情報処理学会論文誌　コンピューティングシステム　Vol. 1　No. 2　70–82 (Aug. 2008)

ツリー型オンチップネットワークにおける適応的アクティベーション制御

松谷宏紀[†1]　鯉渕道紘[†2,†3]
王代涵[†1]　天野英晴[†1]

Network-on-Chip（NoC）のスタンバイ電力を減らす方法として，オンチップルータのランタイムパワーゲーティングが注目されている．本論文では NoC において広く利用されているツリー型トポロジを対象に，1) トラフィック負荷に応じて適応的にルータチャネルを On/Off させるスリープ制御と，2) ウェイクアップ遅延に起因する性能劣化を緩和するためのルータアーキテクチャを提案する．このルータアーキテクチャでは，物理的に近隣に配置され，異なるツリーに属するルータ間にバイパスリンクを設ける．ウェイクアップ遅延などによって一方のツリーで混雑が生じた場合，このバイパスリンクを用いてトラフィックをもう一方のツリーへと横流しする．評価の結果，横流し機構を持ったツリーはたかだか 5.2% の面積オーバヘッドで通常のツリーより高いスループットを実現し，リーク電力の削減量でも有利となった．

An Adaptive Activation Scheme for On-chip Tree-based Networks

Hiroki Matsutani,[†1] Michihiro Koibuchi,[†2,†3]
Daihan Wang[†1] and Hideharu Amano[†1]

The run-time power gating of individual channels in a router is one of attractive solutions to reduce the standby power of Network-on-Chips. Assuming on-chip tree-based networks, 1) we propose a sleep control method that adaptively activates the router channels in response to the traffic load, and 2) we propose two router architectures that can cope with the traffic congestions induced by the wakeup delay of sleeping channels. In these architectures, bypass links are newly introduced between routers, each of which belongs to different trees but is placed very closely, in order to divert a traffic from a congested tree to an uncongested one. Evaluation results show that one of the proposed architectures outperforms an original tree in terms of peak performance and leakage power reduction, though its area overhead is at most 5.2%.

1. はじめに

半導体技術の進歩によって単一チップ上にプロセッサやメモリ，I/O など複数の設計モジュールをタイル状に実装することが可能になり，このようなタイルどうしの結合手法として Network-on-Chip（NoC）が注目されている[1)–3)]．

タイルアーキテクチャの主要なアプリケーションは携帯機器や情報家電などの組み込み機器であり，バッテリ寿命を延ばすため，また，パッケージングや放熱にかかるコスト削減のため，消費電力の削減が重要である．消費電力はスイッチング電力とリーク電力に大別され，スイッチング電力の削減にはクロックゲーティングやオペランドアイソレーションなどの手法が代表的であり，広く利用されている．一方，リーク電力に関しては，回路が動作していなくても電源が供給されている限り電力を消費してしまううえに，半導体技術の微細化にともない，全消費電力に占める割合が近年著しく増加している．このような状況を背景に，とりわけ，プロセッサコアにおけるリーク電力の削減手法の研究が最近さかんであり，パワーゲーティングなどの手法を用いることで文献 4) では携帯電話向けプロセッサのリーク電流を 11 μA まで抑えることに成功している．このようにプロセッサコアの省電力対策が進むほど，ルータのリーク電力が相対的に大きくなってくる．したがって，コアのパワーゲーティングが行われるような環境ではルータのパワーゲーティングも同様に必須である．

そこで，NoC において一般的に使われているツリー型トポロジを対象に，オンチップルータの物理チャネル単位のパワーゲーティングについて検討する．このようなネットワークでは，スリープ中のチャネルが利用可能になるまで一定の遅延がかかるため，このウェイクアップ遅延によってパケットが経路上のネットワーク資源を占有したまま停止することがあり，後続パケットの通信に影響が生じる．本論文では，このような Head-of-Line（HoL）ブロッキングの影響を軽減するためのルータアーキテクチャを提案する．

本論文の構成は次のとおりである．まず，2 章でオンチップルータの消費電力を解析し，ルータチャネルのランタイムパワーゲーティングが有効なことを示す．3 章ではランタイム

†1 慶應義塾大学大学院理工学研究科
Graduate School of Science and Technology, Keio University
†2 国立情報学研究所
National Institute of Informatics
†3 総合研究大学院大学
The Graduate University for Advanced Studies

© 2008 Information Processing Society of Japan

パワーゲーティングのための技術を紹介し，4 章で Fat Tree を対象とした，適応的パワーゲーティングのためのルータアーキテクチャを提案する．5 章で評価を示し，6 章で本論文をまとめる．

2. オンチップルータ

本章では，文献 5) と同様の方法を用いてオンチップルータの消費電力を解析する．

2.1 アーキテクチャ

消費電力の解析のために，文献 6) のワームホールルータをもとに典型的な仮想チャネルルータ回路を 90 nm スタンダードセルライブラリを用いて実装した．

図 1 に実装したルータ回路の概要を示す．このルータはクロスバ，アービタ回路，ポート数分の物理チャネルから構成される．物理チャネルごとに 2 本の仮想チャネルを持ち，仮想チャネルごとに 4-flit 分の入力バッファを持つ．バッファの深さが 4-flit 分と浅いため，入力バッファはメモリマクロではなく，フリップフロップを用いて実装した．物理チャネル数を 5 本，仮想チャネル数を 2 本としたとき，ルータ面積の約 64% を入力バッファが占める結果となった．ルータのデータ（フリット）幅は 64-bit とした．

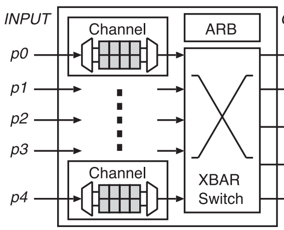
このルータは 3 段のパイプラインステージから構成される．入力されたヘッダフリットは，1) routing computation（RC）ステージで出力ポートを計算，2) virtual channel and switch allocation（VSA）ステージで仮想チャネルが割り当てられ，出力ポートの調停を行い，3) switch traversal（ST）ステージで隣接ルータへ転送される．各ステージの機能は文献 7) のルータと同じである．

2.2 消費電力の解析

上記のルータの消費電力を求めるために，1) Design Compiler でルータ回路を合成，2) Astro で配置配線（CTS でバッファを挿入），3) Verilog-XL で配置配線後シミュレーションを行い Switching Activity Interchange Format（SAIF）を生成し，4) Power Compiler でこの SAIF をもとに消費電力を解析した．なお，クロックゲーティングやオペランドアイソレーションなどの省電力対策は十分に施してある．

手順 3) では一定のトラフィック負荷を与え，200～500 MHz で配置配線後シミュレーションを行った．文献 5) にならい，リンクの最大帯域の 30% の負荷になるように一定間隔で注入されるパケットの流れを **stream** と呼ぶ．ここでは，n 個のポートを持つルータに対し，0 から n 本の stream を与えたときの消費電力を測定した．ただし，$3 \le n \le 6$ とする．

仮想チャネルを 2 本持たせたルータを 200 MHz で動作させたときの消費電力を**図 2** に示す．当然，負荷が高くなるに従い消費電力が増えるが，スタンバイ時（stream 数が 0 のとき）においても一定の電力が消費されている．**図 3** にスタンバイ電力の内訳を示す．スタンバイ電力の 53.8%～55.8% を入力チャネルとそのバッファのリーク電力が占めている．



Router parameters	
Process	ASPLA 90nm
Voltage	1.0V (25 C)
Speed	Max 500 MHz
Flit size	64-bit
# of ports	3 - 8 (configurable)
# of VCs	2
Buf size	4-flit for each VC
Packet len	4-flit data + 1-flit header

図 1　ルータの概略図と実装パラメータ
Fig. 1　Router overview and implementation parameters.

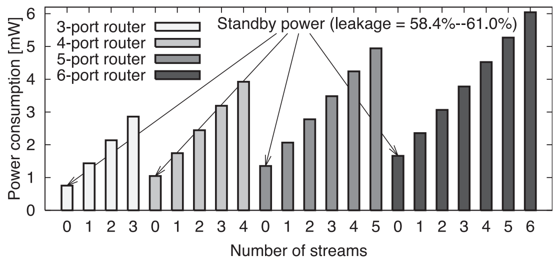


図 2　負荷を与えたときのルータの消費電力（200 MHz 動作）
Fig. 2　Power consumption at various workloads (200 MHz).

