Design Alternatives and Performance Trade-offs for Implementing MPI-2 over InfiniBand

Wei Huang, Gopalakrishnan Santhanaraman,
Hyun-Wook Jin and Dhabaleswar K. Panda

Technical Report
OSU-CISRC-5/05-TR28

Design Alternatives and Performance Trade-offs for Implementing MPI-2 over InfiniBand *

Wei Huang, Gopalakrishnan Santhanaraman,
Hyun-Wook Jin, and Dhabaleswar K. Panda

Department of Computer Science and Engineering
The Ohio State University
{huanwei,santhana,jinhy,panda}@cse.ohio-state.edu

Abstract. MPICH2 provides a layered architecture to achieve both portability and performance. For implementation of MPI-2 over InfiniBand, it provides the flexibility for researchers to choose the RDMA channel, CH3 or ADI3 layer. In this paper we analyze the performance and complexity trade-offs associated with implementation at these layers. We describe our designs and implementations, as well as optimizations at each layer. To show the performance impact of different design schemes and optimizations, we evaluate our implementations with different micro-benchmarks, HPCC and NAS test suite. For example, our experiments show that the CH3 layer design can show up to 49% improvement on one sided communication throughput test comparing with the design at the RDMA channel layer; and with optimizations at the ADI3 layer, the throughput can be improved by up to another 8.1%. Although the ADI3 layers adds complexity in implementation, the benefits achieved through optimizations justify moving to the ADI layer to extract the best performance.
keywords: MPI-2, InfiniBand, RDMA channel, CH3, ADI3

1 Introduction

In the last decade, MPI (message passing interface) has become the *de facto* standard for programming parallel scientific applications. MPI-1 standard [14] was proposed by the MPI forum to provide a uniform standard for MPI developers. MPI-2 [11] standard aims to build on MPI-1 specification. It provides extension to MPI-1 in the areas of one sided communication, I/O and dynamic process management.

MPICH2 [12] from Argonne National Laboratory is one popular implementation of the MPI-2 standard. MPICH2 aims to combine portability with performance over different interconnects. It tries to achieve this by maximal sharing of common code like MPI datatypes, groups, attributes, and communicators, etc., which are platform independent and calls the interface named Abstract Device Interface (ADI) for platform dependent code. The porting to different interconnects is achieved by having a separate ADI implementation for each interconnect. To provide ease of porting, the ADI itself is also layered and can be implemented in terms of lower level interfaces.

In the field of High Performance Computing, InfiniBand [6] interconnect is emerging as a strong player. InfiniBand supports several advanced hardware features including RDMA capability, atomic operation, multi-cast, etc. To implement MPI-2 on InfiniBand, MPICH2 provides flexibility of implementation at ADI3, CH3, or RDMA channel layers. Understanding the benefits and limitations of implementing at each layer (ADI3,

* This research is supported in part by Department of Energy's Grant #DE-FC02-01ER25506, National Science Foundation's grants #CNS-0204429, and #CUR-0311542, and a grant from Intel.

CH3, and RDMA channel) is very important while coming up with an efficient MPI-2 design. The lower layer interfaces are easier to port at the cost of some performance penalties. This is rather expected, but it would be of more interest to quantitatively understand the performance impact and try to come up with different levels of optimizations at each layer. To the best of our knowledge, there is no literature that does an in-depth study on this topic.

In this paper, we attempt to do an detailed analysis of the performance/complexity trade-offs for implementating MPI-2 over InfiniBand at the RDMA channel, CH3, and ADI3 layer. We focus on point to point and one sided operations in the current work. For fair comparison, we provide our design and implementation at each of these layers.

The rest of the paper is organized as follows. In Section 2 we describe the background of our work. Section 3 describes our design choices and implementations. In Section 4 we do the performance evaluation and analyze the trade-offs. Conclusions and future work are presented in Section 5.

