Vol. 35 No. 4 情報処理学会論文誌 Apr. 1994

分割再構成可能なトーラスネットワーク

林 憲一[†] ISAAC CHUANG[††] 堀江健志[†]

メッシュやハイパーキューブなどのネットワークで接続された並列計算機は分割して複数のシステムとして利用することが可能である．しかし，トーラスネットワークの場合にはメッシュの端と端を接続するラップアラウンドパスが存在するために，ネットワークを分割再構成するのが困難であった．本論文では，トーラスネットワークにスイッチを追加することで分割再構成を可能とする方法を提案する．また，ネットワークに追加したスイッチを使って，効率的なグローバル演算や放送が可能であることを示す．

A Reconfigurable Torus Network

KENICHI HAYASHI,[†] ISAAC CHUANG[††] and TAKESHI HORIE[†]

Independent subgroups of a hypercubic or mesh DMPP may be accessed simultaneously by multiple users. However, partitioning of a torus network is complicated by having to deal with wraparound paths. In this paper, we present a novel architecture for a dynamically reconfigurable torus; our design arises from geometrically folding the torus and interspersing switches between appropriately designated partitions. We avoid excess additional wiring and switching complexity and allow subgroups of processors to be utilized independently by different users. We describe our routing algorithm and show that efficient global reduction and broadcast can be performed using network switches.

1. はじめに

2次元や3次元などの低次元のトーラスネットワークはハードウエアコストを一定とした場合には，k-ary n-cube の中で，低レイテンシ，高スループットを実現する優れたネットワーク・トポロジであることが知られている[2]．

しかし，トーラスには，

(a) 長いラップアラウンドパスが存在すること
(b) デッドロックの可能性があること
(c) ノード間の平均距離が長いこと
(d) 分割再構成が困難であること

などの問題点があった．

本論文ではネットワークにスイッチを付加するだけで (a)，(c)，(d) の3つの問題を解決し，かつ効率的なグローバル演算や放送が可能なネットワークを提案する．

(b) のデッドロックの問題に関しては，構造化チャネル[7]やバーチャルチャネル方式[3]を用いることで，解決することができる．

本論文ではまず，分割再構成可能なトーラスネットワークを提案する．次にネットワーク分割の2つのモードを提案する．さらにネットワークを分割するための機構を利用した効率的なグローバル演算や放送について評価する．最後にネットワークに追加するハードウェアと外部とのインターフェースについて議論する．

2. 関連研究

ネットワークの再構成の問題に関してははこれまで様々な研究が行なわれてきた．ハイパーキューブに対しては，Efe[5]が，ハイパーキューブにスイッチを付加し，Crossed Cube (or Twisted Cube) とハイパーキューブを動的に切替えることで，放送やソートの効率化について検討している．

メッシュに関しては，Miller[10] や Chen[1] らがバスを用いた Reconfigurable Mesh を提案している．またこの Reconfigurable Mesh のための様々なアルゴリズムも検討されている[6),8),12]

トーラスに関しては中村[11]や松山[9]らによって再構成可能なトーラスが提案されている．11) では，予め冗長なパスを用意しておいて，構成に応じてパスを選

† (株) 富士通研究所
Fujitsu Laboratories Ltd.
†† スタンフォード大学
Stanford University

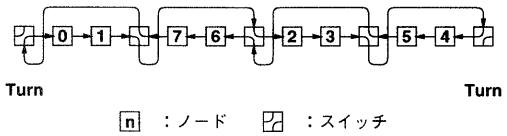


図1 分割再構成可能なトーラスネットワーク
Fig. 1 A reconfigurable torus network.