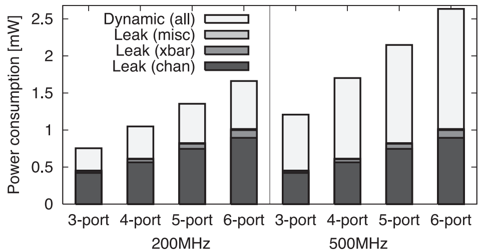


図 3　スタンバイ電力の内訳（200 MHz/500 MHz 動作）
Fig. 3　Breakdowns of the standby power (200 MHz/500 MHz).

© 2008 Information Processing Society of Japan

残りのスイッチング電力に関してはクロックツリーバッファによるものであるため，動的電力のこれ以上の削減は難しい．

1 章で述べたとおり，コアのパワーゲーティングが必要となるような環境ではルータのパワーゲーティングも同様に求められる．ルータのパワーゲーティングには，1) ルータ単位のパワーゲーティングと，2) 物理チャネル単位のパワーゲーティングの 2 種類の粒度が考えられる．2) の場合，クロスバスイッチにはつねに電力が供給され，通電されるチャネルと非通電のチャネルが混在することになる．一方，1) の場合はクロスバスイッチもパワーゲーティングの対象になるが，図 3 で示したようにクロスバのリーク電力は物理チャネルに比べてかなり小さい．しかも，1) ではすべての物理チャネルにパケットが存在しないときしか，ルータへの通電を止めることができず，パワーゲーティングの機会が制限されてしまう．以上より，オンチップルータの各物理チャネルを動作時に動的にパワーゲーティングする手法が NoC のスタンバイ電力を減らすために有効と考えられる．次章ではランタイムパワーゲーティングについて調査する．

3.　省電力技術のサーベイ

3.1　パワーゲーティング

パワーゲーティングでは，スリープ対象の回路と Vdd ラインの間，もしくは，スリープ対象の回路と GND ラインの間にスリープトランジスタを挿入し，このスリープトランジスタを ON/OFF することで回路への電力供給を制御する．通常，何らかの方法でパワーゲーティング対象の回路のアイドル状態を検出し，その回路がアイドル状態の間だけ電源供給を遮断することでリーク電力を減らす．文献 8) で報告されているように，パワーゲーティングの単位ごとにパワードメインを設け，非通電のパワードメインから他のパワードメインに不定値が伝搬しないようパワードメイン間の I/O に isolation cell を挿入する必要がある．

パワーゲーティング対象の回路の粒度は様々である．たとえば，文献 4) では携帯電話用プロセッサをコア単位でパワーゲーティングしている．これより粒度の細かい例として，文献 9) ではマイクロプロセッサの ALU や FPU などの演算器レベルでパワーゲーティングを行っている．本研究の目的はオンチップルータにおける物理チャネル単位のパワーゲーティングであるため，粒度的に最も近いと考えられる演算器レベルでのパワーゲーティングに着目することにする．

パワーゲーティングでは，リーク電力削減の見返りにコストを払うことになる．実際，パワーゲーティングのためにパワースイッチを挿入する必要があり回路面積が増大する[10]．また，スリープ中の回路をウェイクアップさせるには一定の遅延がかかるため性能に影響が出る．たとえば，文献 3) では FPMAC にパワーゲーティングを適用しており，電流スパイクを抑えるためウェイクアップ処理に 6 サイクルを要している．ウェイクアップ遅延に加え，スリープおよびウェイクアップ操作のためにエネルギーが消費されるため，短期間のパワーゲーティングは逆に消費電力の増加を招くことが知られている[9]．

文献 9) では，演算器レベルのパワーゲーティングのコストモデルを提案している．このモデルでは，次の 3 つのパラメータがパワーゲーティングの性能や消費電力に影響を与える．

- T_{wakeup}：スリープ中の回路が再び使われることを検出し，回路への電源供給を再開，実際に動作可能になるまでのサイクル数．
- $T_{idledetect}$：動作中の回路が今後しばらく使われないことを検出し，パワースイッチを操作して回路への電源供給を止めるまでのサイクル数．
- $T_{breakeven}$：パワーゲーティングによるリーク電力の削減量が，パワースイッチ操作に要すオーバヘッドエネルギーより大きくなる最小のサイクル数（**損益分岐点**）．

スリープ中の回路にデータが来ると T_{wakeup} サイクルだけ待たされる（パイプラインストール）ので T_{wakeup} は性能に影響を与える．また，$T_{idledetect}$ が長いと，スリープするかどうかの判断に時間がかかり実際にスリープできる期間が減る．

$T_{breakeven}$ がパワーゲーティング適用の損益分岐点となる．パワーゲーティングできる期間が $T_{breakeven}$ より長ければパワーゲーティングによって消費電力が減るが，$T_{breakeven}$ より短ければパワーゲーティングによって電力が増えてしまう．文献 9) では，この $T_{breakeven}$ を求める計算式を提案しており，典型的なマイクロプロセッサの $T_{breakeven}$ は 10 サイクル前後であると報告している．本論文でも，この式を用いてオンチップルータの $T_{breakeven}$ を計算している（5.4 節）．

3.2　ルータ/リンクの On/Off 制御

文献 11) では Drowsy キャッシュを用いてルータの入力バッファのリーク電力を削減している．文献 12) では，動的にリンクの電源をシャットダウンする際のルーティングアルゴリズムについて検討している．ただし，これらの研究ではキャッシュメモリのような比較的大きなバッファをルータに持たせることを想定している．もしくは，文献 13) のようにいかなる場合にも電源を Off しないリンクをあらかじめ用意しておく．一方，文献 6) で用いられるような NoC 向けの小規模なワームホールルータの場合，パワーゲーティングの際のウェイクアップ時間が通信性能に直に影響を及ぼす可能性がある．

© 2008 Information Processing Society of Japan

4. On/Off Fat Tree アーキテクチャ

NoC のネットワークトポロジとして，メッシュやトーラスなどのグリッド型トポロジのほかに，ツリー型トポロジが広範囲に利用されている．最も一般的かつ実用的なツリー型トポロジは Fat Tree である．本論文では Fat Tree ベースの NoC を対象に，ルータの物理チャネル単位の動的なパワーゲーティングについて検討し，ウェイクアップ遅延に起因する HoL ブロッキングを隠蔽するルータアーキテクチャを提案する．

4.1 節で Fat Tree に関する用語を説明する．次に Fat Tree トポロジにおけるルータチャネルの段階的なパワーゲーティングについて検討し，スループットとパケット遅延を維持しつつ行うウェイクアップ制御の難しさを指摘する．この問題を解決するためのルータアーキテクチャを 4.4 節で提案する．

4.1 Fat Tree トポロジ

図 4 に代表的な Fat Tree トポロジとその 2 次元レイアウトを示す．図中の四角はルータ，丸は計算コアをそれぞれ表す．

定義 1　Fat Tree (p, q, c) は，各ルータ（最上位ランクを除く）の上向きリンク数が p 本，下向きリンク数が q 本，計算コアのポート数が c 本の Fat Tree である．

たとえば，図 4 (d) の各ルータは 2 本の上位リンク，4 本の下位リンク，各コアは 2 本のポートを持つため，これは Fat Tree (2,4,2) となる．このような $q = 4$ のツリーが $2^n \times 2^n$ 個のコアを持つとき，ツリーの階層数は $\log_4(4^n) = n$ となる．

