Recursive Diagonal Torus: an interconnection network for massively parallel computers

Yulu Yang, Akira Funahashi, Akiya Jouraku, Hiroaki Nishi, Hideharu Amano, Member, IEEE, Toshinori Sueyoshi, Member, IEEE

Abstract— **Recursive Diagonal Torus (RDT), a class of interconnection network is proposed for massively parallel computers with up to 2^{16} nodes. By making the best use of recursively structured diagonal mesh (torus) connection, the RDT has a smaller diameter (eg., it is 11 for 2^{16} nodes) with smaller number of links per node (i.e., 8 links per node) than those of the hypercube. A simple routing algorithm called vector routing, which is near-optimal and easy to implement is also proposed. Although the congestion on upper rank tori sometimes degrades the performance under the random traffic, the RDT provides much better performance than that of 2-D/3-D torus in most cases, and under hot spot traffic, the RDT provides much better performance than that of 2-D/3-D/4-D torus.**

The RDT router chip which provides message multicast for maintaining cache consistency is available. Using the $0.5\mu m$ BiCMOS SOG technology, versatile functions including hierarchical multicasting, combining acknowledge packets, shooting down/restart mechanism, and time-out/set-up mechanisms work at 60MHz clock rate.

Keywords— **Interconnection Network, Massively Parallel Computer, routing algorithm, router chip, mesh network, torus network, message multicast.**

I. Introduction

THE communication network is one of the critical components of a highly parallel multicomputer. Recently, multicomputers providing more than a thousand computation nodes are commercially available, and efforts have been exerted to implement Massively Parallel Computers (MPCs) with tens of thousands nodes.

In these systems, the connection topology often dominates the system performance. Instead of hypercube used in first-generation multicomputers, most recent machines take the 2D or 3D mesh (torus) network[1][2][3]. Although the diameter of a mesh network is large ($O(\sqrt{M})$ or $O(\sqrt[3]{M})$ for M nodes), it only requires four or six links per node unlike the hypercube which requires $log_2 M$ links per node. Architectural supports for fine grain processing [4] and the wormhole routing [5] reduce the performance degradation caused by a large diameter. Moreover, mesh networks are suitable for wide area of scientific calculations including flow dynamics, QCD, and structural analysis.

Y. Yang is with the Department of Computer Science, Nankai University, Tianjin, China. E-mail: parallel@public.tpt.tj.cn .

A. Funahashi is with the Department of Information Engineering, Mie University, Mie, Japan. E-mail: funa@info.mie-u.ac.jp .

A. Jouraku and H. Amamo are with the Department of Computer Science, Keio University, Yokohama, Japan. E-mail: {jouraku, hunga}@am.ics.keio.ac.jp .

H. Nishi is with the Real World Computing Partnership, Tsukuba, Japan. E-mail: west@rwcp.or.jp .

T. Sueyoshi is with the Department of Computer Science, Kumamoto University, Kumamoto, Japan. E-mail: sueyoshi@cs.kumamoto-u.ac.jp .

However, in an MPC with more than ten thousands nodes, the large diameter of the mesh network is intolerable. A lot of connection topologies (De Bruijn[6], fat tree[7], Star graph[8] and others) have been proposed for such MPCs. Although these networks support a small diameter with a small degree, embedding the mesh network is difficult. On the current machines with mesh structure, parallel computation algorithms and message handling algorithms have been accumulated and refined. To make the best use of them, a network including the mesh structure is advantageous even for future MPCs.

In this paper, we propose a novel class of networks called Recursive Diagonal Torus (RDT)[9], which consists of recursively structured mesh (torus) connection. It was originally designed as a network of a massively parallel processor[10] for efficient hierarchical multicasting. However, it provides a lot of interesting properties as a general purpose network. In Section 2, the structure of RDT and assignment of torus are defined. In Section 3, a simple routing algorithm called “vector routing[11]” is defined. In Section 4, performance evaluation under the random traffic is reported, and other properties of this network are discussed in comparison with other networks for MPCs. In Section 5, hierarchical multicasting is introduced, and implementation of a router chip for the RDT are described.

II. Interconnection Network: RDT

Recursive Diagonal Torus (RDT) is a novel class of networks which consists of recursively structured mesh (torus) connections of tori with different sizes in the diagonal directions[9].

A. Definitions of the RDT

First, a two-dimensional square torus is defined as the basis of RDT.

Definition 1: : **Base torus**
The base torus is a two-dimensional square array of nodes each of which is numbered with a two-dimensional number as follows:

$$\begin{array}{ccccc}
(0,0) & (1,0) & (2,0) & \cdots & (N-1,0) \\
(0,1) & (1,1) & (2,1) & \cdots & (N-1,1) \\
(0,2) & (1,2) & (2,2) & \cdots & (N-1,2) \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
(0,N-1) & (1,N-1) & (2,N-1) & \cdots & (N-1,N-1)
\end{array}$$

where $N = n^k$. The n and k are natural numbers. The torus network is formed with four links between node (x, y) and neighboring four nodes:

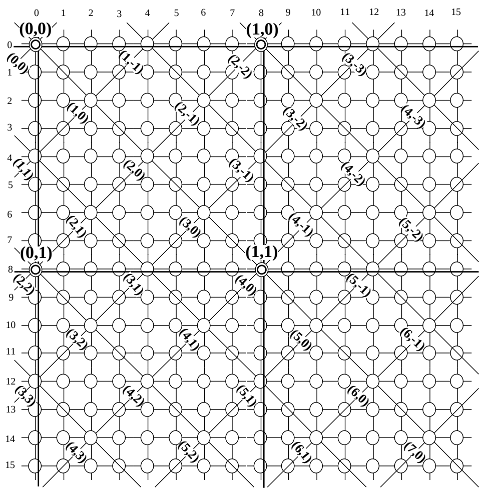


Fig. 1. Upper rank tori

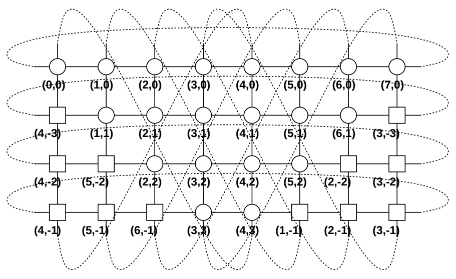


Fig. 2. Structure of the rank-1 torus

$$(\mathrm{mod}(x \pm 1, N),\ y)\ \mathit{and}\ (x,\ \mathrm{mod}(y \pm 1, N))$$

This base torus is also called the rank-0 torus. □

In order to reduce the diameter, the best way for the torus network is to provide bypass links for the diagonal direction. Assume that four links are added between a node (x, y) and nodes $(x \pm n, y \pm n)$. Then, the additional links form a new torus-like network. The direction of the new torus-like network is at an angle of 45 degrees to the original torus, and the grid size is $\sqrt{2}n$ times of the original torus. Here, this torus-like network is called the rank-1 torus. On the rank-1 torus, another torus-like network (rank-2 torus) can be made by providing four links in the same manner. Thus the rank-(r+1) torus can be formatted on the rank-r torus in the same way. Figure 1 shows rank-1 and rank-2 tori when n is set to be 2. Our new network called the RDT consists of such recursively formed tori.

Each odd rank torus-like network is a 2:1 rectangle one which provides spiral circular links as shown in Figure 2, while each even rank network is a common square nearest neighbor torus. Here, both types of networks are called the "torus" or "tori".

Definition 2: : **Upper rank torus**
Assume that the rank-r torus satisfies the following condition:

$$N_{x(r+1)} \times N_{y(r+1)} \geq 2$$

where

$$N_{x(r+1)} = \frac{N_{yr}}{\mathrm{gcd}(N_{xr}, n)}$$

$$N_{y(r+1)} = \frac{N_{xr}}{2\mathrm{gcd}(N_{yr}, n)}$$

and, $N_{xr}, N_{yr}, N_{x(r+1)}$ and $N_{y(r+1)}$ are the sizes of the rank-r and rank-(r+1) in x and y axes.

Links between node (x, y) and nodes (x', y'), (x'', y'') on the rank-r torus form the rank-(r+1) torus:
Case 1: r is an even number, and thus, an odd upper rank tori on the even rank tours is defined. (x, y), (x', y'), (x", y") are the co-ordinates on the even rank tours.

$$x' = (x+n) - N_{xr}\left\lfloor \frac{x+n}{N_{xr}} \right\rfloor$$

$$y' = (y+n) - N_{yr}\left\lfloor \frac{y+n}{N_{yr}} \right\rfloor$$

$$x'' = (x+n) - N_{xr}\left\lfloor \frac{x+n}{N_{xr}} \right\rfloor$$

$$y'' = (y-n) + N_{yr}\left\lfloor \frac{n-y+N_{yr}}{N_{yr}} \right\rfloor$$

Case 2: r is an odd number, and thus, an even upper rank tori on the odd rank tours is defined. (x, y), (x', y'), (x", y") are the co-ordinates on the odd rank tours.

$$x' = (x+n) - (N_{xr} - N_{yr})\left\lfloor \frac{x+y}{N_{xr} - 2n} \right\rfloor$$

$$y' = (y+n) - (N_{xr} - N_{yr})\left\lfloor \frac{x+y}{N_{xr} - 2n} \right\rfloor$$

$$x'' = (x+n) - (N_{xr} - N_{yr})\left\lfloor \frac{x-y}{N_{xr} - 2n} \right\rfloor$$

$$y'' = (y-n) + (N_{xr} - N_{yr})\left\lfloor \frac{x-y}{N_{xr} - 2n} \right\rfloor$$

Nodes on the rank-(r+1) torus can be identified by another two-dimensional number (x_{r+1}, y_{r+1}). The number is given in the following manner:

1. Let node (0,0) of the rank-r torus be (0,0) of the rank-(r+1).
2. For an even rank torus, the axis is set as the same direction of the base torus. For an odd rank, the direction from (x, y) to (x', y') is set to be an x-axis of the rank-(r+1) torus.
3. Give a two-dimensional number to each node on the rank-(r+1) torus according to Definition 1.

