

Maestro2: A new challenge for high performance cluster network

Shinichi Yamagiwa* Keiichi Aoki** Masaaki Ono**

Tetsuya Sakurai** Koichi Wada** Luis Miguel Campos*

*PDM&FC, Rua Latino Coelho, 87, 1050-134, Lisboa, Portugal

**Institute of Information Sciences and Electronics, University of Tsukuba,
1-1-1 Tennodai, Tsukuba, Ibaraki, 305-8573, JAPAN

ABSTRACT

Cluster computers have been gaining popularity as a high performance computing environment for parallel/distributed applications. Commonly, the network hardware and protocols used to build a cluster are based on WAN or LAN technologies, which add overheads to guarantee reliability in wide-area communications. However the communication patterns of parallel applications running on clusters require low latency and high throughput. Thus, there is a disparity between the performance obtained by using conventional network and protocol technologies and the one required by clusters. To address this issue, we have been developing a high-performance network optimized for cluster computing, called Maestro2. This paper presents the overall architecture of Maestro2 and shows preliminary performance evaluation results. Several novel techniques to extract potential performance of a physical medium are described. From the simulation-based experiment, we confirm that Maestro2 achieves better performance than the currently available gigabit-based networks.

Keywords: cluster computing, high performance communication, network hardware, link protocol, performance evaluation.

1. INTRODUCTION

Fast performance increases in commodity computing resources such as processors, memories, networks, and its peripherals has led to a surge of interest in cluster computers[1]. However the tendency among researchers building such cluster computers is to use off-the-shelf components. As such the most popular clusters use Ethernet and TCP/IP as its communication device and protocol, respectively. Although there are obvious advantages (for instance price and simplicity) to build clus-

ters using off-the-shelf components especially for computation, there are also serious disadvantages especially when using conventional networking technology. Those conventional WAN or LAN-based network devices and protocols are not optimal for clusters, since they include overheads to guarantee reliability in wide area communications[3]. Given that clusters are very often organized in geographically localized spaces such as laboratories or offices, hardware and software for communication can be optimized to improve the overall performance of the system.

The communication patterns of cluster applications require low latency and high throughput. For instance, distributed parallel simulations that exchange small messages among processors at each simulation step require low latency in communication. Many matrix computations such as FFT that exchange large messages among processors require high throughput.

As mentioned above, there is a disparity between the obtainable communication performance using conventional network technology and the requirements of cluster applications. Trying to solve this disparity, we have developed a high-performance network for cluster computing, called *Maestro*[8]. Additionally, a high-performance message passing library was also developed on *Maestro*[7]. The minimal latency and the maximum throughput of *Maestro* are 8 μ s between any two processors' memories, and 20Mbytes/sec, respectively. The maximum throughput of *Maestro* is 80% of the throughput of the physical medium (200Mbps IEEE1394 physical layer),

As a follow-on research to the *Maestro* project, we are now developing *Maestro's* next generation network, called *Maestro2*. Through the experience gained during the *Maestro* project, we have identified the two issues that restricted the performance. The first issue occurs at the link layer. In [8] we proposed the *network burst transfer* at the link layer. In the network burst, the sender aggregates multiple packets that compose part of a message, and sends them in burst to the receiver. We con

firmed network burst was able to keep the utilization ratio of the physical medium higher than conventional link layers. However, when the sender finished transferring the number of packets decided at the beginning of the transfer, the burst transfer is inevitably terminated. This termination increases arbitration overhead in acquiring the physical medium. The other issue occurs in the switch. Maestro's switch transfers one message at a time. In addition, the transfer is stopped when the source or destination network buffer becomes full or empty. This mechanism decreases the dynamic communication performance. This paper describes the overall architecture of Maestro2 and shows preliminary performance evaluation results. In the next section we describe the design and implementation of Maestro2 cluster network and propose two novel techniques to solve the two problems that restricted performance in Maestro. In section 3, we show preliminary performance evaluation results for Maestro2 through communication experiments and compare the performance of Maestro2 against other high performance networks. We conclude this paper and describe future research in section 4.

