Liao XK, Pang ZB, Wang KF *et al.* High performance interconnect network for Tianhe system. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 30(2): 259–272 Mar. 2015. DOI 10.1007/s11390-015-1520-7

High Performance Interconnect Network for Tianhe System

Xiang-Ke Liao [1,2] (廖湘科), *Fellow, CCF, Member, ACM,* Zheng-Bin Pang [1,2] (庞征斌), *Member, CCF, ACM*
Ke-Fei Wang [1] (王克非), Yu-Tong Lu [1,3] (卢宇彤), *Member, CCF, ACM,* Min Xie [1,3] (谢 旻), Jun Xia [1] (夏 军)
De-Zun Dong [1,2] (董德尊), *Member, CCF, ACM, IEEE,* and Guang Suo [1,3] (所 光)

[1] *College of Computer, National University of Defense Technology, Changsha 410073, China*

[2] *Science and Technology on Parallel and Distributed Processing Laboratory, National University of Defense Technology Changsha 410073, China*

[3] *State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha 410073 China*

E-mail: {xkliao, zhengbinpang, kefeiwang, ytlu, xiemin, xiajun, dong, suoguang}@nudt.edu.cn

Received November 30, 2014; revised January 15, 2015.

Abstract In this paper, we present the Tianhe-2 interconnect network and message passing services. We describe the architecture of the router and network interface chips, and highlight a set of hardware and software features effectively supporting high performance communications, ranging over remote direct memory access, collective optimization, hardware-enable reliable end-to-end communication, user-level message passing services, etc. Measured hardware performance results are also presented.

Keywords Tianhe-2 supercomputer, interconnect network, router architecture, network interface architecture, user-level message passing

1 Introduction

The Tianhe-2 (TH-2 or Milkyway-2) system, the second generation of massively parallel supercomputers in the TH series designed by National University of Defense Technology (NUDT), was crowned as the fastest supercomputer in the world on the 41st TOP500 list in June 2013①[1], having a peak performance of 54.9 petaflops in theory and 33.86 petaflops in Linpack benchmark. The system is installed in the National Supercomputing Center in Guangzhou, China, and currently has remained on top for two years. The Tianhe-2 project is sponsored by both the National High-Tech Research and Development Program (863 Program), administered by the Ministry of Science and Technology of China, and the government of Guangzhou. It aims to break barriers of scale and complexity in high-performance computing systems and continuously improve computing efficiency in post petaflops era, and plans to deliver a supercomputer achieving theoretical peak performance of 100 petaflops by 2015.

Tianhe-2, like its predecessor, Tianhe-1A, still employs accelerator-based architectures. Each compute node is equipped with two Intel® Xeon® E5-2600 processors and three Intel® Xeon® Phi™ accelerators based on the many-integrated-core (MIC) architecture, delivering a peak performance of 3.432 teraflops. The system is packaged in a compact structure. Each compute rack has four compute frames, and each frame contains one switch board, one monitor board, and 32 compute nodes packaged in 16 compute broads. The system can be maximally configured with 144 racks, up to 18 304 compute nodes and dedicated I/O nodes. The operating system is a 64-bit Kylin OS optimized to effectively support massive applications. Such massive parallelism in TH system puts high pressure on

Regular Paper

Special Section on Applications and Industry

This work was partially supported by the National High Technology Research and Development 863 Program of China under Grant No. 2012AA01A301 and the National Natural Science Foundation of China under Grant No. 61120106005.

① TOP500. http://www.top500.org/, Jan. 2015.

the design of interconnection network[2-3], which needs to provide efficient data movement and integrate these computational resources into a single system.

In order to fulfill the high communication requirement, Tianhe-2 uses proprietary interconnect, called TH Express-2 network for high-bandwidth and low-latency inter-processor communications. All network logic is developed and integrated into two specific ASIC chips, i.e., high-radix router chip and network interface chip. Both of them adopt efficient mechanisms to achieve high performance communications with regard to bandwidth, latency, reliability, and stability. The high-radix router chip has 24 8-lane ports with the bidirectional bandwidth of 224 Gbps, and delivers an aggregate bandwidth of 5.376 Tbps. The network interface chip contains a full width 16-lane PCI-E 3.0 interface, and provides an interface between software and hardware, enabling applications to access the high-performance network fabric efficiently.

TH Express-2 network highlights a set of hardware and software features effectively supporting high performance communications. In the network interface, we implement an efficient host interface upon PCI-E 3.0 that supports virtual ports and descriptor submission and achieves user-level message passing in virtual address mode. We implement mini-packet (MP) for low-latency and short massages, remote direct memory access (RDMA) for transfer of large amounts of block data, and conduct collective optimization for barrier and broadcast operations. In the network level, we use many techniques to improve the reliability of high-speed communications. We ensure reliable link-level packet delivery through credit-based flow control, link-level CRC, and packet re-transmission. Both deterministic and adaptive routings are performed to obliviously balance network traffics. Moreover, the network interface provides a reliable end-to-end connection mechanism. We develop high performance message passing services, leveraging the features of the router and network interface chips, to efficiently support the execution of massively parallel programs from user space with minimal software overhead.

This paper presents the state-of-the-art of our proprietary interconnect. Section 2 introduces the chip set and presents the architecture of high-radix router and network interface. Section 3 describes the network topology and Section 4 presents the detail of network protocols. Section 5 introduces message-passing services. Section 6 presents a summary of performance results on the Tianhe-2 interconnect fabric. Section 7 highlights prior related work. Finally, Section 8 summarizes the key features of TH Express-2 network.

2 Chip Set

Two high performance ASIC chips, network interface chip (NIC) and network router chip (NRC), are designed for TH Express-2 network using 40 nm process, which enables indirect network topology, specifically fat tree used in TH systems. NIC is a host interface chip connecting a host node into the network. NRC is a router chip with 24 high performance ports and implements high throughput switching among all its ports.

2.1 High Radix Router

The schematic of NRC is depicted in Fig.1. NRC mainly contains three types of blocks: 24 identical network ports (NPs), a big network-on-chip (NoC), and a module for chip configuration and management (CCM).

2.1.1 *High-Bandwidth Network Port*

Compute nodes in TH systems adopt a heterogeneous architecture, and each node is composed of two Intel® Xeon® CPUs and three Intel® Xeon® Phi™s, delivering a computing performance up to 3.4 teraflops. This makes it a challenge for the network to provide a well-balanced bandwidth in terms of bytes-to-flops ratio, that is, the amount of network bandwidth relative to each node's floating-point capability.

In order to get ultra-high bandwidth, eight high speed serial lanes, running up to 14 Gbps per lane, are integrated in one network port. This is much more than other router chips in current Top10 HPCs and hence results in a 112 Gbps (224 Gbps for bi-direction bandwidth) network port, twice greater than commercial off-the-shelf InfiniBand FDR. Twenty-four network ports are integrated in the NRC router chip, and thus the throughput of the router reaches 5.376 Tbps, supporting the TH Express-2 network to be the most powerful network in the world in 2014.

To make these eight independent lanes work cooperatively, we adopt some novel technologies, including elastic-buffer based jitter tolerance, lanes de-skew, lanes testing and selection based on built-in signal integrity testing, lanes re-ordering based on a small crossbar, rolling-CRC based packet checking, etc.