The rank-(r+1) torus is called the upper rank torus based on the rank-r. n is called as a **cardinal number**. □

Note that there are several independent upper rank tori formed on a torus (This problem will be discussed in Section II-B). By forming upper rank tori recursively on the base torus (Definition 1) according to Definition 2, the Recursive Diagonal Torus (RDT) is defined.

Definition 3: : **Perfect RDT**
Let form upper rank tori recursively according to Definition 2 on the base torus defined in Definition 1 as many as possible. A network in which every node has links to form all possible upper rank tori (i.e. RDT(n, R, R))is called

the perfect RDT (**PRDT**(n, R)) where n is the cardinal number and R is the maximum rank. □

Although the PRDT is unrealistic because of its large degree ($4(R+1)$), it is important as a basis for establishing message routing algorithms of the RDT theoretically.

Definition 4: : **RDT**
Recursive Diagonal Torus **RDT**(n, R, m) is a class of networks in which each node has links to form m upper tori (the maximum rank is R) with the cardinal number n in addition to the links of the base torus. □

According to this definition, the degree of the RDT(n, R, m) is $4(m+1)$.

B. *Torus assignment*

Various structures of the RDT can be formed by changing n and m. Although large n is sometimes advantageous, the cardinal number n is set to be 2 so as to enable easy implementation of algorithms based on binary tree or cube. Since an upper torus requires four links, the RDT with a large m requires too much hardware. Here, a system with tens of thousands nodes (for example, array of 128×128 nodes or 256×256 nodes) is assumed, and m is set to be 1 (degree = 8). For such number of nodes, the upper most rank of tori is 4. Thus, the RDT(2,4,1) is mainly treated here.

In the RDT(2,4,1), one of upper rank tori is assigned to each node. Thus, the structure of the RDT(2,4,1) also varies with which rank of tori are assigned to each node. This assignment is called the *torus assignment*. For the torus assignment, the identification of upper rank tori is required.

Theorem 1:
$2n^2$ independent rank-$(r+1)$ tori are formed on a rank-r torus while there are $N_{xr} \times N_{yr} \geq 4n^2$ nodes in the rank-r torus and there are $N_{x0} \times N_{y0}$ nodes in the system. Where $N_{x0} = N_{y0} = c(2n)^\gamma$, and c, n, γ are natural numbers.
The proof is shown in *Proof 1* of Appendix.

Theorem 2:
Let be the following $2n \times n$ array of nodes in the rank-i torus of the PRDT(n,R).

$$\begin{array}{cccccc}
(0,0) & (1,0) & (2,0) & (3,0) & \cdots & (2\text{n}-1,0) \\
(0,1) & (1,1) & (2,1) & (3,1) & \cdots & (2\text{n}-1,1) \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
(0,\text{n}-1) & (1,\text{n}-1) & (2,\text{n}-1) & (3,\text{n}-1) & \cdots & (2\text{n}-1,\text{n}-1)
\end{array}$$

If rank-$i+1$ torus can be formed on the rank-i torus, every node in the above array fraction is a member of an independent rank-$i+1$ torus. We call nodes in this array fraction a minimum node set. A node which has links for rank-i torus ($i \neq 0$) is called a node with the rank-i torus. "A node has the rank-i torus" means that the node has links for forming the rank-i torus.
The proof is shown in *Proof 2* of Appendix.

Definition 5: : **Identification of upper torus**
A rank-1 torus is called the (i_0, j_0) torus if a node (i_0, j_0) in the minimum node set of the base torus has the torus. Similarly, a rank-2 torus is called the $(i_0, j_0)(i_1, j_1)$ torus if a node (i_1, j_1) in the minimum node set of the (i_0, j_0) torus has the torus. In general, a rank-k torus is called the $(i_0, j_0)(i_1, j_1)\ldots(i_{k-1}, j_{k-1})$ torus if a node (i_{k-1}, j_{k-1}) in the minimum node set of the $(i_0, j_0)(i_1, j_1)\ldots(i_{k-2}, j_{k-2})$ torus has the torus.

Here, (*,*) means all nodes in the minimum set. Thus, $(i_0, j_0)(i_1, j_1)\ldots(i_{k-2}, j_{k-2})(*, *)$ represents all rank-k tori on the $(i_0, j_0)(i_1, j_1)\ldots(i_{k-2}, j_{k-2})$ torus. □

For example, the rank-1 torus in Figure 3 is called (1,0) torus. The rank-2 torus in Figure 3 is (1,0) torus formed on (0,0) torus, and thus, called ((0,0)(1,0)) torus.

Various torus assignment strategies can be selected considering the traffic of the network. If the local traffic is large, the number of nodes which have low ranks should be increased. However, complicated torus assignment introduces difficulty to the message routing algorithm and implementation.

A relatively simple torus assignment is selected here.

Definition 6: : **The RDT(2,4,1)/α**
Here, RDT(2,4,1) with the following torus assignment is called the RDT(2,4,1)/α .

- *rank*-1: (1,0) , (3,1)
- *rank*-2: ((0,0)(*,*)) , ((2,1)(*,*))
- *rank*-3: ((1,1)(*,*)(*,*)) , ((3,0)(*,*)(*,*))
- *rank*-4: ((0,1)(*,*)(*,*)(*,*)) , ((2,0)(*,*)(*,*)(*,*)) □

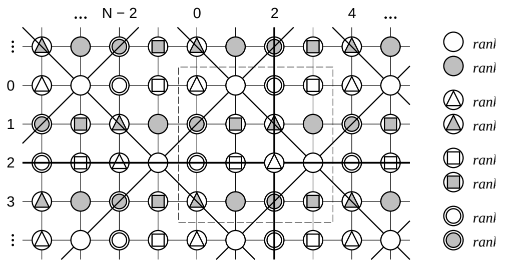


Fig. 3. Torus assignment for the RDT(2,4,1)/α

Torus assignment used in the RDT(2,4,1)/α is shown in Figure 3. In this assignment, a node has eight links, four for the base (rank-0) torus and four for rank (1-4) torus (Most of links for upper rank tori are omitted in Figure 3). With the cardinal number $n = 2$, 8 independent rank-1 tori can be formed on the base torus. Two tori (1,0) and (3,1) are used directly as the rank-1 torus. Other rank-2 tori are formed on two rank-1 tori (0,0) and (2,1). Similarly, two rank-1 tori are used in forming rank-3 tori, and two rank-2 tori for rank-4 tori.

Note that a node with an upper rank torus has neighboring nodes with tori of other three upper ranks. Therefore, a packet sent from any node can be passed to any rank torus with a single message transfer between neighboring nodes. This property reduces the diameter and average distance between nodes.

Another torus assignment called RDT(2,4,1)/β is proposed[32]. It improves the average distance compared with

RDT(2,4,1)/α when local communication is dominant.

III. Routing Algorithm

A. Vector routing

First, a routing algorithm called the vector routing[11] is introduced for the PRDT. In this routing algorithm, the route of a message is represented with a combination of unit vectors each of which corresponds to each rank of tori. On the torus structure, a vector from a source node to the destination node is represented with a vector $\vec{A} = a\vec{x_0} + b\vec{y_0}$ where $\vec{x_0}$ and $\vec{y_0}$ are unit vectors of the base (rank-0) torus. The goal of the routing algorithm is to represent the vector $\vec{A}$ with a combination of vectors each of which corresponds to a unit vector of each rank of torus (Figure 4(a)).

First, the direction of the unit vector corresponding to each rank torus must be defined. The direction of the unit vector for each rank torus rotates clockwise at an angle of 45 degrees as the rank increases as shown in Figure 4(b). The unit vectors of rank-(r+1) torus $\vec{x}_{r+1}, \vec{y}_{r+1}$ is represented with the unit vectors of rank-r ($\vec{x_r}, \vec{y_r}$) as follows:

$$\vec{x}_{r+1} = n\vec{x_r} + n\vec{y_r} \qquad \vec{y}_{r+1} = -n\vec{x_r} + n\vec{y_r}$$

. First, the target vector $a\vec{x_0} + b\vec{y_0}$ is represented with a

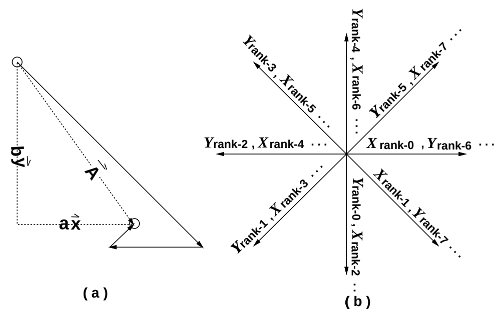


Fig. 4. Directions or the coordinate axes

combination of $\vec{x_1}$,$\vec{y_1}$,$\vec{x_0}$ and $\vec{y_0}$ as follows:

$$a\vec{x_0} + b\vec{y_0} = g\vec{x_1} + f\vec{y_1} + j\vec{x_0} + k\vec{y_0}$$

Here, we select maximum g and f in order to use the upper torus as possible. From previous equations, maximum integers for g and f are represented as follows:

$$g = \frac{a+b}{2n} \qquad f = -\frac{a-b}{2n}$$

In order to minimize j and k corresponding to the remaining unit vectors of the rank-0 torus (and so required message transfer using the rank-0 torus), the integer divisor used here is rounded to the nearest whole number (if the remainder is greater than n, increment the divisor).