2. Maestro2 CLUSTER NETWORK

During the development and testing of Maestro we have identified two issues that restricted the performance of the network. In this section we introduce two new techniques that successfully solve both problems.

2.1. New techniques

In Maestro2, two new techniques are introduced into the link layer and the switch as follows:

Continuous network burst transfer: The first technique is the *continuous network burst transfer*. This technique allows for the continuation of the network burst as long as there are packets in the network buffer. This can be done by inserting the *continue command* at each boundary of the burst. We use a frame configuration

similar to the one used for the network burst transfer. That is, the frame consists of one or more fine-grained *packets* and a header. A packet consists of user data. At the beginning of the transfer, the header of the frame is dispatched to the physical layer, and then arbitration is done to acquire the physical medium. When there are multiple packets to be sent in the network buffer, the continuous network burst is invoked for all or a part of those packets. As the burst transmission progresses, it is possible to append subsequent packets to the network buffer. The conventional network burst never continues the transmission beyond the number of packets predetermined at the beginning of the transmission because the frame length has to be determined at the beginning of the header transfer. However, the continuous network burst is able to continue the transmission by inserting the *continue command* into the first burst transmission without having to reacquire the physical medium.

Figure 1 compares the conventional network burst transfer with the continuous network burst transfer. It assumes that messages are written into the sender's network buffer one after another. The conventional network burst transfer terminates soon after it finishes transferring the amount of packets determined at the beginning of the transfer. Therefore, the arbitration time for acquiring physical medium accumulates. On the other hand, the continuous burst transfer can reduce the number of arbitrations.

This technique increases the utilization ratio of the physical medium, and achieves higher throughput than the conventional network burst.

Out of order switching: While routing messages at a switch, the message might be blocked when the destination buffer becomes full. Such occurrences force subsequent messages to be blocked if the switch only routes messages in-order. This in-order switching decreases performance when the next message to route is for a different destination and when the destination buffer is not full.