丸で描かれた計算コアには 2 次元座標 (x_{2D}, y_{2D}) が割り当てられる．以降，計算コアを rank 0 ルータと呼ぶ．

定義 2　rank 0 ルータに対し次式で示す **tree 座標** $T(r_0, r_1, ..., r_{n-1})$ を割り当てる．

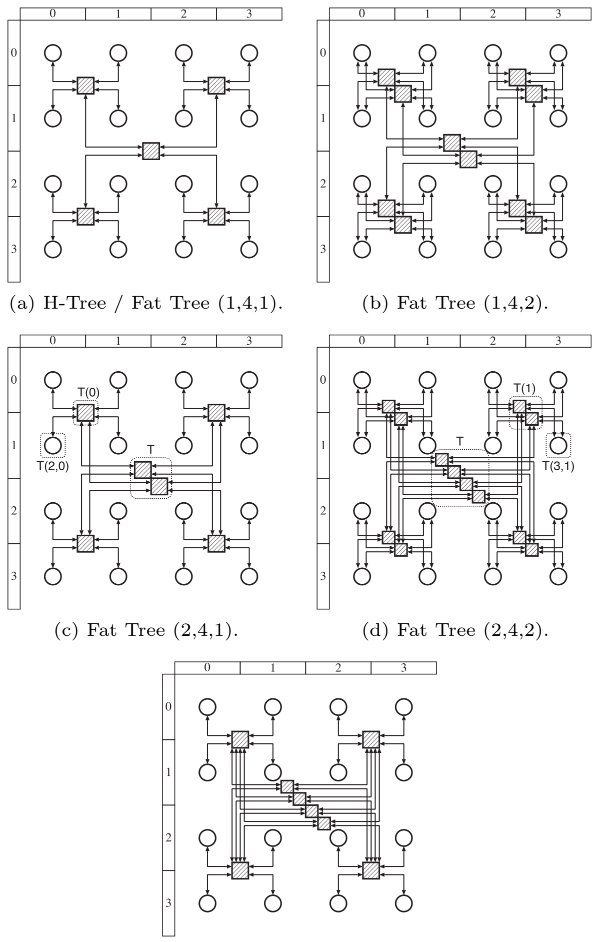
$$r_i = ((x_{2D}/2^i) \bmod 2) + 2 \times ((y_{2D}/2^i) \bmod 2) \qquad (1)$$

さらに，tree 座標 $T(r_0, r_1, ..., r_{n-1})$ を持つ rank 0 ルータの上位ルータを rank 1 ルータと呼び，そのすべてに $T(r_1, ..., r_{n-1})$ の tree 座標を割り当てる．同様に rank 2 ルータから rank n ルータにも tree 座標を割り振る．ただし，最上位ランク（rank n）のルータの tree 座標は T とする．

図 4 (c) と 図 4 (d) にいくつかの tree 座標の例を示す．図 4 (d) の $T(1)$ のように複数のルータが同じ tree 座標を共有する場合もある．

定義 3　同じ tree 座標を共有する 1 個以上のルータの集合を **brothers** と呼ぶ．

たとえば，図 4 (d) の T と $T(1)$ はそれぞれ 4 個，2 個のルータを持つ brothers である．



(a) H-Tree / Fat Tree (1,4,1).　(b) Fat Tree (1,4,2).
(c) Fat Tree (2,4,1).　(d) Fat Tree (2,4,2).
(e) Fat Tree (4,4,1).

図 4　様々な Fat Tree トポロジ（16 コア）
Fig. 4　Various Fat Tree topologies (16-core).

Fat Tree を 図 4 のように 2 次元レイアウトした場合，brothers 内のルータどうしは隣接して配置されることになる．

© 2008 Information Processing Society of Japan

4.2　Fat Tree におけるパワーゲーティング

Fat Tree は変数 p，q，c に応じてツリーを多重化したトポロジである．たとえば，Fat Tree (2,4,2) は (2,4,1) の 2 倍の数の最上位リンクを持つが，ルータの個数も 2 倍となり，それに従いスタンバイ電力も増える．つまり，多重度の高い Fat Tree を採用することはスタンバイ電力の増加というデメリットをかかえることになる．

一方，Fat Tree において，ルータ内のいくつかのチャネルを停止させることでツリーの多重度を縮退化できる．たとえば，Fat Tree (2,4,2) は (2,4,1) へ，Fat Tree (2,4,1) は (1,4,1) へ縮退させることができる．つまり，Fat Tree の各ルータチャネルに対しパワーゲーティングを施し，負荷に応じてツリーの多重度を変えていくことで，リーク電力はネットワークトポロジではなく，実際の負荷に応じて決まるようになる．

ルータチャネルのパワーゲーティングでは，パケット転送処理が完了して入力チャネルが空になった場合に，スリープモードへ移行するものとする．

ここで，スリープ中のチャネルをトラフィック負荷に応じてウェイクアップさせる方法として，多重化したツリーの特徴を利用した conservative と aggressive の 2 種類を提案する．

- **conservative：** ルータの output selection function（OSF）[⋆1]として，空いている出力ポートのうちチャネル番号が一番小さいものを選ぶ．
- **aggressive：** あるルータの入力ポートが混雑してきた場合，brothers 内の隣接ルータの同じ位置の入力ポートをウェイクアップさせる．OSF はランダムなどでトラフィックを分散させるようにする．

conservative では，負荷が小さい場合，つねに同じ出力ポートが選ばれるのでアクティブなツリーの多重度を小さく保つことができる．一方，負荷が高くなると残りの出力ポートも使われるようになる．ただし，このような OSF はトラフィックの分散能力に乏しく，性能が落ちる可能性がある．

aggressive では，混雑状況に応じて brothers 内の他のルータの入力ポートを投機的にウェイクアップさせておき，手前のルータにどちらかをランダムに選択させる．トラフィックの分散能力は conservative より優れるが，投機的にウェイクアップさせたチャネルが実際には使われないケースが生じる．

⋆1 Fat Tree では目的地への経路が複数存在する場合がある．このような場合に利用可能な出力ポート群から 1 つを選択する機能．

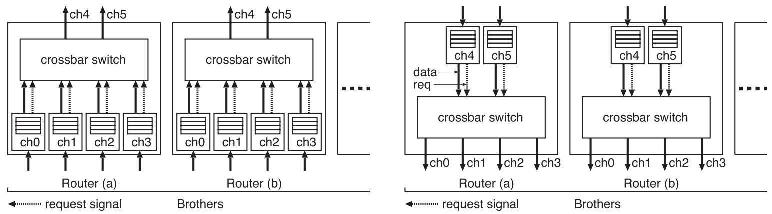
4.3　ウェイクアップ遅延の影響

3.1 節で述べたように，スリープ中の回路をウェイクアップさせるには数サイクルのウェイクアップ遅延がかかる．ルータチャネルのパワーゲーティングの場合，スリープ中のチャネルにパケットが到達すると，そのパケットはチャネルが利用可能になるまで 1 つ手前のルータの出力ポートで待たされる．

Fat Tree では，送信元から least common ancestor（LCA）[⋆2] へのパケット転送では複数の出力ポートの中から 1 つを選択できる．そのため，他にアクティブなツリーがあればスリープ中のツリーを使わないで済む．一方，LCA から宛先への下方向の経路は 1 つだけであるため，前方のルータの入力チャネルがスリープ中であれば，ウェイクアップするまで必ず待たなければならない．

図 5 は通常のルータにおける上方向と下方向へのパケット転送を表している．たとえば，図 5 (b) において，Router (a) の ch4 から入力されたパケットを ch2 に転送したいとする．このとき ch2 の前方のルータの入力チャネルがスリープ中であると，パケットは Router (a) の ch4 のバッファを占有したままストールしてしまう．とくにワームホールスイッチングの場合，ストールしているパケットによって他のパケットの進行が阻害されるため，性能への影響が拡大する．また，パケットが格納されているチャネルの電源を落とすことはできないため，ストールによってリーク電力削減の機会も減る．

そこで，我々は look-ahead ルーティングを応用して，ウェイクアップ遅延を削減する方法を提案した[6]．この方法では 2-hop 手前のルータからのウェイクアップ信号を監視するため 2-hop にまたがる長い制御線を用いる．メッシュのように 1-hop の距離が均等かつ短い



(a) 上方向へのデータ転送　(b) 下方向へのデータ転送

図 5　オリジナルルータ（Orig）
Fig. 5　Original router (Orig).

