Proceedings of International Conference
on Computer Architecture

An SCSI-Based Adaptive Router in Torus Network

Kae-Fen Hwang, Jiunn-Wen Jou, Chia-Cheng Liu and Ted. C Yang*

Department of Information Engineering,
Feng Chia University, Taichung, 407, Taiwan,
R.O.C.
E-Mail : tcyang@fcu.edu.tw

Abstract

This paper presents the design of an adaptive router in a torus-connected computer network. The design can be divided into two parts: (1) The design of the hardware interface in the router. (2) Software support.

Every node uses an SCSI interface card to connect to the neighboring nodes to transmit data, and also needs some hardware to receive data from the neighboring nodes. We have finished the design of the hardware. The router only needs 4 FPGA chips, EPF 8636A of Flex 8000 series produced by Altera, plus some SRAMs and TTL devices. All of the components are inexpensive and readily available.

The adaptive router also needs software supports, such as the DMA driver, the SCSI driver and a routing algorithm. Because SCSI interface is used, the switching technique is packet-switching. In order to improve network performance, we design an adaptive routing algorithm to distribute the traffic load. Besides, we divide the bidirectional torus into 4 virtual networks, X+Y+ 、 X+Y- 、 X-Y+ and X-Y-, to decrease the complexity of the adaptive routing algorithm and increase the communication throughput. As a result, our routing algorithm is deadlock-free and minimally adaptive.

The router is experimentally evaluated using different traffic patterns and message lengths. Preliminary results show that a short packet length of 128 bytes is appropriate to the torus through our design, and the saturation point is about 0.55 for random routing and 0.35 for transpose routing.

1. Introduction

A recent trend in supercomputer design has been moving towards scalable parallel computers, which offer proportional gains in performance as the number of processors is increased. Many such systems, e.g. MIT J-Machine [Dall90], Connection Machine CM-5 [Thin93], and Cray T3D [Kess93], are characterized by the distribution of memory among an ensemble of processing nodes. Each node has its own processor, local memory, and other supporting I/O devices.

Communication has been a major issue in parallel-processor systems. Internode communication in MPP systems is carried out by passing messages through some static connection network. To increase the bandwidth and reduce the message latency, many parallel machines use dedicated hardware router. The routing algorithms, the flow control and the switching techniques are important factors that affect the network performance. The switching techniques can be classified into three types: circuit switching[Kerm79], packet switching, and wormhole switching [Dall87]. Beside the switching techniques, routing algorithm is also of critical importance. A good routing algorithm can determine the routing path in short time, avoid the deadlock and distribute evenly the network traffic. A large number of routing algorithms have been proposed, for example, [Chie95], [Dall93], [Glas91], [Lind91], and [Smar91].

We, at Feng Chia University, proposed a massively parallel (MPP) system and call it MPP/FCU. The system is designed with scalability in mind and its configuration is shown in Figure 1. Still at the early stage, the MPP/FCU system is an experimental platform with only 16 nodes. The M-NET in Figure 1 indicates the message network and uses torus as its network topology.

In this paper, a router is presented for the construction of the M-NET in MPP/FCU. Our goal is to use an adaptive routing algorithm to avoid contention and improve network performance.

In section 2, we will discuss the design considerations of our adaptive router as well as the techniques to incorporate the SCSI into the router to reach the goal. In section 3, the design of the adaptive router, the system interface and the functions of the adaptive router are presented

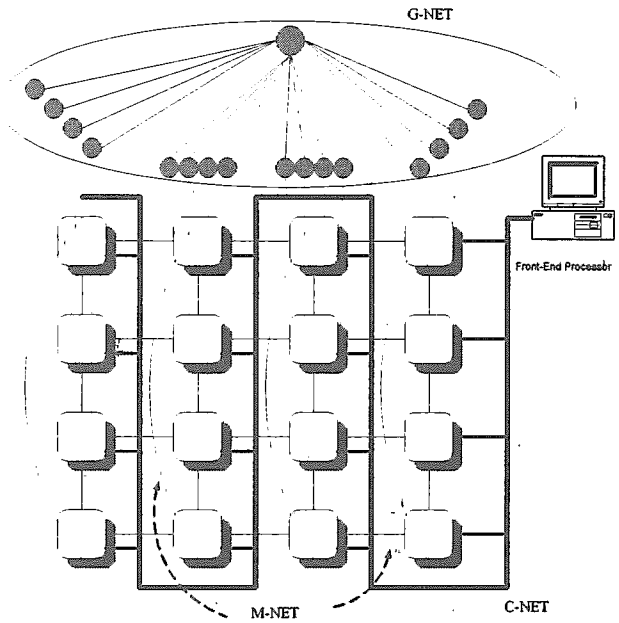

Figure 1. The architecture of MPP/FCU

in detail. Besides hardware, routing algorithm and software supports are also described. In section 4, analysis of the performance is presented. Finally, a brief conclusion is given in section 5.