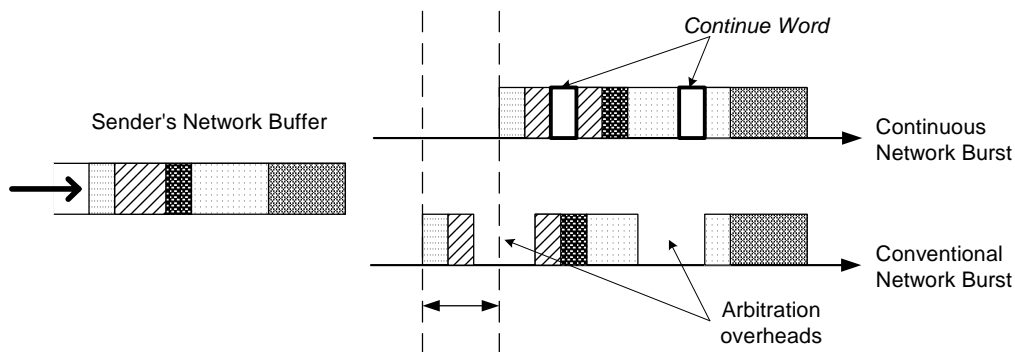


Figure 1 : Continuous network burst vs. conventional network burst

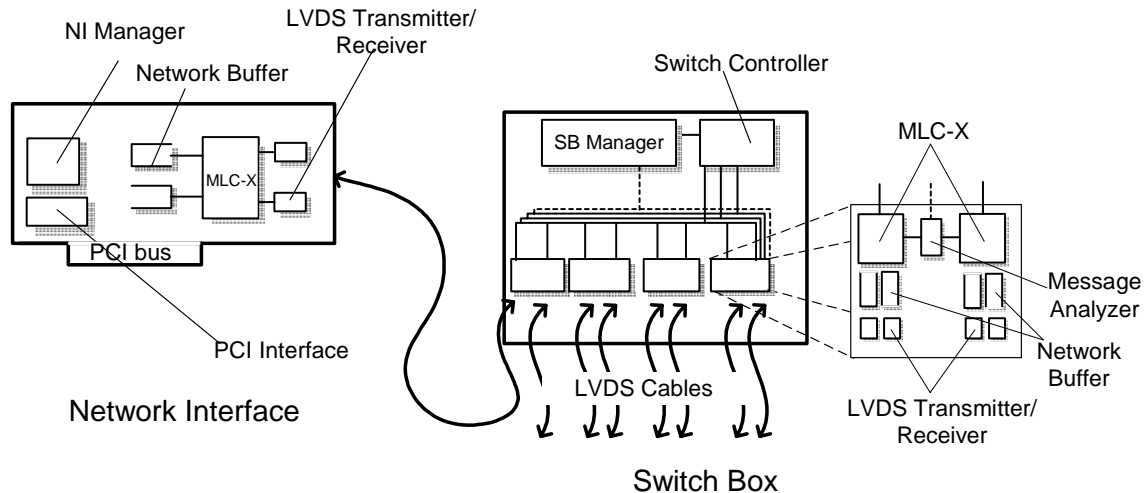


Figure 2 : Maestro2 cluster network

To improve efficiency in switching, we propose an *out-of-order switching*. In the out-of-order switching, multiple transfer requests are stored in a *transfer reservation buffer*. For each request it is examined whether or not its destination buffer is full. If the buffer is not full, the corresponding request is marked as *active*. The switch works by picking up active requests and routing the messages to their destinations. The out-of-order switching is applicable to various kind of switch architecture, such as crossbar, single or multiple buses-based, and so on.

2.2. Implementation

A simple Maestro2 network is composed of network interfaces (NI) and a switch box (SB) as shown in **Figure 2**. Each network interface is connected via two LVDS (Low Voltage Differential Signaling) [2] cables for transmission and reception, and is connected to a commodity processing element such as a PC or a WorkStation via a 64bit@66MHz PCI bus. The connection between NI and SB is full-duplex, and the peak bandwidth is 6.4Gbps. Currently, the SB has eight ports for connections to NIs. One or more ports can increase the fan out of the switch by cascading SBs.

The NI includes a NI manager, a PCI interface, network FIFO buffers, a link controller (MLC-X) and LVDS transmitter/receiver. The NI manager works as a processor element in charge of handling communications, and is composed of a PowerPC603e@300MHz and 64Mbyte SDRAM. The PCI interface maps the address space of the SDRAM and host processor's memory into the PowerPC's address space. The 8Kbyte network FIFO buffers store incoming and outgoing messages. The MLC-X is a full duplex link layer controller on which the continuous network burst transfer is implemented.

MLC-X supports two communication channels between the network FIFO buffers. And finally, LVDS transmitter/receiver drives its physical medium under control of MLC-X. It transmits and receives data via its 6.4Gbps full duplex link. The PCI interface, network buffers, and MLC-X are implemented into the Virtex-II FPGA chip[6].

The SB consists of four SB interfaces, an SB manager and a switch controller. Each SB interface manages two ports and includes a message analyzer. The message analyzer extracts each message header of an incoming message, and passes it to the SB manager. The SB manager consists of a PowerPC603e, 32Mbyte SDRAM, and a routing circuit that generates and writes requests to the switch controller. The switch controller transfers messages from source to destination(s) using the out-of-order mechanism mentioned earlier.

3. EVALUATION

We are currently designing the RTL (Register Transfer Level) implementation of Maestro2. Out-of-order switching is still being implemented. Therefore, we will show the preliminary simulation-based performance without the out-of-order switching in this section. **Figure 3** shows the simulation model used for the evaluations performed in this section. The hardware modules are implemented in RTL, but control software and peripheral controllers are implemented at the behavioral level. The part implemented in RTL can emulate the actual hardware precisely down to the clock-level. On the other hand, the parts implemented at the behavioral level are the peripheral components such as a PC and devices for LVDS.

