2009 International Conference on Computational Science and Engineering

Hardware Supported Multicast in 2-D Mesh InfiniBand Networks

*Jiazheng Zhou**, *Shen-En Liu*[+] and *Yeh-Ching Chung**[1]
**Department of Computer Science*
National Tsing-Hua University, Hsinchu, Taiwan, R.O.C
Email: {jzzhou, ychung}@cs.nthu.edu.tw

[+]*Department of Flight Simulation*
Aerospace Industrial Development Corporation, Taichung, Taiwan, R.O.C
en.arg@msa.hinet.net

Abstract

The multicast operation is a useful operation in parallel applications. With the hardware supported multicast of the InfiniBand Architecture (IBA), we propose a multicast scheme for m×n mesh InfiniBand networks based on the XY routing scheme. The basic concept of the proposed multicast scheme is to find the union sets of the output ports of switches that are in the paths between the source node and each destination node in a multicast group. We have implemented the proposed multicast scheme on a 2-D mesh InfiniBand network simulator. Several multicast cases with different message size and different traffic workload are simulated. The simulation results show that the proposed multicast scheme outperforms their corresponding unicast scheme for all simulated cases. The larger the message size, the number of multicast source nodes, and the size of the multicast group, the better speedup can be expected from the proposed multicast scheme.

Index Terms - multicast, unicast, InfiniBand, mesh, union operation.

1 Introduction

Interconnection networks in cluster systems have great impact on the performance of communication-bounded applications. Therefore, a high speed, low latency, and high throughput network is essential for a cluster system. The InfiniBand architecture (IBA) [7] is a new industry-standard architecture for server I/O and inter-server communication. IBA defines a switch-based, point-to-point interconnection network that enables high-speed, low-latency communication between connected devices. Due to the characteristics of IBA, it is very suitable to use IBA as the interconnection network of a cluster system.

The multicast operation is a very commonly used operation in parallel application programs [15]. It can be used to implement many collective communication operations as well. Therefore, its performance will affect application programs and collective communication operations greatly. Many research results on multicast operations have been proposed in the literature [1,3,4,11-13]. Since the IBA supports hardware multicast, application programs and the collective communication operations can take advantage of this feature to speed up their execution.

In this paper, we propose a hardware supported multicast scheme for 2-D mesh InfiniBand networks. To perform multicast operations correctly on a 2-D mesh InfiniBand network, our scheme consists of three parts, the node addressing scheme, the path selection scheme, and the forwarding table assignment scheme. In the node addressing scheme, each node in the mesh network is assigned a Local Identifier (LID). In the path selection scheme, we develop different methods to associate a port with different number of virtual lanes in a switch. The *XY* routing [5] is applied to avoid deadlock. In the forwarding table assignment scheme, a two-phase forwarding table setup is used. In the first phase, the one-to-one forwarding table is set according the node addressing scheme and the path selections scheme. The second phase is to set up the multicast forwarding table based on the union operation.

To evaluate the proposed scheme, we have implemented a 2-D mesh InfiniBand network simulator. We simulate the proposed multicast scheme and the corresponding unicast scheme. Several multicast cases with different message size and different traffic workload are simulated on a 16×16 mesh InfiniBand network with 1, 2 and 4 virtual lanes. The simulation results show that the proposed multicast scheme outperforms the corresponding unicast scheme for all test cases. The larger the message size, the number of multicast source

[1] The corresponding author.

978-0-7695-3823-5/09 $26.00 © 2009 IEEE
DOI 10.1109/CSE.2009.158

nodes, and the size of the multicast group, the better speedup can be expected from the proposed multicast schemes. For the port association methods, for both unicast schemes and multicast schemes, given the same number of virtual lanes, the better performance can be expected if we do not associate a port (or direction) with dedicated virtual lane(s).

The rest of this paper is organized as follows. Section 2 will introduce the related work of routing on mesh topology. The proposed multicast scheme will be described in Section 3. Section 4 will give the simulation results for the proposed multicast scheme.

2 Related Work

In a 2-D-mesh network, how to avoid the deadlock is an important issue. There are many methods proposed in the literature to offer deadlock-free routing on 2-D mesh topology. These methods can be divided into deterministic and adaptive routings. Deterministic routing algorithms always use the same path between every pair of nodes. For example, *XY* routing [5] is for 2-D mesh and e-cube is for hypercubes [16]. Most multicomputer architectures like Cray T3D [9] and Stanford DASH [10] use deterministic routing. Most partially adaptive algorithms are based on the absence of cyclic dependencies between channels to avoid deadlock. Planar-adaptive routing [2] provides adaptivity in only two dimensions at a time and can minimize the needed resources. The turn model [6] provides a systematic approach to develop partial adaptive routing algorithms. Fully adaptive routing algorithms are based on the virtual networks to avoid deadlock [8,14]. A virtual network is a subset of channels used to route packets toward a set of destinations.

