THE EVALUATION OF MBP CORE ARCHITECTURE FOR JUMP-1

KEN-ICHIRO ANJO† INOUE HIROAKI† JUN TANABE† MITSURU SATOH‡
HIDEHARU AMANO† KEI HIRAKI††

† Keio University ‡ Fujitsu Laboratories LTD †† The University of Tokyo
3-14-1 Hiyoshi, Kouhoku, Yokohama, Kanagawa, Japan 223
TEL:+81-45-560-1063, FAX:+81-45-560-1064
e-mail: mbp@aa.cs.keio.ac.jp

Abstract
A massively parallel processor JUMP-1 has been developed for building an efficient cache coherent-distributed shared memory on a large system with thousands of processors. In this paper, we evaluate *buffer-register architecture* adopted by the core processor in the Memory Based Processor(MBP)-light which manages the distributed shared memory on JUMP-1. It can manage the distributed shared memory with a smaller amount of hardware more effciently than the case using 32bit RISC processor as a core processor.

Keywords:
JUMP-1, MBP-light, buffer-register architecture

1 INTRODUCTION

Cache Coherent Non-Uniform Memory Access machine (CC-NUMA) is one of hopeful candidates for future common high performance machines. Unlike bus-connected multiprocessors, the system performance can be enhanced scalable to the number of processors. Moreover, parallel programs developed in small multiprocessors can be transported easily.

A number of CC-NUMA systems have been developed: Stanford DASH[1] / FLASH[2], MIT Alewife[3], SGI Origin2000[4] and the Sequent NUMA-Q[5] are representative. Although such systems work efficiently with tens or hundreds of processors, a large amount of memory and hardware are required to manage distributed shared memory (DSM) in case that thousands of processors are connected.

JUMP-1 is a prototype of a massively parallel processor with cache coherent-DSM developed by collaboration of 7 Japanese universities[6]. The major goal of this project is to establish techniques required to build an efficient DSM on a massively parallel processor. A lot of novel techniques are introduced in the DSM of JUMP-1 for this purpose. In order to satisfy both high degree of performance and flexibility, a dedicated processor called Memory Based Processor(MBP)-light is proposed to manage the DSM of JUMP-1. MBP-light consists of a simple processor core and hardwired controllers which manage memory systems, bus and network packets.

In this paper, a core processor of MBP-light is mainly described and evaluated. Although it is based on a simple 16-bit RISC processor, it can handle packets efficiently by using a special instruction set called *buffer-register architecture.* In Section 2, the structure of JUMP-1 is introduced. The architecture of MBP-light and its core processor are described in Section 3, and evaluation results are presented in Section 4.

2 A MASSIVELY PARALLEL PROCESSOR – JUMP-1 –

JUMP-1 consists of 256 clusters connected each other with an interconnection network called RDT (Recursive Diagonal Torus)[7]. RDT includes both torus and a kind of fat tree structure with recursively overlayed two-dimensional square diagonal tori structure. On the other hand, each cluster provides a high speed point to point I/O network[8] connected with disks and high-definition video devices.

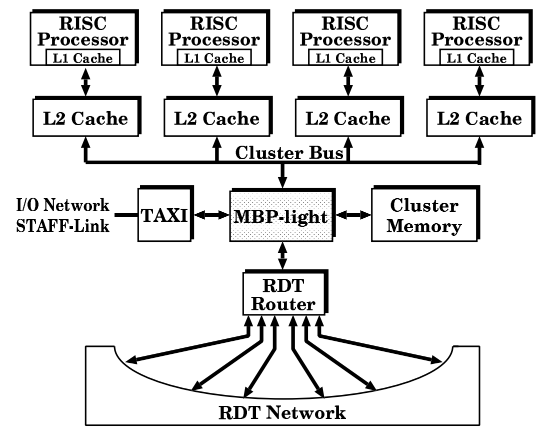

FIGURE 1: THE STRUCTURE OF JUMP-1 CLUSTER

Each cluster is a bus-connected multiprocessor, as shown in Figure 1, including four RISC processors (SuperSPARC+), MBP-light which is directly connected to a cluster memory, and RDT router chip for interconnection network.

MBP-light, the heart of JUMP-1, is a custom designed processor which manages DSM, synchronization, and packet handling.