2.1.2 *Tile-Based Switch Fabric*

Traditionally, the input-queued crossbar organization is often used in low-radix routers. As the port num-

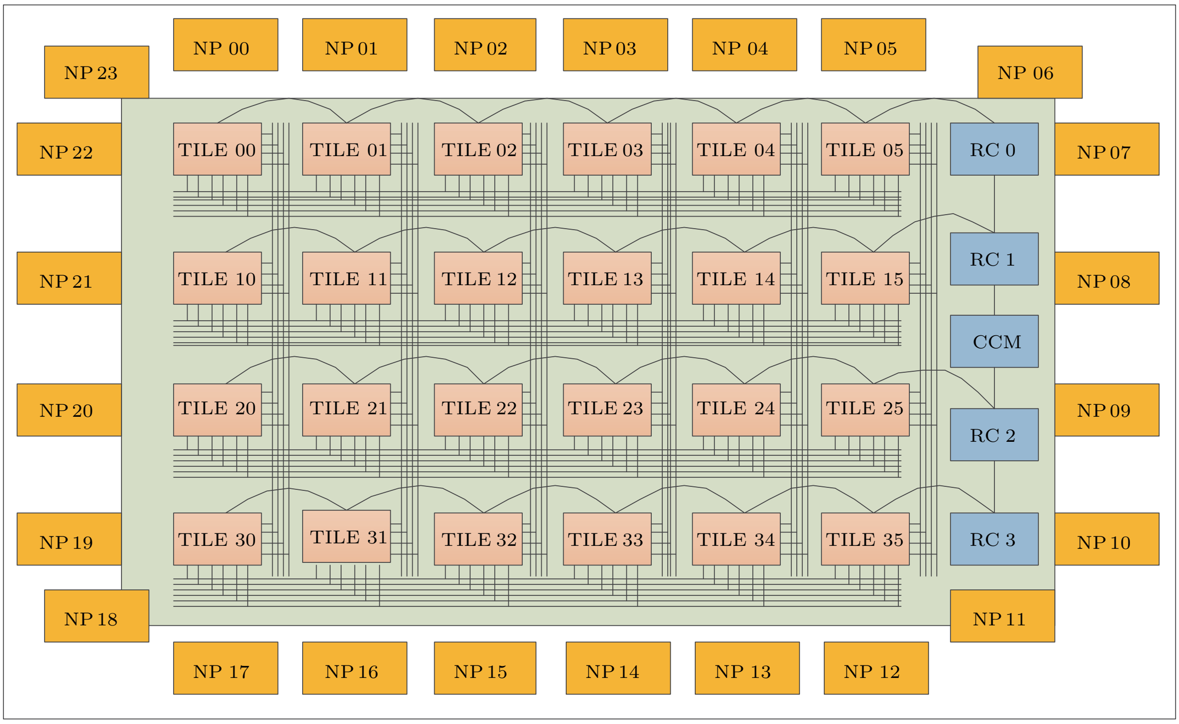

Fig.1. Schematic for network router chip.

ber of the router chip and port bandwidth increases, input-queued based crossbar does not scale efficiently due to both the arbitration logic and the wiring complexity growing quadratically with the number of inputs. To overcome these complexities, we use a hierarchical organization similar to that proposed by [4]. The 24-port crossbar is split into 24 tiles. As shown in Fig.1, the NRC router chip is organized as a 4×6 array of tiles. Such a tile-based microarchitecture produces regular structures and facilitates physical implementation.

Each tile is related to one bi-directional network port, and contains all of the arbitration logic and buffering associated with that port. Once a packet enters a tile, the destination output port is selected according to its destination address and the routing strategy. Then the packet is dispatched, through its row bus, to the "crossing-point" tile which sites on the same row with the input port and the same column with the selected output port. Each tile dispatches data packets to the six tiles on the same row, and receives data packets from them. A 6×4 small crossbar is implemented inside each tile, routing packets from the six input ports on the same row to the four output ports on the same column. Packets destined for the same output port from four tiles on the same column are collected and routed along the row buses and reach the destination port, where 4-to-1 multiplexer is implemented and the packets are routed to the right output port.

Using this tile-based fabric, the large crossbar of size 24×24 is simplified to 24 6×4 small crossbars. All the 24 tiles can be designed identically. The front-end logic design, verification and the back-end on-chip wiring are dramatically simplified. Another advantage of tile-based switch fabric is high throughput. As the load from one input is distributed to six small crossbars, each small crossbar and the whole fabric can get high throughput up to nearly 100%. According to our experiment results, the 24-port NRC can get 96% throughout for uniform random traffics.

It is worth noting that tile-based fabric consumes more packet buffers compared with traditional input-queue based crossbar. To mitigate the buffer usage overhead, NRC uses customized dynamically allocated multi-queue (DAMQ) to decrease the buffer size and improve the buffer utilization, while still meeting tight latency budgets. A novel DAMQ buffer management mechanism is used to construct input buffers, row buffers, and column buffers, and the test result shows that memory requirements decrease by more than 30% compared to statically allocated multi-queue (SAMQ).

2.2 Network Interface Chip

The NIC architecture is showed in Fig.2. It includes function modules of PCI-E interface, network interface, descriptor submission, protocol engine, address translation cache (ATC), connection management, and receive path.

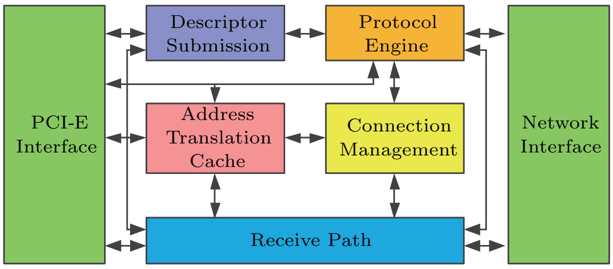

Fig.2. NIC architecture.

PCI-E interface is used to connect NIC to the host. It includes IP controller supporting 16-lane PCI-E 3.0 and other specific logics. Its main function is to serve read and write requests coming from the internal modules of NIC to the host memory.

Network interface is used to connect NIC to the network. It contains an 8-lane 14 Gbps SerDes at physical layer and link layer logics. It realizes the function of sending and receiving the network packets.

2.2.1 *Descriptor Submission*

The descriptor submission module is responsible for descriptor management and implements the submission queues of virtual ports. Eight hardware-based descriptor queues (HDQs) are organized as first-in-first-out (FIFO) structures on the chip independently while 32 software-based descriptor queues (SDQs) are grouped into eight DAMQs which are used to buffer descriptors fetched from the memory. HDQs are designed for low latency descriptor submission using limited on-chip buffer resource, and SDQs are designed for high submission rate for many-core CPUs and accelerators. Once the descriptors arrive at NIC and are buffered in the corresponding queues, they will be scheduled in round-robin fashion and submitted to the protocol engine. The execution of collective descriptor sequence is also realized in this module.

2.2.2 *Protocol Engine and Address Translation Cache*

The protocol engine is used to process MP and RDMA descriptors. To improve bandwidth utilization for short messages, multiple fully pipelined protocol processing units are implemented to handle descriptors in parallel.