択することで，トーラスの再構成を実現している．9)ではトーラスにスイッチを付加し，そのスイッチを切替えることで，再構成を可能にしている．しかし，11)では冗長なパスの割合が高くなるという問題がある．また9)では構成を大きくしたときに，長いリターンパスが存在するという問題がある．

ネットワークの平均距離を短くする方法としては，Express Cubes[4]が提案されている．これはネットワークにスイッチとエクスプレス・チャネルを付加することで低次元の k-ary n-cube の平均距離を短くする方法である．しかし，Express Cubes では，元のネットワークにスイッチとバイパス用のチャネルの両方を追加する必要がある．

本論文で提案する方法は，トーラスネットワークにスイッチだけ追加し，冗長なパスは必要ない．またトーラスを折り畳んでいるため，構成が大きくなった場合にも長いリターンパスが存在しない．また，付加したスイッチを経由することで，エクスプレス・チャネルを構成することが可能で，トーラスの平均距離を短くすることができる．

デッドロックに関しては，構造化チャネル[7]やバーチャルチャネル方式[3]を用いることで，解決することができる．

3. トーラスネットワークの分割

n 次元トーラスの分割再構成の問題は，一次元トーラスの場合と本質的に同じである．そこで，最初に一次元の場合について考察する．

図1 に8ノードからなる分割再構成可能なトーラスの例を示す．このデザインの2つの重要な特徴は，

(1) 折り畳んだトーラス（folded torus）で各ノードを接続していること．
(2) 分割再構成のためのスイッチを含んでいること．

である．

3.1 トーラスの折り畳み

トーラスを単純にインプリメントするとメッシュの端と端を繋ぐための長いラップアラウンドパスが必要

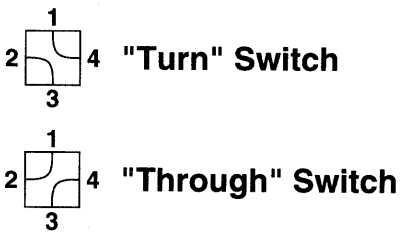


図2 スイッチの2つの状態
Fig. 2 Two kinds of switches.

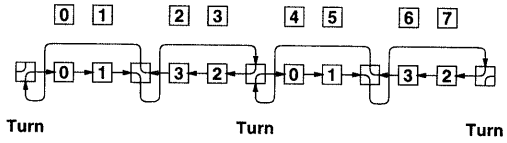


図3 2つのトーラスへの分割
Fig. 3 The 8-processor torus partitioned into two 4-processor tori.

となる．これを避けるにはノードの物理的なレイアウトを折り曲げて，端どうしが近くになるようにしなければならない．

本論文で提案する分割再構成可能なトーラスではネットワークを折り畳むことで長いラップアラウンドパスをなくすとともに，ネットワークにスイッチを配置することで，後で述べるような，エクスプレス・トーラスを実現し，効率的なグローバル演算や放送を可能とする．

3.2 ネットワークの再構成

図1のトーラスでは，**図2** に示されているようなスイッチの2つの状態が使われている．スイッチを切替えることによって，トーラスを分割することが可能である．2つの例を**図3，4** に示す．図3，4で，上の数字は物理的なノードID，下の数字は論理的なノードIDを示している．物理的ノードIDは各ノードに固有の値で，その値は次節で述べるように，動的ルーティングに使われる．論理的ノードIDは分割された時に各ノードに与えられるIDである．

図1，3，4では，分割再構成のためのスイッチを2つのノードにつき1つずつ入れている．この場合，ネットワークは2ノード以下には分割できないので，これを分割の最小単位と呼ぶことにする．分割の最小単位は2次元以上の場合にもあてはまる．

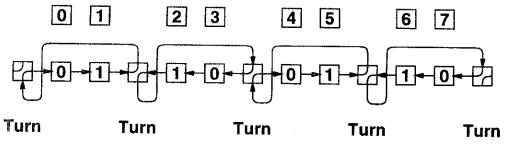


図4 4つのトーラスへの分割
Fig. 4 The 8-processor torus partitioned into four 2-processor tori.

4. 分割再構成可能なトーラスの使用

分割再構成可能なトーラスは次の2つのモードで利用することができる.

(1) 静的モード
(2) 動的モード

4.1 静的モード

静的モードはマルチユーザ環境を実現する簡単でかつ有効な方法である. 静的モードでは各スイッチの状態は固定されている. 例えば, システム構成の変更は管理者によって行なわれ, ユーザはそのシステム構成で計算機を利用する. 具体的には, ある時間は64×64の2次元トーラスとして全システムが1人のユーザによって利用され, またある時間では4つの16×16の2次元トーラスとして分割再構成され, 4人のユーザに利用される. 静的モードでの分割再構成は全スイッチの状態を希望するシステム構成にセットすることで実現される.

