BCL-3: A High Performance Basic Communication Protocol for Commodity Superserver DAWNING-3000

Ma Jie(马捷)   He Jin(贺劲)   Meng Dan(孟丹)   Li Guojie(李国杰)

Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, P. R. China

{majie, jhe, md, lig}@ncic.ac.cn

Abstract

This paper introduces the design and implementation of BCL-3, a new high performance low level communication software running on a cluster of SMPs (CLUMPS) called DAWNING-3000. BCL-3 provides flexible and sufficient functionality to fulfill the communication requirements of fundamental system software developed for DAWNING-3000 while guaranteeing security, scalability, and reliability. Important features of BCL-3 are presented in the paper, including special support for SMP and heterogeneous network environment, semi-user-level communication, reliable and ordered data transfer and scalable flow control. Performance evaluation of BCL-3 over Myrinet is also given.

Keywords: Cluster Communication System, User-level Communication, Scalable Flow Control, CLUMPS

1. Introduction

Clusters have been popular platforms for high performance computing in recent years. They are widely used in scientific and engineering computing, business computing, and Internet information services. Communication performance is one of the most critical factors determining the performance of a whole cluster system. So how to improve the performance of communication is a hot research topic in cluster computing. Meanwhile, building a cluster with commodity SMPs (CLUMPS) is becoming trend since this approach can increase the packing density of system while reducing cost. CLUMPS provide different hardware for intra and inter-node communication, that is, shared memory and message-passing network. Communication software should take the difference into account and provide corresponding protocols so that the capacity of underlying communication hardware can be fully used for various applications[1].

BCL-3 is a high-performance low-level communication protocol designed and implemented on DAWNING-3000. It provides reliable and ordered message passing with flow control and error correction. DAWNING-3000 is a CLUMPS consisting of 70 nodes. Each node is a 4-way 375MHz Power3 SMP, running IBM AIX4.3.3. All nodes are connected with Fast Ethernet, Myrinet and Dnet. Dnet is high performances interconnect developed by us. Both Myrinet and Dnet are supported by BCL-3. It consists of a library in user space, a device driver in kernel space, and a control program called MCP running on network interface. A few important features of BCL-3 are described as follows:

This work is supported by National '863' High-Tech Program of China.

- Flexible and sufficient functionality

BCL-3 provides a channel-based asynchronous communication mechanism, which supports point-to-point message-passing and remote memory access (RMA) operations. It provides efficient communication support for the implementations of high-level communication software and other system software on DAWNING-3000, including PVM, MPI, JIAJIA (a software DSM) and a cluster file system called COSMOS. BCL-3 also implements automatic node status detection, node isolating/rejoining functionality, and application program exception handling to guarantee system availability and stability. Although more functionality can decrease communication performance, especially latency, it is necessary for a commodity system.

- Support for SMPs

BCL-3 provides different protocols for intra and inter-node communication respectively. Intra-node communication is implemented via shared memory while inter-node via message-passing network. This improves intra-node communication performance dramatically.

- Zero memory copy and pin-down cache

BCL-3 implements direct user-to-user data transfer within inter-node communication. Sending and receiving data buffers provided by users can be pinned down in physical memory and therefore become DMA-able during communication process. Pin-down Cache is implemented, which can reuse the pinned down area. All these make contributions to bandwidth improvement. BCL-3 provides almost all hardware bandwidth to its users.

- Semi-user-level communication

BCL-3 implements user-level communication on receiving side. But on sending side, there is need for senders to trap into OS kernel and invoke message-sending operations. Compared with user-level communication, only one kernel trapping is incorporated into the communication path. Although it can increase communication latency a little bit, this combination has a few advantages, which will be discussed in later section.

- Heterogeneous network environment support

Because the BCL-3 library in user space is independent of underlying network interface, binary code written in it or in a high-level communication library such as PVM, MPI, and JIAJIA on top of it can run on any combination of networks. Applications written in BCL-3 need not be recompiled. This feature is especially useful for applications running over a cluster of clusters.

- Scalability and reliability

BCL-3 adopts an ACK/NAK mechanism to guarantee reliable and ordered data communication. ACK/NAK-based flow control improves the scalability in comparison with credit-based flow control. Memory consumption is no longer proportional to system scale. Currently, ACK/NAK mechanism is implemented on network interfaces of Myrinet and Dnet.