In traditional CC-NUMAs – DASH / FLASH, Alewife and NUMA-Q –, the DSM is managed with a cache line size, and data in other clusters is copied into a cache attached to each processor. The consistency protocol is a simple invalidation protocol, the interconnection network is a simple mesh or ring, and the directory scheme is based on one-to-one data transfer.

Although such mechanisms work efficiently in those systems with a limited number of processors, it is not suitable for a system with thousands of processors. For example, a large amount of memory for cache and directory is required. The invalidation protocol based on one-to-one data transfer often causes a network congestion when many processors share the same data.

In order to address these problems, the following methods are used in JUMP-1.

1) Each processor (SuperSPARC+) shares a global virtual address space with two-stage TLB implementation, and the directory is attached not to every cache line but to every page, while the data transfer is performed by a cache line. Some parts of cluster memory are available as L3 (Level-3) cache which stores the copies of other cluster memory.
2) Various types of cache coherence protocols can be utilized, including not only invalidate type protocols but also update type protocols for applications which require frequent data exchange.
3) Reduced Hierarchical Bitmap Directory schemes (RHBDs) are introduced[9] for efficient directory management.
4) Each processor provides a custom sophisticated snoop cache as the L2 cache. Various cache protocols including cache injection are supported by this chip. A relaxed consistency model is implemented using write buffers in the L2 cache[10].

The detail scheme of distributed shared memory management for JUMP-1 is described in [11].

3 THE STRUCTURE OF MBP-LIGHT

3.1 THE DESIGN POLICIES

The following design policies are adopted.

1) Generation and collection of acknowledgement packets are managed with a dedicated hardwired logic. Another custom logic is provided for the tag checking in the cluster memory.
2) The protocol processes require a complicated large hardwired logics. Therefore, a simple 16bit core processor (MBP Core) is adopted to reduce the amount of total hardware. Although the core processor is simple, a packet can be processed quickly by introducing the *buffer-register architecture* which can treat packet buffers as special 68-bit registers.

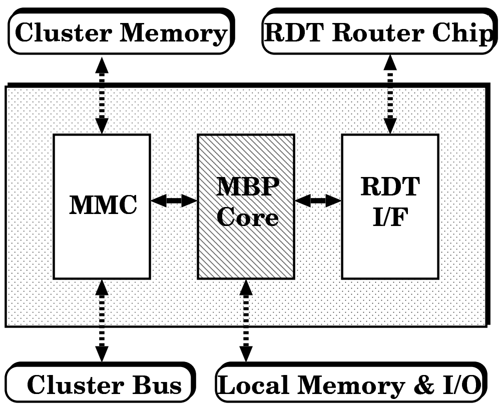

FIGURE 2: THE WHOLE STRUCTURE OF MBP-LIGHT

Figure 2 shows the structure of MBP-light depending on above policies. MBP-light consists of three modules: RDT Interface for treating network packets, MMC (Main Memory Controller) which controls the cluster memory and cluster bus, and MBP Core processor. RDT Interface and MMC provide their own hardwired controller, and work independently from MBP Core.

3.2 MBP CORE

Since jobs to be processed quickly are mostly managed by the hardwired logic in RDT Interface and MMC, MBP Core only processes a complicated part of the DSM protocol. It mainly decodes a received packet, accesses a table, transforms the accessed address, and generates a sending packet. A header of a packet in JUMP-1 is sometimes complicated and occupies several flits of the packet. Also, tags included in data flits of a packet are related with a protocol control. Therefore, it is convenient to treat all packet buffers as a register. Since a width of packet buffers is 68-bit, it requires an enormous hardware to treat such buffers as common general purpose registers in the processor.

To solve this problem, MBP Core provides 16 GPRs (General Purpose Registers) in 16bit width and 112 PBRs (Packet Buffer Registers) of 68bit width. The PBR is indicated by the content of the GPR, and accessed in the processor pipeline as a common register. While data transfer between PBRs as well as normal operations are allowed, the content of the PBRs is transferred directly as a packet from/to MMC or RDT Interface. As operations are mainly done between such packet buffers and registers, we call this structure the *buffer-register architecture*. As an example of this architecture, the add operation between GPR and PBR (*i.e.* `ADDPG R1 R2(0)`) is shown in Figure 3.