静的モードでの問題点は, 分割された部分どうしではメッセージのやりとりができないことである. そのため, 分割された各パーティションごとにホスト(外部)とのインターフェースを持たなければならない.

4.2 動的モード

動的モードでは, 各スイッチはそれぞれのスイッチを通過するメッセージによって状態をダイナミックに変化させる. すなわち, 動的モードではメッセージヘッダーに含まれる宛先ノードの情報を元にして, スイッチを切替える. この時, 宛先ノードには物理的ノードIDを用いる. メッセージがスイッチに到着した時, スイッチの状態は3つの情報, 宛先ノードID *nid*, 両隣の物理ノードID i, jから決定される(**図5**). ノードiの方からメッセージが入ったスイッチは, 例えば, 以下の条件に従ってスイッチを切替える. ただし, スイッチは両隣のノードの物理IDを知っているものとする.

if $nid \geq j$ **then** *through* **else** *turn*

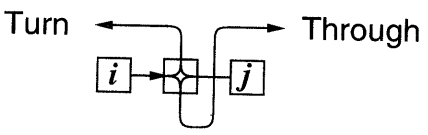


図5 スイッチの状態決定
Fig. 5 Port nomenclature used in describing the dynamic routing algorithm.

このように動的モードでは, 各スイッチはヘッダー情報を解析してスイッチの状態を変化させ, ルーティングを行なう.

4.3 2つのモードの有効利用

これまで述べてきたように, 従来のトーラスにスイッチを追加することで, トーラスの分割再構成が可能となることが分かった. また分割のモードとしては静的モードと動的モードの2つを挙げた. マルチユーザ環境で使用する場合には, 静的モードを利用するのが簡単かつ有効な方法である. 動的モードは静的モードに比べて, ハードウェアが複雑になるが, より柔軟な処理が可能となる. 次節以降では動的モードを利用した, 高速なグローバル演算や放送の可能性について検討する.

5. エクスプレス・トーラス

本論文で提案する分割再構成可能なトーラスネットワークではネットワークの分割再構成の他に, ネットワークに追加したスイッチを利用して, エクスプレス・トーラスを形成することができる. このエクスプレス・トーラスはExpress Cubes[4]と同様に, 遠く離れたノードへのエクスプレス・チャネルを提供する.

図6には2次元トーラスにスイッチを追加した例を示している. この図では2×2の4つのノードが分割の最小単位になっている.

図6のStep 2やStep 3で0番のノード間の距離は通常のルーティングでは2または4であるのが, スイッチを経由することで, 隣接するノードのように通信できる. 通信のレイテンシにはワイアでの遅延とノード内での遅延があるが, 通常ノード遅延の方が大きい[4]. このため, 途中に含まれるノードの少ない通信経路を用いれば遅延の小さい通信が可能である. スイッチを通る経路は途中のノードを通らずに遠くのノードにメッセージを高速に送ることができる. この経路をエクスプレス・トーラスと呼ぶことにする.