Thus, j and k are represented with g and f:

$$a\vec{x_0} + b\vec{y_0} = g(n\vec{x_0} + n\vec{y_0}) + f(-n\vec{x_0} + n\vec{y_0}) + j\vec{x_0} + k\vec{y_0}$$

$$a = ng - nf + j \qquad b = ng + nf + k$$

$$j = a - ng + nf \qquad k = b - ng - nf$$

. Then, $g\vec{x_1} + f\vec{y_1}$ are represented with a combination of vector $\vec{x_2}$, $\vec{y_2}$, $\vec{x_1}$, and $\vec{y_1}$ in the same manner. By iterating this process to the maximum rank, vectors for message routing are obtained.

This algorithm is specified with a simple C program fragment as follows:

Algorithm: The simple vector routing

```
for (rank=0; rank < MAX_RANK; rank++) {
    g=div_2n(a+b); f=div_2n(-(a-b));
    vector[rank].x= a-(n*g - n*f);
    vector[rank].y= b-(n*g + n*f);
    a=g;   b=f;
}
vector[MAX_RANK].x=g; vector[MAX_RANK].y=f;
```

where div_2n means division with $2n$ rounding to the nearest number. The routing vectors for each rank are obtained in the array *vector*[*rank*].

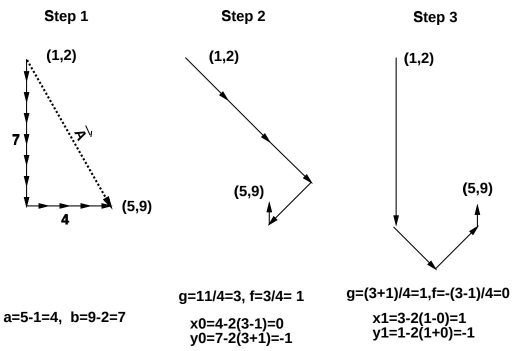


Fig. 5. An example of the vector conversion

Figure 5 shows an example of a vector from (1,2) to (5,9) converted into a combination of unit vectors of rank-0, rank-1, and rank-2.

Theorem 3: The diameter of the RDT(n,R,R)(i.e. PRDT(n,R)) with the simple vector routing is as follows:

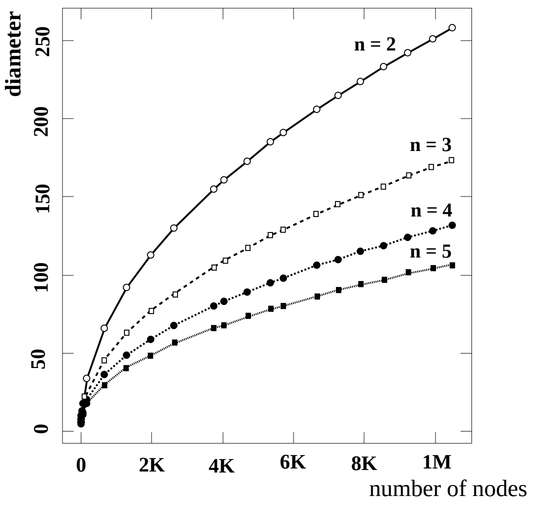
$$D = \frac{N2^{\lceil \frac{R-1}{2} \rceil}}{(2n)^R} + nR$$

where D is the diameter, N is the size of the base torus, R is the maximum effective rank number.

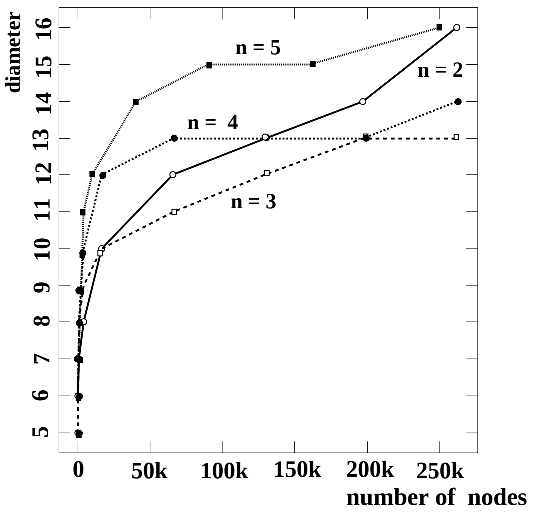
The proof is shown in the *Proof 3* of Appendix.

In the simple vector routing, vectors of a rank are decided with already fixed vectors of lower ranks. Better routes may be found by considering the vectors of all lower ranks again. However, the difference between the diameter with the simple vector routing and the theoretical one is at most 1. Considering that the simple vector routing only requires simple add and shift operations, it is advantageous in most cases.

According to the Theorem 3, diameters of RDTs with various n and R can be obtained. Figure 6 (a) shows diameters of the RDT(n,1,1) in which a node has the base torus and only rank-1 torus. In this case, the diameter is decreased with a larger n. It shows that a large n is advantageous if it is not possible to make higher rank tori.



(a) RDT(n,1,1)



(b) RDT(n,4,4)

Fig. 6. Diameter vs. the number of nodes

However, if higher rank tori are provided, a smaller n is suitable. Figure 6 (b) shows the diameters of the RDT(n,4,4) in which a node has the base torus and rank-1,2,3,4 upper torus links. In this case, $n = 2$ is optimal with up to 16000 nodes. For larger number of nodes, $n = 3$ is optimal.

B. Floating vector routing

Unlike the PRDT, only one upper torus is connected to each node in the RDT(2,4,1)/α . Therefore the vector routing algorithm for the PRDT must be extended for the RDT(2,4,1)/α so as to transfer the message to upper rank tori using the rank-0 torus. Since the vectors from the source node to the destination node are determined before routing, this algorithm is called the fixed vector routing.

For the fixed vector routing of the RDT(2,4,1)/α , the following theorem is easily found.

Theorem 4:
The diameter of the RDT(2,4,1)/α ($D_{RDT(2,4,1)/\alpha}$) is as follows:

$$D_{RDT(2,4,1)/\alpha} \leq D_{PRDT(2,R)} + R - 1$$

where $D_{PRDT(2,R)}$ is the diameter of the PRDT(2,R).

Proof: Since every torus can use rank-0 (base) torus, at most $R - 1$ times additional steps of message transfers are required for the use of every upper rank torus. ■

However, the algorithm becomes complicated for considering the routing on the rank-0 torus links required to use upper rank tori in the RDT(2,4,1)/α . Moreover, this routing is not suitable for bypassing faulty links or hot spot nodes. To cope with these problems, the floating vector routing is proposed.

This routing method is torus assignment independent, and the vector calculated by the vector routing algorithm for the PRDT can be directly used. In this method, the vector reduction is done in the source node, according to the vector routing for the PRDT. Then, the routing tag for routing vectors is stored in the packet header. A router attached to each node checks the tag, and determines the link to send the packet. From Theorem 3, the maximum number of vectors used for a direction is equal to n. In the RDT(2,4,1)/α , 3bits (-2 to 2) are required for horizontal and vertical directions respectively, and thus, 6bits are required for a rank in total. Here, the bit map for the rank-r vectors is represented as (v_{rh}, v_{rv}).

Each router is provided with a simple table indicating the direction of the nearest node for every rank of torus. In the RDT(2,4,1)/α , each node can use any rank of torus by only a single step of message transfer to a neighboring node, and thus the local routing table is quite simple.

The algorithm for the floating vector routing is as follows.

1. If the node i (i_x, i_y) has the rank-r torus and $(v_{rh}, v_{rv}) \neq (0, 0)$, send the packet according to (v_{rh}, v_{rv}). Otherwise, choose rank p whose $(v_{ph}, v_{pv}) \neq (0, 0)$. If there are no upper ranks to be routed, goto (4).
2. Choose the node j (j_x, j_y) which has the rank-p torus and minimizes $abs((j_x - i_x) - v_{0h}) + abs((j_y - i_y - v_{0v})$. This node selection is done by the local table reference.
3. Send the packet to node j through links for the rank 0 torus.
   When the packet reaches to the node j, replace i by j, and (v_{0h}, v_{0v}) by $(v_{0h} + (j_x - i_x), v_{0v} + (j_y - i_y))$. Then, goto (1).
4. Send the packet according to (v_{0h}, v_{0v}) through links of the rank 0 torus.

In this method, the packet transfers on the rank-0 torus which is required to use upper rank tori are automatically adjusted, and so the vectors calculated for the PRDT can be directly utilized. This method can be applied for any torus assignment of the RDT, and also useful as the basis of the fault tolerant or adaptive routing. Since the node is selected so as to minimize the routing with rank-0

TABLE I
Diameter and average distance of the RDT

Number of nodes			1024	4096	16384	65536
PRDT(2,R)	Theoretical diameter		5	6	8	10
	Simple Vector Routing		6	7	9	10
RDT(2,4,1)/α	Floating Vector Routing	Diameter	7	8	9	11
		Average distance	5.56	6.67	7.82	8.95
	Fixed Vector routing	Diameter	9	11	15	19
		Average distance	5.67	7.07	8.60	11.21
	Deadlock-free	Diameter	11	12	17	19
		Average distance	6.68	8.19	9.96	11.69
	Closed Partitioning	Diameter	13	17	19	23
		Average distance	6.69	8.37	9.93	11.74

torus, both the diameter and average distance are also minimized. Table I shows the diameter and average distance of the RDT(2,4,1)/α with the floating vector routing and fixed vector routing. As shown in Table I, the diameter and average distance of the RDT(2,4,1)/α with the floating vector routing are not so increased compared with that of the PRDT(2,R), and greatly improved compared with the fixed vector routing.

IV. Performance evaluation and comparison with other networks

As shown in section V, the RDT is originally proposed for hierarchical multicasting in a massively parallel processor. However, it is a satisfactory performance for a general purpose network in many aspects. In this section, the performance of the RDT is evaluated and its properties as a general purpose network are discussed.

A. *Comparison with other networks*

First, the diameter and degree of the RDT(2,4,1)/α is compared with other direct networks.