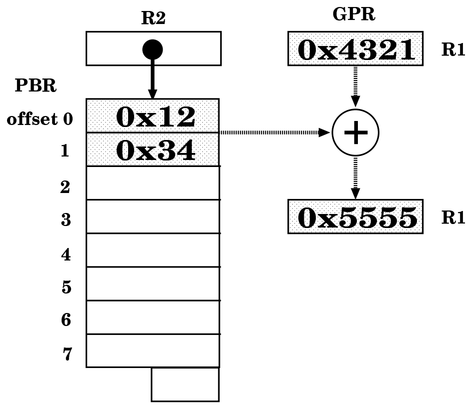

FIGURE 3: THE ADD OPERATION BETWEEN GPR AND PBR

MBP Core consists of a pipeline with four stages treating 21-bit instructions and 16-bit data. 21-bit × 64K local memory which stores instructions and local data is connected. MBP Core takes the I/O mapped approach, and another 64K address is provided for I/O devices including STAFF-Links and dedicated hardware for the barrier operation. The cluster memory and the tag memory are also accessed by using the PBR.

MBP Core also provides the 16-bit × 256 internal memory which is used for the table jump and storing bitmaps of acknowledge packets.

4 A PERFORMANCE EVALUATION

Here, the *buffer-register architecture* proposed as a core architecure is evaluated in detail. We evaluate MBP Core with a famous 32bit RISC processor called *DLX*.

The PBRs used in MBP Core are divided into two internal memory modules for RDT Interface and MMC respectively. We assume that the packet handling of RDT Interface or MMC are processed by hardware attached to those internal memory as well as MBP-light.

4.1 PROTOCOL PROCESSING TIME

Table 1 shows comparison results using the practical program which manages DSM.

TABLE 1: PROCESSING TIME FOR VARIOUS PRIMITIVES

Type	Home	MBP(μs)	DLX(μs)
Read Miss	Valid	5.9	7.0
Read Miss	Invalid	14	17
Invalidate	Valid	11	13

In order to compare the core structure including the instruction set, an access time to cluster memory and a loss of DLX version which may be caused by communication with MMC and RDT Interface are omitted. Here, we assumed that both processors work at 50 MHz.

MBP Core version works about 20 % faster than the DLX version. This shows advantages of MBP Core version overcomes the loss caused by short (16bit) data width.

4.2 INSTRUCTION MIX

In order to demonstrate the efficiency of MBP Core architecture, the instruction mix in 8 clusters is analyzed by some applications of SPLASH-2.

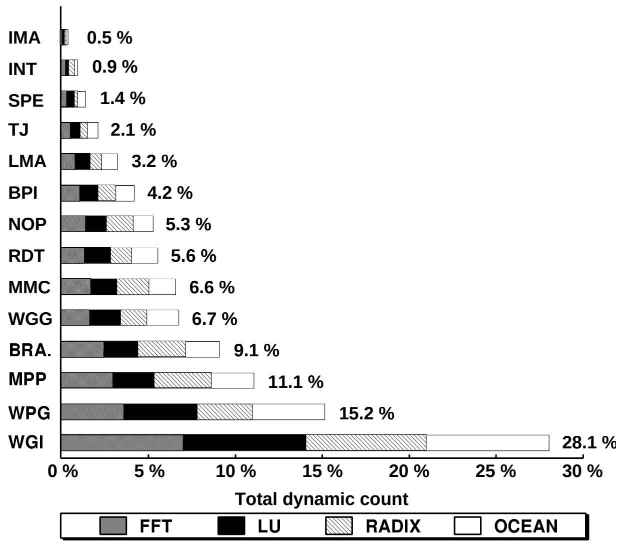

FIGURE 4: THE INSTRUCTION MIX

As shown in Figure 4, instructions related to buffer-register architecture are WPG, MPP and BPI. The total ratio of those classes is 29.7%. This result shows that instructions for the buffer-register architecture adopted by MBP Core are efficiently used.

4.3 HARDWARE AMOUNT

Here, the critical path and number of gates are evaluated as shown in Table 2.

TABLE 2: CRITICAL PATH AND GATE NUMBERS

Core Part	Critical Path (ns)	Gates
MBP Core	12.7	22,973
DLX	13.3	27,405

(Gates used for connections between MMC and RDT Interface are excluded in both designs.)