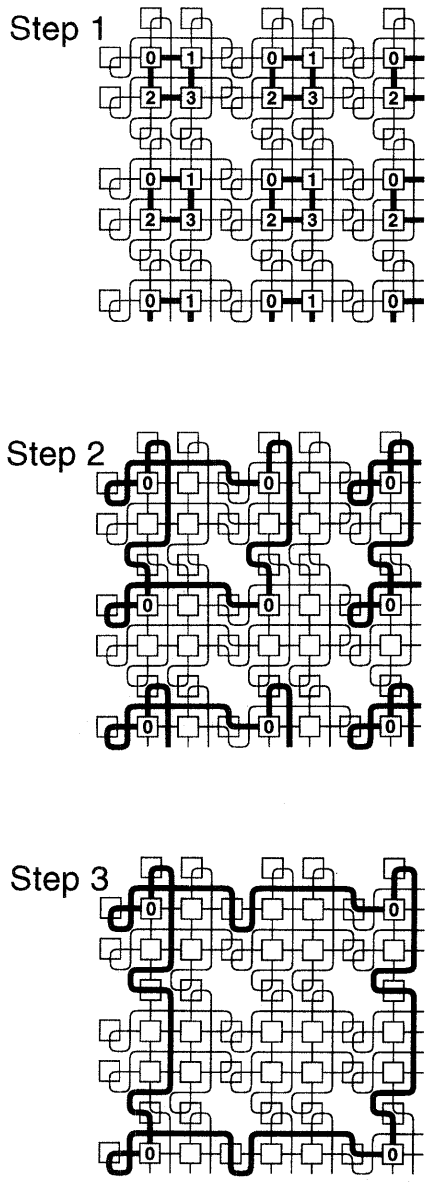


図6 エクスプレス・トーラス
Fig. 6 Express torus.

エクスプレス・トーラスでは遠くのノードとスイッチを経由して，直接繋げるため，システム全体の平均距離は通常のトーラスと比べて小さくなる．すなわち，トーラスネットワークの問題点の内，(c) のノード間の平均距離が長いという問題が解決される．**図7** にエクスプレス・トーラスと通常のトーラスの平均距離

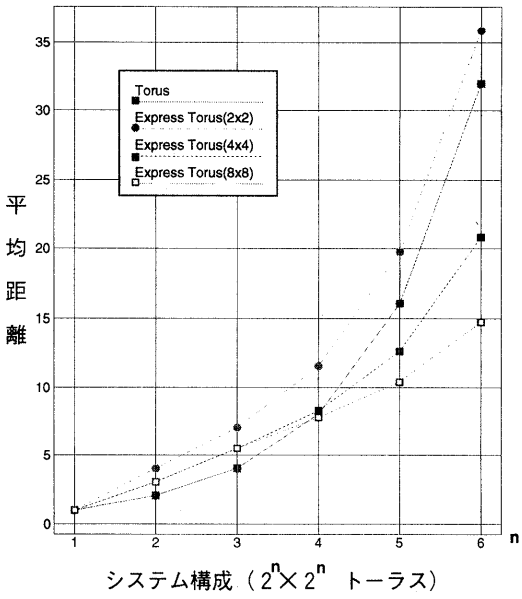


図7 エクスプレス・トーラスの平均距離
Fig. 7 Average distance of express tori.

の比較を示す．トーラスは2次元で，エクスプレス・トーラスは分割の最小単位を 2×2，4×4，8×8 の3つの場合について考えている．

図7に示されているようにエクスプレス・トーラスの平均距離は，分割の最小単位を 2×2 した場合を除いて，通常のトーラスと比べて小さくできる．このため，通常距離が長くて通信遅延時間が大きくなるものを短時間で終らせることが可能となる．

次にグローバル演算と放送についてエクスプレス・トーラスの効果を評価する．

5.1 グローバル演算

グローバル演算は全てのノードで，ある変数の値の総和や最大値，最小値などを求める演算で，並列計算機では良く用いられる計算パターンである．

ここではエクスプレス・トーラスを利用して，グローバル演算を行なう方法を提案する．ここでのグローバル演算の基本的なアルゴリズムは階層的演算である．まず，ネットワーク分割の最小単位（ユニット），図6では 2×2 の4ノード，でグローバル演算を行ない，代表のノード，例えば各ユニットの0番のノード，がその結果を持つ（Step 1）．次に4つのユニットの代表がエクスプレス・トーラスを使ってデータを交換し，大代表がその値を持つ（Step 2）．これを階層の頂上まで繰り返す（Step 3）．計算の結果は，演算の時と

表1 評価に用いる変数
Table 1 Symbols.

T_w	ワイア遅延時間
T_n	ノード遅延時間
T_o	演算時間
T_s	スイッチ遅延時間
T_{uin}	分割最小単位での fan-in の時間
T_u	分割最小単位でのグローバル演算時間
T_t	通常のトーラスでのグローバル演算時間
T_e	エクスプレストーラスでのグローバル演算時間

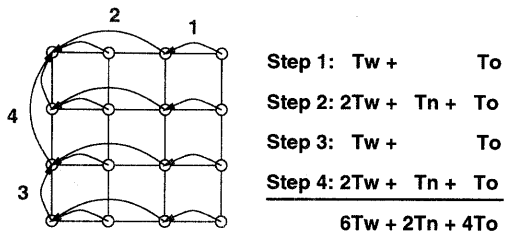


図8 トーラスにおける fan-in
Fig. 8 fan-in on torus.