2. Adaptive Router Design Considerations

While designing the network topology of the MPP/FCU, our first option was common bus (see Figure 2). For common bus, the diameter is very low, i.e. at most 2. It only requires an SCSI interface card in a node to construct the network, therefore the cost is low. However, there are drawbacks. Firstly, the nodes would compete with one another for using the common bus to transmit data and the communication performance would be degraded. Secondly, the scalability of MPP systems is low due to the limits on SCSI ID. Additional SCSI interface cards in some nodes are needed to extend the number of nodes in the row or column bus. This will cause hot-spot in intermediate nodes and increase the diameter. Thirdly, the SCSI ID of each node uniquely defines the node priority for using the common bus and this could cause starvation. Based on all these reasons, the common bus is thus disregarded.

The network architecture in MPP/FCU is now presented. The architecture is depicted in Figure 3. Nodes on the network architecture play both roles of an initiator and a target. The initiator that is supported by an SCSI interface card uses its unique SCSI bus to transmit data to one of the 4 neighbor nodes. The target is supported by additional receiving hardware and its function is to receive data from some initiator. The cost of the hardware router is inexpensive. Because there is no priority difference between nodes in SCSI bus, the network does not cause starvation.

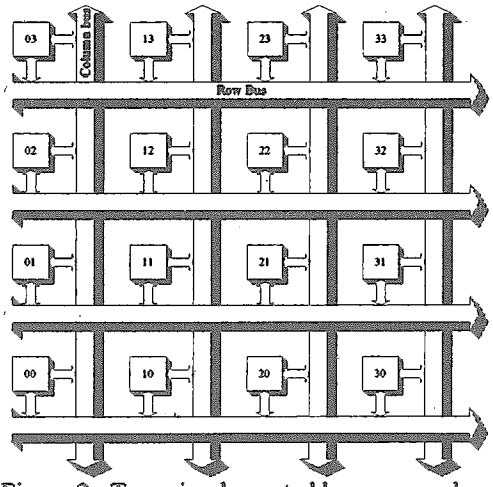

Figure 2. Torus implemented by common bus

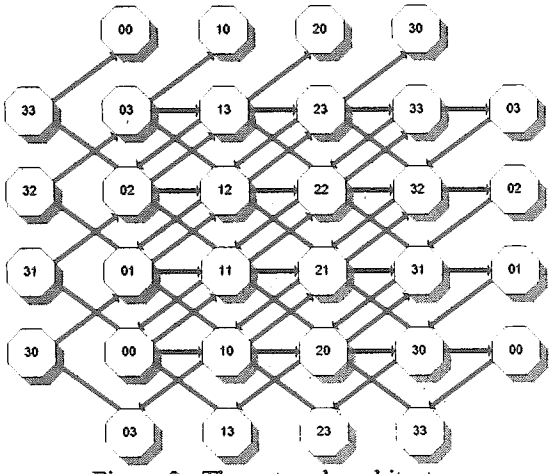

Figure 3. The network architecture

The additional receiving hardware includes the SCSI controller, which is responsible for executing the SCSI commands, processing the protocols on SCSI bus and receiving data from SCSI bus. In order to reduce the development time and allow easy modification of the receiving circuit, we use an FPGA device to design the controlling circuit.

3. Router Hardware Design and Software Supports

The design of the router can be divided into two parts: hardware interface and software supports. The hardware implementation contains the design of the SCSI controller, the interrupt unit and the PC-board-design of the entire router. The software supports include the DMAC (Direct Memory Access Controller) driver, the routing algorithm and the SCSI driver.

3.1. Hardware Interface Design

Functional blocks of the router are depicted in Figure 4. The major functions of the blocks in the router are described in the following paragraphs.