⋆2 送信元ノードと宛先ノードの共通の祖先となるノードのうち，最も下位にあるもの．

© 2008 Information Processing Society of Japan

場合には有効であると考えられるが，ツリーのように長いリンクを含むトポロジではさらに長いリンクを生じてしまう．近年は配線遅延が大きな問題となっているため，このような解決策はツリーには適さない．

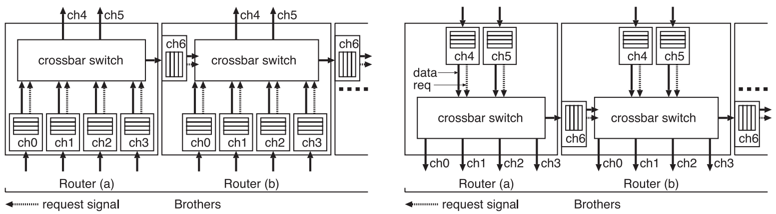
4.4　Brothers 内のデータ横流し機構

ウェイクアップ遅延に起因する HoL ブロッキングを軽減させるために，我々は brothers 内でデータの横流しを行う方式を提案する．そのための機構として，buffered bypass port（BPort）と bufferless bypass port（LPort）を順に説明する．

4.4.1　Buffered Bypass Port 付きルータ

図 6 に示した buffered bypass port 付きルータでは，同じ brothers 内のルータ間でデータの横流しができるように入力チャネルを増やす（図 6 の ch6）．たとえば，図 6 (b) において，Router (a) の ch4 から入力されたパケットを ch2 に転送したいとする．このとき ch2 の前方のルータの入力チャネルがスリープ中であると，パケットを ch4 で待たせるのではなく，Router (b) の ch6 を経由して，Router (b) の ch2 の利用を試みる．これを**横流し**と呼ぶ．Router (b) の ch2 も利用できない場合，さらなる隣接ルータへ横流しすることもできる．これによってストールしたパケットが Router (a) の ch4 およびその上位ルータを占有し続けることを回避でき，ウェイクアップ遅延による HoL ブロッキングを緩和できる．

しかし，この方法には 2 つの欠点がある．1) buffered bypass port を追加するために面積が増える．2) buffered bypass port を使ったパケットの横流しでは，横流しのたびに，パケットの bypass port バッファへの read/write が発生し，電力を消費する．このようなスイッチング電力はリーク電力に比べて大きいため，buffered bypass port を用いた横流しによって，トータルの消費電力が逆に増加する可能性がある．



(a) 上方向へのデータ転送　(b) 下方向へのデータ転送

図 6　Buffered bypass port 付きルータ（BPort）
Fig. 6　Router with buffered bypass port (BPort).

4.4.2　Bufferless Bypass Port 付きルータ

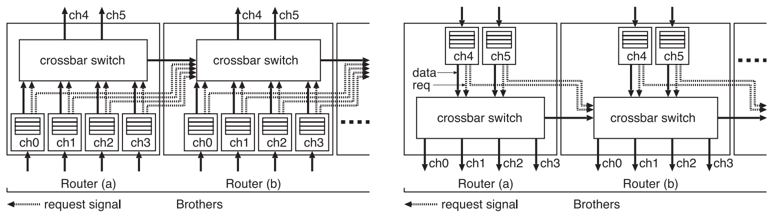
buffered bypass port の問題を解決する方法として bufferless bypass port を提案する（**図 7**）．これは，同一 brothers 内のルータが集約配置されている点を利用して，buffered bypass port からバッファを取り除いた構造をとる．通常，入力チャネルは同じルータのクロスバに対して，switch allocation（図中の request 線）を行うが，bufferless bypass port では，隣接ルータのクロスバに対しても switch allocation を行う．たとえば，図 7 (b) において，Router (a) の ch4 から入力されたパケットを ch2 に転送したいとする．このとき ch2 の前方のルータの入力チャネルがスリープ中であると，Router (a) の ch4 は Router (b) のクロスバに対し ch2 の switch allocation を実行する．Router (b) の ch2 の利用権を取得後，Router(a) の ch4 は (b) の ch2 に対しパケットを転送する．

ただし，この方法にも次の欠点がある．1) 各入力チャネルは自ルータに加え，隣接ルータの switch allocation を実行するため，隣接ルータのクロスバへの request 信号が必要となり，アービタが複雑になる．2) 2 回以上の横流しを実行するには 2 個隣りのルータへの switch allocation が必要となるが，制御の複雑さを考えると現実的ではない．そのため，bufferless bypass port では横流しの回数を 1 回までとする．

それでも，bufferless bypass port の場合，横流しによってパケットのバッファへの read/write が増えないため，buffered bypass port と違い，動的電力の増加を抑えることができる．

4.4.3　デッドロックフリールーティング

Fat Tree では通常 up*/down* ルーティング[14)]が用いられる．横流し機構の導入により，up*/down* ルーティングで得られる経路集合に加え，bypass port を経由した経路が利用できるようになる．これにより，これまでツリー型トポロジで問題にならなかったパケット間



(a) 上方向へのデータ転送　(b) 下方向へのデータ転送

図 7　Bufferless bypass port 付きルータ（LPort）
Fig. 7　Router with bufferless bypass port (LPort).

© 2008 Information Processing Society of Japan

のデッドロックが生じる可能性が出てくる．そこでルーティングについて検討する．bypass port の利用を整理すると次の 2 種類に分類することができる．

a) ツリー間の移動：$c \geq 2$ の Fat Tree では，複数の Fat Tree を重ね合わせたトポロジとなる．bypass port を用いることで，複数ツリー間の移動が可能になる．

b) ツリー内の移動：$p \geq 2$ の Fat Tree では単一ツリー内に複数の枝が生じる．bypass port を用いることで，これらの枝の間を移動できる．

定理　bypass port を経由する経路を含む up*/down* ルーティングはデッドロックフリーである．

証明　単一ツリー内のルーティングはデッドロックフリーであるため，b) は循環依存を生じない．bypass port の利用は一方通行であり U-turn を生じないため，a) も循環依存を生じない．よって，デッドロックフリーである．∎

5. 評　　価

本論文では Fat Tree の拡張として，buffered bypass port（BPort）付き Fat Tree と bufferless bypass port（LPort）付き Fat Tree を提案した．BPort や LPort 付きに加え，bypass port を持たない通常の Fat Tree（Orig）を面積，スループット，消費エネルギー，スリープ率，リーク電力について評価する．

5.1　結合網のハードウェア量

結合網の面積はルータとネットワークインタフェース（NI）からなる．通常のルータ回路は 2.1 節で紹介したものをそのまま使った．NI 回路は 2-flit 分の FIFO を入力側と出力側にそれぞれ持たせるだけの単純なものを使用した．

BPort 付きルータは物理チャネルを 1 個増やすだけで実現できる．LPort 付きルータの場合にも物理チャネルを 1 個追加するが，この物理チャネルにはバッファは持たせない．その代わり，隣接ルータの switch allocation を実行するために，入力チャネルから隣接ルータのクロスバへの request 信号を追加実装した（図 7）．BPort と LPort 付きルータでは，ルータのクロスバのポート数が 1 つ増えることになる．なお，BPort や LPort はルータ回路のみに実装し，NI 回路への変更は不要である．2.2 節と同様の手順で，配置配線後のゲート数を見積もった[⋆1]．

図 8 が 16 コア，**図 9** が 64 コアの場合のゲート数である．それぞれ Orig，BPort，LPort

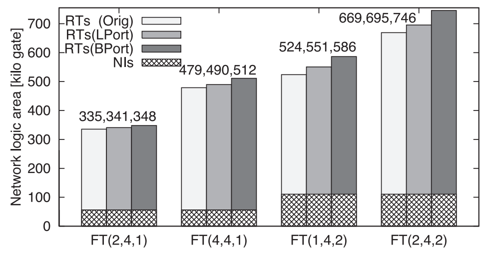


図 8　ルータ（RT）と NI のゲート数（16 コア）
Fig. 8　Gate counts of routers (RTs) and NIs (16-core).

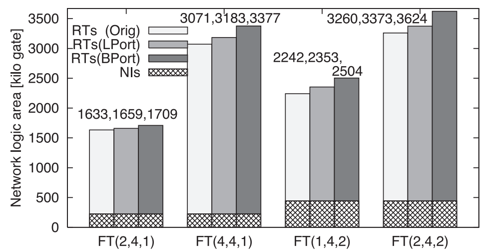


図 9　ルータ（RT）と NI のゲート数（64 コア）
Fig. 9　Gate counts of routers (RTs) and NIs (64-core).