は逆に，上の階層から下の階層へデータを送り，全てのノードが結果を持つ．このようにして，グローバル演算は実現できる．

5.2 従来の方法との比較

エクスプレス・トーラスによるグローバル演算を通常のトーラスの場合と2次元の場合について比較する．評価に用いる変数を**表1**に示す．

5.2.1 通常のトーラスの場合

図8に示されるように 4×4 のトーラスでは fan-in に $6T_w+2T_n+4T_o$ 時間必要となる．一般に $2^n\times2^n$ のトーラスでは fan-in に次に示すような時間がかかる．

$$2\sum_{i=1}^{n}\{2^{i-1}T_w+(2^{i-1}-1)T_n+T_o\}$$

fan-out は fan-in の逆の順序で行なうとすると，図8の場合，fan-out には $6T_w+2T_n$ 時間必要となる．ただし，fan-out の時は演算時間 T_o はないものとする．一般に通常のトーラスでグローバル演算にかかる時間 T_t は，fan-in，fan-out を合わせて，次の式で表される．

$$T_t=4\sum_{i=1}^{n}\{2^{i-1}T_w+(2^{i-1}-1)T_n+\frac{1}{2}T_o\}$$

表2 比較する3つの場合の T_u
Table 2 Three cases of T_u.

ユニット	T_u
$2\times2=4$	$4T_w+2T_o$
$4\times4=16$	$12T_w+4T_n+4T_o$
$8\times8=64$	$28T_w+16T_n+6T_o$

5.2.2 エクスプレス・トーラスの場合

エクスプレス・トーラスでグローバル演算を行なう場合には，まずユニット内で通常のトーラスと同じように fan-in を行ない，その後で代表のノードがエクスプレス・トーラスを使って通信および演算を行なう．例えば，ユニットを4つ（2×2）使ったシステムでは fan-in にはユニット内での fan-in 時間 T_{uin} に加えて $6T_w+3T_s+2T_o$ 時間かかる．一般に $2^N\times2^N$ のユニットを使った場合には fan-in に次に示すような時間がかかる．

$$T_{uin}+2\sum_{i=1}^{N}\{(2^{i-1}+2)T_w+(2^{i-1}+1)T_s+T_o\}$$

これよりエクスプレス・トーラスを用いたグローバル演算にかかる時間 T_e は次の式で与えられる．

$$T_e=T_u+4\sum_{i=1}^{N}\{(2^{i-1}+2)T_w+(2^{i-1}+1)T_s+\frac{1}{2}T_o\}$$

5.2.3 評 価

ユニットに含まれるノードの数を変えて，通常のトーラスの場合とグローバル演算にかかる時間を比較する．ここで仮定として $T_w=T_n=T_o=T_s=1$ とする．トーラスは2次元で，エクスプレス・トーラスは分割の最小単位を 2×2，4×4，8×8 の3つの場合について考える．それぞれの場合のユニットでのグローバル演算の時間 T_u は**表2**に示す通りである．

この結果を**図9**に示す．図9に示されているようにエクスプレス・トーラスを使ったグローバル演算は通常のトーラスを用いた時より短い時間で終了することがわかる．ユニットに含まれるノードの数を多くすれば，大きなトーラスでのグローバル演算が高速に行なえ，追加するスイッチの数が減るのでハードウェアのコストも小さくなる．ただし，この場合，分割の最小単位が大きくなるので，分割可能なパーティションの数が少なくなる．