Based on the deadlock-free routing algorithms mentioned above, many multicast algorithms have been proposed for 2-D mesh topology. In [11], a tree-based multicast algorithm, the double-channel *XY* multicast routing algorithm, is proposed for 2-D mesh. The method uses double channels to avoid the deadlock. In this method, a 2-D mesh network is divided into four subsets for a given source node. Each subset can be viewed as a virtual network where the *XY* routing is used. The tree-based multicast with branch pruning [16] allows any deadlock-free routing algorithms of unicast message. It can be applied to any topology. The deadlock can be recovered by pruning the branches. This scheme outperforms other mechanisms when the multicast traffic consists of short messages. Tree-based multicast routing can also be applied to multistage interconnection networks (MINs) [1].

Besides the tree-based multicast routing algorithms, path-based multicast routing algorithms can be applied to 2-D mesh topology. Based on Hamiltonian path, the dual-path multicast routing and the multi-path multicast routing are presented in [13]. In [13], the network is separated into two sub-networks, the high-channel sub-network and the low-channel sub-network. The high-channel sub-network contains all the channels from lower label nodes to higher label nodes and the low-channel sub-network contains all the channels from higher label nodes to lower label nodes. In the dual-path routing algorithm, for a given source node, it sends messages to destination nodes with lower labels and higher labels through the low-channel and the high-channel sub-network, respectively. The multi-path routing algorithm further partitions the two sub-networks in the dual-path routing into four sub-networks and can utilize the four channels. In [12], the label-based dual-path (LD) adaptive multicast routing algorithm is proposed. It is similar to the dual-path routing algorithm and allows both minimal and non-minimal paths. In other adaptive routing algorithms [3, 4], the messages can also use the alternative minimal paths.

3 The Proposed Multicast Scheme

An InfiniBand network can be divided into subnets. In an InfiniBand subnet, packet source/destination is called endport. A Local Identifier (LID) is an address assigned to an endport by the subnet manager during the subnet initialization process. LID is unique within an InfiniBand subnet. Since the InfiniBand network is a packet-switching based network, routing in an InfiniBand subnet is deterministic, based on the forwarding table lookup. For a packet, the LIDs of its source and destination nodes are stored in SLID and DLID fields of the Local Route Header (LRH), respectively. A packet within a switch is forwarded to an output port based on the packet's DLID field and the switch's forwarding table.

The IBA supports hardware multicast. In the IBA, each multicast group is assigned a multicast LID and a GID by the subnet manager. The subnet manager will set up the forwarding table of each switch for each multicast group according its LID and GID. To execute a multicast operation in an InfiniBand network, the source node uses the multicast LID and the GID of a multicast group to send packets. When a switch receives a multicast packet, it replicates the packet and forwards the packet to the corresponding output ports according to its forwarding table.

In a 2-D mesh topology, a deadlock may occur as shown in Figure 1. In Figure 1, each big block presents a node and there is one port (including input buffer and output buffer) on each edge of the block. Source nodes A , B , C , and D want to send messages to their

destination nodes *A*', *B*', *C*', and *D*', respectively. The deadlock may occur if the four nodes *A* , *B* , *C* , and *D* send messages to their destination nodes simultaneously.

To avoid the deadlock, the *XY* routing is proposed in [5]. In the *XY* routing, the packet will be forwarded first in the *X*-direction followed by the *Y*-direction. In this way, the cyclic resource dependency shown in Figure 1 does not occur as shown in Figure 2. The deadlock is avoided since there are no sufficient conditions. Since the IBA uses the virtual cut-through technique, the packet will stay in channel (buffer) until the next required channel is free. The deadlock can be avoided when we apply *XY* routing. The proposed multicast can be divided into three schemes: the node addressing scheme, the path selection scheme, and the forwarding table assignment scheme.

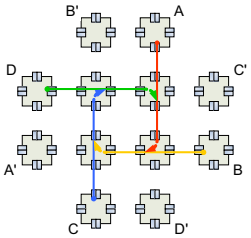


Figure 1: A deadlock in 2-D mesh.

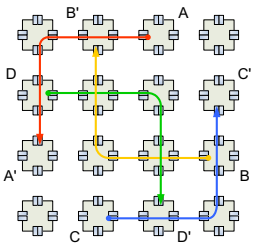


Figure 2: The XY routing can avoid deadlocks.

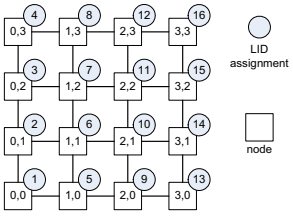


Figure 3: The node addressing scheme in IbaMesh(4, 4).

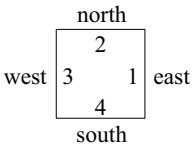


Figure 4: Directions and port number mapping in mesh topology.