2 Background

2.1 InfiniBand Architecture

The InfiniBand Architecture (IBA) [6] defines a System Area Network (SAN) to interconnect processing nodes and I/O nodes. In addition to send/receive semantics it also provides RDMA semantics which can be used to directly access/modify the contents of remote memory. RDMA operations are one sided and do not incur software overhead on remote side. Further, InfiniBand verbs provide scatter/gather features to handle non contiguity. Remote atomic operations is another useful feature of InfiniBand verbs specification. VAPI is the Mellanox' implementation of InfiniBand verbs interface, which is currently being used on many systems. A new open source consortium, OpenIB [3], is defining a new Gen2 verbs level interface for next generation InfiniBand systems.

2.2 Layered Design of MPICH2

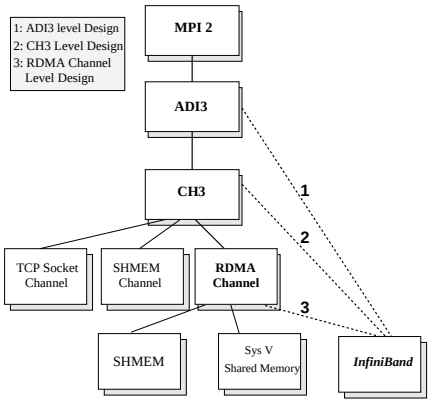

Fig. 1. Layered Design of MPICH2

MPI-2 supports both point to point and one sided operations which are also referred to as Remote Memory Access (RMA) operations. MPICH2 is a portable implementation of MPI-2. Figure 1 describes the layered approach provided by MPICH2 for designing MPI-2 over RDMA capable networks like InfiniBand. Implementation of MPICH2 on InfiniBand can be done at one of the three layers in the current MPICH2 stack: RDMA channel, CH3 or ADI3. One of the objectives of such kind of a design is to get a better balance between performance and flexibility.

RDMA channel is at the bottom most position in the hierarchical structure of MPICH2. All communication operations that MPICH2 supports are mapped to just five functions at this layer. In the five-function-interface only two (*put* and *read*) are communication functions, thus it

provides the least porting overhead. The interface needs to conform to stream semantics. It is especially designed for architectures with RDMA capabilities, which directly fits with the InfiniBand's RDMA semantics.

The CH3 provides a *channel* device that consists of a dozen functions. It accepts the communication requests from the upper layer and informs the upper layer once the communication has completed. The CH3 layer is responsible to make communication progress. This is the added complexity associated with implementing MPI-2 at this layer. From a performance perspective, it has more flexibility to improve the performance since it can access more performance oriented features than RDMA channel layer. There are a few known implementations of MPICH2 over InfiniBand that have been developed at this level. Argonne National Lab[12] and University of Chemnitz[4] have developed CH3 devices for Infiniband.

The ADI3 is a full featured, abstract device interface used in MPICH2. It is the highest layer in MPICH2 hierarchy. A large number of functions must be implemented to bring out a ADI3 design, but it can also provide flexibility for many optimizations, which are hidden from lower layers.

2.3 MVAPICH2

MVAPICH2 is our high performance implementation of MPI-2 over InfiniBand [7] based on MPICH2. The latest release version is 0.6.0. In this version we implement the point to point communication at the RDMA channel layer [9] and optimized the one sided communication at the ADI3 level [8, 5]. Currently, MVAPICH2 together with MVAPICH are being used by more than 200 organizations in 26 countries. Even though MVAPICH2 provides good performance and functionalities, its performance is little lower compared to existing MVAPICH [10] (MPI-1 implementation based on ADI2). This has motivated us to carry out the proposed in-depth study of design alternatives and performance trade-offs so that we can design the next version of MVAPICH2 with higher performance.