Communication performance of BCL-3 has been measured on DAWNING-3000 over Myrinet. As a reliable and ordered message passing protocol, it provides 2.7μs one-way latency and 391MB/s bandwidth of intra-node communication while 18.3μs and 146MB/s of inter-node's. Performance of MPI on top of BCL-3 is also presented.

This paper introduces the design and implementation of BCL-3 over Myrinet in detail. Then the evaluation of BCL-3 on DAWNING-3000 platform is shown. Finally, we present our conclusions and discuss ongoing research.

2. Design issues

- Communication services

Services provided by a communication system can greatly impact its performance. Many communication protocols only provide very limited services and show excellent performance. However, limited services will make the implementation of other system software on high layers more difficult. BCL-3 provides reliable and ordered message passing with flow control and error correction. Although reliable and ordered delivery will reduce the performance of BCL-3, it will reduce more performance when it is implemented on higher layers.

Furthermore, RMA and special message arrival inform mechanism are also provided to support DSM and distributed file system. RMA makes it possible to deliver message without cooperation on the other side. It is essential for software DSM and distributed file system. Also, BCL-3 provides "select" mechanism to inform the arrival of message so that the application can use BCL-3 message passing as a socket. It makes the applications using TCP/IP easy to be ported to using BCL-3.

- Semi-user-level communication

In traditional communication protocols, such as TCP/IP, the communication main path involves OS kernel trapping and interrupts handling. All these overheads cause high end-to-end message latencies. User-level communication allows applications directly access the network interface cards (NIC) without operating system intervention on both sending and receiving sides. Messages are transferred directly to and from user-space by the network interface card while observing the traditional protection boundaries between processes. It reduces the communication overhead by avoiding kernel traps on the data path.

However, in commodity systems, security is the most important requirement. User level communication protocol exposes all the control data structures to user space so that any mistake or malice operation will cause the system broken. Kernel level communication protocol protects the important data structures in kernel space.

To provide a low-latency and secure communication, BCL-3 uses a combined method. It ensures the system security by protecting the most important data structures in kernel space and reduces communication overhead by diminishing some unnecessary kernel traps. Furthermore, it can make the application binary portable, suitable in heterogeneous network environment and more efficient. BCL-3 is separated to two parts, user level library and kernel extension. All the system-related operations are hidden in the kernel extension, so that the user level library can be the same on different platforms. Application, which linked the same user level library, can be ported to different platforms with binary code. Also, the difference of heterogeneous network environment is hidden in the kernel extension.

- Communication mechanism and semantic

There are three commonly used message passing modes, synchronous message passing, blocking message passing and non-blocking message passing. It is clearly that synchronous and blocking message passing can be implemented by non-blocking message passing. So BCL-3 only provides non-blocking sending and receiving semantics.

Channels are used to identify the message transfer paths in BCL-3. System channel is used to transfer small messages. Rendezvous semantic is used to transfer large messages. Rendezvous means the send is guaranteed to complete only if a corresponding receive has been posted. RMA operations are presented in BCL-3 by using a special type of channel. A message arrival inform mechanism, which is compatible with TCP/IP socket, is also provided so that application can use "select" system call to waiting for the incoming message.

- Reliability and scalability

Although the reliable protocol can be implemented in the higher levels, it is more efficient to implement the reliable protocol in this communication layer. BCL-3 provides a reliable and ordered message passing data path. It uses an ACK/NAK based reliable protocol. By using CRC verification and sequential number checking on each packet transferred on the net, BCL-3 can ensure the correctness and order of the packet. Since the communication hardware is highly reliable, this ACK/NAK based protocol is very efficient on our platform. Compared with credit based reliable protocol, the ACK/NAK based protocol is scalable. It is easily to be used in large-scale cluster.

- Zero memory copy and buffer management

To implement zero memory copy, user buffer should be pinned in physical memory and the virtual address should be translated to physical address before starting communication. This operation will increase the communication overhead. In order to reduce the communication overhead, pin-down cache[2] and three types of buffer are introduced in BCL-3. They can reduce the overhead and improve the system performance. These two techniques will be discussed in detail in the next section.