を 4 種類の Fat Tree に適用した．Fat Tree (1,4,2) や (2,4,2) のように多重度が高いツリーほど bypass port の数が多くなる．Orig と比較すると，追加のバッファを必要とする BPort では最大 11.8%，バッファを必要としない LPort では最大 5.2% 面積が増えた．面積の増加率が最大となったネットワーク構成は BPort，LPort ともに 16 コアの Fat Tree (1,4,2) であった[⋆2]．すべての bypass port に通常のチャネルと同じだけのバッファを持たせる BPort の面積の増分と比べ，LPort のオーバヘッドは半分以下である．そのため，LPort は BPort より軽量なハードウェアで実現できたといえる．

5.2　平均ホップ数とスループット性能

本節では，パワーゲーティング適用時の Orig，BPort，LPort のパケット転送性能（スループット）を評価する．さらに，パワーゲーティング非適用時（NoPG）のスループット

⋆1 結合網の面積に加え，実際にはスリープトランジスタなどが必要となる．

⋆2 Fat Tree (2,4,2) は最上位ランク以外のルータが 6 個のチャネルを持つのに対し，Fat Tree (1,4,2) のルータのチャネル数は 5 個である．このように Fat Tree (1,4,2) は結合網全体の面積が小さい分，bypass port による面積の増加率が他より高くなった．

© 2008 Information Processing Society of Japan

との比較も行う．

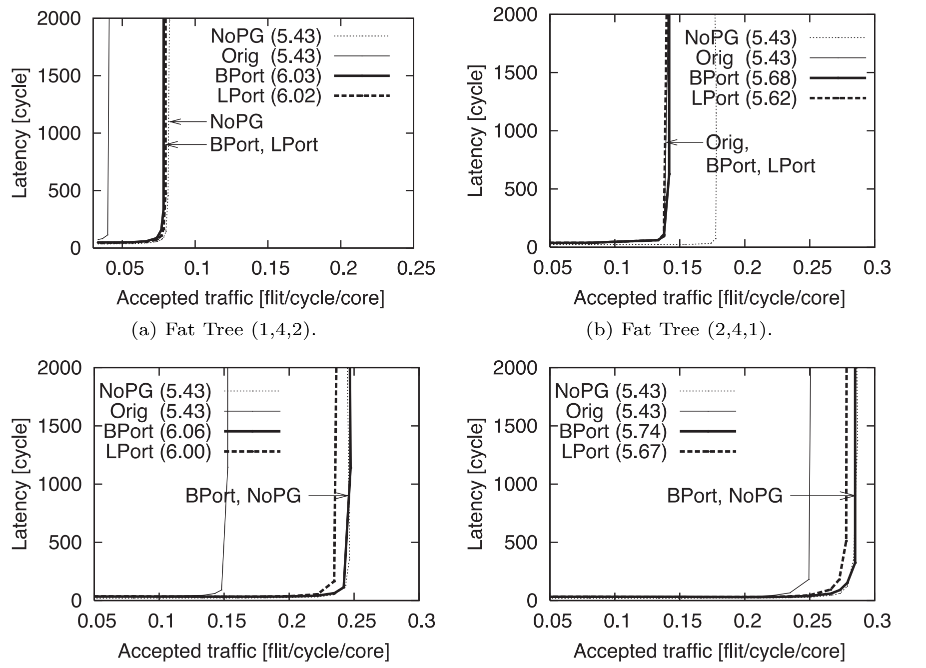
評価のためにフリットレベル・ネットワークシミュレータを用いた．ルータのスイッチング機構として，I/O バッファ，クロスバ，アービタを単純化したモデルを採用しており，2.1 節で示したワームホールルータ回路と同じ挙動をするように設計した．ここでは 4-flit 分のデータに 1-flit 分のヘッダを付与したものを 1 パケットとし，ヘッダフリットが隣接ルータまたは計算コアに転送されるのに最低 3 サイクルかかるものとした．ネットワークトポロジには，16 コアおよび 64 コアの Fat Tree (1,4,1)，(1,4,2)，(2,4,1)，(2,4,2)，(4,4,1) を用いた．ルーティングには Fat Tree で一般的に用いられる up*/down* ルーティングを用い，仮想チャネル数は 2 本とした[⋆1]．シミュレーションに用いる通信パターンとしてユニフォームトラフィックを用いた．T_{wakeup} は 3 サイクルと 6 サイクルとし，$T_{idledetect}$ は 2 サイクルとした．ウェイクアップ制御の方法として 4.2 節で述べた conservative を採用した．なお，NoPG はウェイクアップ遅延の影響を受けない．

図 10 と **図 11** に，64 コアの各種 Fat Tree におけるスループット（accepted traffic）とレイテンシのグラフを示す．前者が $T_{wakeup} = 3$，後者が $T_{wakeup} = 6$ の場合の結果である．グラフ中の括弧内に平均ホップ数を示した．

まず，$T_{wakeup} = 3$ の場合の Fat Tree (1,4,2) に着目する（図 10 (a)）．Orig では経路の途中でツリーを切り替えることができず，NoPG と比べて大きくスループットが落ちている．一方，BPort と LPort では bypass port により負荷分散ができており，NoPG なみのスループットが出ている．それでも 図 11 (a) のようにウェイクアップ遅延がさらに増えると，ウェイクアップ遅延のスループットへの影響を隠蔽できなくなり BPort と LPort のスループットも低下する．このような傾向は Fat Tree (2,4,2)，(4,4,1) においても見られた．

BPort と LPort のスループットは bypass port をどれだけ活用できたかで決まる．たとえば，Fat Tree (2,4,1) では rank 1 の brothers 内にルータは 1 個しかなく，横流しできる箇所は一部の上位ルータに限られている．一方，Fat Tree (1,4,2) や (2,4,2) ではすべての brothers が複数のルータを持つため，横流しできる機会が多い．Fat Tree (4,4,1) のように rank 1 の brothers では横流しができなくても，rank 2 以上の brothers で横流しすることでスループットの劣化を抑えている例もある．実際，Fat Tree (1,4,2) や (2,4,2) における BPort と LPort の平均ホップ数は，(2,4,1) よりも多く，より頻繁に bypass port が使われたことが分かる．

⋆1 2 本の仮想チャネルを有効に使うため，宛先アドレスに応じて使用する仮想チャネルを均等に振り分けた．



(a) Fat Tree (1,4,2).　(b) Fat Tree (2,4,1).　(c) Fat Tree (2,4,2).　(d) Fat Tree (4,4,1).

図 10　各種 Fat Tree における Orig，BPort，LPort のスループット（$T_{wakeup} = 3$; 64-core; uniform traffic）

Fig. 10　Throughputs of Orig, BPort and LPort on various Fat Trees ($T_{wakeup} = 3$; 64-core; uniform traffic).

図 10 と 図 11 の評価結果をとおして，LPort のスループットは BPort に若干劣るものの，その差は小さいといえる．

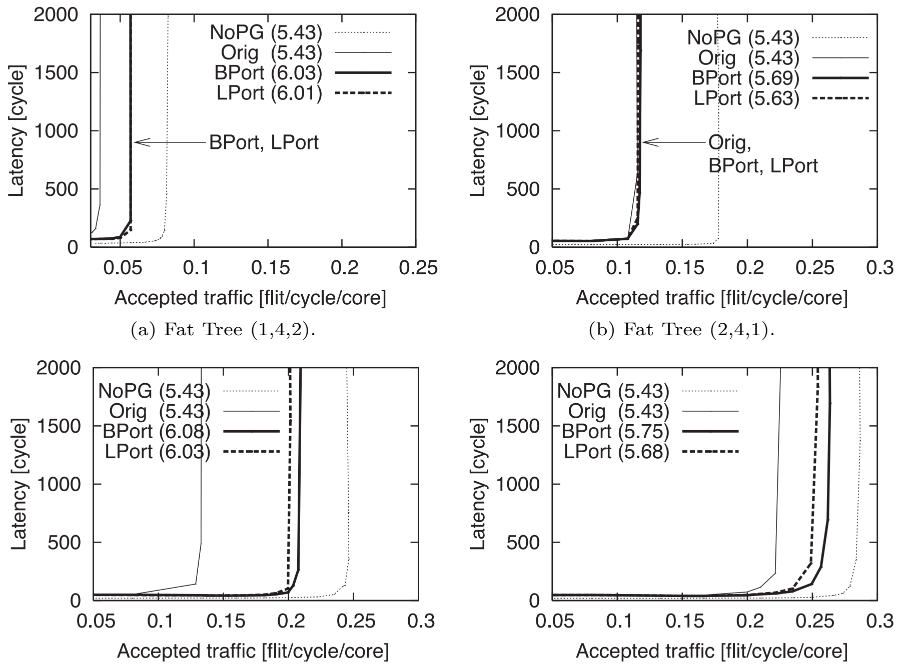
5.3　フリット転送エネルギー

1-flit のデータを送信元から宛先コアに転送するとき，これに要する平均転送エネルギー E_{flit} は次式で計算できる．

$$E_{flit} = wH_{ave}(E_{sw} + E_{link}) \quad (2)$$

ただし，w を 1-flit のビット幅，H_{ave} を平均ホップ数，E_{sw} をルータまたは NI が 1-bit のデータ転送に消費するエネルギー，E_{link} をリンクが 1-bit のデータ転送に消費するエネ

© 2008 Information Processing Society of Japan



(a) Fat Tree (1,4,2).　(b) Fat Tree (2,4,1).