3.1 The Node Addressing Scheme

Given an $m \times n$ mesh InfiniBand network *IbaMesh*(m, n), nodes in *IbaMesh*(m, n) are labeled as $N(x, y)$, where $x \in \{0, 1, \ldots, m-1\}$ and $y \in \{0, 1, \ldots, n-1\}$. It has the following characteristics:

1. There are $m \times n$ nodes in *IbaMesh*(m, n).
2. For each node, there are four directions (or ports) north (*N*), east (*E*), south (*S*), and west (*W*) for the message routing.

For each node $N(x, y)$ in *IbaMesh*(m, n), it is assigned an LID $lid = x \times m + y + 1$.

In Figure 3, there are 16 nodes in a 4×4 mesh InfiniBand network *IbaMesh*(4, 4). For node $N(3,2)$ in *IbaMesh*(4, 4), it is assigned an $lid = 3 \times 4 + 2 + 1 = 15$.

3.2 The Path Selection Scheme

Based on the *XY* routing, the packet will be forwarded first in the *X*-direction and then in the *Y*-direction. As shown in Figure 4, we define the east port as port 1, the north port as port 2, the east port as port 3, and the south port as port 4. Given the source node $N(x_{source}, y_{source})$ and the destination node $N(x_{dest}, y_{dest})$ in *IbaMesh*(m, n), for each node $N(x_{current}, y_{current})$ in the traversed routing path, we can determine the output port of $N(x_{current}, y_{current})$. We have the following cases:

Case 1. If $(x_{dest} - x_{source} \geq 0)$ and $(y_{dest} - y_{source} \geq 0)$, the path of the packet will be $N(x_{source}, y_{source})$, $N(x_{source}+1, y_{source})$, $N(x_{source}+2, y_{source})$, …, $N(x_{dest}, y_{source})$, $N(x_{dest}, y_{source}+1)$, $N(x_{dest}, y_{source}+2)$, …, $N(x_{dest}, y_{dest})$.

Case 2. If $(x_{dest} - x_{source} \geq 0)$ and $(y_{dest} - y_{source} < 0)$, the path of the packet will be $N(x_{source}, y_{source})$, $N(x_{source}+1, y_{source})$, $N(x_{source}+2, y_{source})$, …, $N(x_{dest}, y_{source})$, $N(x_{dest}, y_{source}-1)$, $N(x_{dest}, y_{source}-2)$,…, $N(x_{dest}, y_{dest})$.

Case 3. If $(x_{dest} - x_{source} < 0)$ and $(y_{dest} - y_{source} < 0)$, the path of the packet will be $N(x_{source}, y_{source})$, $N(x_{source}-1, y_{source})$, $N(x_{source}-2, y_{source})$, …, $N(x_{dest}, y_{source})$, $N(x_{dest}, y_{source}+1)$, $N(x_{dest}, y_{source}+2)$, …, $N(x_{dest}, y_{dest})$.

Case 4. If $(x_{dest} - x_{source} < 0)$ and $(y_{dest} - y_{source} \geq 0)$, the path of the packet will be $N(x_{source}, y_{source})$, $N(x_{source}-1, y_{source})$, $N(x_{source}-2, y_{source})$, …, $N(x_{dest}, y_{source})$, $N(x_{dest}, y_{source}-1)$, $N(x_{dest}, y_{source}-2)$, …, $N(x_{dest}, y_{dest})$.

An example is shown in Figure 5. In Figure 5, the source node $N(2, 2)$ sends messages to destination nodes $N(0, 3)$, $N(0, 4)$, $N(3, 3)$, $N(4, 2)$, and $N(4, 0)$ through *P*, *Q*, *R*, *S*, and *T*, respectively. All the packets will be forwarded first in the *X*-direction, then in the *Y*-direction.

Besides the routing between switches, we have different methods to associate a port with virtual lane(s) in a switch. With one virtual lane, the packet will always be forwarded to the output port using the only one virtual lane shown in Figure 6(a). With two virtual lanes,

the packet can be forwarded to the output port through virtual lane VL0 or VL1. Therefore, we can have two configurations for two virtual lanes. One configuration is that when the packet is forwarded to the output port, the packet can use VL0 or VL1 by SL-to-VL mapping function shown in Figure 6(b). By the Service Level (SL) and SL-to-VL mapping table, the IBA can offer more traffic control and ensure the QoS. The other configuration is that we associate each output port with one virtual lane by packet relay function. It means that when the packet is forwarded to the output port(s), it will always use the selected virtual lane. In our configuration, we associate the north port and east port with VL0, the south port and the east port with VL1 as shown in Figure 6(c). Figure 6(d) and Figure 6(e) show the configurations for four virtual lanes. In Figure 6(e), we associate the east port, the north port, the west port, and the south port with VL0, VL1, VL2, and VL3, respectively.