ここでは2次元のトーラスの場合について評価を行なったが，この結果は一般に n 次元の場合にもあてはまる．n 次元トーラスの場合のグローバル演算にかか

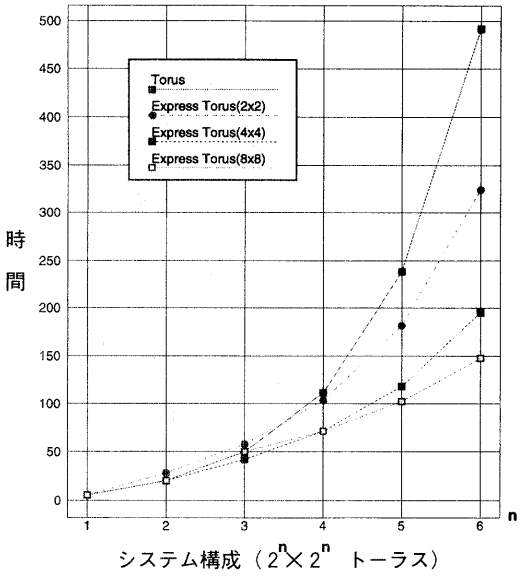


図9 グローバル演算による
エクスプレス・トーラスの評価
Fig. 9 Evaluation of express tori
by global reduction.

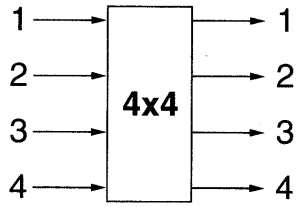


図10 スイッチの構成
Fig. 10 Configuration of network switch
in the dynamic mode.

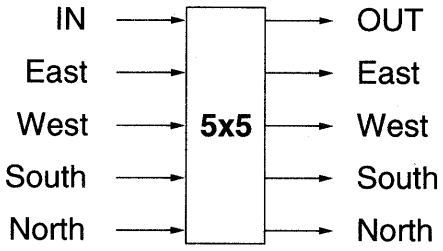


図11 ルーティング・コントローラの構成
Fig. 11 Configuration of routing controller.

る時間は，1次元の場合の n 倍なので，通常のトーラスとエクスプレス・トーラスでグローバル演算にかかる時間の相対的な関係は変化がない．言い替えると，図9のグラフを縦方向に伸び縮みさせるだけである．また，ここでは，$T_w = T_n = T_o = T_s = 1$ としたが，これらの値をそれぞれ現実的な範囲で変化させてもエクスプレス・トーラスの優位性は変わらない．

このようにエクスプレス・トーラスでは，一般に n 次元の場合について通常のトーラスよりもグローバル演算を高速に実行することが可能である．

5.3 放　送

放送はグローバル演算の fan-out と同様に実現できる．放送には以下の手順が必要となる．

(1) 放送を要求したノードの含まれるユニットでの放送
(2) ユニットの代表間の階層的データ分配
(3) 1以外のユニットでの放送

1で放送を要求したノードがユニットの代表でない場合は，データが代表まで伝わってから，ユニット間でデータの階層的分配を行なうことにする．ユニット内での放送は通常のトーラスの場合と同様に行なう．エクスプレス・トーラスによって階層的にデータを分配することができるので，グローバル演算の場合と同様に，通常のトーラスの放送より高速に放送を実現できる．

6. ハードウェアの構成

トーラスに簡単なスイッチを追加することで，トーラスが分割再構成可能となるばかりか，高速なグローバル演算や放送が可能となることがわかった．ここでは，追加するハードウェアの構成について検討する．

分割再構成を行なうためにトーラスに追加するスイッチは**図10**に示すような4入力4出力のスイッチになる．しかし，このためのスイッチを新たに開発する必要はない．トーラスネットワークの各格子点にはメッセージのルーティングをコントロールするためのスイッチ（ルーティング・コントローラ）が使われるが，このスイッチは2次元トーラスの場合，**図11**に示すように，東西南北の4方向とCPUへのIn/Outを合わせて，5入力5出力，3次元トーラスの場合は上下方向を合わせて7入力7出力である．ルーティング・コントローラと分割再構成のために追加するスイッチは機能的に同じものなので，共用することが可能で素子開発のコストが削減できる．