(c) Fat Tree (2,4,2).　(d) Fat Tree (4,4,1).

図 11　各種 Fat Tree における Orig，BPort，LPort のスループット ($T_{wakeup} = 6$; 64-core; uniform traffic)

Fig. 11　Throughputs of Orig, BPort, and LPort on various Fat Trees ($T_{wakeup} = 6$; 64-core; uniform traffic).

ルギーとする．

5.1 節で実装した Orig，BPort，LPort ルータおよび NI を，500 MHz での動作を仮定してゲートレベルでシミュレーションした．その結果，Orig ルータの E_{sw} は 0.183pJ，LPort ルータの E_{sw} は 0.193pJ，NI の E_{sw} は 0.092pJ となった．BPort の bypass port は Orig ルータのチャネルと同じ構造を持っているため，BPort の bypass port を 1 回経由するたびに Orig の E_{sw} と同じ 0.183pJ が余計に消費されるものとした．一方，E_{link} は次式で計算できる．

$$E_{link} = dV^2C_{wire}/2 \quad (3)$$

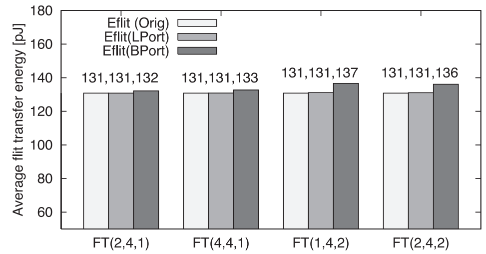


図 12　平均フリット転送エネルギー E_{flit}（16 コア）

Fig. 12　Average flit transmission energy E_{flit} (16-core).

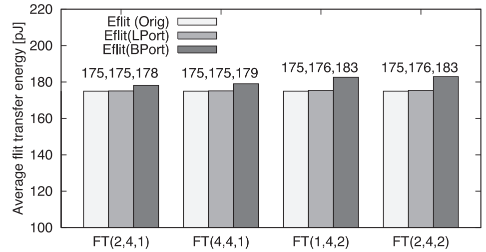


図 13　平均フリット転送エネルギー E_{flit}（64 コア）

Fig. 13　Average flit transmission energy E_{flit} (64-core).

ただし，d を 1-hop あたりの平均距離，V を動作電圧，C_{wire} を配線容量とする．ここでは V を 1.0 V とし，C_{wire} は 90 nm プロセスを仮定して 300 fF/mm とした[15]．8 mm 角のチップを想定し，上記のパラメータをもとに E_{flit} を計算した．

16 コアの結果を **図 12**，64 コアの結果を **図 13** に示す．5.2 節で見たように bypass port を使うことで平均ホップ数が増える．とくに BPort の場合は，増えたホップ数分だけバッファへの read と write が余計に発生するため消費エネルギーが最大で 4.6% 増えた．LPort の E_{sw} は Orig より 5.5% 大きいものの，そもそも全パケット転送に占める bypass port の利用はそれほど頻繁ではないため LPort の消費エネルギーは Orig と同程度となった．

ここまでの評価結果より，1) LPort の面積オーバヘッドは BPort の半分以下である（5.1 節），2) LPort は BPort に若干スループットで劣るものの，両者のスループットの差は小さい（5.2 節），3) BPort を採用するとフリット転送エネルギーが増えるが LPort ではほとんど増加しない（5.3 節）ということが分かった．以上より，LPort は BPort と同等のスループットで，なおかつ省電力性に優れることから，LPort のほうが有利であると考え

© 2008 Information Processing Society of Japan

られる．そこで，以降の評価では Orig と LPort の比較を中心に行う．

5.4　パワーゲーティングの損益分岐点

チャネル単位のパワーゲーティングを行った際のスリープ率やリーク電力を評価するため前準備として，本節では，パワーゲーティングの損益分岐点（$T_{breakeven}$）を計算する．

パワーゲーティングの損益分岐点を求めるために，文献 9) で提案された解析モデルを用いる．損益分岐点の計算は次の 3 ステップからなる．1) N サイクル間スリープすることで削減できるリークエネルギーの合計値（E^N_{saved}）を計算する．2) パワースイッチの ON/OFF 操作に要すエネルギー（$E_{overhead}$）を計算する．3) E^N_{saved} と $E_{overhead}$ が等しくなる N の値を求める．こうして求めた N の値が $T_{breakeven}$ である．

E^N_{saved} は次式で計算できる[9]．

$$E^N_{saved} = E_{leak}\frac{\mathrm{DIBL}}{mV_t}\frac{N^2}{2}\frac{\alpha L V_{dd}}{2(\frac{1}{2}+\frac{C_D}{C_S})} \tag{4}$$

ただし，V_{dd} を供給電圧，E_{leak} をパワーゲーティング対象回路（本論文の場合はルータのチャネル）のリークエネルギー，E_{sw} を対象回路のスイッチングエネルギー，$L = E_{leak}/E_{sw}$，α を対象回路のスイッチング率とする．また，DIBL は drain-induced barrier lowering であり，ここでは 0.1 としている．$V_t = kT/q \approx 25\,\mathrm{mV}$ は熱電圧（thermal voltage）であり，ここでは $m \approx 1.3$ としている．W_H は対象回路に占めるパワースイッチの面積であり，ここでは 0.1 としている．C_S は対象回路のスイッチング容量の合計，C_D はパワースイッチを含むローカルの電源ライン容量の合計であり，ここでは $C_D/C_S = 0.5$ としている．

$E_{overhead}$ は次式で計算できる[9]．

$$E_{overhead} \approx 2\frac{W_H}{\alpha}E_{leak}. \tag{5}$$

2.2 節で示したルータの消費電力をもとに，まず，入力チャネル 1 個あたりのリークエネルギー E_{leak}，スイッチングエネルギー E_{sw}，スイッチング率 α を抽出した．そして，入力チャネル 1 個を N サイクルの間パワーゲーティングしたときの消費電力（リーク電力およびパワースイッチの ON/OFF 電力の和．以降,「補正済みリーク電力」と呼ぶ）を式 (4) と式 (5) を用いて計算した．

図 14 にこの計算結果のグラフを示す．グラフ中の NoPG(25°C) と NoPG(75°C) は，パワーゲーティングを行わないルータチャネルのリーク電力を示しており，それぞれ動作温度が 25°C と 75°C の場合である．一方，PG(25°C) および PG(75°C) は，それぞれ 25°C と 75°C で動作させたときの補正済みリーク電力を示す．当然，スリープ時間 N が大きくなるに従い，パワースイッチを ON/OFF する際に消費されるオーバヘッドエネルギーの影響が小さくなるので，平均消費電力が減少する．PG(25°C) と NoPG(25°C) のラインが交差するときの N の値が 25°C で動作させたときの損益分岐点である．グラフより，25°C 動作の場合 $T_{breakeven} \approx 9$，75°C 動作の場合 $T_{breakeven} \approx 5$ となった．

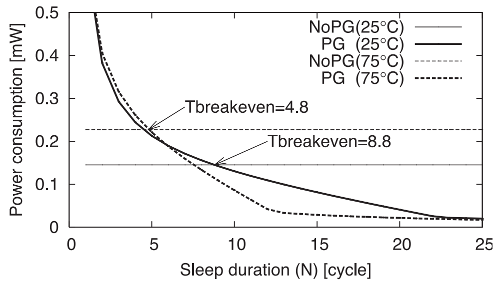


図 14　スリープ時間 vs. チャネルの補正済みリーク電力
Fig. 14　Sleep duration vs. compensated leakage power of a channel.