During RDMA data transfer, the protocol engine uses ATC to translate the virtual address into the PCI address of physical pages. There is an address aligning unit in the protocol engine, which helps byte alignment between the local and the remote buffer of RDMA. Because the buffers in the kernel module are often allocated in contiguous physical memory, RDMA can also be set to bypass ATC. Thus the PCI address of physical page can be contained directly in RDMA descriptors.

In order to support RDMA in virtual address mode, NIC implements virtual address translation mechanism to translate virtual address into physical address. The whole address translation table (ATT)[5] is allocated in host memory, and NIC provides an on-chip cache named ATC to reduce the long latency of main memory access.

Before address transformation, the legality of each virtual address must be checked. A memory checker inside ATC is employed to detect the validity of virtual address. ATC can store one million physical address items recently used and adopts 8-way set associative structure and least-recently-used (LRU) replacement strategy. A deep non-blocking pipeline is implemented to improve performance and bandwidth. When the total number of physical address items used by software is smaller than one million, ATC can be configured to a large buffer to achieve substantial improvement on the performance of accessing the physical address. This buffer is initialized by writing all physical address items that will be used later, and then the latency of reading these items can be largely shortened.

2.2.3 *Connection Management and Receive Path*

The connection management module implements dynamic-context based reliable end-to-end communication mechanism. There are 256 sender contexts and 256 receiver contexts. Two fully pipelined context processing engines are realized to handle data packets in non-blocking mode. In order to reduce the negative effect of connection setup on data transfer delay, the data fetch from the memory operations can overlap with the processing of connection setup, which means that data fetch can start before the connection setup.

The receive path module is used to parse network packets and forwards them to appropriate modules. The legality of network packets is checked here and any illegal packets will be discarded. Pipelined and non-blocking packet processing maximizes the bandwidth utilization of the network and the host. This module also implements the management of the receive queues of virtual ports.

3 Topology

It is important and challenging for the TH system to adopt a high performance and scalable topology. Based on high performance networking ASICs, the TH HPC system can be constructed by following almost all practical topologies used by HPCs, such as mesh, torus and fat tree. The main reason why the fat tree is chosen is that, among all candidate topologies, the fat tree can get the highest bisection bandwidth. The bisection bandwidth per node in the fat tree is always equal to the injection bandwidth of a node, no matter how large the network is and how many levels of the tree should be used.

Another merit for fat tree is that it can be used to construct high density network with lowest cost. For any subtree in a fat tree, the number of upstream ports always equals the number of downstream port. This is much less than any other candidate topologies. This feature reduces the global links, which are usually used in cabinet-to-cabinet connection and use expensive active optical cables (AOCs), to a reasonable cost. For example, a compute cabinet contains 128 compute nodes in the TH system, and thus requires at most 256 AOCs for upstream connection due to two AOCs per port.

To fit for the physical structure of TH supercomputers, TH Express-2 network uses 3-level fat tree as shown in Fig.3.

In the first level, 32 compute nodes are connected to one switch board, named NRM, in the compute frame, forming a level-1 subtree. Routing within this level-1 subtree is carried electrically via PCB traces and the mid-plane. In the second level, 12 compute frames packaged in three racks, containing a total of 384 compute nodes, are connected with one leaf switch using active optical cables and make up a level-2 subtree. In the third level, these level-2 subtrees are further connected by top level switches. Leaf switches and root switches are contained in standalone router cabinets.

All the switches, including NRM, leaf and root switches, are built by NRC chips. The level-1 switch, NRM, consists of six NRC chips that are connected to form a 2-level fat tree, including three lower ones and three upper ones. Each lower NRC chip connects up to 12 computing nodes and connects its 12 ports to the upper NRC chips. Each upper NRC chip outputs up to 12 optical ports using AOCs. Totally, one NRM connects 32 compute nodes and outputs up to 36 optical ports. These 32 computing nodes and one NRM are physically installed in a compute frame, and four compute frames physically construct a cabinet. Due to network cost-cutting, the number of output ports in one frame can be reduced according to the system size. The level-2 leaf switch consists of only one NRC chip and outputs its all 24 ports using 48 AOCs. One level-2 switch connects its 12 ports to NRMs in 12 different compute frames, or a group of three cabinets, and other 12 ports to a level-3 switch. The level-3 root switch outputs 48 optical ports, and consists of six NRC chips connected in a 2-level fat tree manner. Four of them are at the lower level and two at the upper level. Up to 48 level-2 switches can be connected to a level-3 switch.

Using this 3-level fat tree topology, up to 144 cabinets can be connected to construct a supercomputer

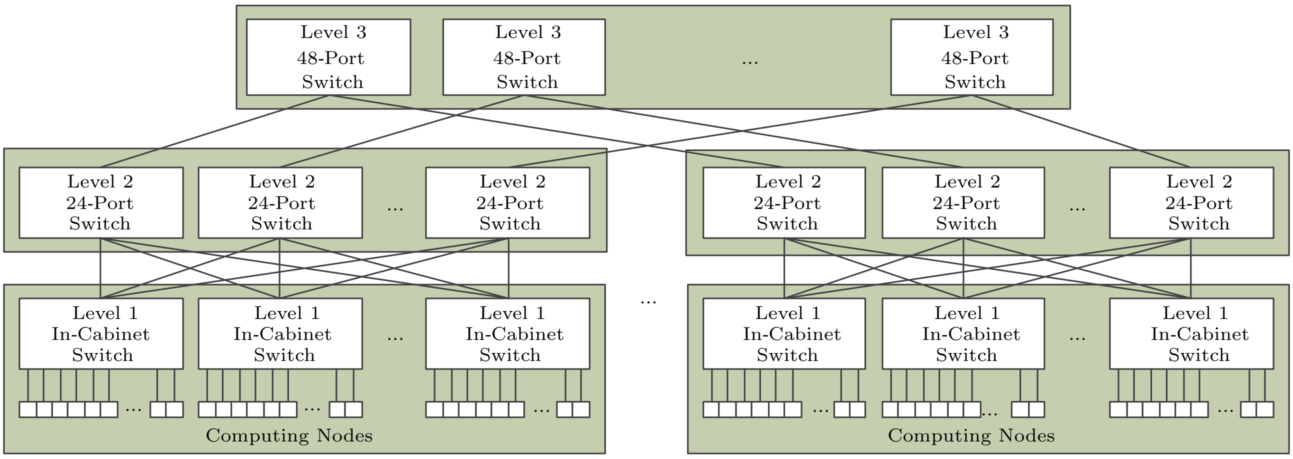

Fig.3. Topology of TH Express-2 network.

with up to 18 432 nodes. Replacing the 24-port level-2 switches with the 48-port switches, or some other customized high density switches, the system can be extended to even larger scale.

4 Protocol

The TH-2 hardware protocol hierarchy is composed of host interface protocol, transport protocol, and network protocol. Fig.4 shows the relationship of these three protocols. The host interface protocol defines how software submits requests to hardware and how software receives responses from hardware. The transport protocol defines how user data is transported between network ends. The network protocol defines how packets are transferred among the network.