3 Design Alternatives at Different Layers and Implementations

For InfiniBand, MPICH2 design provides implementation choices at three different layers. In this sense, understanding the exact trade-offs of the performance constraints and implementation complexity at each layer will be critical to have an efficient design of MPI2 over InfiniBand. In order to study the implementation and performance issues associated with each level of the MPICH2 hierarchy and to carry out a fair comparison, we have designed and implemented MPI2 over InfiniBand at the RDMA channel, CH3 and ADI3 layers, respectively.

The following subsections present our design and implementation strategies at RDMA channel, CH3, and ADI3 layer, respectively.

3.1 RDMA Channel Level Design and Implementation

At RDMA channel, as mentioned in Section 2.2, all architecture dependent communication functionalities are encapsulated into a small set of interfaces. The interface needs to provide only stream semantics and the communication progress of MPI messages are left to the upper layers.

Our design at the RDMA channel is purely based on the RDMA capability provided by InfiniBand [9]. Fig. 2 illustrates design issues related with this layer. For short messages, eager protocol is used to achieve good latency. It copies messages to

pre-registered buffers and sends them through RDMA write operations. For large messages, using pre-registered buffers will introduce high copy overhead, so a zero-copy rendezvous protocol is used. However, InfiniBand requires the memory to be registered before the RDMA operations can take place, which is an expensive operation. So we implement a registration cache [15] to reduce the registration overhead.

The RDMA channel provided by MPICH2 receives the communication requests from the CH3 layer above it. The current CH3 layer makes only one outstanding request to the RDMA channel and will not issue the next request until the previous one has completed. This results in serialization of communication requests to the RDMA channel, which adversely affects the throughput. For small messages, an optimization would be to copy the message to the pre-registered buffer and immediately report completion to the CH3 layer so that the CH3 layer can issue the next communication request. By this early completion method, multiple communication requests are issued to the RDMA layer, thus avoiding the serialization described earlier. But for large messages which are sent through rendezvous protocol, we can only report completion after the whole rendezvous process finishes since we need to hold the user buffer for zero-copy send. Because of this reason, it is difficult to obtain higher bandwidth for medium-large messages using the RDMA layer. We show this performance impact in Section4.

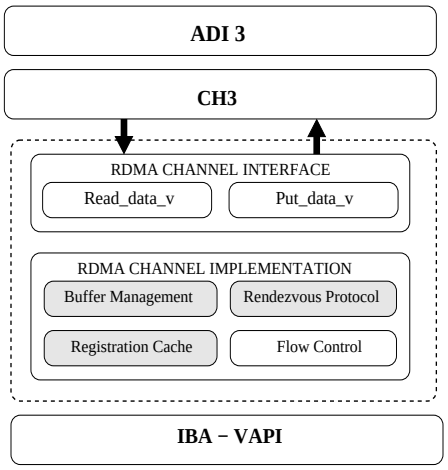

Fig. 2. RDMA Channel level design and implementation

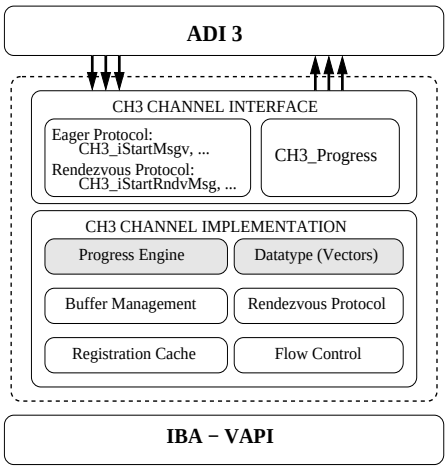

Fig. 3. CH3 level design and implementation

3.2 CH3 Level Design and Implementation

Fig. 3 shows the basic function blocks that are needed by a CH3 level design. Messages can be sent through eager or rendezvous protocol similar to the approach taken by the RDMA channel implementation. Hence functionalities such as buffer management, registration cache, etc., need to be also implemented at the CH3 layer. In addition to that, the CH3 layer needs to handle communication progress. Further, datatype communication can be optimized at this layer.