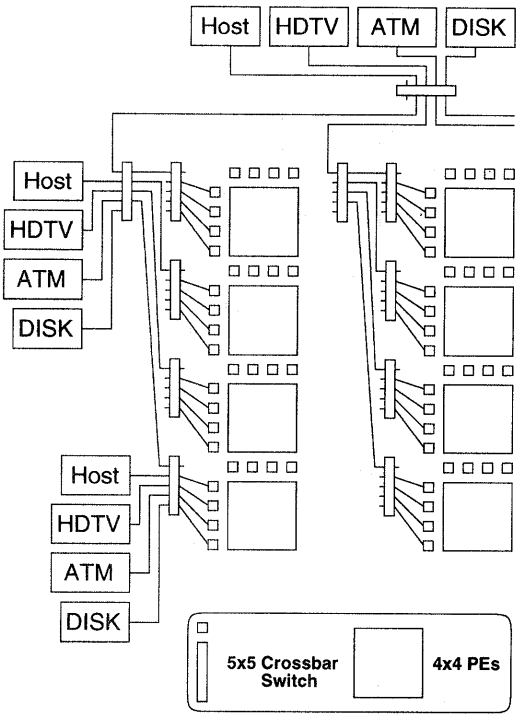


図 12 外部とのインターフェース
Fig. 12 Interface to peripherals.

7. ホストインターフェース

次にネットワークを分割するためのスイッチにルーティングコントローラを利用した場合の外部とのインターフェースについて考察する．前に述べたように，ルーティング・コントローラには2次元トーラスの場合，東西南北とCPUの5入力5出力，3次元トーラスの場合，東西南北と上下とCPUの7入力7出力のスイッチをそれぞれ利用している．ネットワークを分割するためのスイッチは，2次元トーラスの場合，4入力4出力，3次元トーラスの場合，6入力6出力必要であるので，ルーティング・コントローラを利用するとスイッチは2次元の場合も3次元の場合も1入力1出力余る．これを利用して，外部とのインターフェースを行なうことを考える．**図 12** にホストコンピュータや外部装置を含めた2次元の場合の全体構成の例を示す．

図12では $4\times4=16$ 個のノードを分割の最小単位としている．これは，図12の大きな正方形として示

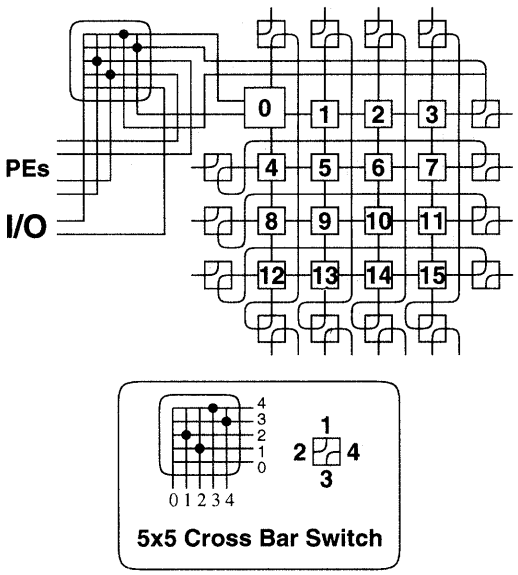


図 13 スイッチの接続例
Fig. 13 Example of switch connection.

されている．また小さな正方形と長方形で示されているのは 5×5 のクロスバースイッチで，小さな正方形は単方向，長方形は双方向とする．システム全体としては4分木を構成している．ホスト計算機やその他の外部装置はI/O用のスイッチの内，余っている端子のどこにでも接続できる．I/Oのための双方向のクロスバースイッチは単方向のものを2つ合わせればいいので，ルーティング・コントローラのためのスイッチと分割のためのネットワークに入れたスイッチとI/Oのためのスイッチは全て同じもので，新たな開発コストを必要としない．

図 13 に1つのクロスバースイッチの内部を拡大したものを示す．

ホスト計算機や他の外部装置は各パーティションに1つずつ繋ぐこともできるし，4つのパーティションに1つ，また16個のパーティションに1つなど，自由に変更できる．

ホスト計算機は複数のパーティションを制御できるので，分割できるパーティションの数はホスト計算機の数に制限されない．

8. ま と め

トーラスネットワークに簡単なスイッチを追加することで，ネットワークを分割再構成する方法を提案し

た．これによって，トーラスネットワークにおける分割によるマルチユーザ環境が可能であることを示した．また，ネットワークに追加したスイッチを動的に切替えることで，高速なグローバル演算が可能であることも示した．そして，このスイッチを従来のトーラスネットワークに追加するためのハードウェア・コストについて検討した．