From Table II, it appears that RDT(2,4,1)/α supports smaller diameter than most of direct networks considering its degree. Especially, for the system with 65536 nodes, the diameter of the RDT(2,4,1)/α is smaller than that of the hypercube with only a half degree.

Although De Bruijn, Kautz, and Pradhan provide a comparable diameter and degree with RDT(2,4,1)/α , they are difficult to use to embed the mesh structure. Diagonal Mesh [20] which also consists of a torus and bypass links is equivalent to the PRDT(n,1,1). It provides no recursive structure. There are other networks (base-m n-cube or hypermesh[21] [22] and fat tree[7]) which are suitable for MPCs. However, the comparison is difficult because they are indirect networks. Although the latency of the message passing in these networks is smaller than that of the RDT, large size of crossbar switches are required if the size of the network is larger than ten thousand.

Bisection bandwidth of the RDT is difficult to represent with a simple formula, since it depends on both the network parameters (n,m,R) and torus assignment. Assuming that the node number is N and channel bandwidth is W, bisection bandwidth of k-ary n-cube ($N = k^n$) is represented as $2Wk^{n-1}$. Since in RDT(2,4,1)/α each node provides a 2D torus as a base torus and another 2D torus with different size as an upper torus, bisection bandwidth is sum of those of two 2D tori. For RDT(2,4,1)/α , bisection bandwidth b_{rdt} is represented as follows:

$$b_{rdt} = \begin{cases} 2 \times 13 \times W \times k, & \text{for } k \leq 64 \\ 2 \times 28 \times W \times k, & \text{for } k > 64 \end{cases}$$

From this equation, the order of RDT(2,4,1)/α 's bisection bandwidth is just as same as 2D torus($O(k)$).

TableIII represents examples of bisection bandwidth of direct networks ($N = 4096$ and $N = 16777216$).

TABLE III
Bisection bandwidth of direct networks

network	4096 nodes (64×64)	16777216 nodes (4096×4096)
2D Torus	$128W$	$8192W$
3D Torus	$512W$	$131072W$
4D Torus	$1024W$	$524288W$
Hypercube	$2048W$	$8388608W$
RDT(2,4,1)/α	$1664W$	$229376W$

From TableIII, it is shown that the bisection bandwidth of RDT(2,4,1)/α is larger than that of 4D torus when node number is 4096. For a larger number of nodes such as 16777216, the bisection bandwidth of 4D torus becomes larger than that of RDT(2,4,1)/α since the order of RDT(2,4,1)/α 's bisection bandwidth is $O(k)$.

B. *Performance evaluation*

In this paper, an interconnection network simulator is utilized to compare the performance of the RDT with other networks. This simulator has been developed to estimate the performance of interconnection networks for massively parallel computers which has tens of thousands processor elements.

The interconnection network simulator used here is a flit-level simulator written in C++. Network size, the number of virtual channel, and packet length are selected just by changing parameters. As shown in Figure 7, each node consists of a processor, request queue and the router which provides bidirectional channels. Each node is connected with neighboring nodes by using bidirectional channels attached to the router.

As shown in Figure 8, a simple router model consisting of channel buffers, crossbar, link controller (LC), virtual channel controller (VC) and control circuits is used.

TABLE II
Diameter (degree) of direct networks

Nodes number	4096	Notes	65536	Notes
2D Torus	64(4)	64*64	256(4)	256*256
3D Torus	24(6)	16*16*16	64(6)	32*32*64
4D Torus	16(8)	8*8*8*8	32(8)	16*16*16*16
Hypercube	12 (12)		16(16)	
De Bruijn[6]	12 (4)		16 (4)	
Kautz[13]	11 (4)	3072 Nodes	15 (4)	49152 Nodes
Pradhan[14]	12 (5)		16 (5)	
Circular omega[15]	20 (4)	5120 Nodes	26 (4)	53246 Nodes
n-Star graph[8]	7 (6)	5040 Nodes	8 (7)	40320 Nodes
CCC[16]	40 (3)	9-9 4608 Nodes	66 (3)	12-12 49152 Nodes
Hypernet[17]	19(5)	4D3-hierarchy	17(6)	5D3-hierarchy
Crossed Cube [18]	7(12)		9(16)	
Midimew [19]	46(4)	64*64	181 (4)	256*256
RDT(2,4,1)/α	8(8)	64*64	12(8)	256*256

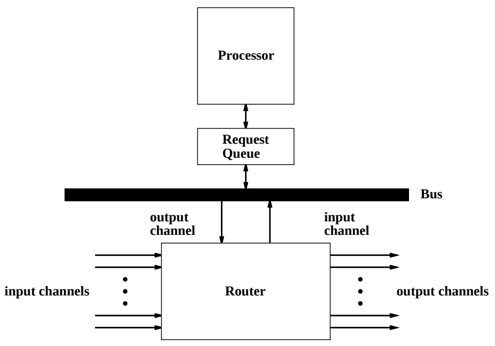


Fig. 7. The construction of each node

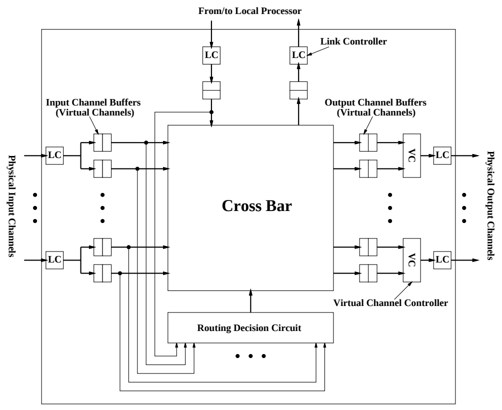


Fig. 8. Router model

In this paper, we will consider five interconnection networks with 4096 nodes, i.e., the RDT(2,4,1)/α , the 2D torus (64×64), the 3D torus (16×16×16), the 4D torus ($8 \times 8 \times 8 \times 8$) and the 12-dimensional hypercube. In terms of implementation, the simulation parameters for each interconnection network are set as TableIV.

The destination node of a packet is determined by the traffic pattern in the simulator. Two traffic patterns are used in this simulation:

- *uniform*
  All destination nodes are selected randomly, and so distributed uniformly.
- *hot spot*
  Only 128 nodes (i.e. $\frac{1}{32}$ of all nodes) are selected for destination node, thus causes hot spot traffic.

TABLE IV
Simulation parameters

Network size	4096 nodes
Packet length	16 flits or 128flits(fixed)
Packet header length	2 flits (fixed)
Routing method	wormhole
Packet generation time	1 clock
Routing and cross bar setup time	1 clock
Flit transfer time (from input buffer to output buffer)	1 clock
Flit transfer time (at physical link)	1 clock
Simulation time	10000 clock (ignore the first 1000 clock)
The number of virtual channels	2

The following two measures are used for evaluations.

B.0.a Average path length:. The average distance between nodes can be measured without any conflict in the network.

B.0.b Network latency:. Let the time when a node p inserts the first flit of a packet into the input buffer be t_0, and the time when the tail flit of the packet is sent to the processor at destination node q be t_1. Here, we call $T_{lat}(p,q) = t_1 - t_0$ the *network latency*, and use the measure of the network performance.

B.0.c Throughput:. Throughput is the maximum amount of information delivered per time unit. Here, throughput could be measured in flits per node in each clock cycle.

When the network is saturated, the execution of simulation is aborted.

TABLE V
Average path length of direct networks

network (4096 nodes)	average path length	diameter
2D Torus	32.0078	64
3D Torus	12.0029	24
4D Torus	8.0020	16
Hypercube	6.0015	12
RDT(2,4,1)/α	8.1902	12

Table V shows the average path length of each direct networks (4096 nodes). In this simulation, since a simple deadlock free routing algorithm shown in the sectionIV-C is used, diameter of RDT(2,4,1)/α is larger than that of the 4D torus which provides the same number of links a little.

Figure 9 shows the average message latency as a function of network throughput for the RDT(2,4,1)/α , 2D torus, 3D torus, 4D torus, and the 12-dimensional hypercube under uniform traffic. In this case, packet length is 16 flits.

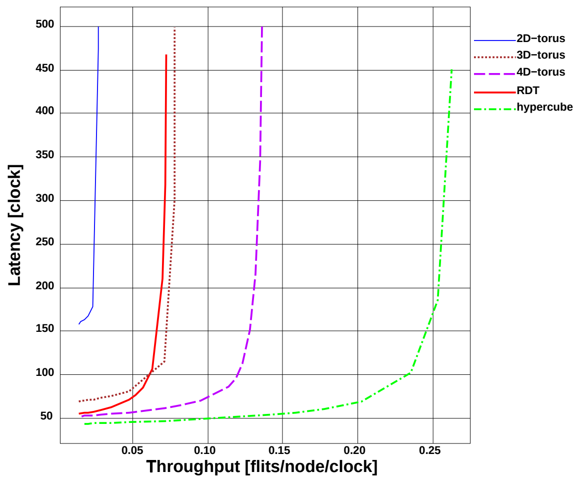


Fig. 9. Network Throughput vs. Average message latency

The latency of 2D torus is large because of its small degree. In contrast, the latency of higher dimensional topologies: the hypercube and 4D torus is low. The RDT(2,4,1)/α also has a low latency under low throughput. However, the latency of the RDT(2,4,1)/α becomes worse than that of the 3D torus when the throughput is high, thus, the network is congested. Under the uniform traffic, when the network is congested, messages transferred with upper rank tori will increase. This causes severe congestion on upper rank tori, and stretches the latency in RDT(2,4,1)/α .