Therefore, we assume the latencies of the LVDS physical layer and the DMA transfer's startup via PCI bus are $30\eta\text{s}$ and $300\eta\text{s}$ respectively. In addition, the send and receive mechanism of the NI manager is not able to run simultaneously because the PowerPC cannot process a send request and a receive request at the same time.

In the following discussion, we will show the basic performance of Maestro2 comparing with Maestro. The comparison with other commercial networks will be shown in the next subsection.

3.1. Basic performance

Figure 4 shows the basic performance by using PING-PONG and PINGPING transfers. PINGPONG transfer is a round-trip memory-to-memory transfer between two host processors. In case of PINGPING transfer, two NIs send a message simultaneously between their host processors' memories. **Figure 4** shows the how latency and

throughput of Maestro and Maestro2 vary when the message size increases.

In the case of PINGPONG transfers, the minimum latency of Maestro2 is $1\mu\text{sec}$ when the message size is 32bytes. This is 1/20 of Maestro. The maximum throughput of Maestro2 is 336Mbytes/sec, which is 84% of peak performance of the physical link. Moreover, the throughput is 17 times higher than Maestro's.

PINGPING results show the worst case scenario where the message transfers are serialized. In Maestro, transfers are inevitably serialized when congestion occurs. In the case of Maestro2, the serialization also occurs, when the source and the destination of messages are identical. The throughput of Maestro and Maestro2 is one half of their PINGPONG performance. However, in Maestro2, when the out-of-order switching is introduced, the performance for PINGPING should increase.