The need to implement progress engine is the most significant difference between the CH3 level design and RDMA channel design. The CH3 layer must keep track of all requests coming from ADI3 layer, finish all of them and report completions to the

ADI 3 layer. ADI3 may keep giving requests to send messages but the underlying network resources are limited. Here we implement a queue to buffer the requests which cannot be finished immediately because of the lack of network resources. Requests in the queue will be retried when network layer reports new available resources. The benefit of implementing the progress engine is obvious. Now we can get access to all the requests at the sender side. So for large messages that need to be sent through rendezvous protocol, we can start multiple rendezvous progresses at the same time so that the throughput is expected to be greatly improved as compared to the RDMA channel level design.

Datatype communication can also be optimized at this layer. ADI3 flattens the datatype and provides the datatype information to the CH3 layer as a vector list. With this information, a CH3 level design can have a global picture of all the buffer vectors that need to be sent in a particular MPI message. So optimizations such as zero-copy datatype [13, 16] can be applied at this level.

3.3 ADI3 level Design and Implementation

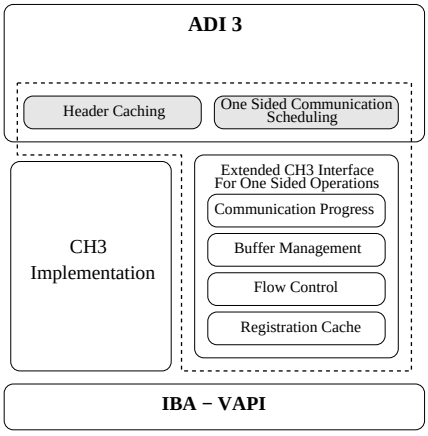

Fig. 4. Optimizations at ADI3 layer

Compared to the CH3 and RDMA channel which provide minimal interfaces, the ADI3 interface is full-featured. This allows the MPI implementor to take advantage of opportunities for more efficient communication. Since reimplementing the whole ADI3 is very complicated, we decided to reuse most of the CH3 level implementation described in section 3.2 and perform several optimizations at the ADI3 layer, shown in Fig 4. These optimizations are possible at the ADI3 layer since it allows direct access to several global data structures which are abstracted out for the lower layers.

Header caching is an optimization that can be implemented at this layer to reduce the point to point latency for small messages. The basic idea here is to cache some of the MPI header information for each connection at the remote side. So that if the next message between a pair of sender and receiver has the same header information in those cached fields, we can reduce the size of header embedded in the messages being sent. If these fields differ, there is a copy overhead for the message header, which is quite negligible according to our experience. It is be noted that MPI header caching cannot be performed at lower layers since only ADI3 is supposed to know the contents in a message header.

Another significant optimization is in the area of one-sided communication. Originally in MPICH2, one sided operations are performed through the point to point interfaces provided by CH3. Our previous work [8] has shown that by directly using RDMA features provided by InfiniBand instead of going through the point to point path, the performance for RMA operations can be greatly enhanced. What is more, at the ADI layer, we can schedule one sided operations to achieve much better latency

and throughput. The scheduling schemes are described in more detail in [5]. Optimizations for one sided operations are done by extending the CH3 interface [8]. At the CH3 layer we cannot distinguish between data for two sided and one sided operations and hence cannot perform such optimizations. MPICH2's latest release version 1.0.1 also has extended the CH3 one sided interface for shared memory architectures, which also reflects the potential to optimize one sided operations at the ADI3 layer.

4 Performance Evaluation

In this section we evaluate our implementations at RDMA channel, CH3 and ADI3 layers by a set of micro-benchmarks, HPC Challenge Benchmark [2] and NAS test suite [1].