- Special support for SMPs

In the context of SMPs, communication protocol needs to support several processes on one node. When the memory copy bandwidth is lower than the network bandwidth, it will provide a high performance to transfer messages via NIC even if the two processes are in the same node. But the memory copy bandwidth is improved heavily now. It is costly and has no meaning to transfer intra-node messages as the same manner to transfer inter-node messages. BCL-3 uses direct memory copy instead of transfer via network. Shared memory and direct copy from another process's memory space can be used to implement intra-node message passing. BCL-3 uses the former mechanism, which will be discussed later[3][4][5].

3. BCL-3 protocol

BCL-3 is a low level communication software used on DAWNING-3000. It has two versions. One is implemented on Myrinet and the other is implemented on Dnet. The upper levels can make full use of the hardware performance via

BCL-3. BCL-3 supports point to point message passing. All other collective message passing should be implemented in the higher level software.

3.1. Architecture

Figure 1 is the protocol stack from BCL-3 applications' point of view. Applications can directly access the BCL-3 level or use other functionality levels. BCL-3 is the lowest level software in DAWNING-3000's communication software. MPI is implemented directly on BCL-3. PVM is implemented on ADI-2 (Abstract Device Interface) of MPI. Notice that this may be changed according to the different implementations that are done. In order to keep high performance, we can also port PVM directly on top of BCL-3. TCP/IP is designed to port onto our BCL-3 in the next step. A prototype will be completed early next year. The other two components in the stack are JIAJIA and COSMOS. The former is a DSM software, and the latter is a distributed file system of DAWNING-3000.

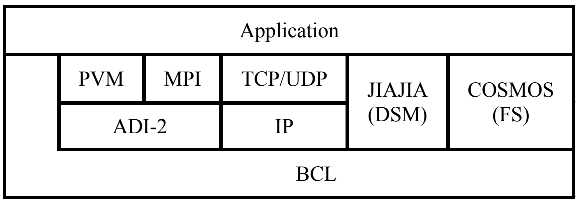

Figure 1 Protocol stack of DAWNING-3000 software

BCL-3 is divided to three parts, the on-card control program (MCP, Myrinet Control Program), the device driver (DD) and the BCL library. Myrinet has an on-card processor, which can control the card to transfer packets by using three DMA engines. In BCL-3, MCP controls all the inter-node packet transfers. MCP completes a sending operation by reading send request in the card's local memory, sending/receiving message with DMA engines and informing user process the completion. Device driver is a kernel extension, which provides several *ioctl()* calls to control the hardware. It posts operation requests to the on card memory. These requests include the sending requests and other requests, such as initialize/close communication port request and initialize channel requests. Device driver also implements some functional operations, which need to be executed in the kernel environment. Such operations include the memory pin/unpin operation and physical memory address conversion. BCL library includes all implementations of BCL API.

3.2. Communication mechanism

Communication is occurred between ports. Each process can create only one port to communicate with others. A process is labeled with its node number and port number. The pair of node number and port number is the unique identifier of the process within the application. Each port has a sending request queue, a receiving buffer pool and the corresponding event queues (sending event queue and receiving event queue). Receiving buffer pool is composed of several channels. There are three types of channels, system channel, normal channel and open channel.

When a process (sender) wants to send a message to another process (receiver), the sender should compose a send request and put it into the send request queue. Destination is specified by its node number, port number and channel number in the request. The receiving channel should be ready before the message arrived the receive side. A receive event will be generated when a complete message arrived. On the send side, a send event will be put into the send event queue when a sending operation is over.

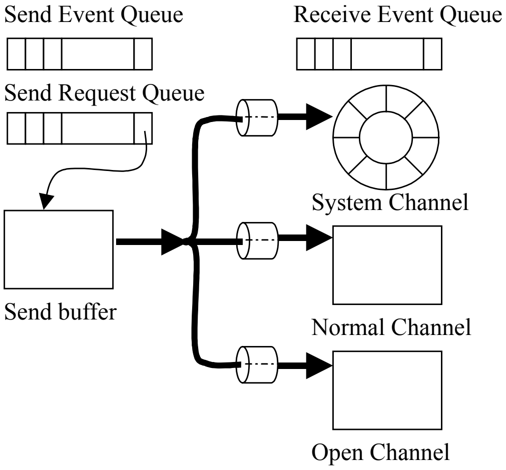

Figure 2 Communication mechanism of BCL-3

As shown in Figure 2, there are three types of channels: system channel, normal channel and open channel. System channel is designed to transfer small messages. Each process has one system channel. Every system channel has a buffer pool, which is initialized when the process starts. It is organized as a FIFO queue. When a short message is arrived, it is automatically put to the first free buffer in the buffer pool. The incoming message will be discarded if there is no free buffer in the pool. After the receiver gets the message, the buffer will be returned to the buffer pool.