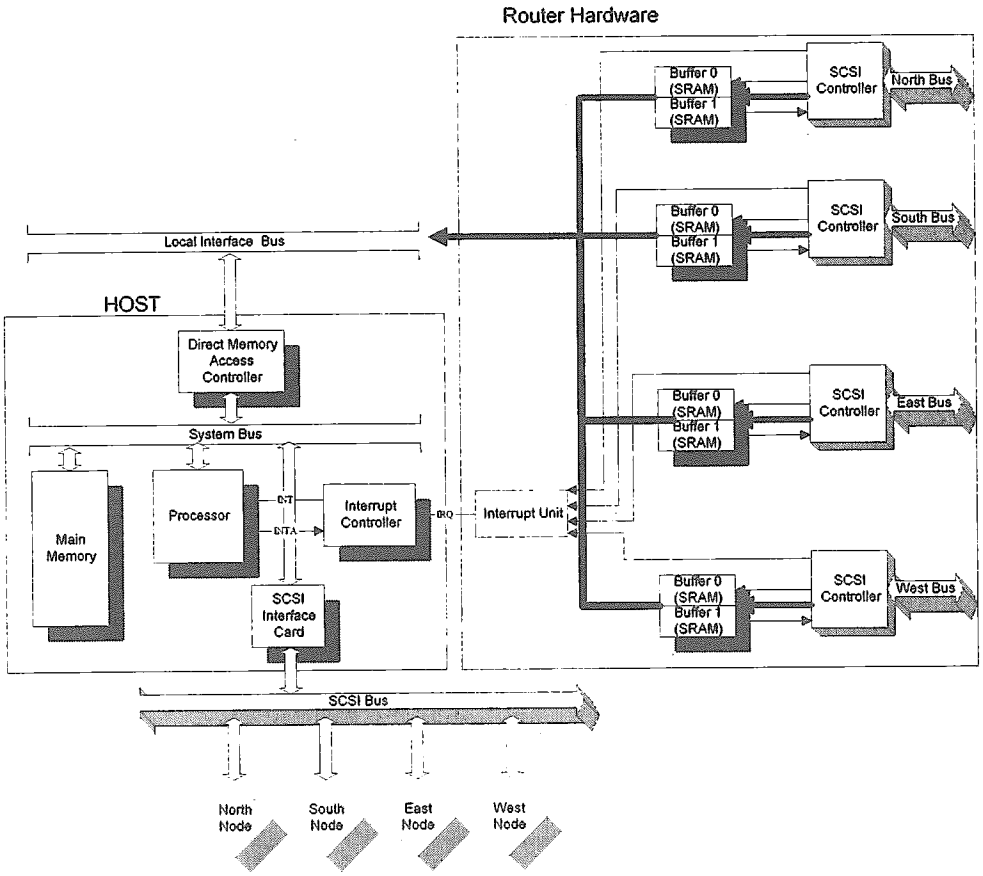

Figure 4. Block diagram of the adaptive router

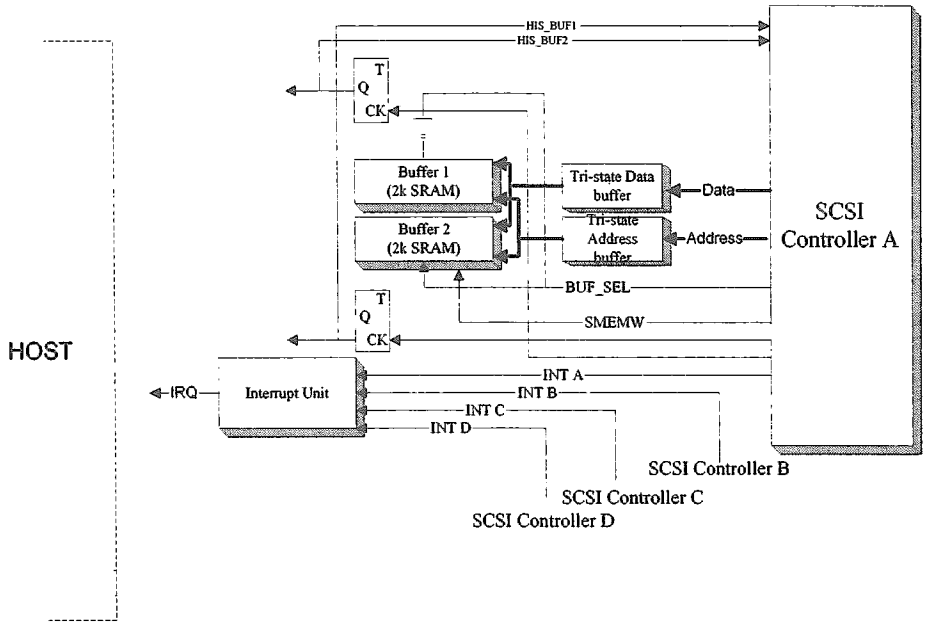

Figure 5. Receiving hardware in a router

1. SCSI controllers: There are 4 SCSI controllers. Every controller is responsible for processing the communication protocol in the SCSI bus, executing SCSI commands and receiving data through the SCSI bus from some neighboring node.
2. Buffer: The data received by the SCSI controller is stored in the buffer and waits for further processing.
3. Interrupt unit: The interrupt unit accepts the interrupt signals from the 4 SCSI controllers and coordinates the interrupt signals to notify the processor that there are some received data in the buffer waiting to be processed.
4. Direct Memory Access Controller: The DMAC reads the messages out of the buffer and stores the messages into the memory associated with the host.
5. Host: The host executes the routing algorithm to determine the direction to route the message. Besides this, the host also executes the SCSI driver to control the SCSI interface card and transmit the message to the next node through the SCSI interface card.