We evaluated our schemes on two different clusters. The first cluster (Cluster A) consists of computing nodes with dual Intel Xeon 3.0 GHz processors, 512 KB L2 cache, and PCI-X 64-bit 133 MHz bus. They are equipped with MT23108 HCAs and are connected by an InfiniScale MTS2400 switch. The second cluster (Cluster B) is equipped with dual Intel Xeon 2.66 GHz processors, 512 KB L2 cache, and PCI-X 64-bit 133 MHz bus. They are equipped with MT23108 HCAs and are connected through an InfiniScale MTS14400 switch.

4.1 Point to Point Communication

Point to point communication test are conducted on Cluster A. Fig. 5 shows the unidirectional bandwidth test results for our implementation at RDMA channel, CH3 and ADI3 layers. For medium-large messages, the bandwidth is significantly improved by up to 28% by moving from RDMA channel to CH3 layer, because CH3 handles multiple send requests simultaneously. ADI3 layer shows similar numbers as CH3 layer. It is to be noted that all bandwidth numbers in this paper are reported in MillionBytes/Sec (MB/s).

Fig. 6 shows the point to point ping-pong latency. By getting rid of the stack overhead, the one byte message latency drops from 5.7us at RDMA layer to 5.3us at CH3 layer. By performing header caching at ADI3 layer, the number drops further to 4.9us. Header caching technique is applied to messages smaller than 256 bytes. So for this range the ADI3 level numbers consistently outperform CH3 numbers.

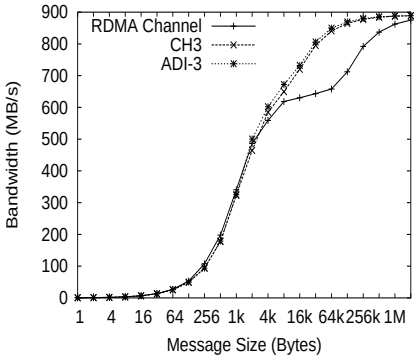

Fig. 5. Unidirectional bandwidth

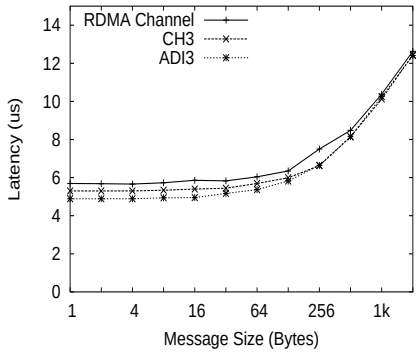

Fig. 6. Point to point latency

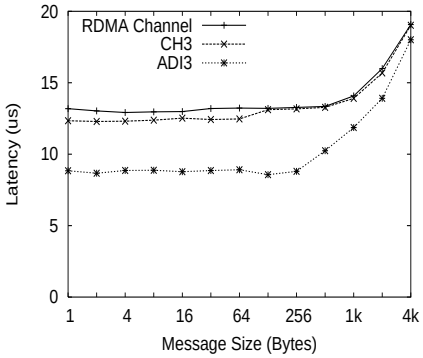

Fig. 7. MPI_Put latency

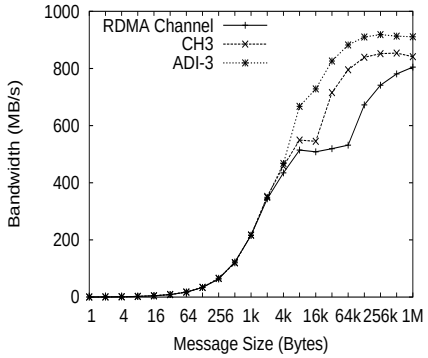

Fig. 8. One sided throughput test

4.2 One Sided Operations

Evaluation with one sided operations are also conducted on Cluster A. The results for MPI_Put test are shown in Fig. 7. The test times the ping-pong latency time for performing the put operation followed by synchronization. For the CH3 and the RDMA Channel level design, one sided operations are implemented based on point to point communication. Their numbers are similar, with the CH3 level design performing slightly better because point to point communication is optimized. By optimizing one sided operation at ADI3, we observe 30% reduction in MPI_Put latency as compared to the CH3 level design.