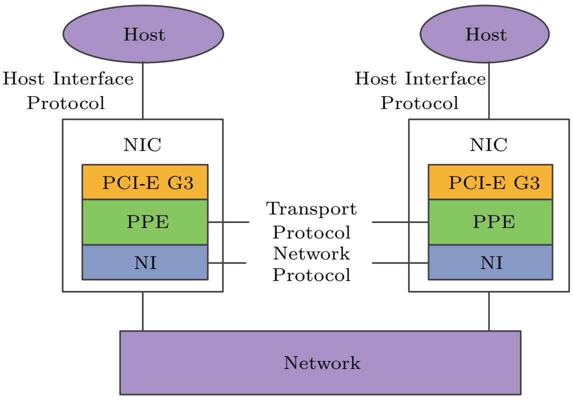

Fig.4. Hardware protocol hierarchy.

4.1 Host Interface Protocol

Hardware resources virtualization plays an important role in implementing protected user-level communication[6-8], which provides each process an exclusive programming view for using communication hardware. When several processes run concurrently, communication operations from different processes are isolated without interference.

In order to support protected user-level communications, the host interface protocol exploits a mechanism named virtual port (VP), which is a combination of a small set of memory-mapped registers and a set of in-memory data structures. All the data structures can be mapped into user space so that it can be accessed in user space concurrently with protection. The related data structures are organized in several queues, including on-chip HDQs, SDQs, mini-packet queue (MPQ), event queue (EQ), interrupt queue (INTQ), and error packet queue (EPQ).

4.1.1 Submission Queues

To support multiply processes submitting descriptor simultaneously, the host interface protocol supports up to 40 VPs in hardware. Among them, eight VPs receive descriptor through programmable IO (PIO) write and the other 32 VPs fetch descriptor through direct memory access (DMA). The two different types of VP are organized in HDQ and SDQ respectively. HDQ and SDQ have their specific advantages and it is up to system software to choose the proper VPs to submit descriptors.

HDQ is implemented in NIC directly and organized as an FIFO structure. Software maintains the write pointer of the FIFO structure and hardware maintains the read pointer. Descriptor submission through PIO is easy by writing descriptors to the HDQ queue directly via PCI-E interface. Compared with SDQ, HDQ implementation has less PCI-E transactions and relatively lower latency from submitting descriptor to processing it.

SDQ is implemented in memory and also organized as an FIFO structure. However, the steps required to fetch descriptors through DMA in SDQ are more complicated. First, the software informs NIC to fetch descriptor by writing the number of descriptors prepared in host memory to a specific register within VP. Second, NIC schedules the candidate VPs which have descriptors to submit. Third, NIC requests to read descriptor in host memory through PCI-E interface. Finally, NIC receives the descriptor and stores it in the SDQ buffer inside NIC. Compared with HDQ, SDQ could accommodate much more descriptors.

4.1.2 Receive Queues

In order to support polling hardware status in software, NIC supports managing several queues in host memory, which includes MPQ, EQ, INTQ and EPQ. MPQ and EQ are independent for each VP while INTQ and EPQ are shared by all VPs. Each memory queue is organized as a FIFO structure, its write port is managed by NIC, through which NIC can put various packets into the queue, and the read port is managed by software.

MPQ is a queue for receiving a mini-packet (MP), a short two-sided communication packet. MPs are sent by descriptors, and queued in the MPQ at the destination virtual port. To support large queue size, MPQ supports virtual address mode.

EQ is used to handle various types of events reporting the completion of communication. EQ also supports virtual addressing mode.

INTQ is leveraged to save the interrupt information, i.e., the interrupt type, source address, VP number, etc. When INTQ is not empty, NIC triggers hardware interrupt command to system software, and then the latter could access INTQ to check which type of the interrupt is triggered. NIC supports several types of interrupt, including non-empty queue, error, descriptor defined interrupt, etc.

EPQ is mainly for hardware debugging purpose. When NIC receives an error network packet, NIC can be configured to save this packet in EPQ, and then system software can fetch the error packets from EPQ to check the specific errors in network.

4.2 Transport Protocol

In transport level, TH Express-2 provides three remote memory access methods: small data transfer, block data transfer, and collective data transfer. To improve reliability further, an end-to-end reliable communication mechanism is also implemented in transport level.

4.2.1 MP

MP is used for mini-packet transfer. When the data size is no more than 120 bytes, the whole data can be put into the descriptor instead of the memory. After an MP descriptor is submitted, NIC can get the data from the descriptor and send out the packet directly. When the MP packet arrives at the destination, it will be written into the corresponding MPQ in memory. Software can detect the MP packet arriving by polling or interrupt. As the latency of source memory access is saved, MP transfer can support fast remote communication for small data.

4.2.2 RDMA

RDMA is used for offload block data transfer and supports RDMA GET and RDMA PUT operations. A RDMA communication can support maximum 128 MB data transfer.

When an RDMA PUT descriptor is submitted, NIC will read the data from the source memory. After that, the data will be packaged into a sequence of data packets and sent out to the network. After the data packets arrive at the destination, the data will be written into the destination memory. Event or interrupt can be used to indicate the completion of the transfer. If the data size is no more than 96 bytes, the data can be put into the RDMA PUT descriptor directly and immediate RDMA PUT operation will be executed. Like MP transfer, immediate RDMA PUT can transfer small data with low latency.

When an RDMA GET descriptor is submitted, NIC will package the descriptor into a request packet and send it to the destination. After the request packet arrives at the destination, the data will be read from the destination memory and be packaged into a sequence of data packets. Finally, these data packets will be returned to the source and written into the source memory. Event or interrupt can be used to indicate the completion of the transfer.

4.2.3 Collective Communication

In order to speed up collective communications in MPI, offload mechanisms are provided in transport level to accelerate collective operations[9-10]. The software is required to construct a collective algorithm tree and generate a collective descriptor sequence for NIC to initialize collective communication.

The execution of collective descriptor sequence is triggered upon receiving a special control packet. When the descriptor sequence is executed, NIC may perform a series of swap operations to modify the memory address and VP information of the descriptor in the descriptor sequence using the data from the control packet.

In the broadcast procedure, a non-leaf node in the collective algorithm tree needs to submit a series of descriptors with the same data element to transmit data to its child nodes. In order to reduce the descriptor submission transactions, NIC defines a special descriptor named collective descriptor which can be used for multiple times in hardware. In this case, the software need submit only one collective descriptor to facilitate the broadcast from parent nodes to its child nodes in a broadcast tree.

4.2.4 Reliable End-to-End Communication

Although link level retransmission can guarantee point-to-point reliability, there still exists the possibility that some errors cannot be found by link level CRC check. Moreover, soft errors happening on router chips may not be corrected by their chip level reliable mechanisms. Reliable end-to-end communication can recover the errors that cannot be solved by reliable mechanisms in link level or in router-chip level, and therefore can improve the reliability of the network further.

A dynamic context based reliable end-to-end communication mechanism is introduced in transport level and implemented in NIC. The contexts include sender contexts and receiver contexts. They are all saved on the chips and allocated on demand dynamically. A connection needs to be set up between a sender and a receiver before the sender wants to communicate with the receiver. Accordingly, a sender context and a receiver context are allocated for the connection to save the corresponding information about it. When data transfer is ended, the sender context and the receiver context will be freed to close the connection. User level CRC is used in data packets to ensure the data integrity. When user level CRC errors occur, data packets will be re-transferred. A timeout counter is used in the context to detect any packet missing error. Data packets will also be re-transferred when packet missing errors occur. The contexts provided by NIC are limited. The sender contexts constrain the maximum connections which can be set up with the receivers simultaneously and the receiver contexts constrain the maximum connections which can be accepted by the senders simultaneously. The descriptor processing will be paused when the sender runs out of the sender contexts and the connection setup requests will be NACKed and retried when the receiver runs out of the receiver contexts.