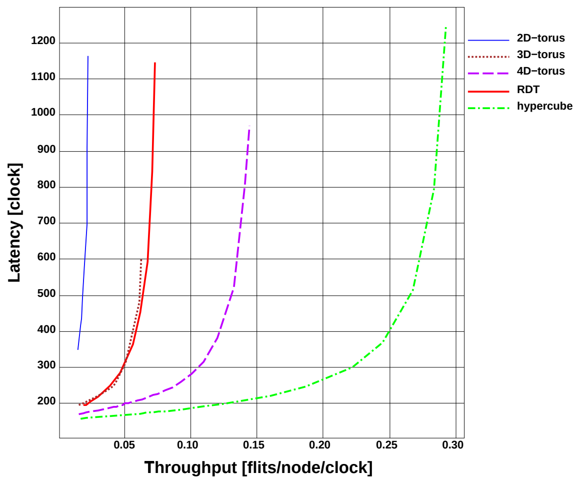


Fig. 10. Network Throughput vs. Average message latency (long messages)

Figure 10 shows simulation results in the case of long messages (packets with 128 flits) are transferred. In this case, the latency of RDT(2,4,1)/α is better than 3D torus, even under a heavy traffic load. It comes from that the average distance of RDT(2,4,1)/α is smaller than that of 3D torus as shown in Table V.

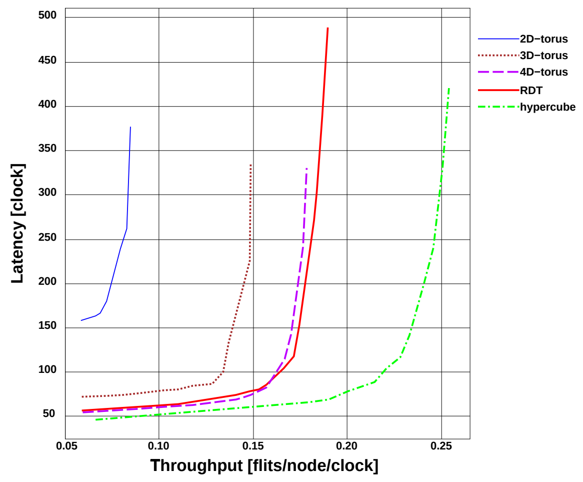


Fig. 11. Network Throughput vs. Average message latency (hot spot traffic)

Figure 11 shows simulation results under "hot spot" traffic, that is, packets are concentrated to only a part of nodes. Under the hot spot traffic, RDT(2,4,1)/α performs better performance than 4D torus. Since RDT(2,4,1)/α consists of recursive structured tori, RDT(2,4,1)/α contains a hierarchical structure and this will cause performance improvement under the hot spot traffic.

Through these evaluations, the latency of the RDT(2,4,1)/α is better than those of 2D and 3D torus except the case using short packets, and under hot spot traffic, RDT(2,4,1)/α is better than that of 4D torus. However, the congestion of the upper tori degrades performance under uniform traffic. In order to avoid the partial congestion of the network, adaptive routing is required.

C. Other issues of the RDT

The RDT provides other features which are useful as interconnection networks for MPCs.

C.0.d Deadlock free routing:. The RDT consists of hierarchical connected tori, that is k-ary 2-cubes. For the PRDT, the e-cube routing [12] which is a common deadlock-free routing algorithm for k-ary n-cubes is almost directly applied and the deadlock can be avoided with two virtual channels for each link[11].

For the RDT(2,4,1)/α , the order of rank usage is restricted so as the dimension ordering of e-cube routing. The packet must traverse the highest rank torus first, and traverse each rank torus in descending order. Another virtual channel is required for links to the X-dimension of the rank-0 torus[11] for moving between different ranks without deadlock.

C.0.e Closed partitioning:. The network partitioning is important for a massively parallel computer, since multiple users might use the machine independently. One of the major disadvantages of 2-D or 3-D torus network is that a partitioned torus must be used as a 2-D or 3-D mesh (not a torus) as the wrap-around links cannot be used. This causes a lot of packets for distant nodes.

However, in the RDT, by using links for upper rank tori, the network is partitioned without severe performance degradation. In the RDT(2,4,1)/α , the quad-sectional (2×2) partitioning is possible. For, example, the 16×16 torus shown in Figure 1 can be divided into four 4×4 tori. The minimum size of the partition is 4×4. The diameter of the partitioned RDT(2,4,1)/α is stretched 2-4 steps compared with the same scale RDT(2,4,1)/α [11].

C.0.f Hypercube embedding:. Hypercube can be easily embedded by the RDT(2,4,1)/α if the number of nodes is the same. Even rank tori can be directly used for the hypercube embedding, while two step message transfers are required in odd rank tori. Since these transfers are performed without any conflict, the hypercube embedding is performed with 2 steps. With the RDT(2,4,1)/α , $2(R+1)$ steps ($R \leq 4$) are required[11].

C.0.g Adaptive Routing:. Although the vector routing proposed here is a simple and near-optimal method, it cannot use alternative paths for avoiding congestion in the network. Since RDT(2,4,1)/α provides multiple paths between nodes, adaptive routings can be applied by simple modification of methods for torus networks[24][25].

Two algorithms are proposed for RDT(2,4,1)/α . The first algorithm is based on Duato's necessary and sufficient condition[23]. With this method virtual channels are effectively used while paths with redundant routing steps are prohibited[29]. Another algorithm is based on the turn model[26]. By prohibiting certain turns on RDT(2,4,1)/α , it permits paths with additional hops. Both algorithms are proved to be deadlock free, and from evaluation, it is known that adaptive routing on RDT(2,4,1)/α greatly improves the performance[24][25].

C.0.h Fault tolerance:. From the Theorem 1, the RDT(2,4,1)/α provides independent tori for each rank: 2 for rank-1, 16 for rank-2, 128 for rank-3, and 1024 for rank-4. Since upper tori have a degree of redundancy, there exists an alternative torus which can be used when a torus cannot be used by a fault. By using the floating vector routing, a faulty node can be bypassed just by rewriting the table which stores location of a node with each upper rank torus.

V. The RDT router chip

A. A massively parallel processor prototype JUMP-1

The RDT was originally proposed as a network of a massively parallel processor prototype JUMP-1 [31][10] which is developed by collaboration between 7 Japanese universities.

The major goal of this project is to establish techniques for building an efficient distributed shared memory on a massively parallel processor. JUMP-1 consists of clusters which is a bus-connected multiprocessor including 4 coarse-grained processors(CPU), 2 fine-grained processors (Memory Based Processor or MBP) each of which is directly connected to a main memory and the RDT router chip. A CPU is an off the shelf RISC processor (SUN SuperSparc+) which performs the main calculation of the program.

In JUMP-1, each node processor 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. Unlike other CC-NUMAs, update type cache coherence protocols can be utilized in JUMP-1 for applications which require frequent data exchange.

For supporting this approach, Reduced Hierarchical Bitmap Directory schemes (RHBDs) were introduced[27][28]. In the RHBD, the bit map directory is reduced and carried in the packet header for quick multicasting without accessing directory in each hierarchy.

Now, JUMP-1 with 64 processing elements (Figure 12) is working and the RDT is actually used as the interconnection network. Here, the RDT router chip for efficient implementation of the RHBD is described.

Fig. 12. JUMP-1 with 64 processing element

B. Message broadcast/multicast

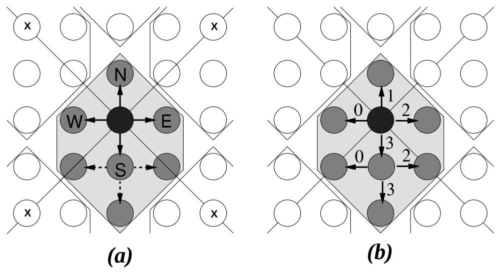


Fig. 13. The message transfers for broadcast and deadlock-free

In the RHDB, messages for keeping the cache consistency are required to multicast through the hierarchical network structure. Upper rank tori of the RDT can be used for this purpose. The pattern of message transfers from a node with the rank-i torus to nodes with the rank-(i-1) torus is shown in Figure 13(a). Two steps are required: (1) each node transfers a message to four neighbors, (2) a

TABLE VI
The number of steps required for the broadcast/broad-call

Number of nodes	1024	4096	16384	65536
RDT(2,4,1)/α	10	11	12	13
Hypercube	10	12	14	16
4-ary tree	8	10	12	14

neighbor (South in Figure 13(a)) transfers the message to three neighbors except the source node (dotted lines in Figure 13(a)). As shown in this figure, the eight nodes which receive the data never receive the message from other nodes (marked X in Figure 13(a)) on the same rank-i torus. By repeatedly applying this data transfer from the maximum rank to the rank-0, the data is sent to every node of the PRDT (i.e. broadcast). Since two steps are required for each rank, message broadcast on the PRDT(2,R) requires 2(R+1) steps.

On the RDT(2,4,1)/α , the following steps are required:

$$2(R+1)+(R-1)+1=3R+2.$$

The second term $(R-1)$ corresponds to the additional steps of message transfers on the rank-0 torus to use other rank tori, and the third term (1) is a message transfer to the node with the maximum rank of torus when the broadcast starts.

Table VI shows required steps for broadcasting on the RDT(2,4,1)/α , 4-ary tree and hypercube. In the RDT(2,4,1)/α , there are many nodes with the maximum rank torus, and all of them are used as a root node for broadcast. Therefore, broadcast is performed on the RDT(2,4,1)/α with less steps than the hypercube and sometimes the 4-ary tree.

This broadcast method is advantageous for multicasting data to nodes local to the source node. In the RDT, nodes which receive the packet through the tree whose root rank is 'i' are located around the source node. For larger 'i', the number of such nodes becomes large, thus the area to which a message is multicast becomes wide. We call such an area "territory" of a multicast. Figure 14 shows territories of a multicast of rank-1 and rank-0. Since the shape of a territory is always formed surrounding a source node, message multicast to local nodes can be performed as a lower rank multicast (thus, forms just a small territory).