We also measure the throughput of one sided communication. The one sided communication throughput test involves two processes. The first process issues 16 MPI_Put and 16 MPI_Get operations of the same size. The second process just starts an exposure epoch. We measure the maximum throughput we can achieve (in terms of MillionBytes/sec) for multiple iterations of the above sequence. Fig. 8 shows the results. The improvement on point to point bandwidth makes the CH3 level design outperform the RDMA channel level design by up to 49%. And with one sided scheduling at the ADI3 layer, the peak throughput can reach around 920MB/s, which is another 8.1% higher than the CH3 level design numbers.

4.3 HPC Challenge (HPCC) Suite

The HPCC suite [2] contains tests which evaluate the latency for different types of communication patterns. It performs the ping-pong tests between all possible different pairs of processors. It also uses two different ring types of communication patterns to evaluate latency: Naturally Ordered Ring and Randomly Ordered Ring. HPCC tests were performed on 16 nodes of Cluster B and we report 8 bytes latency numbers. As shown in Fig. 9, we clearly observe an performance improvement up to 7% for the CH3 level design over the RDMA Channel level design; and the benefit from header caching at the ADI3 layer enhances the performance by up to another 6%.

4.4 NAS Integer Sort

In this section we show the performance evaluation for the IS benchmark [1]. IS is an integer sort benchmark kernel that stresses the communication aspect of the network. The experiments were conducted for classes B and C on 8 nodes and 16 nodes of Cluster B. The results are shown in Fig. 10. The ADI3 level implementation here shows up

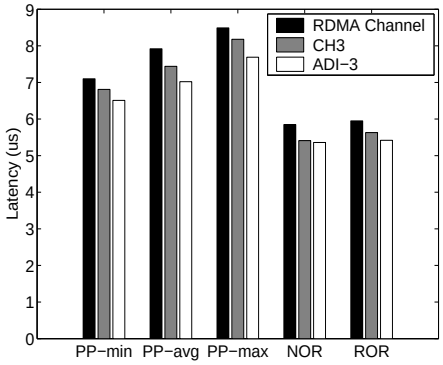

Fig. 9. HPCC 8 bytes latency. PP-min: minimum ping-pong latency; PP-avg: average ping-pong latency; PP-max: maximum ping-pong latency; NOR: Natural ordered ring access latency; ROR: random ordered ring access latency

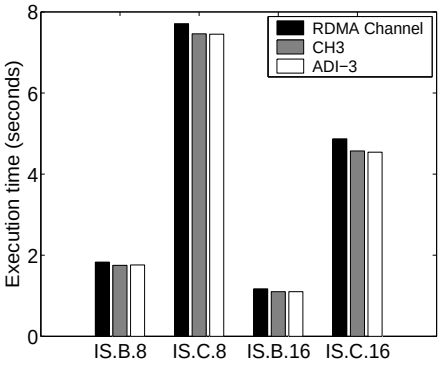

Fig. 10. NAS (IS) results on Cluster B

to 7% improvement comparing with the RDMA channel level design. The ADI3 level optimizations does not directly help the communication pattern that is observed in NAS-IS. Hence the performance seen at the CH3 level is the same as that of the ADI3 level.

5 Conclusions and Future Work

In this paper we have analyzed the trade-offs associated with implementing MPI-2 over InfiniBand at the RDMA channel, CH3 and ADI3 layer of MPICH2. We have also described the various optimizations that are possible at each of these levels.

With respect to design complexity, our work shows that compared to the RDMA channel level design, a CH3 level design needs to implement the progress engine, which is the main cause of added complexity. A fully featured ADI3 level design is very complicated but optimizations like header caching, one sided communication scheduling can be done at this level.