6. SCSI interface card: The SCSI interface card processes the interface protocol of the SCSI bus with the SCSI controller in the next node. The node uses the SCSI interface card to transmit the routing message to the next node through the SCSI bus.

The components of the receiving hardware in the router are shown in Figure 5. Each SCSI controller has two buffers for the received data. The two buffers are independent to each other. The SCSI controller and the host can take turns

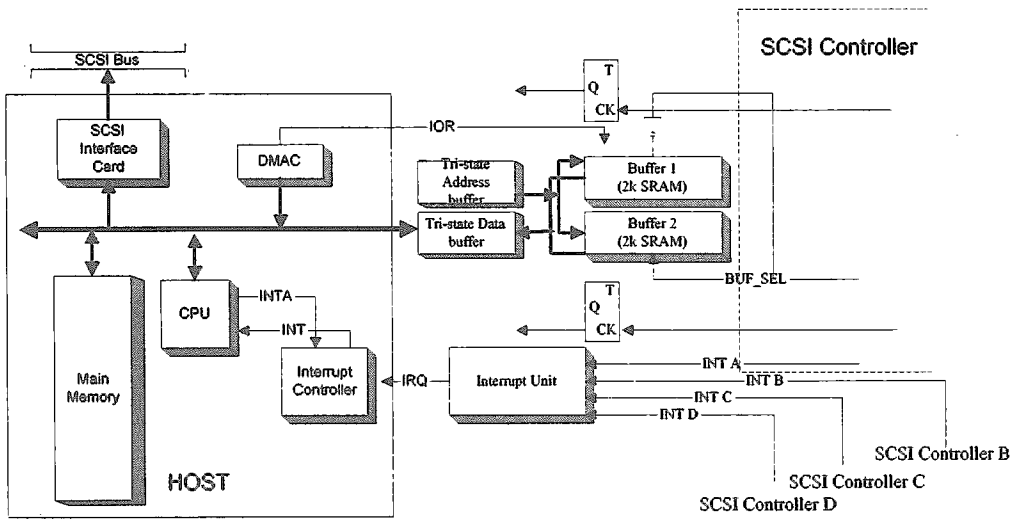

Figure 6. Transmitting hardware in a node

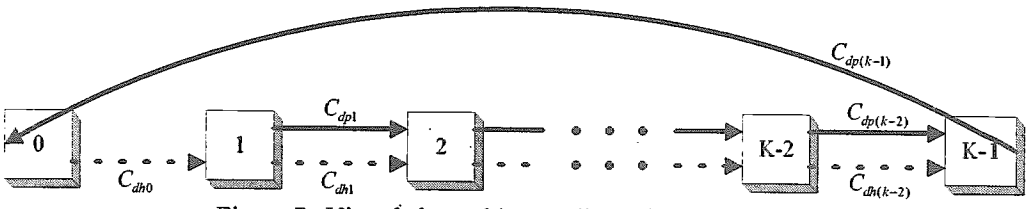

Figure 7. Virtual channel in one dimension of a Torus

at accessing data in the two buffers to increase the network transmission throughput. The type of data accessing in the buffer is memory-mapped I/O. The packet length is from 128 bytes to 1k bytes and the capacity of either buffer is 2k bytes.