Normal channel is designed to transfer messages with rendezvous semantic. Each process has several normal channels. A normal channel has a user-specified buffer. The receiving process needs to prepare the receive channel before the sending process start transmission. A receive event will be generated when a message completed. A normal channel should be initialized before each receiving operation.

Open channel is designed to perform RMA. Each process has several open channels. An open channel has a user-specified buffer. Unlike the normal channel, only once does it need to be initialized before receiving operations. A receiving event will be generated when a RMA operation completed.

3.3. Buffer management

A zero copy message transfer is realized with the help of the Myrinet DMA capability. The user program accesses only virtual addresses while the DMA engines only access physical memory addresses. To realize DMA-issued message transfer, both the sending and receiving buffers must be pinned in physical memory and the physical memory addresses must be known. The pin-down operation is costly in most operating systems. This overhead can be reduced if an application repeatedly transfers data from the same memory area without releasing the pinned down area. When a request to release a pinned down memory area is issued, the actual release operation is postponed until total physical memory reserved for pinned down area is used up. If the release primitive for pinned down area has been issued but in fact the area remains pinned down, a subsequent request to pin down the same area can be answered immediately. This technique is called pin-down cache.

While using pin-down cache technique, there still is a cache-hit overhead. BCL-3 avoids it by using pinned buffer in message passing. If an application needs to transfer data from the same memory area repeatedly, it can request to pin down the buffer before transfer. The pinned down buffer should be released after all transmissions. BCL-3 defines three types of buffer: normal buffer, enabled buffer and DMA buffer. A normal buffer is a user buffer that isn't pinned. The communication primitives need to perform pin/unpin operation when using this type of buffer. An enabled buffer is a user buffer that is already pinned in physical memory. A DMA buffer is a system buffer allocated by a special call, which is pinned in physical memory. There is no need to check the pin-down cache before transmission when using the last two types of buffer. It reduces the overhead when using pin-down cache.

3.4. Intra-node communication

There are several ways to move data from one process to another process within one node (Figure 3). The traditional way is to move data as the same way as between nodes. Process A first transfers the data to NIC (Network Interface Circuit) by DMA. Then NIC transfers them back to process B. While the memory copy bandwidth is much higher than DMA bandwidth, a good solution is to use shared memory to implement intra-node communication. Process A first copies the data to a shared memory area. Then process B copies them out. Another way is to move data directly from user space to user space.

