PAPER *Special Issue on Architecture, Algorithms and Networks for Massively Parallel Computing*

MINC: Multistage Interconnection Network with Cache control mechanism

Toshihiro Hanawa[†], **Takayuki Kamei**[†*], **Hideki Yasukawa**[†**], **Katsunobu Nishimura**[†], *Nonmembers*, *and* **Hideharu Amano**[†], *Member*

SUMMARY A novel approach to the cache coherent Multistage Interconnection Network (MIN) called the MINC (MIN with Cache control mechanism) is proposed. In the MINC, the directory is located only on the shared memory using the Reduced Hierarchical Bit-map Directory schemes (RHBDs). In the RHBD, the bit-map directory is reduced and carried in the packet header for quick multicasting without accessing the directory in each hierarchy. In order to reduce unnecessary packets caused by compacting the bit map in the RHBD, a small cache called the pruning cache is introduced in the switching element. The simulation reveals the pruning cache works most effectively when it is provided in every switching element of the first stage, and it reduces the congestion more than 50% with only 4 entries. The MINC cache control chip with 16 inputs/outputs is implemented on the LPGA (Laser Programmable Gate Array), and works with a 66MHz clock.

key words: MIN, coherent cache, directory scheme, multiprocessor, congestion analysis, VLSI implementation

1. Introduction

Switch-connected multiprocessors have been researched and developed as medium-scale parallel machines. Unlike the shared bus which permits access to one memory module at a time, multiple processors can access the memory modules simultaneously through the switch in such machines. For a small system, a simple crossbar is utilized, while the Multistage Interconnection Network (MIN) that uses a crossbar as a switching element is advantageous for a large system[2]. However, the latency of the switch sometimes bottlenecks the system performance, especially when the number of processors becomes large.

To address this problem, providing a private cache between a processor and the switch is a hopeful approach. However, this gives rise to the coherent problem for the shared writable data. Although the snoop cache technology is commonly used in shared-bus-connected multiprocessors, it cannot be used on switch-connected multiprocessors because of the benefit of the switch, that is, multiple memory modules can be accessed simultaneously.

For the CC-NUMAs with point-to-point network, several directory schemes, including the full-map directory[11], limited pointer[3], chained directory[8], and dynamic pointer[10] are proposed and used. However, these techniques are difficult to implement in the switch-connected multiprocessors. Instead of them, the approach which is supported by the pre-execution analysis with the compiler has been studied[4], [14]. Although the required hardware cost is small, this approach is not suitable for general-purpose use by multiple users.

Instead of the above approaches, a switch which includes a cache or directory has been proposed. Since MINs provide a tree structure, a hierarchical bit-map directory scheme is used. However, traditional methods require not only an external large memory but also extra latency to access the memory outside the element.

In the MHN (Memory Hierarchy Network)[12], a switching element holds both directory and data memory. The shared line moves between hierarchies to minimize the access latency. However, this method requires a large amount of memory in each switching element and it allows only a single copy of shared read/write blocks in the private cache.

In the MIND (MIN with Directory)[13], only directories are stored in the switching element. A hierarchical bit-map directory is implemented so as to minimize the network traffic for cache coherence. For easy multicasting with small hardware, the MBN (Multistage Bus Network)[1], obtained by replacing a switching element of the MIND with a bus, is proposed. However, these methods require a large amount of memory for hierarchical directory outside the switching element. Since multicasting the cache-coherent messages requires access to the directory outside the switching elements, the latency of the message transfer is stretched.

In the approach by Stenström[15], cache coherent messages are multicast according to the directory inside the packet header, and the required hardware is relatively small. However, in this approach, the directory placed on the private cache increases cache coherent messages.

Here, a novel approach, called the MINC (MIN with Cache control mechanism), for the cache-coherent mechanism for switch-connected multiprocessors, is proposed. Although this method is introduced for the MIN in the first step, it is extended for every switch

Manuscript received January 20, 1997.
Manuscript revised May 20, 1997.
[†]The authors are with the Department of Computer Science, Keio University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama, 223 Japan
[*]Presently, the authors are with Toshiba Co.