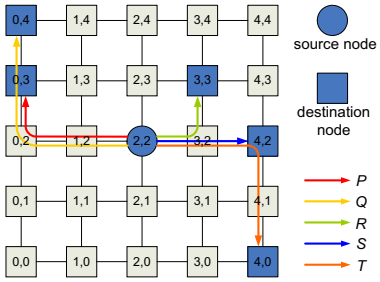


Figure 5: An example of XY routing on 5×5 mesh.

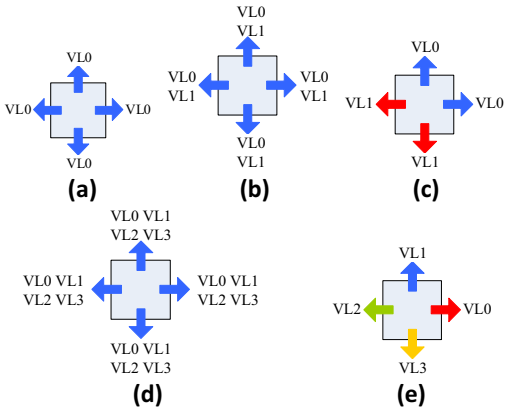


Figure 6: Different configurations for different number of virtual lanes.

3.3 The Forwarding Table Assignment Scheme

After the path selection scheme is performed, the next task is to set up the forwarding. The forwarding table assignment consists of two phases: the one-to-one forwarding table assignment and the multicast forwarding table assignment based on union operation.

3.3.1 The One-to-one Forwarding Table Assignment

Given an $m \times n$ mesh InfiniBand network *IbaMesh*(m, n), a node $N(x, y)$ of *IbaMesh*(m, n), and a packet whose DLID field is *lid*, when the packet arrives in node $N(x, y)$, the output port $N(x, y)_k$ of the packet can be determined based on the node assignment scheme, and the path selection scheme. We have the following four cases.

Case 1: If $\left\lfloor \frac{lid-1}{m} \right\rfloor > x$, the packet will be forwarded in the X-direction to the east port and $k = 1$.

Case 2: If $\left\lfloor \frac{lid-1}{m} \right\rfloor < x$, the packet will be forwarded in the X-direction to the west port and $k = 3$.

Case 3: If $\left\lfloor \frac{lid-1}{m} \right\rfloor = x$ and $lid - x \times m - 1 > y$, the packet will be forwarded in the Y-direction to the north port and $k = 2$.

Case 4: If $\left\lfloor \frac{lid-1}{m} \right\rfloor = x$ and $lid - x \times m - 1 < y$, the packet will be forwarded in the Y-direction to the south port and $k = 4$.

To verify the correctness of these four cases, let us take Figure 5 as an example. In Figure 5, assume that node $N(2, 2)$ wants to send a packet to node $N(0, 3)$ whose *lid* = 4. According to the path selection scheme, the packet sent from $N(2, 2)$ to $N(0, 3)$ will go through nodes $N(2, 2)$, $N(1, 2)$, $N(0, 2)$, and $N(0, 3)$. When the packet arrives in node $N(2, 2)$, *lid* = 4 matches case 2 and the output port of the packet k is 3. When the packet arrives in node $N(1, 2)$, *lid* = 4 matches case 2 and the output port of the packet k is 3. When the packet arrives in node $N(0, 2)$, *lid* = 4 matches case 3 and the output port of the packet is $k = 2$. From the above analysis, we can correctly set up path P for the packet sent from $N(2, 2)$ to $N(0, 3)$. For paths Q, R, S, and T, we can obtain similar results.

3.3.2 The Multicast Forwarding Table Assignment Based on Union Operations

After the one-to-one forwarding table assignment is performed, we can set up the multicast forwarding table for a given source node and a multicast group based on union operations. Let $N(x, y)$ be a source node and $lid = \{lid_1, lid_2, ..., lid_t \mid t \le m \times n\}$ be the DLID of a multicast group, where $\{lid_1, lid_2, ..., lid_t \mid t \le m \times n\}$ is the set of LIDs of destination processing nodes in a multicast group. For the node $N(x, y)$, based on the one-to-one forwarding

table assignment, we can determine the output port of a packet whose DLID is lid_1, lid_2, …, and lid_t as $N(x, y)_{k_1}$, $N(x, y)_{k_2}$, … , and $N(x, y)_{k_t}$, respectively. It means that when the packet whose DLID is lid_1, lid_2, …, and lid_t arrives in node $N(x, y)$, it will be forwarded to port $N(x, y)_{k_1}$, $N(x, y)_{k_2}$, … , and $N(x, y)_{k_t}$, respectively. Since an InfiniBand switch can duplicate a packet to different output ports, the output ports of a multicast packet $lid = \{lid_1, lid_2, ..., lid_t \mid t \le 2 \times (m/2)^n\}$ can be set as the union of $N(x, y)_{k_1}$, $N(x, y)_{k_2}$, … , and $N(x, y)_{k_t}$.