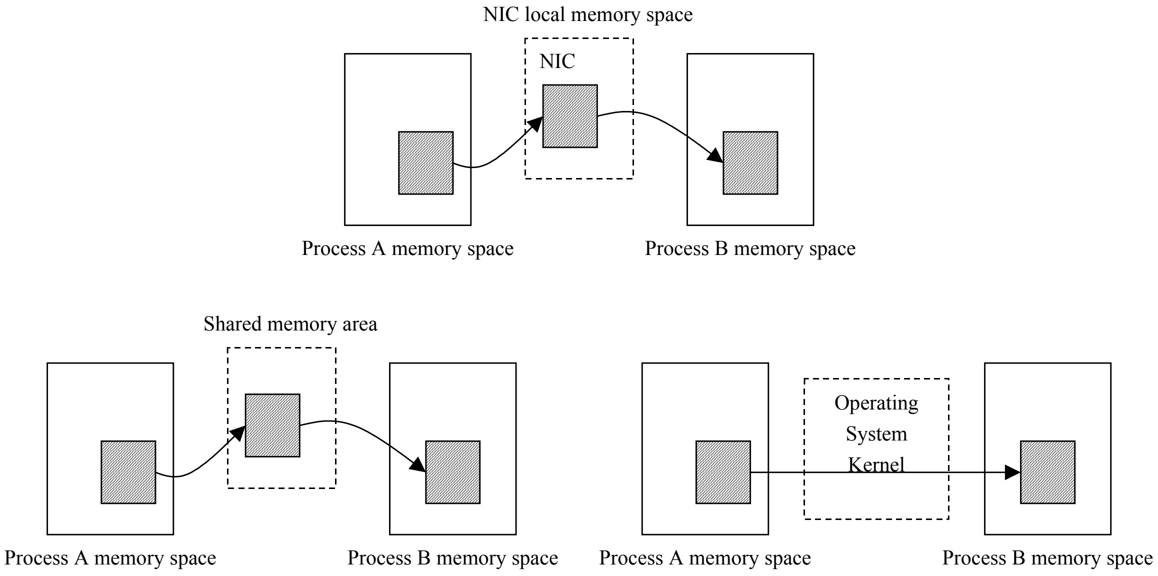

Figure 3 Several ways to move data between processes within one node

Shared memory based intra-node communication can make the system more reliable. Any mistake or malice operation during a directly inter-process memory access can cause the target process crashed. By using shared memory, the sender process can only destroy the shared area. It won't affect other processes' space. This guarantees the independence of each process.

BCL-3 uses shared memory based intra-node communication. The internal buffer queue is used to transfer message from one process to another process within a node. This queue consists of a list of buffers. Each pair of processes has two queues. (Figure 4). To ensure the message sequence, BCL-3 uses the sequential number to decide whether the operation should continue or not. Shared memory based intra-node communication needs an extra memory copy than the direct memory copy solution. BCL-3 reduced the extra overhead by using the pipeline message passing technique.

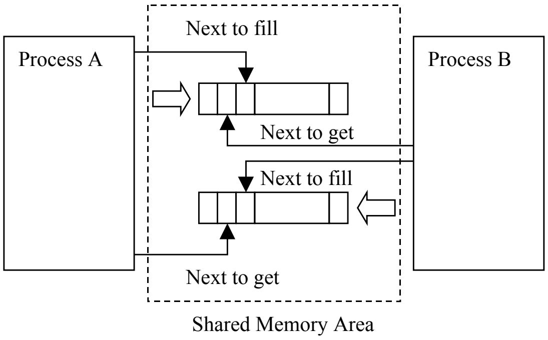

Figure 4 Intra-node communication in BCL-3

3.5. Flow control

BCL-3 provides reliable and ordered delivery. The receiving process will simply discard the packet when it detected an error. The sending process will retransmit it when timeout. When a source node sends two messages to the same destination, a receiving operation executed firstly at the destination will receive the first message. A sequential number based ACK/NAK protocol is used in BCL-3 to implement reliable and ordered delivery. Considered the high reliability of the network hardware, this algorithm is efficient.

Flow control is used to avoid buffer overflow and system deadlock. In BCL-3, the receiving process simply discards messages when there is no enough buffers. The sending process will re-send them when timeout. An overflowed packet is handled as an error packet.

4. Performance and analysis

All the tests[6] are done on our DAWNING-3000, which consists of 70 IBM270 workstations. Each node is a 4-way 375MHz Power3 SMP, which is running IBM AIX4.3.3. Myrinet[7] M2M-PCI64A NICs are used on each node. These nodes are interconnected by M2M-OCT-SW8 switches.

The first tests are raw point to point communications. Latency and bandwidth are measured on our DAWNING-3000. Both inter-node and intra-node communications are tested. The result is shown in Figure 5 and Figure 6. The minimal latency is 2.7μs within one node and 18.3μs between nodes. The bandwidth is 391MB/s within one node (with the affect of cache) and 146MB/s between nodes. And the half-bandwidth is reached with less than 4KB message.