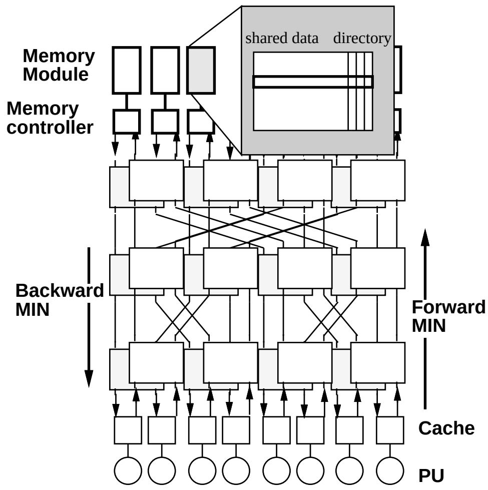

Fig. 1 The structure of the MINC

connected multiprocessor. In this method, the directory is located only on the shared memory using Reduced Hierarchical Bit-map Directory schemes (RHBDs). The reduced bit map allows the directory to be carried in the packet header, therefore, a coherent message can be multicast quickly without accessing the directory in each hierarchy (stage). In order to reduce unnecessary packets caused by compacting the bit map in the RHBD, a small cache called the pruning cache is introduced in the switching element. Since this cache is small enough to be implemented inside the switching element, the packet transferred through the MIN does not require access to the memory outside the switching element.

In section 2, the structure and control of the MINC are introduced using the RHBD and pruning cache. In section 3, results of simulation studies are presented to demonstrate the efficiency of the MINC.

2. MINC (MIN with Cache control mechanism)

2.1 Overview of the MINC

Here, a novel cache-coherent MIN called the MINC is proposed. As in the usual MIN-connected multiprocessors, forward MIN and backward MIN are provided in the MINC, as shown in Figure 1. Caches are provided between the MIN and Processing Units (PUs) which may have a local memory for private data. Any type of MIN can be utilized as the forward MIN which carries address and writing data from processors to memory modules. The reading data and cache-coherent messages are multicast through the backward MIN specialized for the MINC.

The MINC is based on the following two key ideas;

RHBD scheme The bit map of the hierarchical directory is reduced and equipped only in the main memory module. The coherent message is multicast based on the reduced bit map attached to the packet header. This technique is called the Reduced Hierarchical Bit-map Directory scheme (RHBD)[7], [9].

pruning cache In order to reduce unnecessary packets caused by the RHBD, a small cache memory, called the bit-map pruning cache, is introduced inside the switching element in the backward MIN.

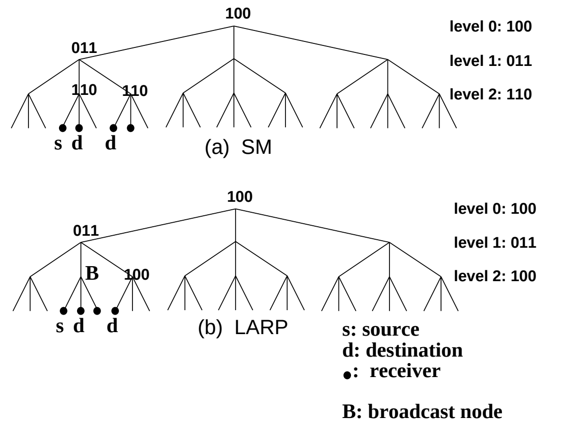

Fig. 2 Hierarchical bit-map directory schemes

2.2 The RHBD

The basic concept of the RHBD was proposed for a massively parallel processor JUMP-1[6]. Although the RHBD is designed for a large-scale cache-coherent NUMA like JUMP-1, this concept can be easily applied to the MIN-connected multiprocessors.

In RHBD scheme, the bit map is reduced using the following techniques:

- use the common bit map for all nodes of the same level of hierarchy, and
- send a message to all children of the node (thus, broadcasting) when the corresponding bit in the map is set.

The reduced directory is not stored in each hierarchy but stored only in the root. Message multicasting is done according to the reduced bit map attached to the message header. Using this method, since multicasting does not require access to the directory in each hierarchy, messages are quickly transferred.

By combining the above two techniques, two schemes, the SM and LARP, are delivered:

SM scheme: Figure 2(a) shows an example of this scheme for the 3-ary tree. In this figure, 's' is the source node, and 'd' is the destination to which the packet is sent. • indicates nodes which receive data. In the SM (Single Map) scheme, all nodes at the same level use a unique bit map, as shown in Figure 2(a), and thus, no broadcast is made (unless a bit map is all 1).