The tree formed on the RDT is a kind of "fat-tree" which provides many root nodes. Therefore, the congestion of root nodes is relaxed even if many source nodes multicast their data simultaneously and independently. Using this property, messages for keeping cache consistency required in the RHBD are multicast effectively to local nodes without causing congestion at root nodes.

C. Structure of the router chip

The RDT router chip which provides message multicast for the RHBD scheme is available. As shown in Figure 15, the core of the chip is a 10×11 crossbar which exchanges packets from/to ten 18-bits-width links, that is, four for the rank-0 torus, four for the upper rank torus, and two for the MBPs of each cluster which manage the distributed shared memory of JUMP-1. In JUMP-1, two RDT router chips are used in the bit-sliced mode to form 36 bits width for each link.

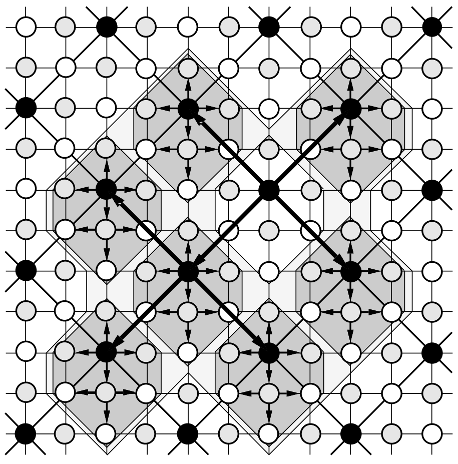


Fig. 14. Territory of a multicast

All packets are transferred between router chips synchronized with a unique 60MHz clock. In order to maximize the utilization of a link, packets are bi-directionally transferred. Maximum packets length is 16flits (36 bits-width 16flits-length) so as to carry a line of the cache. 3-flits header which carries the bit-map of the RHBD is attached to every packet, but the length of the body is variable.

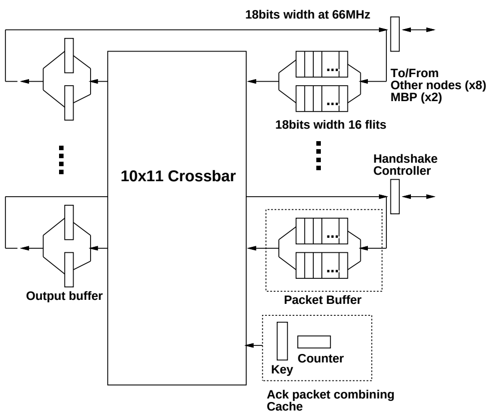


Fig. 15. The structure of the RDT router

Unlike common router chips[4], the following facilities are provided:

- efficient deadlock free multicasting using the asynchronous wormhole routing,
- acknowledge packet combining,
- shoot-down/setup, and
- error/handling mechanism.

Especially, the deadlock free multicasting based on the RHBD and acknowledge packet combining are the most challenging aspect of this design. They reduce the congestion of network traffic drastically[30].

D. Chip Implementation

$0.5\mu m$ Hitachi BiCMOS SOG which provides maximum of 125K gates is utilized. Arbitration of the crossbar and the bidirectional transfer lines are performed simultaneously (It is the critical path of this chip), and it takes six clocks to pass through the chip without conflict. By using internal dual port RAMs, packet buffers can push and pull a flit of a packet simultaneously. The specification of RDT router chip is shown in Table VII. The package of RDT router chip provides 299 pins including 260 signal. In order to drive up to 2m cable directly, we used the ECL I/O buffer, and Bi-CMOS cell is widely used to secure large fanout and high gate speed. 19W maximum power consumption is caused by these Bipolar Cells. To cope with this power consumption, a large heat sink is attached.

The required number of gates are shown in the Table VIII. Random logics require 50,000 gates in total while areas corresponding to about 4,000 gates are required for dual-port RAM. Crossbar and arbiter, which are simple but require high performance, are designed in schematic while the complicated controllers are described with VHDL.

TABLE VII
Specification of RDT router

Power consumption	19.4W
Total Pins	299(Signal 260)
Rate of gate utilization	63
Clock rate	60MHz

TABLE VIII
Number of gates of RDT router chip

Block name	Gates	Blocks	Total	Description
Crossbar	2,927	1	2,927	Schematic
Arbiter	2,736	1	2,736	Schematic
Multicast controller	1,558	10	15,580	VHDL
I/O controller	397	10	3,970	VHDL
Bit-map generator	2,288	10	22,880	VHDL
Acknowledge combining	2,009	1	2,009	VHDL
RAM for buffer	2,021	20	40,420	RAM
Total			90,522	

VI. Conclusion

The RDT is a novel interconnection network for MPCs which achieves a small diameter which is almost half to that of the hypercube. By using the floating vector routing algorithm, the near-optimal routing is possible in any configuration of the RDT. Although the congestion on upper rank tori sometimes degrade the performance under the random traffic, the RDT provides much better performance than that of 2-D/3-D torus in most cases, and under hot spot traffic, the RDT provides much better performance than that of 2-D/3-D/4-D torus.

The RDT router chip which provides message multicast for maintaining cache consistency is available. Using the $0.5\mu m$ BiCMOS SOG technology, the versatile functions including hierarchical multicasting, combining acknowledge packets, shooting down/restart mechanism, and time-out/set-up mechanisms work at 60MHz clock rate. The JUMP-1 prototype with 64 processing units connected with the RDT router chip started to work in the spring of 2000. The communication performance with a real application is now under evaluation.

Appendix (Proofs)

Theorem 1: $2n^2$ independent rank-$(r+1)$ tori are formed on a rank-r torus while there are $N_{xr} \times N_{yr} \geq 4n^2$ nodes in rank-r torus and there are $N_{x0} \times N_{y0}$ nodes in the system. Where, $N_{x0} = N_{y0} = c(2n)^{\gamma}$, and c, n, γ are natural numbers.

Proof: On the rank-r torus whose grid size is a, the rank-(r+1) tori whose grid size is

$$b = \sqrt{(an)^2 + (an)^2} = \sqrt{2}an$$

are formed. When the size of the rank-r torus is $N_{xr} \times N_{yr}$, the size (node number) of rank-(r+1) torus ($N' = N_{x(r+1)} \times N_{y(r+1)}$) is:

$$N' = \frac{aN_{xr} \times aN_{yr}}{(\sqrt{2}an)^2} = \frac{N_{xr} \times N_{yr}}{2n^2}$$

Therefore, the number of independent rank-(r+1) tori (T) is:

$$T = \frac{N_{xr} \times N_{yr}}{N'} = \frac{N_{xr} \times N_{yr}}{\frac{N_{xr} \times N_{yr}}{2n^2}} = 2n^2$$

■

Theorem 2: If the rank-$(r+1)$ tori are formed, every member of any part of the rank-r torus which consists of an $2n \times n$ array of nodes takes an independent upper torus that are different to each other:

$$\begin{array}{cccccc}
(0,0) & (1,0) & (2,0) & (3,0) & \cdots & (2n-1,0) \\
(0,1) & (1,1) & (2,1) & (3,1) & \cdots & (2n-1,1) \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
(0,n-1) & (1,n-1) & (2,n-1) & (3,n-1) & \cdots & (2n-1,n-1)
\end{array}$$

Proof: From the Definition 2, the distance between two nodes on the rank-$(r+1)$ torus is $2n$ in the X-dimension and n in the Y-dimension. So, there are no links of the rank-$(r+1)$ between any two nodes of the above $2n \times n$ array. From the Theorem 1, we know that there are $2n^2$ independent rank-$(r+1)$ tori, that is, any node of the above $2n \times n$ array takes an independent upper torus that is different from each other. ■

Theorem 5:
The diameter D which is of the PRDT(n, R) with the vector routing (Vector Routing Algorithm) is as follow:

$$D = nR$$

where, n is the cardinal number, and R is the maximum rank number.

Proof: On the rank-r of the PRDT, a message from a source node to a destination node can be represented with a vector $\vec{A_r} = \alpha_r \vec{x_r} + \beta_r \vec{y_r}$ and the unit vectors of the neighboring rank have the following relations:

$$\vec{x}_{r+1} = n\vec{x_r} + n\vec{y_r} \tag{1}$$

$$\vec{y}_{r+1} = -n\vec{x_r} + n\vec{y_r} \tag{2}$$

where, $\vec{x_r}$, $\vec{y_r}$ and $\vec{x}_{r+1}$, $\vec{y}_{r+1}$ are the unit vectors of the rank-r and rank-$(r+1)$ respectively, and n is the cardinal number.

Assume that

$$\alpha_r\vec{x_r} + \beta_r\vec{y_r} = \alpha_{r+1}\vec{x}_{r+1} + \beta_{r+1}\vec{y}_{r+1} \tag{3}$$

Using Equation 1 and 2, Equation 3 can be represented as follows:

$$\alpha_r\vec{x_r} + \beta_r\vec{y_r} = \alpha_{r+1}(n\vec{x_r} + n\vec{y_r}) + \beta_{r+1}(-n\vec{x_r} + n\vec{y_r}) \tag{4}$$

Then we can get the result:

$$\alpha_r\vec{x_r} = \alpha_{r+1}n\vec{x_r} - \beta_{r+1}n\vec{x_r} \tag{5}$$

$$\beta_r\vec{y_r} = \alpha_{r+1}n\vec{y_r} + \beta_{r+1}n\vec{y_r} \tag{6}$$

$$\alpha_{r+1} = \frac{\alpha_r + \beta_r}{2n} \tag{7}$$

$$\beta_{r+1} = \frac{-\alpha_r + \beta_r}{2n} \tag{8}$$

Since α_{r+1} and β_{r+1} must be integral numbers, Equation 7 and 8 can get the nearest integral numbers α'_{r+1} and β'_{r+1} as following:

$$\alpha'_{r+1} = \alpha_{r+1} + \frac{t_{r1}}{2n} \tag{9}$$

$$\beta'_{r+1} = \beta_{r+1} + \frac{t_{r2}}{2n} \tag{10}$$

where, $\left|\frac{t_{r1}}{2n}\right| \leq 0.5$ and $\left|\frac{t_{r2}}{2n}\right| \leq 0.5$.