MP, RDMA, and collective transfer can use end-to-end communication mechanism to realize reliable data communication. As a connection must be set up before each transfer, end-to-end communication mechanism based transfer will incur longer latency. It is up to software to trade off between performance and reliability.

4.3 Network Protocol

The network protocol defines the uniform format of messages processed by the transport protocol, and how these messages can be packed, understood, routed, and transferred in the network. The TH Express-2 network protocol is designed for three important aims, simplicity, high efficiency, and high reliability. Similar to 7-level OSI model and TCP/IP protocol stack, the TH Express-2 network protocol is also partitioned into several levels, including physical level, link level, and routing level as shown in Fig.5.

4.3.1 PCS

The physical level protocol includes three sublayer protocols, PHY, SerDes configuration and management, and physical coding sublayer (PCS). The PHY sublayer integrates 8-lane SerDes. Each lane runs at up to 14 Gbps. Therefore, the physical bandwidth of each network port is up to 112 Gbps. SerDes lanes can be configured and downgraded to 10 Gpbs or 5 Gbps mode, mitigating signal integrity issues.

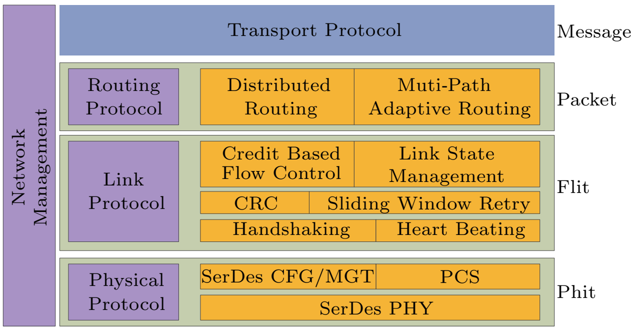

Fig.5. Network protocol stack.

PCS adopts four techniques to unify these eight lanes into a whole channel. First, it codes and decodes data to and from the PHY layer in 64/66 CODEC format. Second, it eliminates clock jitter caused by CDR of the SerDes and lane skew among all eight lanes in the same port using a small elastic buffer. Third, it tests the signal integrity of each SerDes lane using some special characters and frames. Up to four lanes can be picked out and discarded due to signal integrity problems. If four more lanes are bad, PCS should inform the SerDes configuration module to downgrade the data rate of all SerDes in the port. Last, PCS receives packet from link level, partitions packets into some flits, maps these flits into SerDes lanes left, and does flow control between link level and PHY level.

4.3.2 Link Level Protocol

The main target of the link level protocol (LLP) is to make the unreliable channel provided by the physical level into a reliable channel. Handshaking protocol negotiates working data rate between the two connected ports, picks out those decent lanes and discards those misbehavior lanes based on information provided by PCS. Each port transfers heart beating packet periodically when the link is in idle state to notify it is still alive to the other end of the link.

The packet sent by LLP is attached with a 32-bit CRC code, and this CRC code will be checked at the receive end. If some errors occur, the corrupted packet should be resent by the transmitter end using the well-known Go-Back-N sliding window retry protocol.

Four identical virtual channels (VCs) are implemented upon the physical channel in order to enhance channel efficiency. A sophisticated credit based flow control strategy is implemented to archive high efficiency with limited buffer resources. All four VCs share a receiving buffer arranged in DAMQ manner to reduce SRAM resource needed and each VC is assigned to a small part of private buffers to avoid VC deadlock. Credits are partitioned into one shared part and four private parts.

4.3.3 Routing Protocol

The TH Express-2 network is mainly designed for fat tree topology but can also support some other topologies such as mesh, torus. Each node in the network is assigned a 20-bit physical identifier (ID). Therefore, the network can support up to one million nodes. Each network port in the router chip is assigned an elastic routing table buffer (RTB). In order to support large-scale system routing with limited RTB items, RTB is arranged in three levels, and each level is assigned some configurable RTB items to support cabinet level, group level, and system level routing separately.

Two routing protocols, oblivious routing and fat tree based multi-path adaptive routing, are supported. For oblivious routing, each item in RTB includes just one active destination port, and then at each step, only one path can be selected. For multi-path adaptive routing, up to four active destination ports are included in each RTB item, and at each step, the candidate path with the least traffic load will be selected.

5 Software Support

Based on the features provided by TH Express-2 interconnect, we develop high performance message passing services to fulfill the requirement of application and system software.

5.1 Galaxy Express-2

The basic message passing infrastructure on TH Express-2 interconnect is named Galaxy Express-2 (GLEX2), which utilizes the user-level communication technology to drive the NIC network interface. With the support of the memory management unit in CPU and NIC, GLEX2 provides the protected and fully user-level communication operations, bypassing the interference of operating system in critical communication path.

The user-level interfaces of GLEX2 provide users with the MP and RDMA data transfer operation, and the RDMA operation can transfer the data in user space buffers directly. All operations are non-blocking, in order to better support the overlap of communication and computation in the applications. The kernel-level interfaces GLEX2 provides are utilized by other kernel modules for data transfer. TCP/IP driver module is implemented in the kernel, and thus the traditional network services and MPI above TCP/IP can be running on TH Express-2 interconnect. A custom gPXE module is also implemented in BIOS, which uses the TCP/IP protocol for booting the diskless compute node via TH Express-2 interconnect.

Because GPU and MIC have been used widely in high performance systems to accelerate computation, in GLEX2, we also implement the GPU-Direct technology for supporting zero-copy GPU data transfer, and the GLEX-Direct technology for supporting the MIC symmetric mode programming.

5.2 Message Passing Interface

MPI on TH Express-2 interconnect, MPICH-GLEX2, is an optimized port of the Argonne National Laboratory's MPI implementation: MPICH②. MPICH adopts a hierarchical structure with the lowest channel level performing data transmission based on specific interconnect interface. One of the most popular channel implementations is Nemesis channel[11], which specially emphasizes on a highly optimized on-node messaging system and a multi-method capable framework for implementing network modules (Netmod). MPICH2-GLEX2 mainly extends a Netmod for Nemesis, providing high performance message passing through hybrid multiple data transfer channels using MP and RDMA operations.

An MPI process can use a shared RDMA (SR) channel for message passing with all other processes. Communication resources in the SR channel are divided equally into blocks for copying and RDMA transferring the user buffers. The copying and RDMA operations will be performed in pipelining to improve bandwidth. To improve the scalability, a dynamic credit-based flow control protocol is implemented in SR channel[12-13]. Credits between processes can be increased, especially for the processes which have intensive communication between them. Thus we can implement the balance between the resource consumption and the performance improvement. An exclusive RDMA(ER) channel can also be created between two processes for message pass-

②MPICH2: High-performance and widely portable MPI. http://www.mcs.anl.gov/research/projects/mpich2/, Jan. 2015.

ing. Communication resources in the ER channel are used only by these two processes. The message sender manages the communication resources and uses RDMA PUT for data transfer. Thus the ER channel has lower latency. Some MPI applications show a nearest-neighbour communication pattern[14]. ER channels can be created between processes with frequent message passing, and the SR channel is used for communication with other processes. This can improve the performance and scalability.