LARP scheme: In the LARP (Local Approximate Remote Precise) scheme, the bit map is used only for nodes which are root of subtrees far from the source node. Using this scheme, the bit map is used for remote nodes of the source node, while the message is broadcast in the subtree marked B. In the example shown in Figure 2(b), since the multicast starts at level 1 (the top node (level 0) only sends a message to a child), the broadcast (marked **B**) to the subtree which includes the source node starts at level 2. In this case, the broadcast is carried out for local nodes of the source node, while the bit map is used for remote nodes.

Unlike the hierarchical bit-map directory method which requires $\sum_{k=1}^{m} n^k$ bits for each entry, the RHBD only requires nm bits for each entry. However, PUs which are marked with • without 'd' receive packets or messages unnecessary for compaction of the bit map. Although unnecessary messages are merely discarded in the cache controller of the PU, it may cause congestion of the network.

2.3 Pruning cache

The major disadvantage of the RHBD is that it requires unnecessary message multicast since it only uses a single bit map or broadcasting in each level of the hierarchy. From the simple evaluation, the RHBD generates more than 100 times messages as really required messages in the worst case[9].

To cope with this problem, "the bit-map Pruning Cache (P.C.)" is proposed in the switching element of the backward MIN. The bit-map pruning cache is a small cache of the bit map referred to using the line address of multicasting packets. When the line readout packet is transferred through the backward MIN, the pruning cache is referred by the line address in the packet header. If the address is matched, the bit corresponding to the output label of the switching element is set as shown in Figure 3. Otherwise, an entry is discarded and the new entry is registered. After this registration, the bit corresponding to the output label of the switching element is set.

When the invalidation message or updating data are multicast, the bit-map pruning cache is checked. If there is an entry which matches the address, the bit map of the pruning cache is utilized for multicasting, otherwise, multicasting is carried out according

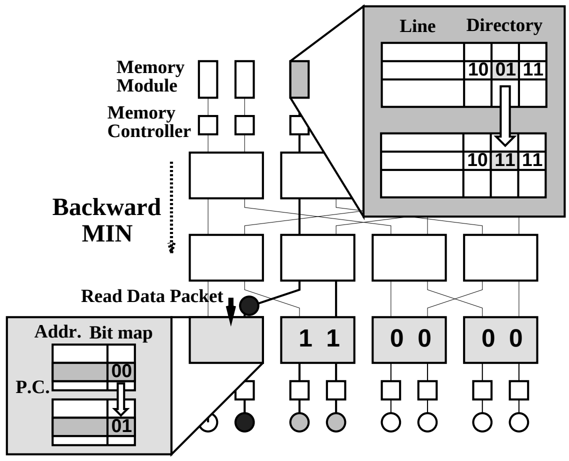

Fig. 3 Bit-map pruning cache (only provided in the first stage)

to usual RHBD schemes. Using this mechanism, unnecessary packets are pruned according to the bit map in the cache. If this buffer is used in each hierarchy, the bit map which is the same as the original hierarchical bit-map directory method, is generated temporarily in each switching element of the backward MIN. In invalidation-type protocols, the entry of the pruning cache is removed after multicasting of the invalidation message. Note that, the miss of the pruning cache just causes unnecessary packets.

Since the bit-map pruning cache must be implemented inside the switching element, the size is limited. If the entry of the cache is full at the registration, an entry is selected and removed. Two selection policies, the FIFO policy for the invalidation-type protocol, and the LRU policy for updating protocol, are used.

For reducing the ammounts of required hardware and latency caused by accessing the pruning cache, the bit-map pruning cache can be omitted in some stages. Based on the simulation results shown later, a satisfactory effect is achieved when the pruning cache is provided in switching elements of only one stage, as shown in Figure 3.

A similar concept has been proposed by Scott and Goodman[16], applied to the k-ary n-cube, and evaluated theoretically. It is based on simple broadcasting, unlike our method which is based on the RHBD. Although there is no detailed description on the protocol management, Scott and Goodman's pruning cache is for a traditional CC-NUMA system with invalidation protocol.