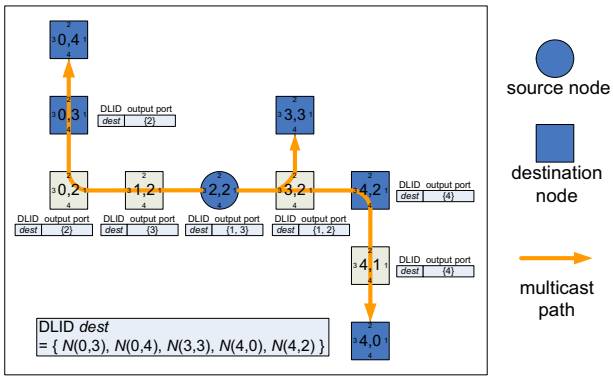


Figure 7: Multicast forwarding table setup.

An example is shown in Figure 7. In Figure 7, assume that processing node $N(2, 2)$ wants to send multicast packets to processing nodes $N(0, 3)$, $N(0, 4)$, $N(3, 3)$, $N(4, 0)$ and $N(4, 2)$. The *lid* set of the multicast group is {4, 5, 19, 21, 23}. From Figure 7, we can see that when a packet is sent from $N(2, 2)$ to $N(0, 3)$, node ports $N(2,2)_3$, $N(1,2)_1$, $N(1,2)_3$, $N(0,2)_1$, $N(0,2)_2$, and $N(0,3)_4$ will be traversed. When a packet is sent from $N(2, 2)$ to $N(0, 4)$, node ports $N(2,2)_3$, $N(1,2)_1$, $N(1,2)_3$, $N(0,2)_1$, $N(0,2)_2$, $N(0,3)_4$, $N(0,3)_2$, and $N(0,4)_4$ will be traversed. When a packet is sent from $N(2, 2)$ to $N(3, 3)$, node ports $N(2,2)_1$, $N(3,2)_3$, $N(3,2)_2$ and $N(3,3)_4$ will be traversed. When a packet is sent from $N(2, 2)$ to $N(4, 0)$, node ports $N(2,2)_1$, $N(3,2)_3$, $N(3,2)_1$, $N(4,2)_3$, $N(4,2)_4$, $N(4,1)_2$, $N(4,1)_4$, and $N(4,0)_2$ will be traversed. When a packet is sent from $N(2, 2)$ to $N(4, 2)$, node ports $N(2,2)_1$, $N(3,2)_3$, $N(3,2)_1$, and $N(4,2)_3$ will be traversed. According the union operations, for a multicast packet whose DLID = {4, 5, 19, 21, 23}, we can determine that its output ports in node $N(2, 2)$ = {1, 3}, $N(1, 2)$ = {3}, $N(0, 2)$ = {2}, $N(0, 3)$ = {2}, $N(3, 2)$ = {1, 2}, $N(4, 2)$ = {4}, and $N(4, 1)$ = {4}, respectively.

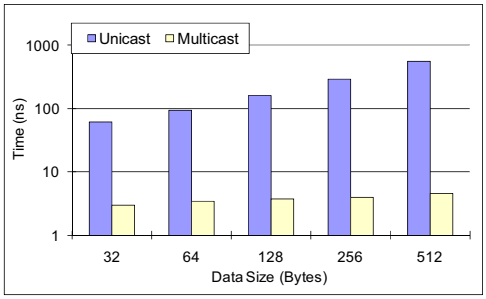
4 Performance Evaluation

To evaluate the performance of the proposed multicast schemes, we design an *m*×*n* mesh InfiniBand network simulator. Multicast schemes and the corresponding unicast schemes are simulated for performance evaluation. Several cases with different message and traffic workload size are simulated on a 16× 16 (256 nodes in total) mesh InfiniBand network with 1, 2 and 4 virtual lanes. The packet size ranges from 32 bytes to 8K bytes. According to the size of source nodes, we have one-source multicast, multi-source multicast, and all-source multicast. That is, there is 1 node, partial nodes, and all nodes perform multicast in the network. The simulation results are shown in Figure 8 to Figure 16. We have the following cases.

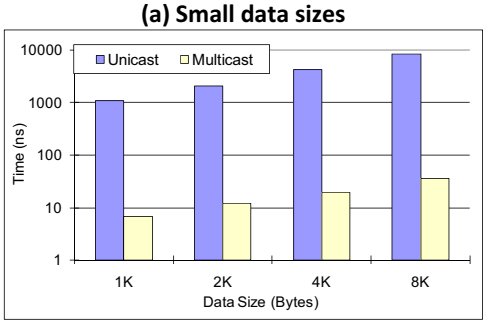
Case 1 (One-source multicast): Figure 8 to Figure 10 show the results of one-source multicast cases. The destination group size is 100% of all nodes. Since there is only one source node, the traffic congestion of two packets using the same buffer is never occurred. From the simulation results, we can see that the multicast schemes outperform their corresponding unicast schemes. Given the same number of virtual lanes, we can get better performance if we do not associate the port (direction) with dedicated virtual lane(s). This is because we can utilize the resources (virtual lanes) efficiently. However, this observation is not clear when the data size is large.