Based on the RDMA features and the Long Message Transfer (LMT) protocol in the Nemesis channel, MPICH-GLEX2 supports zero-copy long message data transfer using RDMA GET operation. A user space registration cache③[15] is utilized to reduce the overhead of memory registration management in RDMA operations.

TH Express-2 interconnect adopts a load-balanced routing based on a hierarchical look-up table. In this routing protocol, data packets may be transmitted out of order through the network. Sequence number is used in all data transfer channels in MPICH-GLEX2 to recover the correct message order according to the MPI specification.

Offloaded optimization of collective communication utilizes the override interfaces of MPICH communicator. Implementation of offloaded collective communication consists of three phases: initialization, posting the operation sequences, and testing for completion. Initialization is performed on the creation of communicator, and the tree topology for the collective algorithm is constructed with the information of group members. Each process knows its parent and children in the tree, getting the addresses of virtual port and the reserved internal buffers of them. In current implementation, the tree topology can be k-nominal tree or k-ary tree. In the tree topology, data transfer in many collective communications such as barrier and broadcast can be executed by NIC automatically. Host CPU can be offloaded to do other computations. This can avoid the affection of system noise and reduce the latency of MPI collective communication.

6 Performance Measurements

Some performance results of TH Express-2 interconnect network are depicted in this section. In the testing, each compute node is equipped with two Intel Xeon E5-2660 processors at 2.2 GHz with 64 GB memory. Benchmark programs include both the customized benchmarks we developed using GLEX2 interfaces and the OSU MPI benchmarks developed by Ohio State University.

6.1 Point-to-Point Performance

Based on the user-level interfaces of GLEX2, we conduct Ping-Pong tests to measure the latency between neighbor nodes under different communication mechanisms.

The minimum point-to-point latency, 0.76 µs, can be acquired when payloads are embedded into the descriptor and are written into NIC using PIO directly, that is, using PIO virtual port and immediate RDMA PUT operation. Any other configuration modes will accumulate other factors that affect latency and lead to higher latencies. For example, when testing using DMA virtual port and immediate RDMA PUT, the latency is 0.96 µs. Additional latency is induced because the descriptor and payload data are fetched through another DMA operation after the PIO operation.

MPI latency between two neighbour compute nodes under two test situations is depicted in Fig.6. The first one uses immediate RDMA PUT through the ER channel to transfer short messages, denoted by PIO-ER, and the second one uses MP through the SR channel to transfer short messages, denoted by PIO-SR. PIO-ER shows a slightly shorter latency than PIO-SR.

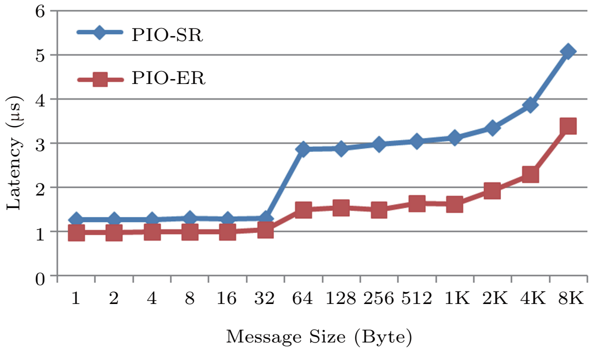

Fig.6. MPI latency between neighbour nodes.

As more and more cores are integrated in one processor and one compute node, the number of MPI processes per node increases rapidly. We test MPI latency between two nodes with multiply processes per node using *osu_multi_lat* benchmark, as shown in Fig.7. When the number of MPI processes increases from 2 to 14, the average MPI latency almost keeps the same when

③MVAPICH: MPI over InfiniBand, 10GigE/iWARP and RoCE, 2013. http://mvapich.cse.ohio-state.edu/, Jan. 2015.

the message length is below 4 KB. The test result indicates that high message rate can be achieved in TH Express-2 interconnect network.

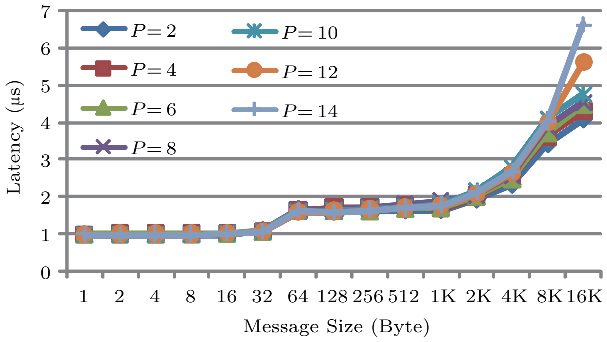

Fig.7. Average MPI latency vs number of MPI processes.

The diameter of fat tree network in TH-2 is 9 hops. Communication between a pair of nodes may go through 1, 3, 5, 7 or 9 hops. We have tested the latency for varying number of hop distances and the results are shown in Table 1. According to the test result, one-hop latency is approximately 100 ns. Some additional latency is induced by long distance transport through optical fiber.

Table 1. Multi-Hop Latency

Hop	Latency (ns)	Fiber Length (m)
1	760	0
3	952	0
5	1 254	10×2
7	1 659	10×2+20×2
9	1 863	10×2+20×2

Fig.8 depicts test results of unidirectional MPI bandwidth between neighbour nodes. Similar to latency tests, the ER channel gets better bandwidth for short messages than the SR channel due to lower protocol overhead. For long messages, the ER and the SR channel show the same result by using zero-copy rendezvous protocol. The peak bandwidth between two neighbour nodes is 12 005 MB/s.

Fig.9 shows test results of bidirectional MPI bandwidth between two neighbour nodes. The ER channel still gets better performance in short messages than the SR channel. The peak bidirectional bandwidth is 23 200 MB/s. The test implies that no performance downgrade could occur in bidirectional communication in TH Express-2 network.

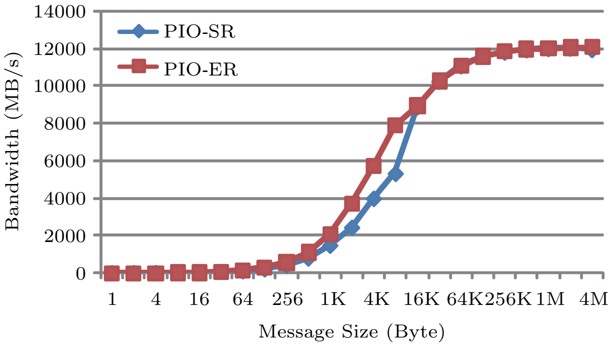

Fig.8. Unidirectional MPI bandwidth.

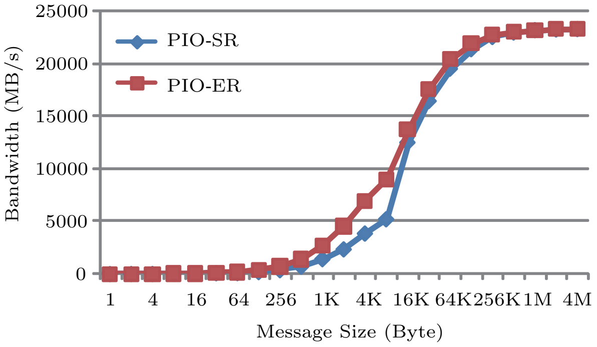

Fig.9. Bidirectional MPI bandwidth.