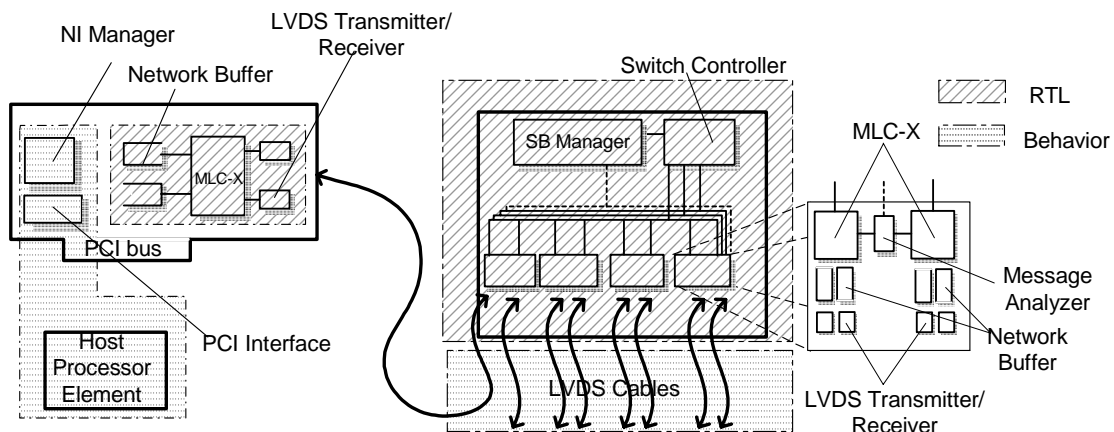


Figure 3 : Simulation model for evaluations

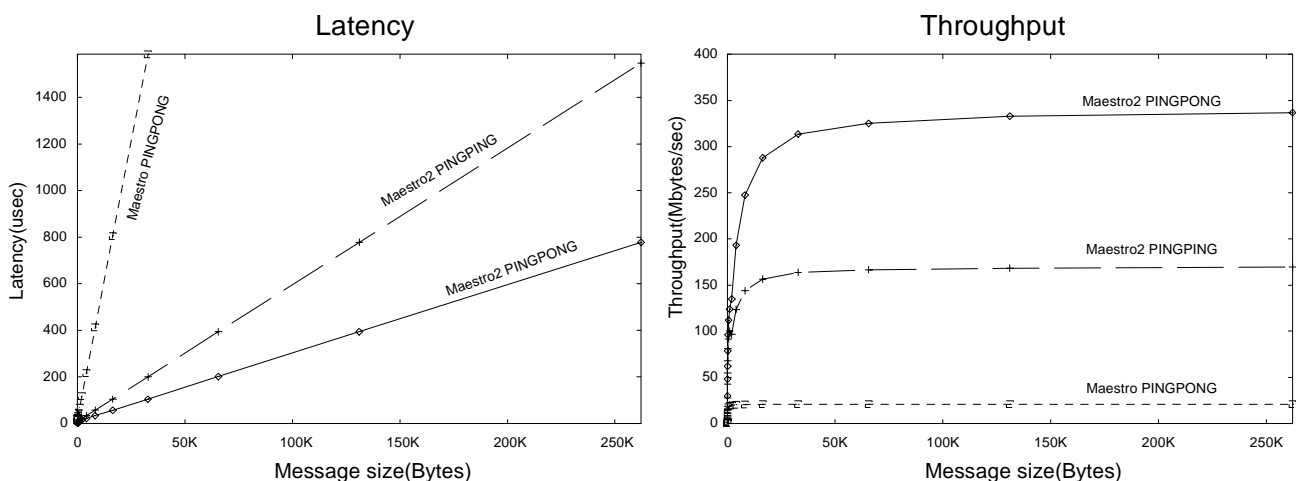


Figure 4 : Basic Performance

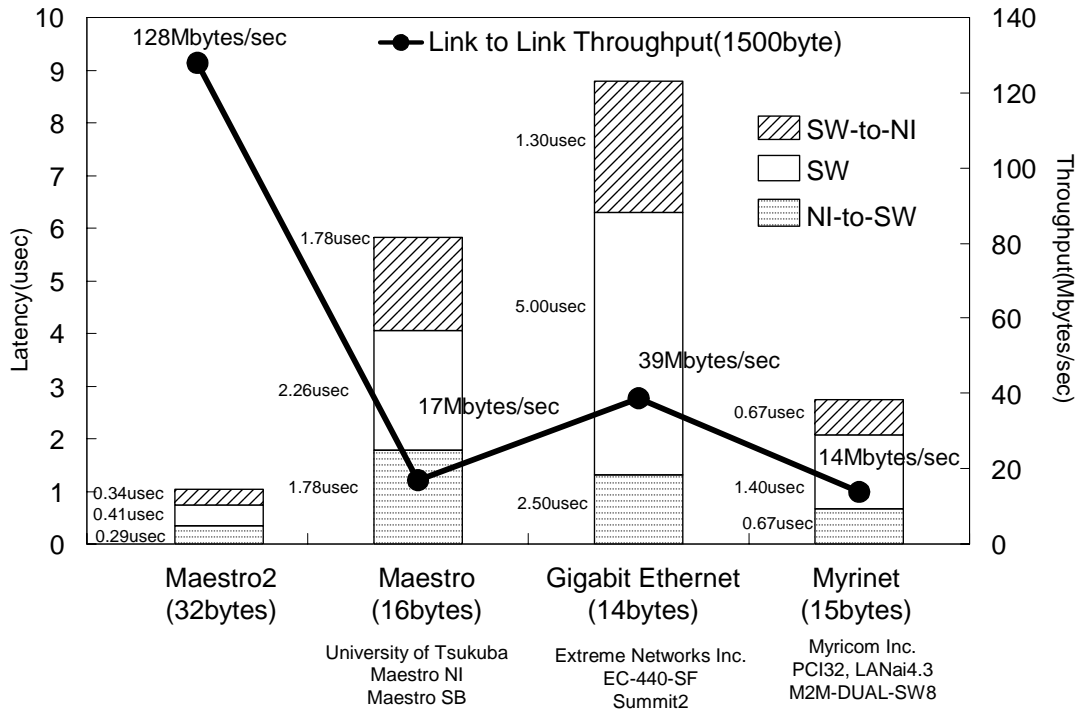


Figure 5 : Comparison with other networks

3.2. Performance comparison with other networks

Figure 5 compares the performance of Maestro2 with Maestro, Gigabit Ethernet, and Myrinet.

The bars in Figure 5 show the latencies when a small message is transferred between two NIs (i.e. link layer to link layer transfer). To obtain these results, we used the information described in [4][5][8]. NI-to-SW, SW, and SW-to-NI indicate the latency from the sender's NI to the Switch, the latency within the Switch and the latency from the Switch to the receiver's NI, respectively.

Maestro2 latency is the smallest of all networks being tested. Especially important is the switching latency of Maestro2 which is only 18% of the first generation Maestro. The overall latency is 18%, 12% and 38% of Maestro, Gigabit Ethernet and Myrinet, respectively.

The solid line in Figure 5 shows the throughput when the message size is 1500 bytes. This message size is chosen according to the packet size of the Gigabit Ethernet. The throughput of Maestro2 ranges from 3.2 to 9.1 times higher than the other networks. This result indicates that the continuous network burst is highly effective.

Through the comparison shown above, we can conclude that Maestro2 achieves low latency and high throughput communication irrespective of the message size.

4. CONCLUDING REMARKS

This paper described the design and implementation of the Maestro2 cluster network, and showed preliminary performance evaluation of Maestro2.

From the evaluation results, we confirmed that Maestro2 achieves very low communication latency (of 1μsec), and high utilization ratio of the physical medium (high throughput).