Case 2 (Multi-source multicast): Figure 11 to Figure 13 show the results of multi-source multicast cases. The size of source nodes is 40% of all nodes. The destination group size is 40% of all nodes. Since there are more than one source nodes send messages to the destination nodes, the traffic congestion occurs. From the simulation results, we can see that the multicast schemes outperform the corresponding unicast schemes. The number of virtual lanes can help to speed up the performance for the unicast schemes, that is, more virtual lanes can help to release the traffic congestion. For both unicast schemes and multicast schemes, with the same number of virtual lanes, we can get better performance if we do not associate the port with dedicated virtual lane(s). However, this observation is not clear when the data size is large.

Case 3 (All-source multicast): Figure 14 to Figure 16 show the results of all-source multicast cases. The destination group size is 100% of all nodes. Since all the processing nodes in the system perform multicast operations, the more traffic congestion we can expect. We observe that all the simulation results of all-source multicast are similar to those of multi-source multicast. Obviously, the cases of all-source multicast spend more time because of more packets need to be transmitted and more traffic congestion occurs.

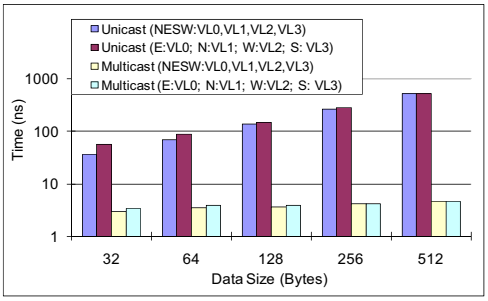


(a) Small data sizes

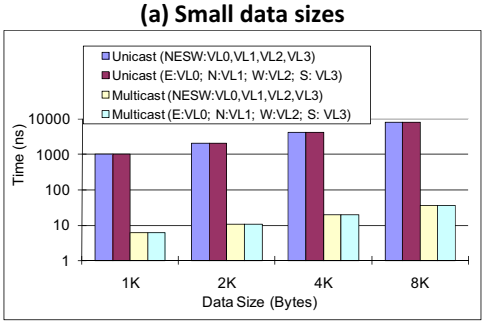


(b) Large data sizes

Figure 8: One-source multicast with 1 virtual lane.

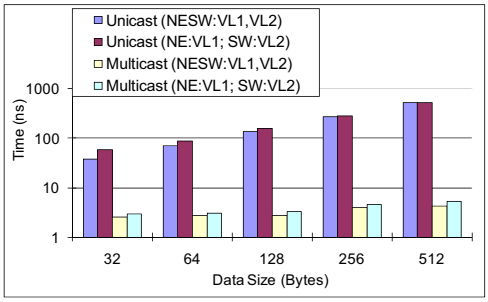


(a) Small data sizes

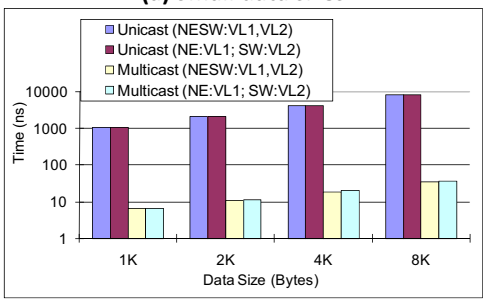


(b) Large data sizes

Figure 10: One-source multicast with 4 virtual lanes.

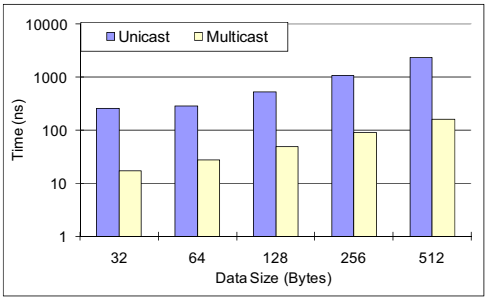


(a) Small data sizes

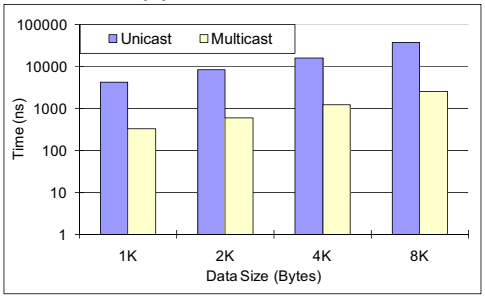


(b) Large data sizes

Figure 9: One-source multicast with 2 virtual lanes.