6.2 Aggregated Bandwidth

We run OSU all-to-all performance tests in different network scales and message sizes. Due to path collision in fat tree topology, optimum results cannot always be achieved. We first test the peak aggregated bandwidth using some node pairs specially selected according to our routing strategy such that no path collision could occur under this situation. The result shows that bandwidth per node is 11.8 GB/s, nearly the same as the peak bandwidth we can get. This means there are no resource bottlenecks inside the NRC switch fabric. 100% throughput can be achieved and the transfer performance of data throughput will not downgrade when flow switching mode is configured, i.e., one input port only transmits data to one output port. We further test the aggregated bandwidth using random node pairs without any path optimization, as shown in Fig.10. The result shows that nearly 68.75% of peak bandwidth can be obtained under random flow traffics.

6.3 Collective Communication Offload

Some collective operations such as barrier and broadcast are supported in TH Express-2 interconnection network. For MPI-barrier, a k-ary tree or a

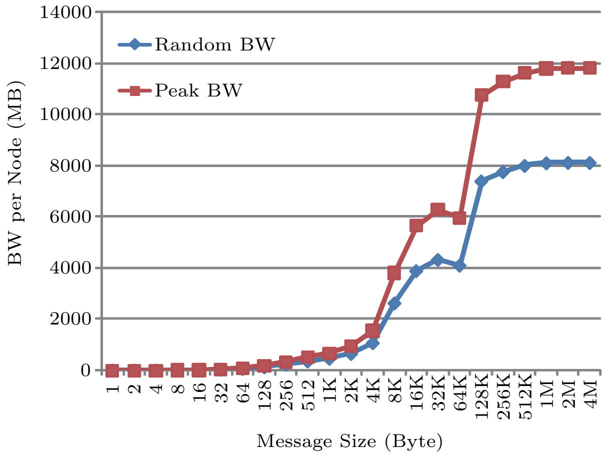

Fig.10. Aggregated bandwidth.

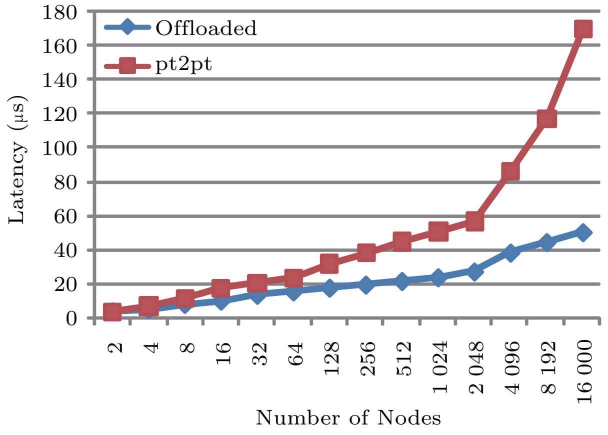

Fig.11. MPI-barrier latency.

k-nominal tree can be constructed in the interconnect network, and synchronization operation between two nodes in the tree is offloaded into the NIC chip and implemented with triggered MP operations automatically. Fig.11 depicts the MPI-barrier latency test result using the algorithm of k-ary tree. The result shows the offloaded barrier implementation gets better performance than that through point-to-point (pt2pt) message passing, and the acceleration ratio increases rapidly especially when nodes included in the barrier operation are at a large scale.

7 Related Work

Interconnection networks keep being a highly innovative area, and striving to keep pace with rapidly increasing levels of concurrency and greater demand for communication performance. User-level operations and zero-copy data transfer are two key state-of-the-art techniques for high performance communication. Most existing interconnect systems adopt these techniques, such as IBM BlueGene/Q[16-17], Fujitsu Tofu[18], Cray Gemini[19], and InfiniBand④. Through the user-level operation and improved logic design, we achieve rather small point-to-point latency in TH Express-2 interconnect.

Reliability becomes more and more crucial for future large-scale computer systems, especially for highly complicated interconnect[20]. Gemini interconnect does not provide end-to-end reliability in hardware, while it resorts to software protocol to maintain reliability. In InfiniBand, a one-to-one connection is required between two processes to deliver end-to-end reliable data transfer, which requires more resources at large scale. In the new generation of TH Express-2 interconnect, we introduce a reliable end-to-end communication in NIC, while simplifying the implementation of scalable message passing services and requiring less resource at large scale. The efficiency of hardware reliability will be further evaluated in the future.

Hardware-assisted collective operations have been investigated extensively in literature. Some of the recent approaches in InfiniBand are [9, 21-22], which optimize collective communication at different layers in the interconnect fabric. In PERCS[23], a special collective acceleration unit is used to speed up collective operation. Our previous work[3] demonstrates that we can achieve good performance results using NIC-assisted collective operations. We believe it is necessary to explore hardware-software co-design to achieve the best result, especially exploiting the hardware offload collective to overlap computation and communication.

8 Conclusions

The performance of HPC systems is often limited by network bandwidth and latency characteristics. TH Express-2 network is designed for applications with high bandwidth and low latency requirements. In this paper, we described the architecture of TH Express-2 network in terms of system topology, network router and interface chips, and message passing services.

The high-radix router chip, NRC, has 24 8-lane ports with the bi-directional bandwidth of 224 Gbps, and delivers an aggregate bandwidth of 5.376 Tbps. NRC uses a hierarchical organization to mitigate the buffer requirements, and uses some novel technologies, such as dynamic buffer allocation, adaptive routing

④InfiniBand architecture specification, volume 1, 2013. http://www.infinibandta.org/, Jan. 2015.

based on hierarchical look-up table, intelligent network management, low-latency scrambler, improved rolling CRC, and so on.

Network interface chip, NIC, provides an interface between software and hardware, enabling applications to access the high-performance network efficiently. It contains a full width 16-lane PCI-E 3.0 interface connected to the compute node. NIC achieves several advanced mechanisms to support scalable high performance computing, including protected user-level communication, RDMA, offloaded collective mechanism, etc.

Reliability becomes more and more important to the high performance interconnect. TH Express-2 network provides a lot of management techniques to improve its RAS capability. The interconnect fabric achieves reliable link-level packet delivery through link-level CRC and packet re-transmission. NIC provides a reliable end-to-end connection mechanism. These RAS features enable both real-time and historical status monitoring and facilitating fault locating.

The system software for message passing services is optimized to efficiently utilize the new generation of TH Express-2 network. GLEX2 communication system provides the basic message passing infrastructures for other software subsystems. An optimized MPI library, MPICH2-GLEX2, is implemented, and extends a Netmod for Nemesis, providing high performance communication with hybrid MP and RDMA data transfer.

In the future, more efforts are needed to study the emerging interconnect topologies and routing protocols for future larger-scale Tianhe systems. It is also important to integrate more network functionalities into a single chip.

Acknowledgements The Tianhe-2 project is a great team effort and benefits from the cooperation of many individuals at NUDT. We would like to thank the entire Tianhe-2 development, applications, and benchmarking teams, and all the people who have contributed to the system in a variety of ways.