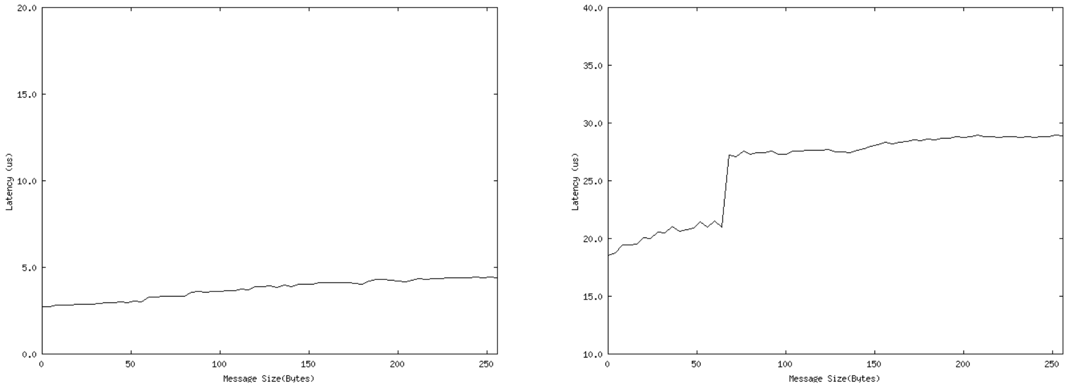

Figure 5 Intra-node (left) and inter-node (right) latency of BCL-3

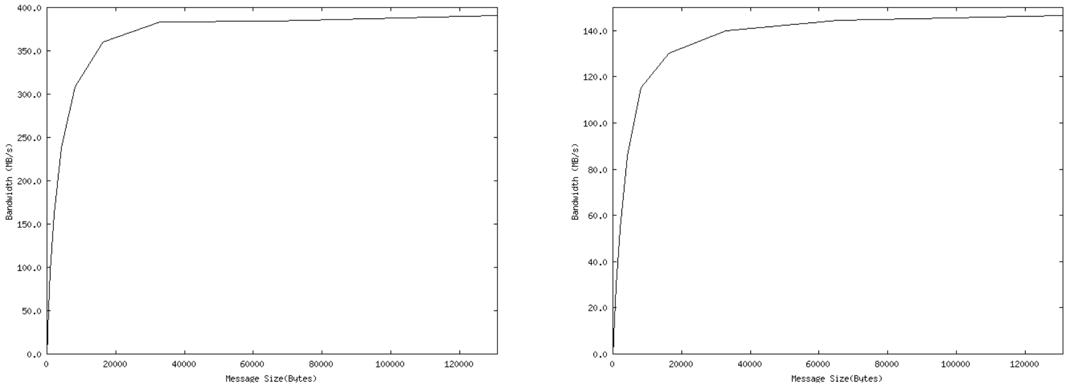

Figure 6 Intra-node (left) and inter-node (right) bandwidth of BCL-3

Table 1 shows the comparison of three protocols. GM is a message-based communication system for Myrinet, which is designed and implemented by Myricom. GM doesn't provide special support for SMP. Only inter-node communication performance data are given in the table. The range for GM's one-way short-message latency on a wide variety of hosts is from 13.37 to 21μs. And the peak bandwidth is over 140MB/s. BCL-3 reaches almost the same performance and provides a more reliable (using kernel level communication) and more complex (special support for SMP) protocol.

Table 1 Comparison of different communication protocols

Protocol	Latency(μs)		Bandwidth(MB/s)	
	Intra-node	Inter-node	Intra-node	Inter-node
GM	—	13.37—21	—	>140
AM-II	3.6	27.5	160	32.8
BIP-SMP	1.8	5.7	160	126
BCL-3	2.7	18.3	391	146

AM-II[8][9][10][11] (Active Messages) is similar to a remote procedure call (RPC) mechanism. Compared with AM-II, BCL-3 has a better latency in both intra-node and inter-node communication. It is meaningless to compare the bandwidth of these two protocols since AM-II needs an extra memory copy when transfer a message while BCL-3 doesn't. BCL-3 reaches a much higher bandwidth.

BIP[12][13][14] (Basic Interface for Parallelism) is a low level message passing system developed by the Laboratory for High Performance Computing in Lyon, France. It has a very low latency. But it doesn't provide the functionality of flow control and error correction. Its bandwidth is lower than that of BCL-3.

Figure 7 and Figure 8 show the performance of MPI over BCL-3. The minimal latency is 6.3μs within one node and

23.7μs between nodes. The bandwidth is 391MB/s within one node (with the affect of cache) and 131MB/s between nodes. When the message length grows to 8MB, the intra-node decreased. This shows the influence of cache. The bandwidth can be very high without cache replacement.