2.4 Coherent control of the MINC

The directory is required for each cache line in the main memory modules, and managed by the memory controller. When the coherent message is required, the bit map is cut out from the directory, pushed into the

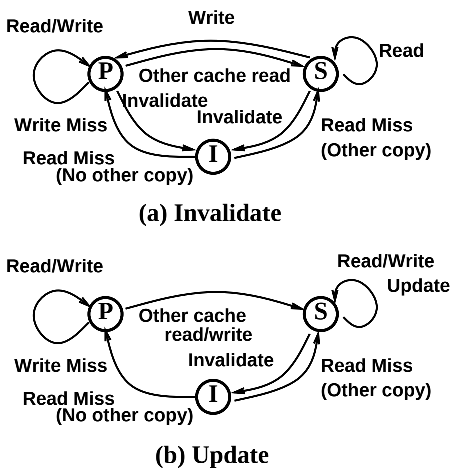

Fig. 4 State transition graph of the MINC

packet header, and directly used as the multicast bit map.

A counter field, which stores the number of PUs holding copies of the cache line, is required when the update protocol is used. If only an invalidation protocol is used, a bit flag indicating whether the line is shared with multiple PUs or not is required instead of the counter.

On the other hand, two status bits are required for the tag of the cache provided between the MIN and PU. Like other cache protocols, the status bits represent

- Valid/Invalid (V/I) and
- Shared/Private (S/P).

As shown in Figure 4, the cache-coherent protocol of the MINC is based on the write-through policy with three states (I: Invalidate, P: Private, and S: Shared). The reason that states P and S exist in spite of the write-through protocol is that write to a line in state P can be completed with immediate updating of its content, while the completion of the write to a line in state S must be postponed until the memory accepts the write operation.

Both invalidate-type protocol (Figure 4(a)) and update-type protocol (Figure 4(b)) are available. Although the update-type protocol requires a lot of traffic, it will work efficiently for scientific computation which requires a lot of data multicasting.

When the read access hits the cache, the data is read out directly from the cache. For other cases, the cache, memory controller, and MINs behave as follows.

(1) Write hit/miss-hit:

If the write request hits to the cache, the cache is updated and the state is changed to ’P’. Since the write-through policy is adopted, the address and data are transferred to the memory module through the forward MIN. After updating the memory module, the *invalidation message* or *data for update* is multicast using the backward MIN according to the bit map of the directory. When the invalidation protocol is used, all bits of the directory are reset and the counter field (flag) is cleared.

(2) Read miss-hit:

If the invalidated (state I) line is not found in the cache, a line is selected and exchanged according to the **swapping out** operation described later. Then, the cache controller sends a *line request packet* to the memory module through the forward MIN. The memory controller receives the requesting PU number in the line request packet, and sets the corresponding bit of the directory. At the same time, the controller reads the line and sends it to the requesting PU with *the line read-out packet* through the backward MIN. Here, the following operations are required depending on the value of the counter (flag).

- If the value of the counter is zero, no other copy exists. In this case, the private bit in the header of the line read-out packet is set. A cache controller, which receives the packet whose private bit is set, changes the line state into ’P’.
- If the value of the counter is one, *the sharing message* is multicast after sending the line read-out packet also through the backward MIN according to the bit map. A cache controller which receives this packet changes the corresponding line state into ’S’.

In both cases, the counter is incremented if update protocol is used.

(3) Swapping out:

Since a bit in the directory may represent multiple copies in the RHBD, the bit in the directory must be reset carefully.

When a line is discarded for replacement by another line, *the discarding message* is sent to the memory module through the forward MIN in the update protocol. The memory controller which receives this message decrements the counter. When the value becomes zero, all bits of the directory are cleared. In the invalidation protocol, since the directory is frequently cleared by each data writing operation, the discard message is unnecessary. In this case, the line to be replaced is just discarded without any operation.

3. Performance Evaluation

3.1 Simulation Model

The network traffic and congestion caused by the coherent messages are evaluated using a simple probabilistic model. The effect of the RHBD and pruning cache in the MINC are sensitive to the message traffic pattern. Here, the following two traffic models are used for deciding the accessing address;

- Nonlocal: a processor accesses every memory module with the same probability.
- Local: the destination memory module of a packet from a processor p follows a normal distribution whose center is p and standard deviation $SD = 1$. Thus, a processor p accesses memory module p or neighbors more frequently than other memory modules.