Using Equation 7 and 8, the following equations can be obtained.

$$\alpha'_{r+1} = \frac{\alpha_r + \beta_r}{2n} + \frac{t_{r1}}{2n} \tag{11}$$

$$\beta'_{r+1} = \frac{-\alpha_r + \beta_r}{2n} + \frac{t_{r2}}{2n} \tag{12}$$

Since rank-r vector is represented with a combination of rank-r and rank-(r+1) vectors by Vector Routing Algorithm, Equation 3 can be modified as follows:

$$\alpha_r\vec{x_r} + \beta_r\vec{y_r} = \alpha'_{r+1}\vec{x}_{r+1} + \beta'_{r+1}\vec{y}_{r+1} + a_r\vec{x_r} + b_r\vec{y_r} \tag{13}$$

Using Equation 1 and 2, we can represent Equation 13 as:

$$\alpha_r\vec{x_r}+\beta_r\vec{y_r} = \alpha'_{r+1}(n\vec{x}_r+n\vec{y}_r)+\beta'_{r+1}(n\vec{y}_r-n\vec{x}_r)+a_r\vec{x_r}+b_r\vec{y_r} \tag{14}$$

Then we can get two equations on the two directions of the unit vectors.

$$\alpha_r = n\alpha'_{r+1} - n\beta'_{r+1} + a_r \tag{15}$$

$$\beta_r = n\alpha'_{r+1} + n\beta'_{r+1} + b_r \tag{16}$$

then

$$a_r = \alpha_r - n\alpha'_{r+1} + n\beta'_{r+1} \tag{17}$$

$$b_r = \beta_r - n\alpha'_{r+1} - n\beta'_{r+1} \tag{18}$$

The $|a_r| + |b_r|$ means the message transfer steps on the rank-r after using rank-(r+1) vectors.

Using Equation 11 and 12, we can get a_r and b_r:

$$a_r = \alpha_r - n\left(\frac{\beta_r + \alpha_r}{2n} + \frac{t_{r1}}{2n}\right) + n\left(\frac{\beta_r - \alpha_r}{2n} + \frac{t_{r2}}{2n}\right) \tag{19}$$

$$b_r = \beta_r - n\left(\frac{\beta_r + \alpha_r}{2n} + \frac{t_{r1}}{2n}\right) - n\left(\frac{\beta_r - \alpha_r}{2n} + \frac{t_{r2}}{2n}\right) \tag{20}$$

With a simple reduction of the equations we can get following results:

$$a_r = -\frac{t_{r1} - t_{r2}}{2} \tag{21}$$

$$b_r = -\frac{t_{r1} + t_{r2}}{2} \tag{22}$$

Since $|t_{r1}| \leq n$, $|t_{r2}| \leq n$

$$t_{r1}^2 + t_{r2}^2 + |t_{r1}^2 - t_{r2}^2| \leq 2n^2$$

$$2(t_{r1}^2 + t_{r2}^2 + |t_{r1}^2 - t_{r2}^2|) \leq 4n^2$$

$$2t_{r1}^2 + 2t_{r2}^2 + 2|(t_{r1} - t_{r2})(t_{r1} + t_{r2})| \leq 4n^2$$

$$(t_{r1} + t_{r2})^2 + (t_{r1} - t_{r2})^2 + 2|(t_{r1} + t_{r2})(t_{r1} - t_{r2})| \leq 4n^2$$

$$|t_{r1} + t_{r2}|^2 + |t_{r1} - t_{r2}|^2 + 2|t_{r1} + t_{r2}||t_{r1} - t_{r2}| \leq 4n^2$$

$$(|t_{r1} + t_{r2}| + |t_{r1} - t_{r2}|)^2 \leq 4n^2$$

$$|t_{r1} + t_{r2}| + |t_{r1} - t_{r2}| \leq 2n$$

$$\left|\frac{t_{r1} - t_{r2}}{2}\right| + \left|\frac{t_{r1} + t_{r2}}{2}\right| \leq n$$

$$\left|-\frac{t_{r1} - t_{r2}}{2}\right| + \left|-\frac{t_{r1} + t_{r2}}{2}\right| \leq n$$

$$|a_r| + |b_r| \leq n$$

that is, the maximum steps of the message transfer on the rank-r is n.

Then, the diameter D which means total steps on all of the ranks is: $D = nR$. ∎

Theorem 6:
The steps of the message transfer on the rank-R of the RDT(n, R, R) *with Vector Routing Algorithm is:*

$$S = \frac{N2^{\lceil\frac{R-1}{2}\rceil}}{(2n)^R}$$

where S is the steps, N is the size of the base torus, and R is the maximum effective rank number.

Proof: This theorem is proved by the inductive method.

The number of the message transferring steps for the vector $\alpha_R\vec{x_R} + \beta_R\vec{y_R}$ on the highest rank-R is:

$$S = |\alpha_R| + |\beta_R|$$

1. In case of $R = 0$
   Since it is a torus structure,

$$|\alpha_0| \le \frac{N}{2} \tag{23}$$

$$|\beta_0| \le \frac{N}{2} \tag{24}$$

$$|\alpha_0| + |\beta_0| \le \frac{N}{2} + \frac{N}{2}$$

$$|\alpha_0| + |\beta_0| \le N$$

   it can be represented as

$$S = \frac{N2^{\lceil \frac{0-1}{2} \rceil}}{(2n)^0}$$

   that is, in case of R=0, Theorem 6 is tenable.
2. In case of $R = k$ (k is an odd number)
   (a) In case of $R = 1$
   From Equation 7 and 8,

$$S = |\alpha_1| + |\beta_1| = \left|\frac{\alpha_0 + \beta_0}{2n}\right| + \left|\frac{-\alpha_0 + \beta_0}{2n}\right|$$

$$S = \frac{|\alpha_0 + \beta_0| + | - \alpha_0 + \beta_0|}{2n}$$

   According to Equation 23 and 24

$$S = |\alpha_1| + |\beta_1| \le \frac{N}{2n} \tag{25}$$

$$S = \frac{N2^{\lceil \frac{1-1}{2} \rceil}}{(2n)^1}$$

   That is, Theorem 6 is tenable when $R = 1$.
   (b) If Theorem 6 is tenable when $R = k$ (k is an odd number, $k \le 1$), then

$$S = |\alpha_k| + |\beta_k| = \frac{N2^{\lceil \frac{k-1}{2} \rceil}}{(2n)^k} \tag{26}$$

   (c) In case of $R = k + 1$

$$S = |\alpha_{k+1}| + |\beta_{k+1}| = \left|\frac{\alpha_k + \beta_k}{2n}\right| + \left|\frac{-\alpha_k + \beta_k}{2n}\right|$$

$$|\alpha_{k+1}| + |\beta_{k+1}| = \frac{|\alpha_k + \beta_k| + | - \alpha_k + \beta_k|}{2n} \tag{27}$$

   Since

$$|\alpha_k + \beta_k| \le |\alpha_k| + |\beta_k|$$

$$| - \alpha_k + \beta_k| \le |\alpha_k| + |\beta_k|$$

   Then we can modify Equation 27 as follow:

$$|\alpha_{k+1}| + |\beta_{k+1}| \le \frac{2(|\alpha_k| + |\beta_k|)}{2n}.$$

   Using Equation 26, we can get

$$|\alpha_{k+1}| + |\beta_{k+1}| \le \frac{2\frac{N2^{\lceil \frac{k-1}{2} \rceil}}{(2n)^k}}{2n}$$

   That is

$$|\alpha_{k+1}| + |\beta_{k+1}| \le \frac{N2^{\lceil \frac{k-1}{2} \rceil + 1}}{(2n)^{k+1}} \tag{28}$$

   Since k is an odd number,

$$\left\lceil \frac{k-1}{2} \right\rceil + 1 = \left\lceil \frac{(k+1)-1}{2} \right\rceil$$

   The Equation 28 can be changed as follows:

$$|\alpha_{k+1}| + |\beta_{k+1}| \le \frac{N2^{\lceil \frac{(k+1)-1}{2} \rceil}}{(2n)^{k+1}}$$

   Then, Theorem 6 is tenable when $R = k + 1$.
   From above three results, theorem 6 is tenable when $R = k$ (k is an odd number).
3. In case of $R = k$ (k is an even number)

$$|\alpha_k| + |\beta_k| = \left|\frac{\alpha_{k-1} + \beta_{k-1}}{2n}\right| + \left|\frac{-\alpha_{k-1} + \beta_{k-1}}{2n}\right|$$

$$|\alpha_k| + |\beta_k| = \frac{|\alpha_{k-1} + \beta_{k-1}| + | - \alpha_{k-1} + \beta_{k-1}|}{2n}$$

$$|\alpha_k| + |\beta_k| \le \frac{2(|\alpha_{k-1}| + |\beta_{k-1}|)}{2n}$$

   Since k-1 is an odd number and Theorem 6 is tenable when k is an odd number, then

$$|\alpha_k| + |\beta_k| \le \frac{2\frac{N2^{\lceil \frac{(k-1)-1}{2} \rceil}}{(2n)^{k-1}}}{2n}$$

$$|\alpha_k| + |\beta_k| \le \frac{N2^{\lceil \frac{(k-1)-1}{2} \rceil + 1}}{(2n)^k} \tag{29}$$

$$|\alpha_k| + |\beta_k| \le \frac{N2^{\lceil \frac{(k-1)}{2} \rceil}}{(2n)^k}$$

   That is, Theorem 6 is tenable when $R = k$ (k is an even number).
4. Because of all above equations, Theorem 6 is tenable when $R = k$ (k is an integral positive number).