There is a T flip-flop associated with each buffer. This data-record-bit indicates whether there is a packet in the buffer. When the SCSI controller receives a SCSI command to receive data, it checks the data-record-bit belonging to the SCSI controller. If both of the record-data-bits are set, then there are packets in both buffers that have not been processed by the processor yet. If there is no room to store the data that the SCSI controller is supposed to receive, the SCSI controller will return a "Busy" message to the previous node. If the data-record-bits are not set at all, the SCSI controller chooses one of the two buffers to store the packet to be received. When the SCSI controller completes receiving the data, it sets the data-record-bit associated with the buffer, and issues an interrupt signal to request the processor to process the packet in the buffer.

When an interrupt occurs, the processor executes the interrupt program. The first action is to check the data-record-bits in router. The interrupt program will find out which buffer stores the packet to be processed. The first 4 bytes in the buffer contains the packet length and the routing information. The interrupt program reads the information and executes the routing algorithm to determine the direction to route the packet. Then, the processor executes the SCSI driver controlling the SCSI interface card to communicate with the SCSI controller in the next node. If there is any buffer left in the next node to store the packet, the SCSI driver reads the packet from the source buffer and uses the SCSI interface card to transmit the packet to the next node through the SCSI bus. Otherwise, the program exercises the DMAC to read the packet from the buffer through DMA, stores the packet into the local memory space in the host and waits for next transmission. The transmission block diagram in a node is shown in Figure 6.

3.2. Software Supports

Software supports include the SCSI driver, the DMAC driver, and the adaptive routing algorithm. Because of the scope of this paper, we shall only describe the most important part, the adaptive routing algorithm. [Hwan96] gives detailed description on the software drivers.

The basic idea of the adaptive routing algorithm is from [Dall87], with modifications. The routing algorithm is called Unidirectional Torus Routing (UTR) which uses channels that transmit data in unidirection. There are two parallel sets of virtual channels, call p-channels and h-channels. The p-channels are used by messages that will eventually use the wraparound channel in the current dimension. After using the wraparound channel, such messages use the h-channel for the remaining journey in the current dimension. The h-channels are used by messages that will not use the wraparound channel in that dimension. Figure 7 illustrates the UTR in a single dimension. In the following discussion, we use the notation of [Dall87] where $C_{d\alpha x}$ represents the node x in d dimension, and the virtual channel set $\alpha \in \{p, h\}$.

There are two virtual channels in every physical channel that are used to avoid any possible cycles in channel dependency graph. The UTR routes messages along the paths under constraints of unidirectional links and is therefore deadlock-free.

In our design, we divide the 2-dimensional torus with bidirectional links into 4 virtual networks, X^+Y^+ 、 X^+Y^- 、 X^-Y^+ and X^-Y^- [Chen96]. The original UTR is modified so that it works for each virtual network and still uses p-channels and h-channels to avoid deadlocks in all virtual networks. Let us define the function $\Re_{UTR}: N \times N \rightarrow C$, which maps a (current node, destination node) pair. In order to define $\Re_{UTR}:(Curt, Dest)$, assume that *Curt* and *Dest* are different nodes in the d dimension, and $\Delta = \sigma_d(Curt) - \sigma_d(Dest)$. When $d = X^+$ or Y^+, then,

$$\Re_{UTR}:(Curt, Dest) = \begin{cases} C_{dpCurt} & \text{if } \Delta < 0 \\ C_{dhCurt} & \text{if } \Delta > 0 \end{cases}$$

When $d = X^-$ or Y^-, then

$$\Re_{UTR}:(Curt, Dest) = \begin{cases} C_{dpCurt} & \text{if } \Delta > 0 \\ C_{dhCurt} & \text{if } \Delta < 0 \end{cases}$$

For each physical channel, there are two independent virtual channels in every virtual network. The packets are routed by using the virtual channels belonging to the virtual network. The packets in different virtual network do not affect one another. From [Dall87], there exist no cycles in the channel dependency graph for every virtual network. Accordingly, the routing algorithm in our design can also be readily proven to be deadlock-free. Before the transmission, the direction of the packet and the virtual network to be used are determined from the addresses of source node and destination node. The packet is routed along any shortest path to the destination. Therefore, the algorithm in our design ,in addition to be deadlock-free, is a minimally adaptive routing algorithm.

