MPUS: A Scalable Parallel Simulator for RedNeurons Parallel Computer

(Work-in-Progress)

Li Hui   Wu Junming   Chen Guoliang   Sui Xiufeng
Department of Computer Science & Technology,
Anhui Province Co-Key Laboratory of High Performance Computing and Application,
University of Science & Technology of China
lihui@mail.ustc.edu.cn   {jmwu, glchen}@ustc.edu.cn   sxf@mail.ustc.edu.cn

Abstract

In this paper, we present a scalable parallel simulator --- MPUS --- for verifying the design of our next generation high performance parallel computer --- RedNeurons(RN) parallel computer. The RedNeurons parallel computer is based on CMP technology, and it adopts an advanced but maybe some complicated architecture and topology. This paper mainly describes the design and implementation of the MPUS.

Categories and Subject Descriptors

C.5.1 [**Computer Systems Implementation**]: Large and Medium (``Mainframe") Computers – *Super Computers*

General Terms

Performance, Design, Experimentation, Verification

Keywords

Parallel Simulator, RedNeurons Parallel Computer, MPICH2

1. Introduction

It is extremely important that before the planned machine is built, we should build a simulator to verify the design and predict the performance of the planned machine.

To this end, we have built a simulator MPUS --- a simulator for RN parallel computer --- to verify our design of the parallel computer and even predict its performance. Our simulator is a scalable parallel simulator, and it has the ability to map multiple processors to one real processor.

2. RN Parallel Computer

This part mainly refers to [2].

The RedNeurons parallel computer is constructed upon the basic unit named MPU16 (i.e., MPU$_{4\times4}$). The RN system is a MIMD system that carries out message passing interface. The memory modules are distributed, and cling to processors. One kind of 2-core processors are used in our system. RN MPU16 will use some I/O ports to provide an interface for router, management network and I/O network.

3. MPUS

MPU$_{4\times4}$ is a nonswitch architecture, the close 4 neighbors communicate via a 2-D torus network. In other words, each process unit (PU) connects directly to 4 neighboring switch units (SU), and vice versa each switch unit connects directly to 4 neighboring process units. This architecture can effectively reduce the network radius to 2. With the help of such tight hardware coupling, we can exploit the high scalability of application programs.

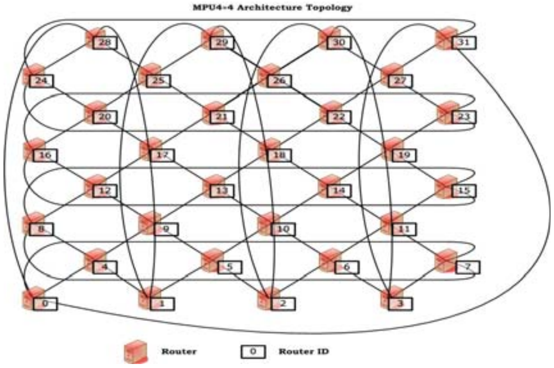

Figure 1 MPU4x4 architecture topology

3.1 MPUS Architecture

Figure 2 shows the architecture of the MPUS. Now we will describe each module in detail.

MPI Application: application programs that based on the standard MPI-2. We chose MICH2.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

INFOSCALE 2007, June 6-8, Suzhou, China
Copyright © 2007 ICST 978-1-59593-757-5
DOI 10.4108/infoscale.2007.919

MPICH2: our implementation surely results in the modification to MPICH2. But here is a principle: there should be as less modification to MPICH2 as possible, and the modification should be transparence to programmers.

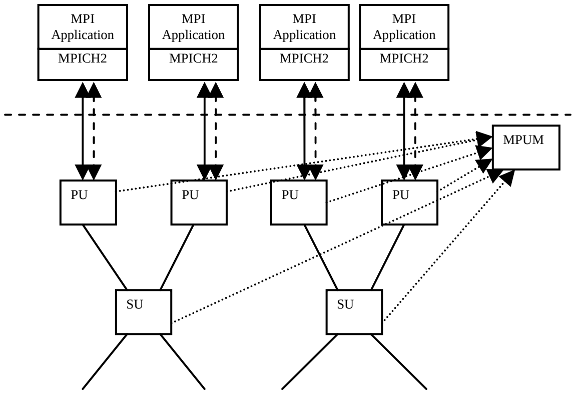

Figure 2 the architecture of MPU simulator

PU: PU is a process that simulates process unit. Its task mainly is to simulate the interprocess communication in our simulator architecture.

SU: SU is a process that simulates switch unit. Its task mainly is to implements the routing function of switch units. SU and PU consists our simulator's hardware layer.

MPUM: MPUM is a manager of PUs and SUs**.** For the number of PUs and SUs may be larger in future, and all simulating processes reside on different nodes, MPUM can help these processes to exchange messages and establish connections.

3.2 Routing

When a PU communicates to other PUs, for there are not any connections between PUs, there would be one SU or two SUs and one PU involved according to the source PU address and the destination PU address.

3.2 Simulating Environment

We choose a blade server with star topology as our simulation platform. All blade servers connect to a router. From the topology of the blade cluster, we know that the communication between any two blades needs one routing operation, and in fact at most one time of routing operation. Because of the high speed of the router, we can omit its effect. So we assume that all blades are connected directly.

4. Experiment Results

The benchmark we adopted in our experiment is NPB which has five core programs. We have tested the IS and EP [1] program on our simulator. As in [1], we use Mflop/s/processor as the metric of the performance and scalability of our simulator.

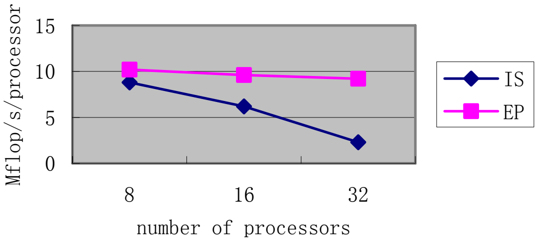

Figure 3 the experiment results

The IS program is very sensitive to the communication latency. Whereas the EP application is not sensitive to the number of processors (Figure 3). Considering the cost of running PU or SU on every node, we can conclude that our simulator, or in other words, our planned machine, can work correctly, and that our simulator has a good scalability.

5. Future Work

In future we will use our MPUS to construct and simulate an integrated RN computer. There will be a lot of work need to do.

In the current version of our simulator, we have not paid much attention to the network contention issue. So in future, we will get the network contention model involved.

6. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China under the Grant No.60533020, the National High Technology Research and Development Program of China (863 Program) under the Grant No.2005AA104031.

7. References

[1] Yuan Wei et al. Performance Analysis of NPB Benchmark on Domestic Tera-Scale Cluster Systems. Journal of Computer Research and Development, 2005, Vol.42, No6, pp.1079-1084.

[2] Alex Korobka et al. RedNeurons MPU Beta System Design. RedNeurons Ltd. Technology Document.