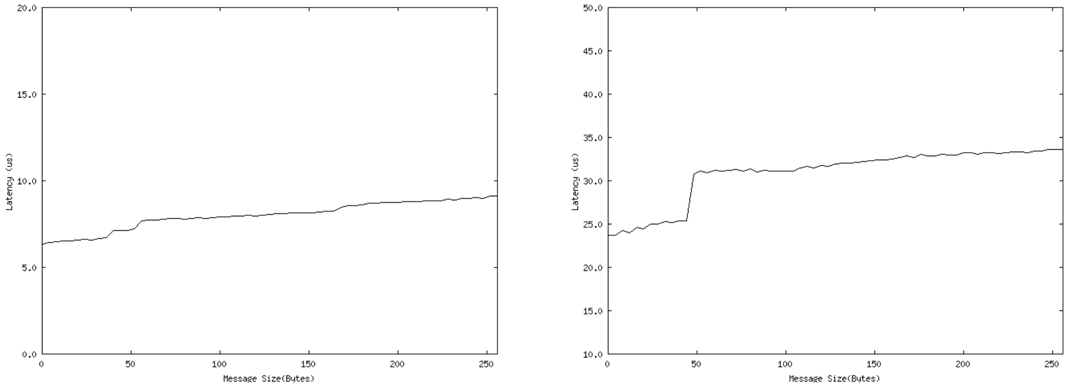

Figure 7 Intra-node (left) and inter-node (right) latency of MPI over BCL-3

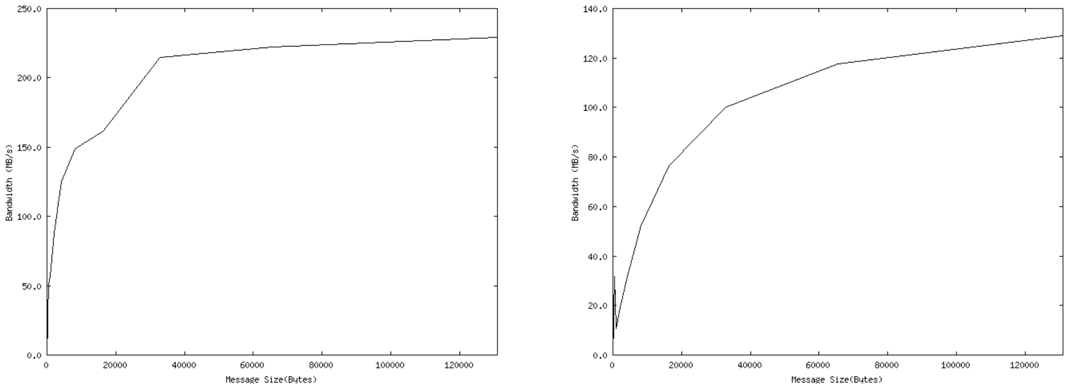

Figure 8 Intra-node (left) and inter-node (right) bandwidth of MPI over BCL-3

5. Conclusion and future work

BCL-3 is available to easily build clusters of commodity SMPs. This communication layer combined with MPI over BCL-3 provides an efficient message-passing interface for a cluster of commodity SMPs. BCL-3 achieves 2.7μs of latency and 391MB/s of bandwidth for intra-node message-passing. The performance of inter-node communication is 18.3μs of latency and 146MB/s of bandwidth. MPI over BCL-3 can achieves 6.3μs of latency and 328MB/s of bandwidth for intra-node message passing, and 23.7μs of latency and 131MB/s of bandwidth for inter-node message passing. BCL-3 is a flexible and reliable protocol. Point to point message passing and RMA are both presented in BCL-3. The combination of kernel and user-level communication makes the system reliable and portable.

Although BCL-3 is at this time implemented on AIX, it can be ported to other platforms easily because it needn't to modify the source codes of the operating system kernel.

The bandwidth provided by the Myrinet network is approximately 160MB/s (1.28Gbit/s). BCL-3's bandwidth is almost reached the hardware limitation. To break the limitation, we plan to design a double-card system that uses two Myrinet cards to perform communication. It ought to get a higher bandwidth by this means.

Finally, we have tested BCL-3 on a large-scale cluster of SMP nodes, DAWNING-3000. There are 70 nodes and each node has four processors. This shows the scalability of BCL-3.

References

[1] Kai Hwang, Zhiwei Xu. Scaleable Parallel Computing: Technology, Architecture, Programming. WCB/McGraw-Hill, 1998.