Next, we define the format of the packet header following the adaptive routing algorithm. It is shown in Figure 8. The 4 fields are current dimension (dim), previously used virtual channel or the currently used virtual channel (V_x and V_y), virtual network to be used (X, Y), and the distance from the current node to the destination node (X-distance and Y-distance). The functions of those fields are presented in Tables 1, 2, and 3.

Table 1. States and virtual channels along the X dimension

dim	V_x	Operations
0	0	The packet is currently using h-channel to be routed along X-dimension
0	1	The packet is currently using p-channel to be routed along X-dimension
1	0	The packet previously used h-channel to be routed along X-dimension
1	1	The packet previously used p-channel to be routed along X-dimension

Table 2. States and virtual channels along the Y dimension

dim	V_y	Operations
0	0	The packet previously used h-channel to be routed along Y-dimension
0	1	The packet previously used p-channel to be routed along Y-dimension
1	0	The packet is currently using h-channel to be routed along Y-dimension
1	1	The packet is currently using p-channel to be routed along Y-dimension

Table 3. Virtual network designations

X	Y	Virtual network
0	0	X^-Y^-
0	1	X^-Y^+
1	0	X^+Y^-
1	1	X^+Y^+

The hardware/software design of the router have been described. The hardware implementation contains the transmitting hardware and the receiving hardware. In transmitting hardware, we only need an SCSI interface card or an SCSI interface on PC board. The receiving hardware only needs 4 SCSI controllers, SRAMs, and some TTL devices. All the components are inexpensive and readily available. These conform to our considerations. The implementation of the router is quick, expandable, and reliable.

We integrate the SCSI interface into the adaptive router, because SCSI is universally accepted. The adaptive routing algorithm needs software supports and scatters traffic load on the network. The bidirectional torus is divided into 4 virtual networks, X^+Y^+ 、 X^+Y^- 、 X^-Y^+ and X^-Y^-. As a result, communication throughput and system throughput are enhanced.

4. Performance Evaluation

A simulator written in C for the router is used for design simulation and performance evaluation. The network performance is affected by two factors: destination node address and network load. In the simulation, the destination

dim	V_x	V_y	X	Y	X-distance	Y-distance
15	14	13	12	11	10	1

dim : 0 indicates X dimension
dim : 1 indicates Y dimension

Figure 8. Format of the packet header

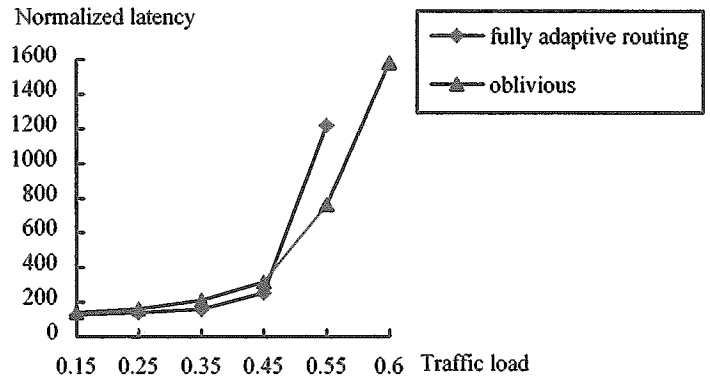

Figure 9. Simulation results of latency versus traffic load for random routing

node address is generated by using a traffic pattern that is called uniform traffic pattern [Ni94] and the average interval to inject packets is computed by Poisson process.

The following quantities are measured:

1. Network latency: The time a packet spends in the network, from the time the packet enters the injection buffer until the packet enters the delivery buffer at the destination.
2. Saturation: The smallest load at which more packets are created than delivered.

The performance of the router in our design are measured for different communication patterns, as based in [Pifa94][Fulg93]:

1. Random Routing: Messages have random destinations. Several distributions are possible, from which two are chosen. The destinations of the packet are uniformly distributed over the set of nodes.
2. Transpose: Transpose is permutation where node (x,y) sends packets to destination (y,x).