The control model of the forward/backward MIN is based on the SSS (Simple Serial Synchronized) MIN[5]. In this control model, packets are transferred through a few bit serial lines synchronized with a common frame clock, from input buffers prepared between PUs and the MIN. Since each switching element stores only one bit (or a few bits) of the packet, the SSS-MIN behaves like a set of shift registers with switching capability.

When a conflict occurs, one of the conflicting packets is routed to an incorrect direction since the SSS-MIN provides no packet buffer in each switching element. The conflict bit in the routing tag is set when the conflicting packet is routed to an incorrect direction. Such a packet is treated as a dead packet and never interferes with other packets. The conflicting packet is inserted again in the next frame from the input buffer.

Although the MINC can be implemented on the traditional MIN which provides the packet buffer inside the switching element, the control mechanism for multicast becomes much simpler in the SSS-MIN. In the backward MIN, the bit map of the directory is copied to the controller when a coherent message is multicast. Since the multicast message may conflict with other messages, only bits corresponding to the successful message are reset, and the multicast is repeated until all bits are cleared.

3.2 Traffic load

First, the number of packets transferred in the backward MIN for coherent control is evaluated. Here, we call the total number of packets which are transferred on links between switching elements in the backward MIN the *traffic load*. If a large number of unnecessary packets is generated with the RHBD, the traffic load increases and the congestion of the backward MIN will degrade the performance.

Figure 5 shows the traffic load versus the number of copies (the number of PUs which share a cache line). The results of four schemes, the LARP, SM, LARP with the pruning cache (LARP/P.C.), and SM with the pruning cache (SM/P.C.) are shown in the figure with the ideal case (IDEAL) which includes no unnecessary packets. In this case, the bit-map pruning cache is only provided in switching elements of the first stage (the nearest stage to the PUs). Here, it is assumed that the size of P.C. is infinite. The *Local* traffic model is used on the system with 4096 PUs. The backward MIN is a 4-stage baseline network with 8×8 switching elements.

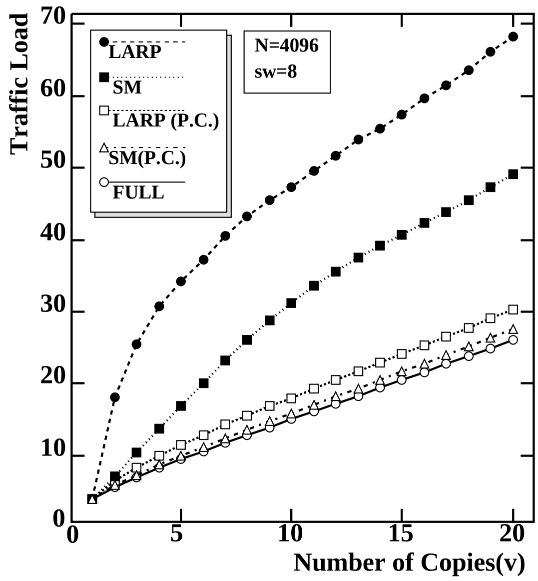

Fig. 5 Traffic load versus the number of copies in the system

Figure 5 shows that the traffic load of the LARP is worse than that of the SM, even with the pruning cache (LARP/P.C.). The traffic load is especially large with a large number of copies. From this evaluation, it appears that the SM is superior to the LARP in the MINC. Unlike the interconnection network used in JUMP-1[9], the backward MIN of the MINC includes complete 8-ary trees. In such trees, depending on the location in the tree, some messages should have been transferred through the very root of the tree just to share the data with the neighboring node. The broadcasting used in the LARP generates a large number of unnecessary packets in such cases.

The bit-map pruning cache greatly improves the traffic load in both cases. The traffic load of the SM with the pruning cache is close to that of the ideal case.

3.3 Congestion of the backward MIN

In order to evaluate the performance, taking into consideration packet conflict, the behaviors of a switching element, memory module, and PUs are simulated in detail. Table 1 shows parameters used in the simulation. Since a large execution time is required for detail simulation, the target system with 256 PUs is evaluated. In this case, the MIN is the 3-stage baseline network with 4×4 switching elements. Considering the implementation, the size of the bit-map pruning cache (PCsz) is

Table 1 Simulation parameters

Packet generation	
Pr	The probability that each processor issues a packet per one frame (0.2 – 0.35)
r	The ratio of reading access in the access which each processor issues (0.8)
System parameters	
N	System size (256)
sw	Switch size (4x4)
PCst	The location of the bit-map pruning cache (0)
PCsz	The size of the pruning cache (4)
Memory parameters	
GM	The size of the shared memory size [bytes](64M)
LC	The size of the processor cache size [bytes](256K)
BS	The size of the cache line size [bytes](16)

set to be 4. Other parameters are fixed as shown in Table 1.