リーク電力は動作温度の増加に応じて増える．現状の 90 nm からプロセスがさらに微細化して消費電力に占めるリーク電力の割合が増えた場合，ここで示した 75°C の場合のようにパワーゲーティングの損益分岐点がさらに小さくなり，パワーゲーティングによる利得が増えると期待される．

5.5　償却可能なスリープ率

パワーゲーティングの観点から，ルータチャネルの状態を次の 3 種類に分類する．

- **Active：** チャネルが動作中の状態．パワーゲーティングをしない状態と同量のリーク電力が消費される．
- **CSS：** compensated sleep（$T_{breakeven}$ 以上の長さのスリープ）の状態．CSS が増えればリーク電力が減る．
- **USS：** uncompensated sleep（$T_{breakeven}$ 未満の長さのスリープ）の状態．USS が増えれば消費電力が増える．

各チャネルにおける CSS の割合を求めるため，5.2 節と同様の方法でシミュレーションを行った．今回はネットワークトポロジとして 64 コアの Fat Tree (2,4,2) を用い，T_{wakeup} を 3 サイクル，$T_{breakeven}$ を 9 サイクル，$T_{idledetect}$ を 2 サイクルとした．シミュレーションに用いる通信パターンとして，ユニフォームトラフィックに加え，NAS Parallel Benchmark（NPB）プログラムから得られた次の通信パターンを用いた．Block Tridiagonal solver

© 2008 Information Processing Society of Japan

(BT)，Scalar Pentadiagonal solver（SP），Conjugate Gradient（CG），Multi-Grid solver（MG），large Integer Sort（IS）．クラスは W とした．

図 15 に評価結果を示す．グラフ中の細い線が CSS の割合を示し，グラフの右軸が CSS の目盛りである．当然 CSS の割合はトラフィック負荷が増えるに従い減少している．なお，これらのグラフでは最大スループットになる手前までの CSS の割合をプロットしている．Orig は LPort より少ない負荷でネットワークが飽和するため，Orig の線は LPort よりも小さい accepted traffic で途切れている．グラフより LPort は Orig より CSS の割合が高く，また，スループットも高いことが分かる．

5.6　リーク電力の削減量

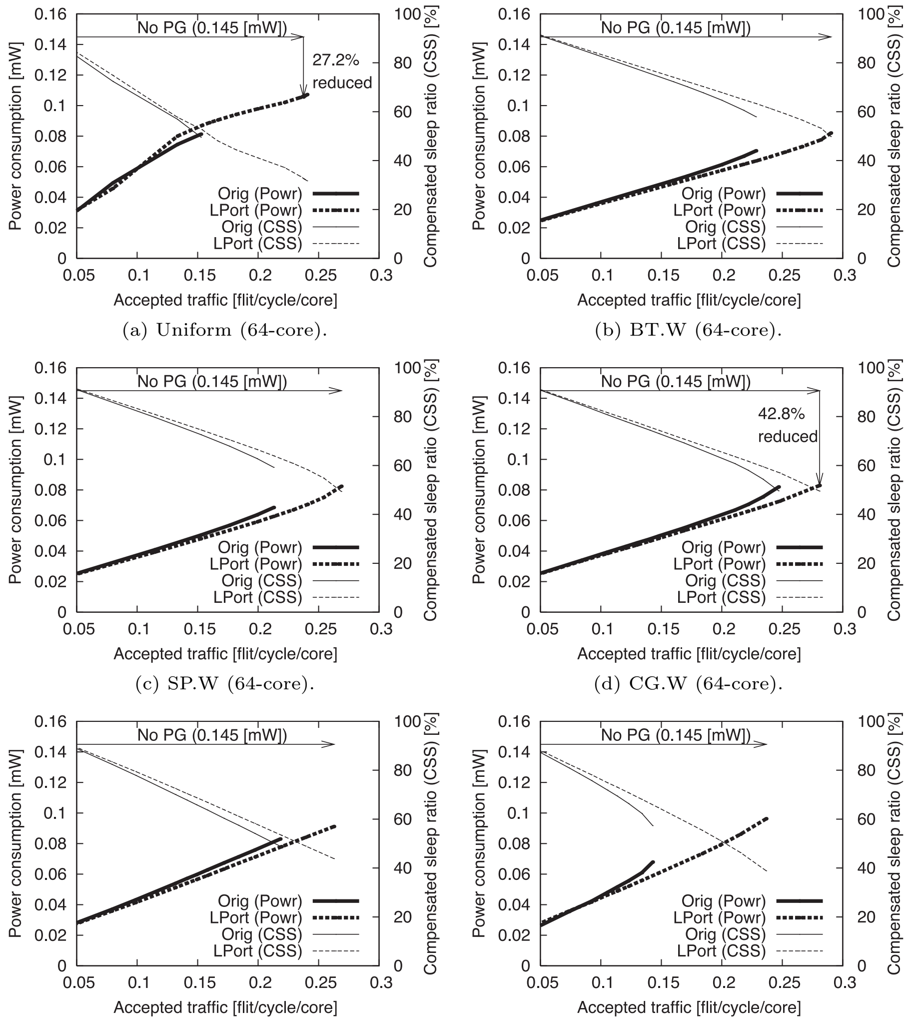
オーバヘッド電力を含めたチャネルの補正済みリーク電力を求めるため，前節と同じ方法でネットワークシミュレーションを実行し，各チャネルごとに発生したすべてのスリープの期間と回数を記録していった（たとえば，15 サイクルのスリープが 1,234 回発生など）．そして，この情報を図 14 で示した「スリープ時間 vs. チャネルの補正済みリーク電力」のグラフにあてはめ，各物理チャネルの補正済みリーク電力を見積もった．当然，$T_{breakeven}$ より長いスリープが多ければ補正済みリーク電力は減り，$T_{breakeven}$ 未満のスリープが多ければ電力は増える．

図 15 に評価結果を示す．グラフ中の太い線がチャネルあたりの補正済みリーク電力を示し，グラフの左軸が補正済みリーク電力の目盛りである．グラフ中にはパワーゲーティングをしない場合のリーク電力（No PG）も表記してある．トラフィック負荷が増えるに従い，償却可能なスリープ（CSS）の機会が減り，チャネルあたりの補正済みリーク電力が増加している．それでも，LPort はトラフィック負荷が最も高いときでも 27.2%〜42.8% のリークを削減できている．LPort の補正済みリーク電力は Orig より小さいため，LPort によって高いスループットを，より少ない補正済みリーク電力で実現できたといえる．

そして，今後，さらにプロセスが微細化されれば，消費電力に占めるリーク電力の割合が増える．それによって，5.4 節で述べたようにパワーゲーティングの損益分岐点が小さくなり，パワーゲーティングの利得がさらに増える．

5.7　ディスカッション

本節では，1) 提案手法の NoPG に対する面積オーバヘッドと，2) NoPG に対するスループット性能の劣化を考慮しても，LPort によるパワーゲーティングが NoPG よりもリーク電力で優れることを示す．



(a) Uniform (64-core).　(b) BT.W (64-core).

(c) SP.W (64-core).　(d) CG.W (64-core).

(e) MG.W (64-core).　(f) IS.W (64-core).

図 15　償却可能スリープ率（右軸）とチャネルあたりの補正済みリーク電力（左軸）（$T_{breakeven} = 9$; $T_{wakeup} = 3$）

Fig. 15　Compensated sleep ratio (right axis) and compensated leakage power of a channel (left axis) ($T_{breakeven} = 9$; $T_{wakeup} = 3$).

