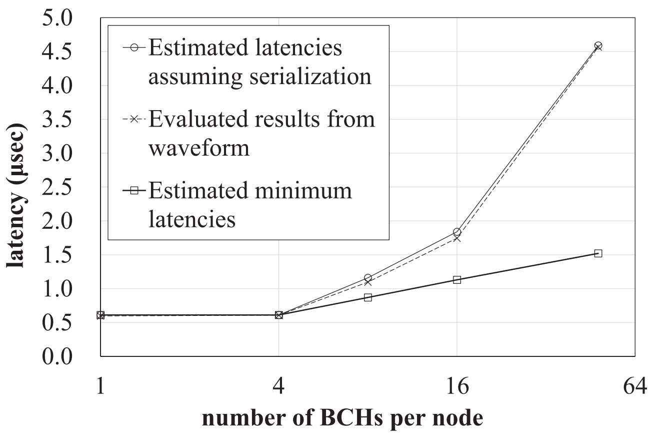

Fig. 8. Estimated and evaluated results of the Tofu barrier test programs

VI. CONCLUSION

In this paper, we introduced a new and highly scalable interconnect called Tofu Interconnect D that will be used in the post-K machine, which will be operational around 2021. The letter D represents high "density" node and "dynamic" packet slicing for "dual-rail" transfer. This paper described the design of TofuD including the package structure of the node, the rack, the link configuration between nodes, the injection rate per node, increased communication resources and a new packet transfer technique. This paper also presented the evaluation results of TofuD. The one-way 8-byte Put latency was 0.49 µs that was reduced by 31% from that for Tofu2. The injection rate per node was 38.1 GB/s which was approximately 83% of the injection rate for Tofu2. The link efficiency was as high as approximately 93%. Additionally, the evaluation results showed the constraints on the in-node usage of the Tofu barrier to avoid performance penalty.

REFERENCES

[1] RIKEN Center for Computational Science – About the Project. [online] Available at: http://www.r-ccs.riken.jp/en/postk/project [Accessed: 06- May - 2018]

[2] Y. Ajima, S. Sumimoto and T. Shimizu, "Tofu: A 6D Mesh/Torus Interconnect for Exascale Computers," in IEEE Computer, vol. 42, no. 11, pp. 36?40, 2009.

[3] Y. Ajima, Y. Takagi, T. Inoue, S. Hiramoto and T. Shimizu, “The Tofu Interconnect,” IEEE 19th Annual Symposium on High Performance Interconnects (HOTI), pp. 87-94, 2011.

[4] H. Miyazaki, Y. Kusano, N. Shinjo, F. Shoji, M. Yokokawa and T. Watanabe, “Overview of the K computer System,” Fujitsu Scientific and Technical Journal, vol. 48, no.3, pp. 255-265, 2012.

[5] Y. Ajima et al. "Tofu Interconnect 2: System-on-Chip Integration of High-Performance Interconnect," In Proceedings of the 29th International Conference on Supercomputing (ISC14), pp. 498-507, 2014.

[6] Y. Ajima et al., “The Tofu Interconnect 2,” IEEE 22nd Annual Symposium on High-Performance Interconnects (HOTI), pp. 57-62, 2014.

[7] InfiniBand Trade Association, InfiniBand Architecture Specification Volume 1 Release 1.2.1, 2007.

[8] Oak Ridge Leadership Computing Facility – Summit. [online] Available at: https://www.olcf.ornl.gov/olcf-resources/compute-systems/summit/ [Accessed: 06- May - 2018]

[9] Jack Dongarra, "Report on the Sunway TaihuLight System." [online] Available at: http://www.netlib.org/utk/people/JackDongarra/PAPERS/sunway-report-2016.pdf [Accessed: 06- May - 2018]

[10] B. Vinaik and R. Puri, “Oracle’s Sonoma Processor: Advanced Low-cost SPARC Processor for Enterprise Workloads,” HotChips 27, 2015.

[11] M. S. Birrittella et al., “Intel Omni-path Architecture: Enabling Scalable, High Performance Fabrics,” IEEE 23rd Annual Symposium on High-Performance Interconnects (HOTI), pp. 1-9, 2015.

[12] Intel – Aurora Fact Sheet. [online] Available at: https://www.intel.com/content/www/us/en/high-performance-computing/aurora-fact-sheet.html [Accessed: 15- May - 2018]

[13] G. Faanes, et al., “Cray cascade: a scable HPC system based on a Dragonfly network,” In Proceedings of the International Conference on High Performance

[14] D. Chen, et al., “The IBM Blue Gene/Q Interconnection Network and Message Unit,” In Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis (SC 2012), Article 26, 2011.

[15] D. Chen et al., “Looking under the hood of the IBM Blue Gene/Q network,” 2012 International Conference for High Performance Computing, Networking, Storage and Analysis (SC), pp. 1-12, 2012.