In the SSS-MIN, if there are many conflicts, multicasting must be retried many times. Thus, the congestion of the MIN is represented by the average number of frames required for the completion of a multicast. Here, the stage number is in ascending order from the one nearest the processor. Here, only the SM scheme is evaluated since it appears superior to the LPRA based on the traffic load analysis. Two possible protocols, invalidate (inv.) and update (upd.), are evaluated under two traffic patterns. The traffic patterns are mixtures of the above-mentioned traffic models:

- global dominant (GD) includes 90% *non local* mode traffic and 10% *local* mode traffic, and
- local dominant (LD) includes 10% *non local* mode traffic and 90% *local* mode traffic.

Figure 6 shows the average number of required frames versus the probability of packet generation by a PU. With a large probability, the number of frames increases. Congestion becomes severe especially when the probability exceeds 0.35. However, this probability is unrealistic in systems with a reasonable amount of private cache. This figure also demonstrates the effect of the bit-map pruning cache. The small number of pruning caches (4 entries) improves the number of required frames from 60% to 80%.

From this figure, it appears that the congestion of the backward MIN is severe when the update protocol is used. While the bit map of the directory is cleared when data are written into the corresponding line in the invalidation protocol, the bit map remains until all copies are swapped out in the update protocol. Therefore, the number of multicasts required for the update protocol is larger than that of the invalidation protocol. Based on this evaluation, the invalidation protocol seems to be better than the update protocol. However, the update protocol will be advantageous in scientific calculations which require a large number of data multicasts. Trace-driven or execution-driven simulation based on practical application programs is required for further comparison.

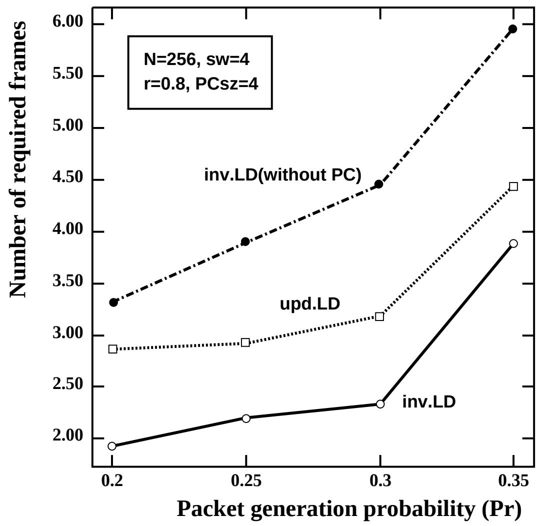

Fig. 6 Average number of required frames versus packet generation probability

Table 2 Pruning cache hit ratio

inv.LD	inv.GD	upd.LD	upd.GD
77.87%	48.52%	54.20%	34.22%

Table 2 shows the hit ratio of the bit-map pruning cache provided in switching elements of the first stage. Needless to say, the hit ratio under local dominant traffic is higher than that of global dominant traffic. Under local dominant traffic, a high hit ratio is achieved with only 4 entries. In the invalidation protocol, the pruning cache is cleared when the invalidation message is multicast. Since only active entries are held in the pruning cache, the hit ratio of the invalidation protocol is higher than that of the update protocol.

4. MINC cache control chip

In previous sections, the MINC is considered as a cache control mechanism dedicated for the MIN. This method can be easily extended to other switches by separating the communication for cache control from other data transfer. Since the data themselves are not transferred in the MINC cache control chip, only invalidation protocol is used.

We designed the MINC cache control chip which manages only address packets of accessing lines and signals for controlling cache. This chip can be used with any other high-bandwidth switch for data transfer. For implementation, the following problems must be solved.

(4) Collision of the multicast packets:

When independent cache control messages are multicast simultaneously, packets may conflict with each other. In this case, only a part of destination processors can receive the control messages. Thus, a mechanism is required for resending messages only to the processors which could not receive them.