[2] Hiroshi Tezuka, Francis O'Carroll, Atsushi Hori, and Yutaka Ishikawa. Pin-down Cache: A Virtual Memory Management Technique for Zero-copy Communication. In *Proc. of IPPS98*, 1998, pp. 308-314.

[3] I. Foster, J. Geisler, C. Kesselman, S. Tuecke. Managing Multiple Communication Methods in High-Performance Networked Computing Systems. *Journal of Parallel and Distributed Computing*, 1997, 40: 35-48.

[4] W. W. Gropp, E. L. Lusk. A Taxonomy of Programming Models for Symmetric Multiprocessors and SMP clusters. In *Proc. of Programming Models for Massively Parallel Computers*, 1995, pp. 2-7.

[5] J. M. Mellor-Crummey, M. L. Scott. Algorithms for Scalable Synchronization on Shared-Memory Multiprocessors. *ACM Transactions on Computer Systems*, 1991, 9(1): 21-65.

[6] Glenn R. Luecke, James J. Coyle. Comparing the Performance of MPI on the Cray T3E-900, the Cray Origin 2000 and the IBM P2SC. *Electronic Journal of Performance Evaluation and Modelling for Computer Systems (PEMCS)*, 1998, available at URL: http://hpc-journals.ecs.soton.ac.uk/PEMCS/.

[7] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, W. Su. Myrinet -- A Gigabit-per-Second Local-Area Network. *IEEE Micro*, 1995, 15(1): 29-38.

[8] T. von Eicken, D.E. Culler, S.C. Goldstein, and K.E. Schauser. Active Messages: a Mechanism for Integrated Communication and Computation. In *Proc. of the 19th Annual International Symposium on Computer Architecture*, 1992, pp. 256-266.

[9] T. von Eicken, V. Avula, A. Basu, and V. Buch. Low-latency Communication over ATM Networks Using Active Messages. *IEEE Micro*, 1995, 15(1):46-53.

[10] S. S. Lumetta, A. M. Mainwaring, and D. E. Culler. Multi-Protocol Active Messages on a Cluster of SMP's. In *Proc. of SC97*, 1997, electronic proceedings only, available at URL: http://www.supercomp.org/sc97/proceedings/.

[11] S. S. Lumetta and D. E. Culler. Managing Concurrent Access for Shared Memory Active Messages. In *Proc. of the International Parallel Processing Symposium*, 1998, pp. 272-278.

[12] Prylli, L., and Tourancheau, B. BIP: A New Protocol designed for High-Performance Networking on Myrinet. In *Workshop PC-NOW*, *IPPS/SPDP98*, 1998, pp. 472--485.

[13] Patrick Geoffray, Loïc Prylli, Bernard Tourancheau. BIP-SMP : High Performance Message Passing over a Cluster of Commodity SMP's. In *Proc. of* SC'99, 1999, electronic proceedings only, available at URL: http://www.supercomp.org/sc99/proceedings/.

[14] Loïc Prylli, Bernard Tourancheau, Roland Westrelin. An Improved NIC Program for High-Performance MPI. In *Proc. of ACM99*, 1999, electronic proceedings only, available at URL: http://www.crhc.uiuc.edu/~steve/wcbc99/.

Ma Jie received his B.S. and M.S. degrees in computer science from Xi'an Jiaotong University in 1995 and 1998, respectively. He is currently a Ph.D. candidate of the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include high performance computer architecture, cluster operating system, high performance communication protocol and parallel computing.

He Jin received his B.S. degree in material science and engineering from Xi'an Jiaotong University in 1996, and M.S. degree in computer science from Huazhong University of Science and Technology in 1999. He is currently a Ph.D. candidate of the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include high performance computer architecture, distributed file system and storage server.

Meng Dan received his B.S., M.S. and Ph.D. degrees in computer science from Harbin Institute of Technology in 1988, 1991 and 1995, respectively. He is an Associate Professor of the Institute of Computing Technology, Chinese Academy of Sciences. His current research interests include high performance computer architecture, cluster operating system, high performance communication protocol, distributed file system and storage server.

Li Guojie received his B.S. degree in Physics from Peking University in 1968, M.S. in computer science from University of Science & Technology of China in 1981, and Ph.D. degree in EE in Purdue University, USA in 1985. He is the member of Chinese Academy of Engineering and director of Institute of Computing Technology, Chinese Academy of Sciences.