The evaluation results showed that the architecture of Maestro2 drastically improves the performance of Maestro. Moreover, by comparing the performance of Maestro2 against recent gigabit-base networks, we confirmed that Maestro2 cluster network is superior at this time.

The board assembly of NI and SB has been completed. Future research will include out-of-order switching performance evaluation. We also plan to use actual hardware instead of a simulation model. We also plan to develop a high performance message passing library on Maestro2 in the next few months.

4.1. Acknowledgements

This research was partially supported by Axell Corporation and the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (B), 14580361, 2002.

5. REFERENCES

- [1] M. Baker, R. Buyya, and D. Hyde. Cluster computing: A high-performance contender. *IEEE Computer*, July 1999.
- [2] IEEE Standard Department. IEEE Standard for Low-Voltage Differential Signals (LVDS) for Scalable Coherent Interface (SCI). 1996.
- [3] Vijay Karamcheti and Andrew A. Chien. Software overhead in messaging layers: Where does the time go? In *Proceedings of International Conference on Architectural Support of Programming Languages and Operating Systems (ASPLOS-VI)*, pp. 526-531, 1994.
- [4] Motohiko Matsuda et al. Network interface active messages on SMP clusters. *Technical Report on IPSJ SIGARC*, Vol.97(No.125):55-60, 1997.
- [5] Sinji Sumiyoshi et al. The design and evaluation of high performance communication library using a gigabit ethernet. *Technical Report on IPSJ SIGHPC*, Vol.72(No.19), 1998.
- [6] Xilinx Inc. Virtex-II Platform FPGA Data Sheet. 2002. <http://www.xilinx.com>.
- [7] Shinichi Yamagiwa and Koichi Wada. Design and Implementation of Message Passing Library on Maestro Network. In *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM'01)*, 2001.
- [8] Shinichi Yamagiwa, Munehiro Fukuda, and Koichi Wada. Design and Performance of Maestro Cluster Network. In *Proceedings of IEEE International Conference on Cluster Computing (CLUSTER2000)*, November 2000.