The critical path of MBP Core is shorter than that of DLX as only 16-bit operation is required. Reuired gates in MBP Core is 20 % smaller than those for DLX. From these evaluations, the *buffer-register architecture* of MBP Core achieves a high performance with a small number of gates compared with a common 32-bit RISC processor. The design of DLX version does not consider the interrupt and exception processing unit. If they are considered, the difference of gates requirement will be increased.

Finally, we show the layout picture of MBP-light in Figure 5.

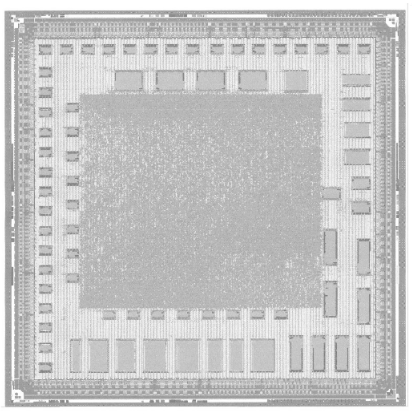

FIGURE 5: THE LAYOUT OF MBP-LIGHT

5 CONCLUSION

A dedicated processor called MBP-light for management of the DSM in JUMP-1 is introduced and its core processor architecture is evaluated. While the jobs which require quick operations are implemented in the hardwired logic, a simple 16-bit RISC processor called *buffer-register architecture* is adopted in MBP-light. Compared with the critical path, the number of gates and protocol processing time, MBP Core is advantageous compared with an educational 32-bit RISC processor (DLX).

Now, a cluster of JUMP-1 with MBP-light chip, four processors and L2 cache is available. The prototype of JUMP-1 with 64 processors is scheduled to be available within this year.

6 ACKNOWLEDGMENTS

The authors would like to express their sincere gratitude to the members of the Joint-University project for their valuable advises and discussions. The authors also express their thanks to Mentor Graphics Japan for supporting design tools.

A part of this research was supported by the Grant-in-Aid for Scientific Research on Priority Areas, #04235130, from the Ministry of Education, Science and Culture. This research is also supported by Parallel and Distributed system Consosium.

References

[1] D. Lenoski et. al., “The Stanford DASH Multiprocessor,” *IEEE Computer*, **25**, pp.63–79, 1992.

[2] J. Kuskin et. al., “The Stanford FLASH Multiprocessor,” *The 21st ISCA*, 1994, pp.302–313

[3] D. Chaiken and A. Agarwal, Software-Extended Coherent Shared Memory: Performance and Cost, *The 21st ISCA*, 1994, pp.314–324

[4] J. Laudon and D. Lenoski, The SGI Origin: A ccNUMA Highly Scalable Server *The 24th ISCA*, 1997

[5] T. Lovett and R. Clapp, STiNG: A CC-NUMA Computer System for the Commercial Marketplace, *The 23rd ISCA*, 1996, pp.308–317

[6] K. Hiraki et. al., Overview of the jump-1, an mpp prototype for general-purpose parallel computations, *Int. Symposium on Parallel Architectures, Algorithms and Networks*, 1994, pp.427–434

[7] Y.Yang et. al., Recursive Diagonal Torus: An interconnection network for massively parallel computers, *Int. Symposium on Parallel and Distributed Processing*, 1993, pp.591-594

[8] H. Nakajo et. al., An I/O Network Architecture of the Distributed Shared-Memory Massively Parallel Computer JUMP-1, *Int. Symposium on Supercomputer*, 1997

[9] T. Kudoh et. al., Hierarchical bit-map directory schemes on the RDT interconnection network for a massively parallel processor JUMP-1, *Int. Conference on Parallel Processing*, 1995, pp.I-186–I-193

[10] M. Goshima et. al., The Intelligent Cache Controller of a Massively Parallel Processor JUMP-1, *Innovative Architecture for Future Generation High-Performance Processors and Systems*, 1997, pp.116–124

[11] T. Matsumoto et. al., Distributed Shared Memory Architecture for JUMP-1: A General-Purpose MPP Prototype, *International Symposium on Parallel Architectures, Algorithms and Networks*, 1996, pp.131–137