To cope with this problem, an acknowledge packet is introduced. When a conflict of the multicast packets occurs, an acknowledge packet is automatically sent to the shared memory controller. The memory controller transfers the multicast packet again only for the branches with conflicts.

(5) Inconsistency when the entry is full:

When a new entry is required to be registered to the pruning cache whose entry is full, inconsistency may occur between the pruning cache and the directory on the memory module.

If an entry corresponding to line **A** is discarded for making a new entry, inconsistency is introduced in the following casses: (1) some entries are purged by a write request, and (2) a new request for line **A** is registered. In this case, the new entry for line **A** is inconsistent with the directory on the memory module since the bit map corresponding to previous requests had been discarded. This situation will occur frequently when two processors share a level-0 switch.

There are two approaches to address this problem.

- In the directory on the shared memory, the 'pruning cache reference bit' is added. This bit is set when inconsistency occurs, and indicates that the corresponding line is never entered in the pruning cache until the write access to the cache line is issued. In order to inform of the inconsistency, a packet is sent to the shared memory when a line is discarded.
- Another acknowledge packet indicating whether the cache is necessary or not is returned when a multicasting packet reaches the cache of each processor.

The former approach is easily implemented even though the hit ratio is degraded by prohibiting the pruning cache, especially when the update protocol is used. Although such degradation can be avoided in the latter approach, the control sequence becomes complicated. In the MINC cache control chip, the former approach is adopted for the following reasons.

- In the MINC cache control chip, only the invalidation protocol is used. In this case, our initial evaluation results based on the trace simulation show a hit ratio greater than 50% can be achieved[17] in the worst case. However, this simulation is based on the trace of small parallel programs, and further extensive research is required.
- The latter approach requires a complicated control sequence of the pruning cache, and it is difficult to implement in the current chip.

Table 3 Specifications of MINC chip

Package	391pin PGA
Process	0.4μm LPGA
Network size	16 inputs/outputs
Clock	66MHz
Number of utilized cells	37372
Number of signal pins	264

The MINC cache control chip is developed on a 0.4μm LPGA (Laser Programmable Gate Array) with 50k gates.

Table 3 shows the utility of the number of gates. In spite of the pin limitation, this chip can be used for a system with 16 processors/memory modules.

Although complicated handling is required for the pruning cache, it operates with a 66MHz clock.

5. Conclusion

A new cache-coherent MIN called the MINC is proposed. By combining the RHBD and bit-map pruning cache, each switching element can multicast coherent messages quickly without accessing outside memory nor generating a large number of unnecessary packets.

Through the simulation, it appears that the SM (Single Map) scheme of the RHBD is suitable for the MINC. The pruning cache works most effectively when it is provided in every switching element of the first stage, and it reduces the congestion more than 50% with only 4 entries. Although the update protocol requires a larger number of messages than the invalidation protocol, it will be advantageous in scientific calculations which require a large number of data multicasts. The trace-driven or execution-driven simulation based on practical application programs is required for further comparison.

For implementation, the MINC mechanism is separated from data transfer, and the MINC cache control chip with 16 input-outputs is implemented on the LPGA. This chip consisting of 37k gates operates with a 66MHz clock.

Acknowledgement

The development of the MINC chip is supported by the VLSI Design and Education Center (VDEC), the University of Tokyo.

The authors also thank the anonymous referee for helpful suggestions on improving this paper.