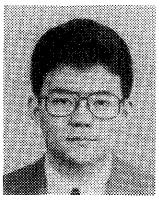
最後に，ネットワークを分割した時に複数のパーティションと複数のホスト計算機やディスク装置などとどのように繋ぐか，という外部とのインターフェースについて検討した．

謝辞 参考文献をお送り頂いた（株）東芝の田辺昇氏，慶応大学の天野 英晴先生，また JSPP での発表に対して貴重な御意見をいただいた皆さんに感謝いたします．さらに，日頃御指導，御助言いただく，並列処理研究センター石井センター長，白石担当部長，池坂主任研究員，研究室の同僚諸氏に感謝いたします．

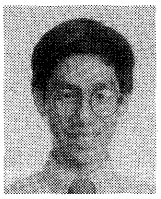
参 考 文 献

1) Chen, W., Liu, C. and Fang, M.: A Massively Parallel Processing Unit with a Reconfigurable Bus System RIPU, *International Conference on Parallel Processing*, pp. I. 431–434 (1991).
2) Dally, W. J.: Performance Analysis of k-ary n-cube Interconnection Network, *IEEE Transactions on Computers*, Vol. 39, No. 6, pp. 775–785 (1990).
3) Dally, W. J.: Virtual-Channel Flow Control, *17th International Symposium on Computer Architecture*, pp. 60–68 (1990).
4) Dally, W. J.: Express Cubes: Improving the Performance of k-ary n-cube Interconnection Networks, *IEEE Transactions on Computers*, Vol. 40, No. 9, pp. 1016–1023 (1991).
5) Efe, K.: Reconfigurable Cube Architecture for Parallel Computation, *The 12th International Conference on Distributed Computing Systems*, IEEE, pp. 218–225 (1992).
6) ElGindy, H. and Wegrowicz, P.: Selection on the Reconfigurable Mesh, *International Conference on Parallel Processing*, pp. III. 26–33 (1991).
7) Horie, T., Ishihata, H. and Ikesaka, M.: Design and Implementation of an Interconnection Network for the AP1000, *Information Processing 92, Volume I*, pp. 555–561 (1992).
8) Jenq, J. F. and Sahni, S.: Reconfigurable Mesh Algorithms for the Hough transform, *International Conference on Parallel Processing*, pp. III. 34–41 (1991).
9) 松山隆司，青山正人：再帰トーラス結合アーキテクチャ，電子情報通信学会技術研究報告，Vol. CPSY91-4-33, pp. 49–58 (1991).
10) Miller, R., Kumar, V. K. P., Reisis, D. I. and Stout, Q. F.: Data Movement Operations and Applications on Reconfigurable VLSI Array, *International Conference on Parallel Processing*, pp. I. 205–208 (1988).
11) 中村定雄：プロセッサ間結合方式，東芝技術公開集，Vol. Vol. 7-22, No. 89-5926, pp. 49–56 (1989).
12) Wang, B. F., Chen, G. H. and Li, H.: Configurational Computation: A New Computation Method on Processor Arrays with Reconfigurable Bus System, *International Conference on Parallel Processing*, pp. III. 42–49 (1991).

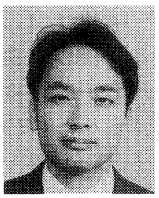
（平成 5 年 9 月 7 日受付）
（平成 5 年 11 月 11 日採録）



林 憲一（正会員）
1967 年生．1991 年 東京大学工学部計数工学科卒業．同年（株）富士通研究所入社．現在並列処理研究センターにて，並列計算機アーキテクチャの研究に従事．



Isaac Chuang
1968 年生．1990 年 マサチューセッツ工科大学卒業．1991 年 同大学院電気工学修了．1992 年 2 月より 9 月まで（株）富士通研究所にて並列計算機アーキテクチャの研究に従事．現在スタンフォード大学大学院博士課程に在学中．



堀江 健志（正会員）
1962 年生．1984 年 東京大学工学部電気工学科卒業．1986 年 同大学院修士課程修了．同年（株）富士通研究所入社，現在に至る．並列計算機に関する研究開発に従事．1992 年 電子情報通信学会論文賞受賞．