■

Theorem 3: The diameter of the RDT(n,R, R) *with Vector Routing Algorithm is as follows:*

$$D = \frac{N2^{\lceil \frac{R-1}{2} \rceil}}{(2n)^R} + nR$$

where D is the diameter, N is the size of the base torus, and R is the maximum effective rank number.

Proof: According to Theorem 5 and 6, the diameter of the RDT(n,R, R) consists of the steps on the highest rank

$$\frac{N2^{\lceil \frac{R-1}{2} \rceil}}{(2n)^R}$$

and the total steps on all other ranks nR. ■

References

[1] Paragon XP/S Product Overview, Intel Corp., 1991.

[2] H. Ishihata, T. Horie, S. Inano, T. Shimizu, S. Kato, and M. Ikesaka, "Third Generation Message Passing Computer AP1000," *International Symposium on Supercomputing*, Nov. 1991, pp. 46-55.

[3] S.L.Scott, G.M.Thorson, "The Cray T3E Network: Adaptive routing in a High Performance 3D Torus," Hot Interconnect Symposium IV, 1996. pp.147-156.

[4] W.J. Dally, A. Chien, S. Fiske, W. Horwat, J. Kenn, M. Larivee, R. Lethin, P. Nuth and S. Wills, "The J-machine: A Fine-Grain Concurrent Computer," *in Proc. IFIP 11th Computer Congress*, Aug. 1989, pp. 1147-1153.

[5] L.M. Ni and P.K. McKinley, "A Survey of Wormhole Routing Techniques in Direct Network," *IEEE Computer*, vol. 26, Feb. 1993, pp. 62-76.

[6] M.R. Samatham and D.K. Pladhan, "The De Bruijn Multiprocessor Network: Versatile Parallel Processing and Sorting Network for VLSI," *IEEE Trans. Comput.*, vol. 38, no. 4, Apr. 1989, pp. 567-581.

[7] C.E. Leiserson, "Fat-trees: Universal Networks for Hardware-Efficient Supercomputing," *IEEE Trans. on Comput.*, vol. 34 no. 10, pp. 892-901, Oct. 1985.

[8] S.B. Akers, D.Harel and B.Krishnamurthy, "The Star Graph: An Attractive Alternative to the n-cube," *Proc. of ICPP87*, pp. 393-400, Aug. 1987.

[9] Y.Yang, H.Amano, H.Shibamura, T.Sueyoshi, "Recursive Diagonal Torus: An interconnection network for massively parallel computers," *Proc. of IEEE the 5th Symposium on Parallel and Distributed Processing,* pp.591-594, Dec. 1993.

[10] H.Tanaka, "The Massively Parallel Processing System JUMP-1," IOS Press (ISBN90-5199-262-9), 1996.

[11] Y.Yang, H.Amano, "Message Transfer Algorithms on the Recursive Diagonal Torus," *IEICE Trans. on Info & Syst.* Vol.E79-D, No.2 1996, pp.107-116.

[12] W.J.Dally, C.L.Seitz, "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks," *IEEE Trans. on Comput.*, vol. 36 no. 5, pp. 547-553, May. 1987.

[13] J.C. Bermond and C. Peyrat, "De Bruijn and Kautz Networks: A Competitor for the Hypercube", *in Hypercube and Distributed Computers*, ed. F. Andre and J.P. Verjus, North-Holland, pp. 279-293, 1989.

[14] D.K. Pladhan, "Fault-Tolerant Multiprocessor Link and Bus Network Architectures," *IEEE Trans. on Comput.*, vol. 34, no. 1, Jan. 1985, pp. 35-45.

[15] S. Sakai, Y. Yamaguchi, K. Hiraki, Y. Kodama, and T. Yuba, "An Architecture of a Dataflow Single Chip Process," *Proc. of the 16th International Symposium on Computer Architecture*, pp. 46-53, 1989.

[16] F.P. Preparata and J. Vuillemin, "The Cube-Connected Cycles: A Versatile Network for Parallel Computation," *Comm. ACM*, vol.24, no. 5 pp. 300-309, May 1981.

[17] K. Hwang and J. Ghosh, "Hypernet: A Communication-Efficient Architecture for Constructing Massively Parallel Computer," *IEEE Trans. on Comput.*, vol. 36, no. 12 pp. 1450-1466, Dec. 1987.

[18] K. Efe, "The Crossed Cube Architecture for Parallel Processing," *IEEE Trans. on Comput.*, vol. 3, no. 5, pp. 513-524, Sept. 1992.

[19] R. Beivide, R. Herrada, J.L. Balcazar, and A. Arruabarrena, "Optimal Distance Networks of Low Degree for Parallel Computers," *IEEE Trans. on Comput.*, vol. 40, no. 10, pp. 1109-1124, Oct. 1991.

[20] K.W. Tang and S.A. Padubidri, "Routing and Diameter Analysis of Diagonal Mesh Networks," *in Proc. of ICPP92*, pp. I143-I150, Aug. 1992.

[21] N. Tanabe, S.Nakamura, T.Suzuoka, and S.Oyanagi, "Base-m n-cube: High Performance Interconnection Networks for Highly Parallel Computer PRODIGY," *in Proc. of ICPP91*, pp. I509-516, Aug. 1991.

[22] T. Szymanski, "O(LogN/LogLogN) Randomized Routing in Degree-LogN "Hypermeshes"," *in Proc. of ICPP91*, pp. 443-450, Aug. 1991.

[23] J. Duato , "A Necessary And Sufficient Condition For Deadlock-Free Adaptive Routing In Wormhole Networks" In *IEEE Trans. on Parallel and Distributed Systems, Vol.6, No.10*, 1995.

[24] A.Funahashi, T.Hanawa, T.Kudoh, H.Amano, "Adaptive routing on the Recursive Diagonal Torus." *Proc. of International Symposium on High Performance Computing* , pp.171-182 Nov. 1997

[25] A.Funahashi, A.Jouraku, H.Amano, "Adaptive routing on the Recursive Diagonal Torus." *Proc. of International Symposium on Parallel and Distributed Computing Systems* , pp.171-177 Aug. 1999

[26] C.J.Glass and L.M.Ni, "Maximally Fully Adaptive Routing in 2D Meshes," In *Proc. of ISCA92.* pp.278-287, 1992.

[27] K.Hiraki, H.Amano, M.Kuga, T.Seuyoshi, T.Kudoh, H.Nakashima, H.Nakajo, H.Matsuda, T.Matsumoto, and S.Mori, "Overview of JUMP-1, an MPP prototype for general purpose parallel computations," Proc. of IEEE International Symposium on Parallel Architecture, Algorithm and Networks, pp.427-434, Dec. 1994.

[28] 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. of ICPP'95 Vol.I, pp.186-193, Aug. 1995.

[29] J.Duato, S.Yalamanchili, L.Ni, "Interconnection Networks an Engineering Approach," IEEE Computer Society Press, 1997.

[30] K.Nishimura, T.Kudoh, H.Nishi, H.Amano, "Pruning Cache: A Dynamic Directory Generation Scheme for Distributed Shared Memory," Proc. of ISMM International Conference on Parallel and Distributed Computing and Networks, pp.89-94, 1998.

[31] H.Inoue, K.Anjo, J.Yamamoto, J.Tanabe, M.Wakabayashi, M.Sato, H.Amano, K.Hiraki, "The Preliminary Evaluation of MBP-light with Two Protocol Policies for A Massively Parallel Processor JUMP-1," Proc. of IEEE Frontiers Massively Parallel Computation, pp.268-pp.275, Feb. 1999.

[32] Q.Fan, Y.Yang, A.Funahashi, H.Amano, "A Torus Assignment for an Interconnection Network Recursive Diagonal Torus," Proc. of IEEE International Symposium on Parallel Architectures, Algorithms and Networks, pp.74-79, 1999.

Yulu Yang received the B.E. from Beijing Agriculture Engineering Univ., China, in 1984. He received the M.E.and Ph.D degrees From Keio univ, Japan, in 1993 and 1996. Now, he is working in the Dept.of Computer Science, Nankai Univ, China. His research interests include interconnection network and parallel processing.

Akira Funahashi received the B.E., M.E. and Ph.D degrees from Keio University, Japan, in 1995, 1997 and 2000. He is now a Research Associate in the Department of Information Technology, Mie University, Japan. His research interests include the area of interconnection network and parallel processing.

Akiya Jouraku received the B.E., M.E. degrees from Keio University, Japan, in 1998 and 2000. He is now a Research student in the Graduate School of Science and Technology, Keio University. His research interests include the area of interconnection network and parallel processing.

Hiroaki Nishi received the B.E., M.E, Ph.D. degree from Keio University, Japan, in 1994, 1996, 1999 respectively. He joined Parallel & Distributed Architecture Laboratory, Real World Computing Partnership in 1999. His research interests include the area of networks for high performance computing.

Hideharu Amano received the Ph.D degree from Keio University 1986. He is now an Associate Professor in the Department of Information and Computer Science, Keio University, His research interests include the area of parallel processing and reconfigurable systems.

Toshinori Sueyoshi has been a professor of Computer Science at Kumamoto University, Japan, since 1997. His research interests include computer architecture, parallel processing, system software, communication network, VLSI device and design. Sueyoshi received the BS and MS degrees in Computer Science and Communication Engineering from Kyushu University in 1976 and 1978 respectively. From 1978 to 1987, he was a research associate at Kyushu University, where he received PhD degree in Computer Science and Communication Engineering in 1986. From 1987 to 1989, he was an associate professor in the Interdisciplinary Graduate School of Engineering Sciences at Kyushu University. From 1989 to 1997, he was an associate professor in the Department of Artificial Intelligence at Kyushu Institute of Technology.