References

[1] L.N. Bhuyan, A.K. Nanda, and T. Askar, "Performance

and reliability of the multistage bus network," Proc. of International Conference on Parallel Processing, pp.I-26–I-33, Aug. 1994.
[2] G. Broomell and J.R. Heath, "Classification categories and historical development of circuit switching topologies," ACM Computing Surveys, Vol.15, No.2, June 1983.
[3] D. Chaiken and A. Agrawal, "Software-extended coherent shared memory: performance and cost," Proc. The 21st International Symposium on Computer Architecture, pp.314–324, 1994.
[4] H. Cheong and A.V. Veidenbaum, "A cache coherence scheme with fast-selective invalidation," Proc. of 15th International Symposium on Computer Architecture, pp.299–307, 1988.
[5] H. Amano, L. Zhou, and K. Gaye, "SSS (Simple Serial Synchronized)-MIN: a novel multistage interconnection architecture for multiprocessors," Proc. of the IFIP 12th World Computer Congress, Vol.I, pp.571–577, Sept. 1992.
[6] K. Hiraki, H. Amano, M. Kuga, T. Sueyoshi, T. Kudoh, H. Nakashima, H. Nakajo, H. Matsuda, T. Matsumoto and S. Mori, "Overview of the JUMP-1, an MPP prototype for general-purpose parallel computations," Proc. IEEE International Symposium on Parallel Architectures, Algorithms and Networks, pp.427–434, 1994.
[7] H. Matsumoto and T. Hiraki. "The shared memory architecture on the massively parallel processor," Technical report of IEICE, CPSY 92-36, pp.47–55, 1992.
[8] D.V. James, A. T. Laundrie, S. Gjessing, and G.S. Sohi, "Distributed-directory scheme: scalable coherent interface," IEEE Computer, 23(6):74–77, 1990.
[9] T. Kudoh, H. Amano, T. Matsumoto, K. Hiraki, Y. Yang, K. Nishimura, K. Yoshimura, and Y. Fukushima, "Hierarchical bit-map directory schemes on the RDT interconnection network for a massively parallel processor JUMP-1," Proc. International Conference on Parallel Processing, Aug. 1995.
[10] J. Kuskin, D. Ofelt, M. Heinrich, J. Heinlein, R. Simoni, K. Gharachorloo, J. Chapin, D. Nakahira, J. Baxter, M. Horowitz, A. Gupta, M. Rosenblum, and J. Hennessy, "The Stanford FLASH multiprocessor," Proc. 21st International Symposium on Computer Architecture, pp.302–313, April 1994.
[11] D. Lenoski, J. Laudon, K. Gharachorloo, W. Weber, A. Gupta, J. Hennessy, M. Horowitz, and M. Lam, "The Stanford DASH multiprocessor," IEEE Computer, 25(3):63–79, 1992.
[12] H.E. Mizrahi, J.L. Baer, E.D. Lazowska, and J. Zahorjan, "Introducing memory into the switch elements of multiprocesssor interconnection networks," Proc. 16th International Symposium on Computer Architecture, pp.158–166, 1989.
[13] A.K. Nanda and L.N. Bhuyan, "Design and analysis of cache coherent multistage interconnection networks," IEEE Trans. Comput., 42(4):458–470, 1993.
[14] A.V. Veidenbaum, "A compiler-assisted cache coherence solution for multiprocessors," Proc. International Conference on Parallel Processing, pp. 1026–1036, Aug. 1986.
[15] P. Stenström, "A consistency protocol for multiprocessors with multistage networks," Proc. The 16th International Symposium on Computer Architecture, pp.407–415, 1989.
[16] S.L. Scott and J.R. Goodman, "Performance of pruning-cache directories for large-scale multiprocessors," IEEE Trans. Parallel and Distributed Syst., vol.4, no.5, pp.520–534, May 1993.
[17] T. Kamei, "An implementation of cache coherent network for MIN based multiprocessor," Master Thesis, Graduate School of Science and Technology, Keio University, 1997 (in Japanese).

Toshihiro Hanawa received his B.E. and M.E. from Keio University, Japan, in 1993 and 1995, respectively. He is a Ph.D. candidate in the Department of Computer Science, Keio University, Japan, and a Research Fellow of the Japan Society for the Promotion of Science. His research interests include analysis of interconnection network.

Takayuki Kamei received his B.E. from Keio University, Japan, in 1995. He is a Master course student in the Department of Computer Science, Keio University, Japan. His research interests includes implementation of interconnection network.

Hideki Yasukawa received his B.E. and M.E. from Keio University, Japan, in 1994 and 1996, respectively. He now works in the System ULSI Engineering Laboratory, Toshiba Corporation.

Katsunobu Nisimura received his B.E. from Tokyo Engineering University, Japan, in 1994, and M.E. from Keio University, Japan, in 1996. He is a Ph.D. candidate in the Department of Computer Science, Keio University, Japan. His research interests include simulation of a massively parallel processor.

Hideharu Amano received his B.E., M.E., and Ph.D. degrees from Keio University, Japan, in 1981, 1983, and 1986, respectively. He is now an associate professor in the Department of Electrical Engineering, Keio University. His research interests include parallel processing.