With respect to performance, the CH3 and ADI3 level design can increase the bandwidth significantly, up to 28% for bandwidth test comparing with the RDMA channel level design. Header caching at the ADI3 can lower the small message latency to 4.9 us, a 12.5% improvement comparing with 5.6 us achieved by the RDMA channel level design. One sided scheduling at the ADI3 level will also greatly improves the performance. We see an enhancement up to 30% in MPI_Put latency and 8.1% in throughput test compared with the CH3 level design, which in turn shows 49% improvement on throughput over the RDMA channel level design. Effects of these optimizations also show benefits at the application level evaluation of HPCC and NAS suite. As a conclusion, although the ADI3 layers adds complexity in implementation, the benefits achieved through optimizations justify moving to the ADI layer to extract the best performance.

As a part of future work we would like to come up with a full fledged MPI-2 design over InfiniBand at the ADI3 layer to deliver good performance. We are planning to support communication through shared memory, and optimize collective operations using InfiniBand's RDMA and multi-cast features.

References

1. D. H. Bailey, E. Barszcz, L. Dagum, and H.D. Simon. NAS Parallel Benchmark Results. Technical Report 94-006, RNR, 1994.
2. HPC Challenge Benchmark. http://icl.cs.utk.edu/hpcc/.
3. Open IB Consortium. http://www.openib.org/.
4. R. Grabner, F. Mietke, and W. Rehm. An MPICH2 Channel Device Implementation over VAPI on InfiniBand. In *Proceedings of the International Parallel and Distributed Processing Symposium*, 2004.
5. W. Huang, G. Santhanaraman, H. W. Jin, and D. K. Panda. Scheduling of MPI-2 One Sided Operations over InfiniBand. Workshop On Communication Architecture on Clusters (CAC), in conjunction with IPDPS'05, April 2005.
6. InfiniBand Trade Association. InfiniBand Architecture Specification, Release 1.2, October 2004.
7. Network Based Computing Laboratory. http://nowlab.cis.ohio-state.edu/projects/mpi-iba/.
8. J. Liu, W. Jiang, H. W. Jin, D. K. Panda, W. Gropp, and R. Thakur. High Performance MPI-2 One-Sided Communication over InfiniBand. International Symposium on Cluster Computing and the Grid (CCGrid 04), April 2004.
9. J. Liu, W. Jiang, P. Wyckoff, D. K. Panda, D. Ashton, D. Buntinas, W. Gropp, and B. Toonen. Design and Implementation of MPICH2 over InfiniBand with RDMA Support. In *Proceedings of the International Parallel and Distributed Processing Symposium*, 2004.
10. Jiuxing Liu, Jiesheng Wu, Sushmitha P. Kini, Pete Wyckoff, and Dhabaleswar K. Panda. High Performance RDMA-Based MPI Implementation over InfiniBand. In *17th Annual ACM International Conference on Supercomputing*, June 2003.
11. Message Passing Interface Forum. MPI-2: A Message Passing Interface Standard. *High Performance Computing Applications*, 12(1–2):1–299, 1998.
12. MPICH2. http://www-unix.mcs.anl.gov/mpi/mpich2/.
13. G. Santhanaraman, J. Wu, and D. K. Panda. Zero-Copy MPI Derived Datatype Communication over InfiniBand. EuroPVM-MPI 2004, September 2004.
14. Marc Snir, Steve Otto, Steve Huss-Lederman, David Walker, and Jack Dongarra. *MPI–The Complete Reference. Volume 1 - The MPI-1 Core, 2nd edition.* The MIT Press, 1998.
15. H. Tezuka, F. O'Carroll, A. Hori, and Y. Ishikawa. Pin-down cache: A virtual memory management technique for zero-copy communication. In *Proceedings of the 12th International Parallel Processing Symposium*. IEEE Computer Society Press, 1998.
16. J. Wu, P. Wyckoff, and D. K. Panda. High Performance Implementation of MPI Datatype Communication over InfiniBand. In *Proceedings of the International Parallel and Distributed Processing Symposium*, 2004.