References

[1] Liao X, Xiao L, Yang C *et al.* Milkyway-2 supercomputer system and application. *Frontiers of Computer Science*, 2014, 8(3): 345-356.

[2] Pritchard H, Gorodetsky I, Buntinas D. A uGNI-based MPICH2 Nemesis network module for the cray XE. In *Proc. the 18th European MPI Users' Group Conference on Recent Advances in the Message Passing Interface*, Sept. 2011, pp.110-119.

[3] Xie M, Lu Y, Liu L *et al.* Implementation and evaluation of network interface and message passing services for TianHe-1A supercomputer. In *Proc. the 19th IEEE Annual Symposium on High Performance Interconnects*, Aug. 2011, pp.78-86.

[4] Kim J, Dally W J, Towles B, Gupta A K. Microarchitecture of a high radix router. In *Proc. the 32nd Annual International Symposium on Computer Architecture*, June 2005, pp.420-431.

[5] Schoinas I, Hill M D. Address translation mechanisms in network interfaces. In *Proc. the 4th International Symposium on High-Performance Computer Architecture*, Feb. 1998, pp.219-230.

[6] Chun B N, Mainwaring A, Culler D E. Virtual network transport protocols for Myrinet. *IEEE Micro*, 1998, 18(1): 53-63.

[7] Araki S, Bilas A, Dubnicki C *et al.* User-space communication: A quantitative study. In *Proc. ACM/IEEE Conference on Supercomputing*, Nov. 1998.

[8] Bhoedjang R A F, Ruhl T, Bal H E. User-level network interface protocols. *Computer*, 1998, 31(11): 53-60.

[9] Graham R L, Poole S, Shamis P *et al.* Overlapping computation and communication: Barrier algorithms and ConnectX-2 CORE-Direct capabilities. In *Proc. IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum*, April 2010.

[10] Kandalla K, Subramoni H, Vienne J *et al.* Designing nonblocking broadcast with collective offload on InfiniBand clusters: A case study with HPL. In *Proc. the 19th IEEE Annual Symposium on High Performance Interconnects*, Aug. 2011, pp.27-34.

[11] Buntinas D, Goglin B, Goodell D *et al.* Cache-efficient, intranode, large-message MPI communication with MPICH2-Nemesis. In *Proc. International Conference on Parallel Processing*, Sept. 2009, pp.462-469.

[12] Lauria M, Pakin S, Chien A. Efficient layering for high speed communication: Fast messages 2.x. In *Proc. the 7th International Symposium on High Performance Distributed Computing*, July 1998, pp.10-20.

[13] Liu J, Panda D K. Implementing efficient and scalable flow control schemes in MPI over InfiniBand. In *Proc. the 18th International Parallel and Distributed Processing Symposium*, April 2004.

[14] Vetter J S, Mueller F. Communication characteristics of large-scale scientific applications for contemporary cluster architectures. *Journal of Parallel and Distributed Computing*, 2003, 63(9): 853-865.

[15] Tezuka H, O'Carroll F, Hori A *et al.* Pin-down cache: A virtual memory management technique for zero-copy communication. In *Proc. Symposium on Parallel and Distributed Processing*, Mar. 30-Apr. 3, 1998, pp.308-314.

[16] IBM Blue Gene team. The IBM Blue Gene project. *IBM J. Res. Dev.,* 2013, 57(1/2): 0:1-0:6.

[17] Chen D, Eisley N A, Heidelberger P *et al.* The IBM Blue Gene/Q interconnection fabric. *IEEE Micro*, 2012, 32(1): 32-43.

[18] Ajima Y, Inoue T, Hiramota S *et al.* The Tofu interconnect. *IEEE Micro*, 2012, 32(1): 21-31.

[19] Alverson R, Roweth D, Kaplan L. The Gemini system interconnect. In *Proc. the 18th IEEE Symposium on High Performance Interconnects*, Aug. 2010, pp.83-87.

[20] Schroeder B, Gibson G. Understanding failures in petascale computers. *J. Physics: Conference Series*, 2007, 78: 012022.

[21] Graham R L, Poole S, Shamis P *et al.* ConnectX-2 InfiniBand management queues: First investigation of the new support for network offloaded collective operations. In *Proc. the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, May 2010, pp.53-62.

[22] Subramoni H, Kandalla K, Sur S *et al.* Design and evaluation of generalized collective communication primitives with overlap using connectX-2 offload engine. In *Proc. the 18th IEEE Annual Symposium on High Performance Interconnects*, Aug. 2010, pp.40-49.

[23] Arimilli B, Arimilli R, Chung V *et al.* The PERCS high-performance interconnect. In *Proc. the 18th IEEE Symposium on High Performance Interconnects*, Aug. 2010, pp.75-82.

Xiang-Ke Liao received his B.S. degree from Tsinghua University, Beijing, in 1985, and M.S. degree from National University of Defense Technology (NUDT), Changsha, in 1988, both in computer science. Currently he is a professor and the dean of College of Computer, NUDT. His research interests include high performance computing systems, operating systems, and parallel and distributed computing. Prof. Liao is a fellow of CCF.

Zheng-Bin Pang received his B.S., M.S., and Ph.D. degrees in computer science from the National University of Defense Technology (NUDT), Changsha. He is a professor in the College of Computer, NUDT, Changsha. His research interests include parallel and distributed computing, and high performance computer systems.

Ke-Fei Wang received his B.S., M.S., and Ph.D. degrees in computer science from the National University of Defense Technology (NUDT), Changsha. He is a professor in the College of Computer, NUDT. His research interests include parallel and distributed computing, and high performance computer systems.

Yu-Tong Lu received her M.S. and Ph.D. degrees in computer science from National University of Defense Technology, Changsha. Currently she is a professor at the university. Her research interests including parallel system management, high speed communication, distributed file systems, and advanced programming environments with MPI.

Min Xie is a professor in the College of Computer at the National University of Defense Technology, Changsha. His research interests include high-speed interconnect, system software and the parallel and distributed computing. Xie got his Ph.D. degree in computer science from the National University of Defense Technology.

Jun Xia received his B.S. and Ph.D. degrees in computer science from the National University of Defense Technology (NUDT), Changsha. He is an associate professor in the College of Computer, NUDT, and a director designer of Tianhe-2 supercomputer. His research interests include parallel and distributed computing, and high performance computer systems.

De-Zun Dong received his B.S., M.S., and Ph.D. degrees from the National University of Defense Technology (NUDT), Changsha, in 2002, 2004, and 2010, respectively. He is an associate professor in the College of Computer, NUDT. His research interests range across high performance computer systems, high speed interconnect networks, wireless networks, and distributed computing algorithms. Currently, he focuses on performance evaluation of high-performance interconnection networks for supercomputers and data centers. He is a member of the ACM, IEEE, and CCF.

Guang Suo received his B.S., M.S., and Ph.D. degrees all in computer science from National University of Defense Technology (NUDT) in 2003, 2005, and 2009 respectively. He is an assistant professor in the College of Computer, NUDT. He has played an important role in the implementation and optimization of MPI library of Tianhe supercomputers. His research interests are in parallel computing, operating system and HPC runtime systems.