© 2008 Information Processing Society of Japan

(1) ハードウェア量：

64 コアの Fat Tree (2,4,2) の場合，Orig に対する面積の増加率は BPort で 11.2%，LPort で 3.5% となった．リーク電力は回路面積にほぼ比例する[16] ため，BPort と LPort ではそれぞれリーク電力が 11.2% と 3.5% ずつ増える．

(2) スループット性能：

64 コアの Fat Tree (2,4,2) で $T_{wakeup} = 3$ の場合，BPort のスループットは NoPG より 0.6% 低く，LPort は NoPG より 3.6% 低い．BPort および LPort が，NoPG と同じスループットを出すには，理論上，BPort のデータ幅を 0.6% 広げ，LPort のデータ幅を 3.6% 広げる必要がある．面積に比例してリーク電力が増える[16] と仮定すると，BPort と LPort ではそれぞれリーク電力が 0.6% と 3.6% ずつ増える．

同様に $T_{wakeup} = 6$ の場合，BPort のスループットは NoPG より 14.8% 低く，LPort は NoPG より 17.9% 低い．NoPG と同等のスループットを期待するには，BPort と LPort のデータ幅をそれぞれ 14.8%と 17.9%広げる必要があるが，それに応じてハードウェア量とリーク電力が増加する．

(3) リーク電力の削減量：

上記の (1) と (2) より，64 コアの Fat Tree (2,4,2) で $T_{wakeup} = 3$ の場合，BPort のリーク電力は NoPG より 11.9% 大きく，LPort のリーク電力は NoPG より 7.2% 大きい．図 15 (a) より，LPort は入力されるトラフィック負荷が最大のときでも 27.2% のリーク電力を削減できている．そのため，(1) と (2) によって LPort のリーク電力が NoPG より 7.2% 大きくなっても，LPort の補正済みリーク電力は NoPG のリーク電力を下回る．さらに，$T_{wakeup} = 6$ の場合であっても LPort は NoPG よりリーク電力で優れる．

6. まとめ

本論文では Fat Tree トポロジにおけるルータチャネルの適応的なパワーゲーティングについて検討し，ウェイクアップ遅延に起因する HoL ブロッキングを隠蔽するためのルータアーキテクチャとして BPort と LPort を提案した．物理的に近隣に配置した異なるツリーに属するルータにデータを横流しするために，BPort では通常のバッファ付きチャネルを追加するが，LPort ではバッファを持たないチャネルを追加する．提案手法は Fat Tree (2,4,2) のように多重度の高いツリーでは，トラフィック負荷を分散させることができたが，Fat Tree (2,4,1) では横流しの効果は小さかった．BPort と LPort のスループット評価の結果，LPort は BPort に若干スループットで劣るものの，bypass port を持たない Orig と比べると両者のスループットの差は小さかった．面積評価の結果，LPort の面積オーバヘッドは BPort の半分以下であり，Orig に対する面積の増加率は 16 コアの Fat Tree (1,4,2) においてたかだか 5.2% 増だった．また，BPort を採用するとフリット転送エネルギーが増加してしまったが，LPort ではほとんど増加しなかった．以上より，面積オーバヘッドが小さく，省電力性にも優れる LPort のほうが BPort よりも有利となった．そこで，チャネル単位のランタイムパワーゲーティングを行う Fat Tree (2,4,2) 上で LPort を評価したところ，Orig よりも高いスループットをより少ないリーク電力で実現できた．

謝辞　本研究は，科学技術振興機構「JST」の戦略的創造研究推進事業「CREST」における研究領域「情報システムの超低消費電力化を目指した技術革新と統合化技術」の研究課題「革新的電源制御による次世代超低電力高性能システム LSI の研究」による．また，本研究は東京大学大規模集積システム設計教育研究センターを通し，シノプシス株式会社・日本ケイデンス株式会社の協力で行われた．

参　考　文　献

1) Dally, W.J. and Towles, B.: Route Packets, Not Wires: On-Chip Interconnection Networks, *Proc. Design Automation Conference*, pp.684–689 (2001).
2) Benini, L. and Micheli, G.D.: *Networks on Chips: Technology And Tools*, Morgan Kaufmann (2006).
3) Vangal, S., et al.: An 80-Tile 1.28TFLOPS Network-on-Chip in 65nm CMOS, *Proc. International Solid-State Circuits Conference* (2007).
4) Ishikawa, M., et al.: A 4500 MIPS/W, 86 μA Resume-Standby, 11 μA Ultra-Standby Application Processor for 3G Cellular Phones, *IEICE Trans. Electronics*, Vol.E88-C, No.4, pp.528–535 (2005).
5) Banerjee, A., Mullins, R. and Moore, S.: A Power and Energy Exploration of Network-on-Chip Architectures, *Proc. International Symposium on Networks-on-Chip*, pp.163–172 (2007).
6) Matsutani, H., Koibuchi, M., Wang, D. and Amano, H.: Run-Time Power Gating of On-Chip Routers Using Look-Ahead Routing, *Proc. Asia and South Pacific Design Automation Conference*, pp.55–60 (2008).
7) Dally, W.J. and Towles, B.: *Principles and Practices of Interconnection Networks*, Morgan Kaufmann (2004).
8) Kanno, Y., et al.: Hierarchical Power Distribution with 20 Power Domains in 90-nm Low-Power Multi-CPU Processor, *Proc. International Solid-State Circuits Conference*, pp.2200–2209 (2006).
9) Hu, Z., Buyuktosunoglu, A., Srinivasan, V., Zyuban, V., Jacobson, H. and Bose,

© 2008 Information Processing Society of Japan

P.: Microarchitectural Techniques for Power Gating of Execution Units, *Proc. International Symposium on Low Power Electronics and Design*, pp.32–37 (2004).
10) Jiang, H., Marek-Sadowska, M. and Nassif, S.R.: Benefits and Costs of Power-Gating Technique, *Proc. International Conference on Computer Design*, pp.559–566 (2005).
11) Chen, X. and Peh, L.-S.: Leakage Power Modeling and Optimization in Interconnection Networks, *Proc. International Symposium on Low Power Electronics and Design*, pp.90–95 (2003).
12) Soteriou, V. and Peh, L.-S.: Exploring the Design Space of Self-Regulating Power-Aware On/Off Interconnection Networks, *IEEE Trans. Parallel and Distributed Systems*, Vol.18, No.3, pp.393–408 (2007).
13) Alonso, M., Coll, S., Martinez, J.-M., Santonja, V., Lopez, P. and Duato, J.: Dynamic Power Saving in Fat-Tree Interconnection Networks Using On/Off Links, *Proc. International Parallel and Distributed Processing Symposium* (2006).
14) Schroeder, M.D., Birrell, A.D., Burrows, M., Murray, H., Needham, R.M. and Rodeheffer, T.L.: Autonet: A High-speed, Self-configuring Local Area Network Using Point-to-point Links, *IEEE Journal on Selected Areas in Communications*, Vol.9, pp.1318–1335 (1991).
15) Ho, R., Mai, K.W. and Horowitz, M.A.: The Future of Wires, *Proc. IEEE*, Vol.89, No.4, pp.490–504 (2001).
16) Moyer, B.: Low-Power Design for Embedded Processors, *Proc. IEEE*, Vol.89, No.11, pp.1576–1587 (2001).

(平成 20 年 1 月 29 日受付)
(平成 20 年 5 月 27 日採録)

松谷　宏紀（学生会員）

平成 16 年慶應義塾大学環境情報学部卒業．平成 20 年同大学大学院理工学研究科開放環境科学専攻博士課程修了．博士（工学）．現在，同大学院理工学研究科訪問研究員．平成 18 年度より日本学術振興会特別研究員．チップ内ネットワークの研究に従事．

鯉渕　道紘（正会員）

平成 12 年慶應義塾大学理工学部情報工学科卒業．平成 15 年同大学大学院理工学研究科開放環境科学専攻博士課程修了．博士（工学）．平成 14 年度より 16 年度まで日本学術振興会特別研究員．現在，国立情報学研究所助教，総合研究大学院大学複合科学研究科情報学専攻助教（兼任）．ハイパフォーマンスコンピューティングとインターコネクトに関する研究に従事．IEEE Computer Society Japan Chapter Young Author Award 2007，平成 19 年度情報処理学会論文賞受賞．IEEE，電子情報通信学会各会員．

王　　代涵

平成 16 年中国武漢大学情報工学部卒業．平成 18 年慶應義塾大学大学院理工学研究科開放環境科学専攻修士課程修了．現在，同大学院理工学研究科開放環境科学専攻博士課程．チップ内ネットワークの研究に従事．

天野　英晴（正会員）

昭和 56 年慶應義塾大学工学部電気工学科卒業．昭和 61 年同大学大学院工学研究科電気工学専攻博士課程修了．工学博士．現在，慶應義塾大学理工学部情報工学科教授．計算機アーキテクチャの研究に従事．

© 2008 Information Processing Society of Japan