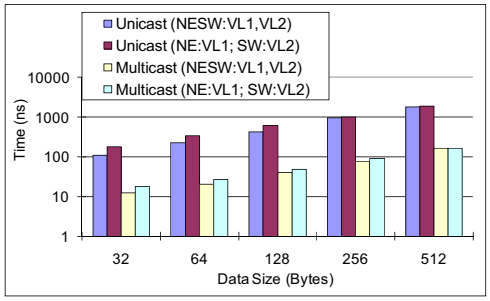


(a) Small data sizes

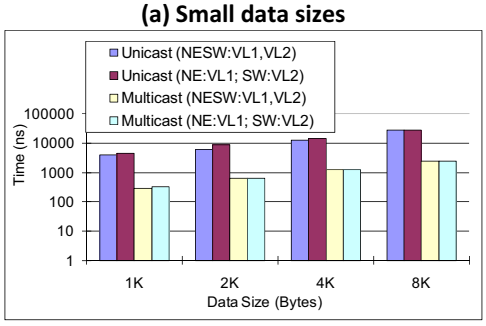


(b) Large data sizes

Figure 11: Multi-source multicast with 1 virtual lane.

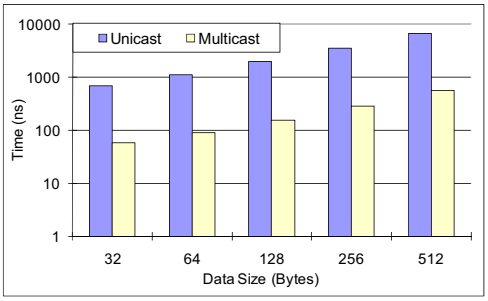


(a) Small data sizes

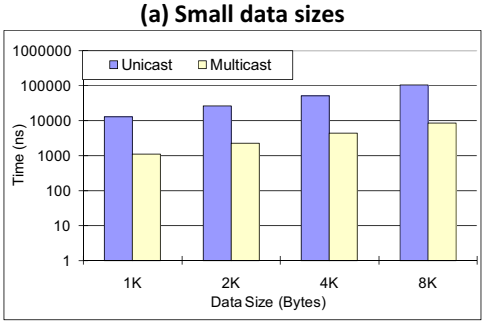


(b) Large data sizes

Figure 12: Multi-source multicast with 2 virtual lanes.

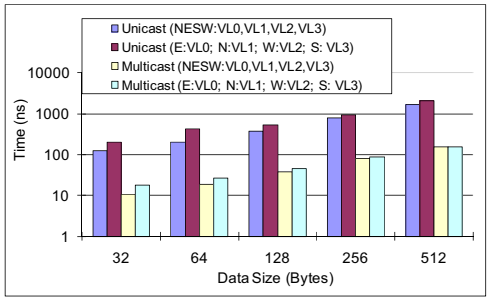


(a) Small data sizes

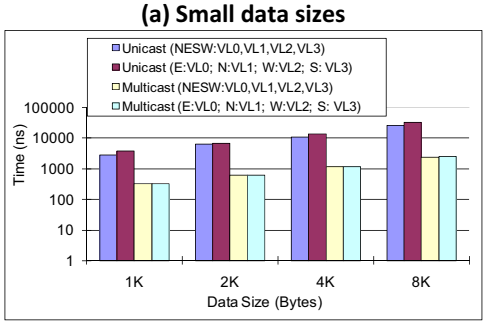


(b) Large data sizes

Figure 14: All-source multicast with 1 virtual lane.

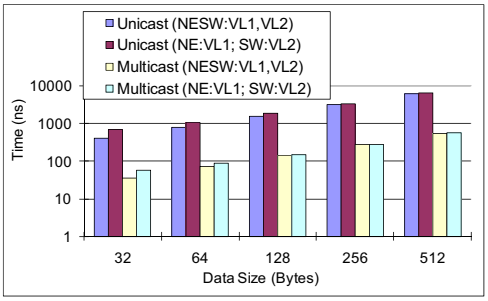


(a) Small data sizes

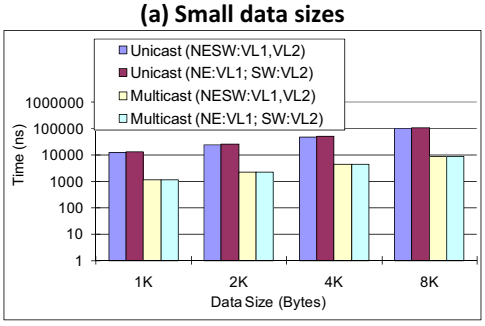


(b) Large data sizes

Figure 13: Multi-source multicast with 4 virtual lanes.