Generally speaking, two sets of experiments have been carried out. One shows the relation between packet size and path lengths and the other shows latency as a function of traffic load. The first experiment involves three different message sizes, namely packets of 128 bytes, 1024 bytes and randomly generated size between 128 and 1024 bytes. We will use uniform random routing patterns to simulate on three different two-dimensional torus with 4 x 4, 8 x 8, and 16 x 16 nodes. The experiment reveals the relation between the packet size and the average path length. Tables 4, 5, and 6 show the simulation results with average latency per byte. From the experimental results, shorter packet lengths are better for the torus which is constructed through the router of our design. Besides, the average latency in cycles per byte with three different packet size in 4 x 4 torus is nearly the same. But the results in 8 x 8 torus and 16 x 16 torus is different. The main reason is that the average path length in either 8 x 8 or 16 x 16 torus is longer than the average path length in 4 x 4 Torus. Therefore, the chances that a packet waits for the SCSI bus to transmit are higher. The above results confirm

Table 4. Simulation results on 4 × 4 torus with random routing

Packet Length	Average latency in cycles per byte
128 bytes	13.17
Random generation between 128~1024 bytes	14.24
1024 bytes	14.53

Table 5. Simulation results on 8 × 8 torus with random routing

Packet length	Average latency in cycles per byte
128 bytes	14.39
Random generation between 128~1024 bytes	16.14
1024 bytes	19.50

Table 6. Simulation results on 16 × 16 torus with random routing

Packet length	Average latency in cycles per byte
128 bytes	16.76
Random generation between 128~1024 bytes	17.07
1024 bytes	27.18

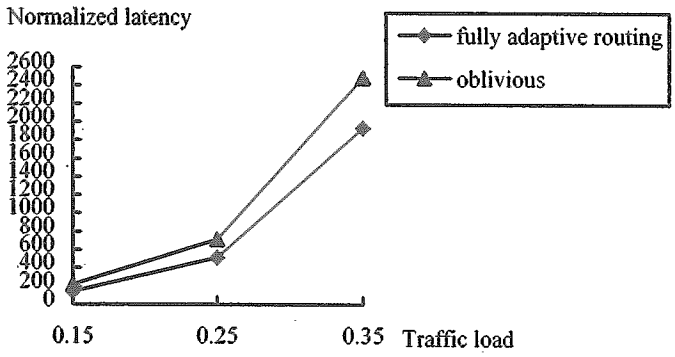

Figure 10. Simulation results of latency versus traffic load for transpose routing

that the packet length and the path length, as expected, are indeed two major factors that affect the network performance.

The second experiment involves packets of 128 bytes on two-dimensional torus with 8×8 nodes and two different traffic patterns: uniform random traffic and transpose traffic. Figures 9 and 10 show the normalized latency as a function of traffic load, for different traffic patterns.

The traffic load where the system saturates is an important measurement since after saturation there is no way to predict the delivery time of messages. We observed that the saturation point is about 0.55 for random routing and 0.35 for transpose routing in our router. As a comparison between the two types of traffic patterns, the simulation results indicate that the uniform random traffic pattern has a higher traffic load and that the transpose traffic pattern induces a higher latency. The reason for this is : the transpose on the torus is a reflection of the source about the line $y = -x$, given a coordinate system through the center of the network. This patterns causes continuous hot spots along the diagonal of the network.

Yet in another experiment, we compare the network performance of the adaptive routing algorithm in our design and a previously published routing algorithm [Dall87]. For the same network architecture, our adaptive routing algorithm is replaced by their algorithm. Table 7 shows the network performance at different traffic load for random routing. It can be observed that the network performance of our adaptive routing algorithm is better than theirs under saturation. Furthermore, since the hardware cost in our design is very low, a reasonable gain in benefit-cost ratio can be expected. The circuit area fits into the FPGA chip, EPF 8636A of Flex 8000 series, produced by Altera. The gate counts and the percentages of the units in each SCSI controller are shown in Table 8.

5. Conclusions

Table 7. Network performance comparison for our routing algorithm and [Dall87].

Normalized latency		
Traffic load	our routing algorithm	[Dall87]
0.15	127.42	140.37
0.25	145.69	162.53
0.35	158.67	211.96
0.45	254.31	318.24

Table 8. Gate counts in SCSI controller

Unit	Gate Counts	Percentages
SCSI interface control unit	505	20.05 %
Phase state control unit	554	22.04 %
SCSI command decoder	461	18.3 %
MEM control unit and Interrupt logic	807	32.04 %
Block counter	113	4.48 %
Registers	78	3.09 %
Total	2518	100 %

The adaptive router in our design integrates the SCSI interface as the communication interface among nodes, or PCs. The design highlights of the adaptive router are :(1) low cost, (2) easy implementation, (3) the MPP/FCU systems can be constructed in a short time through our router, (4) high flexibility in network architecture, and(5) good connectivity to other I/O devices through the SCSI interface of the adaptive router.

In our design, the switching technique is packet-switching. In order to improve the network performance, an adaptive routing algorithm is applied to scatter the traffic load on the network. Simulation results indicate that short packet length ,e.g. 128 bytes, is appropriate to the torus network through the router of our design. The saturation point is about 0.55 for random routing and 0.35 for transpose routing. From the simulation results, reasonable benefit-cost gain in our design is achieved.

In the future, an increase on the width of the SCSI bus in the adaptive router and a change on the transmission protocols of the adaptive router from "nonsynchronous" to "synchronous" to improve the network performance may be considered. We will also make attempts on separating the adaptive router from the nodes in MPP/FCU, and adding a microcontroller into the hardware of the adaptive router for ,among many possibilities, the execution of the routing algorithm to improve network performance.

References

[Altera94] MAX+PLUS II Getting Started. Altera Corporation, 1994

[Chie95] A. A. Chien and J. H. Kim, "Planar Adaptive Routing : Low-Cost Adaptive Networks for Multicomputer," *Jou. of the ACM*, pp. 91-123, Jan. 1995.

[Chen96] C. C. Chen, Wormhole Router in a Torus-Connected MPP Systems. Master's Thesis, Graduate Institute of Information Engineering, Feng Chia University, Jun. 1996.

[Dall87] W. J. Dally and C. L. Sietz, "Deadlock Free Message Routing in Multiprocessor Internection Networks," *IEEE Trans. on Computers*, Vol 36, No. 5, pp. 547-553, May 1987

[Dall90] W. J. Dally, The J-Machine: System Support for Actors. Technical report, AI Laboratory and Laboratory for Computer Science, MIT, 1990.

[Dall93] W. J. Dally and H. Aoki, "Deadlock-free Adaptive Routing in Multicomputer Networks Using Virtual Channel," *IEEE Trans. on Parallel and Distributed Systems*, Vol 4, No 4, pp. 466-475, Apr.1993

[Fulg93] M. L. Fulgham, "Performance of Chaos and Oblivious Routers Under Non-Uniform Traffic", Tech. Rep, University of Washington. Jul. 1993.

[Glas91] C. J. Glass and L. M. Ni, "Turn Model for Adaptive routing," Tech. Rep. Department of Computer Science, Michigan State University, Oct. 1991.

[Hwan96] K. V. Hwang, On the Design of an SCSI-Based Adaptive Router. Master's Thesis, Graduate Institute of Information Engineering, Feng Chia University, Jun. 1996.

[Kerm79] P. Kermani and L. Kleinrock, "Virtual Cut-through : A New Computer Communication Switch Technique," *Computer Network*, Vol. 3, pp. 267-286, 1979

[Kess93] R. E. Kessler and J. L. Schwarzmeier, "CRAY T3D: A new dimension for CRAY research," *Compcon*, pp. 176-182, Spring 1993.

[Lind91] D. H. Linder and J. C. Harden, "An Adaptive and Fault Tolerant Whormhole Routing Strategy For k-ary n cubes," *IEEE Trans. on Computer*, Vol. 40, pp.2-12, Jan. 1991.

[Ni94] L. M. Ni, Y. Gui and S. Moore, " Performance Evaluation of Switch-Based Wormhole Networks," Department of Computer Science, Michigan State University, July 1994.

[Pifa94] G. D. Pifarre', L. Gravano, S. A. Felperin, and J. L. C. sanz, "Fully Adaptive Minimal Deadlock-Free Packet Routing in Hypercubes, Meshs, and Other Networks : Algorithms and Simulations," *IEEE Trans. on Parallel and Distributed Systems*, Vol. 5, No. 3, Mar. 1994.

[Ross90] S. M. Ross, A Course in Simulation, Macmillan Publishing Company, 1990.

[Smar91] S. Konstantinidou and L. Snyder, "The Chaos Router," *IEEE Trans. on Computer*, Vol. 43, No. 12, pp.2-12, Dec. 1991.

[Thin93] Thinking Machines Corporation, "Connection Machine CM-5 Technical Summary," Nov. 1993