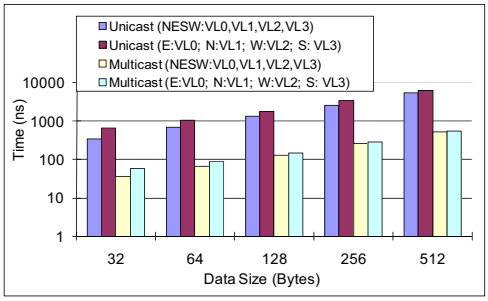


(a) Small data sizes

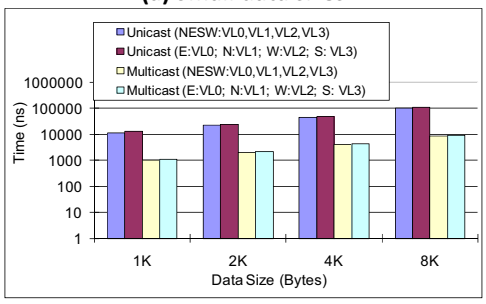


(b) Large data sizes

Figure 15: All-source multicast with 2 virtual lanes.



(a) Small data sizes



(b) Large data sizes

Figure 16: All-source multicast with 4 virtual lanes.

5 Conclusions

In this paper, we propose a hardware supported multicast scheme for InfiniBand network on 2-D mesh topology. We describe how to construct the proposed scheme in details. The simulation results show that the proposed multicast scheme can speed up the execution of multicast operations tremendously. From the simulations results, we have the following remarks:

Remark 1: We observe that the proposed multicast scheme outperforms the unicast scheme for all the simulated cases. The result indicates that the hardware supported multicast of the IBA can help to speed up the execution of multicast operations.

Remark 2: The larger the message size, the number of multicast source nodes, and the size of the multicast group, the better speedup can be expected from the proposed multicast scheme.

Remark 3: For unicast schemes, we can get better performance if we use more virtual lanes. For both unicast schemes and multicast schemes, with the same number of virtual lanes, we can get better performance if we do not associate the port with dedicated virtual lane(s).

Acknowledgement

The work of this paper is partially supported by NSC under grand NSC-96-2221-E-007-130-MY3 and NSC -97-3114-E-007-001.

References

[1] C.-M. Chiang and L. M. Ni, "Deadlock-free multi-head wormhole routing," *Proceedings of the First High Performance Computing*—Asia, 1995.

[2] A. Chien and J. H. Kim, "Planar-adaptive routing: Low-cost adaptive networks for multiprocessors," *Proceedings of the 19th International Symposium on Computer Architecture*, pp. 268-277, May 1992.

[3] J. Duato, "A new theory of deadlock-free adaptive multicast routing in wormhole networks," *Proceedings of the 5th IEEE Symposium on Parallel and Distributed Processing*, pp. 64-71, December, 1993.

[4] J. Duato, "A theory of fault-tolerant routing in wormhole networks," *Proceedings of the International Conference on Parallel and Distributed Systems*, pp. 600-607, December, 1994.

[5] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks - An Engineering Approach*, IEEE CS Press, 1997.

[6] J. Glass and L. M. Ni, "The turn model for adaptive routing," *Proceedings of the 19th International Symposium on Computer Architecture*, pp. 278-287, May 1992.

[7] InfiniBand™ Trade Association, *InfiniBand™ Architecture Specification Volume 1, Release 1.2.1*, January 2008.

[8] R. Jesshope, P. R. Miller, and J. T. Yantchev, "High performance communications in processor networks," *Proceedings of the 16th International Symposium on Computer Architecture*, pp. 150-157, May-June 1989.

[9] R. E. Kessler and J. L. Schwarzmeire, "CRAY T3D: A new dimension for Cray Research," *Proceedings of Compcon*, pp. 176-182, 1993.

[10] D. Lenoski et al., "The Stanford DASH multiprocessor," *IEEE Computer*, vol. 25, no. 3, pp. 63-79, March 1992

[11] X. Lin, P. K. McKinley, and L. M. Ni, "Performance evaluation of multicast wormhole routing in 2-D-mesh multicomputers," *Proceedings of the 1991 International Conference on Parallel Processing*, vol. I, pp. 435-442, August 1991.

[12] X. Lin, P. K. Mckinley, and A. H. Esfahanian, "Aaptive multicast wormhole routing in 2-D mesh multicomputers," *Proceedings of Parallel Architectures and Languages Europe 93*, pp. 228-241, June 1993.

[13] X. Lin and L. M. Ni, "Deadlock-free multicast wormhole routing in multicomputer networks," *Proceedings of the 18th International Symposium on Computer Architecture*, pp. 116-125, May 1991.

[14] H. Linder and J. C. Harden, "An adaptive and fault tolerant wormhole routing strategy for k-ary n-cubes," *IEEE Transactions on Computers*, vol. C-40, no. 1, pp. 2-22, January 1991.

[15] R. J. Littlefield, "Characterizing and tuning communications performance for real applications," *Proceedings of the First Intel DELTA Applications Workshop*, February 1992.

[16] H. Sullivan and T. R. Bashhow, "A large scale, homogeneous, fully, distributed parallel machine," *Proceedings of the 4th International Symposium on Computer Architecture*, March 1977.