Rearrangeable NoC: 配線遅延を考慮した分散ルータアーキテクチャ

松谷宏紀[†] 鯉渕道紘[††]
中村宏[†††] 天野英晴[†]

Rearrangeable NoC（RNoC）では，オンチップルータの機能をバッファと制御ロジックから成る単純なユニットに分解し，リピータバッファの置き換えとして，リンク上に分散配置する．各ユニットは複数のネットワーク間で共有，もしくは，必要に応じてバイパスできるように設計する．リンク上のユニットをバイパスできればバッファリング回数を減らせるため，通信遅延と消費エネルギーが減るが，ラッチされない長い配線が形成されるため配線遅延は増加する．つまり，不要なバッファリングを極力減らして通信遅延と消費エネルギーを削減するには，動作周波数が MET する限り，ネットワークとしての動作に支障の無いユニットをバイパスすれば良い．本論文では，オンチップルータの機能をどのような単位に分解するか，分解されたユニットをどのような間隔でリンク上に分散配置するか，これらをどのようにバイパス，もしくは，利用するかについて議論する．

Rearrangeable NoC: A Distributed Router Architecture for Exploiting Wire Delay

Hiroki Matsutani,[†] Michihiro Koibuchi,[††] Hiroshi Nakamura[†††]
and Hideharu Amano[†]

We propose the Rearrangeable NoC (RNoC) architecture, in which on-chip routers are decomposed into small units, each of which consists of buffers and simple control logic. The decomposed units are distributed on the wire links in order to overcome the wire delay without repeater buffers. They are bypassed or shared by multiple networks on the chip as long as the required operating frequency is met. Bypassing several units on a link can reduce the average buffering counts for each packet transfer, which also reduces the communication latency and energy consumption, while it forms long unbuffered wire lines that would quadratically increase the wire delay. Here we discuss how to decompose on-chip routers into smaller units and how to distribute them on the link. We also discuss how to use (or bypass) each unit on the link in order to bypass more units as long as the required frequency is not harmed.

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，このようなタイル同士の結合手法として Network-on-Chip（NoC）が注目されている[1),2)]．

我々は現状の NoC において以下の課題があると考えている．

(1) トラフィックの局所性（e.g., メモリアクセス）
(2) オンチップルータの面積（e.g., バッファ容量）
(3) 通信遅延（e.g., ルータのパケット転送サイクル数）
(4) 消費電力（e.g., バッファリングに要すエネルギー）

タイルアーキテクチャ向けの NoC を考えるとき，1 点目で挙げたように，メモリアクセスのためのトラフィックが問題になる．一般的に，メモリアクセスには局所性が強く，バースト的に通信が発生する☆．NoC で一般的に用いられている wormhole（WH）スイッチングは軽量，かつ，高い柔軟性を持ち，汎用のネットワークには適しているが，メモリアクセスには circuit switch（CS）のように通信路を予め予約する方式が適している．そのため，**図 1** のように複数系統のネットワークを持ち，用途の応じてネットワークを使い分けるタイルアーキテクチャが今後普及すると考えられる．実際，RAW[3)] や TILE64[4)] は複数系統のネットワークを持ち，a) プログラムのコンパイル時に通信パターンが決定する静的なトラフィックと，b) プログラムの実行時に発生する動的トラフィックをそれぞれ別々のネットワークを使って転送する．加えて，今後は，要求性能や消費電力に応じてネットワークごとに異なる周波数で動作する NoC が実用化されると考えられる．

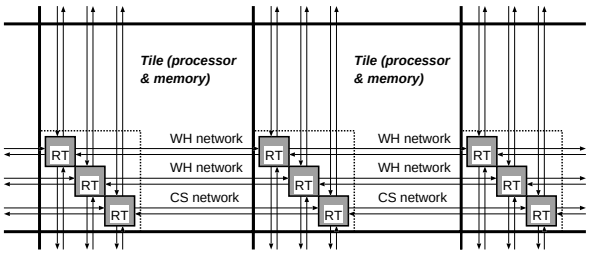


図 1 複数系統のネットワークを持つタイルアーキテクチャの例.

その一方で，課題の 2〜4 点目で挙げたように，オンチップルータのハードウェア量は極力小さくする必要がある．また，パケットをバッファリングするために要す消費エネルギーと通信レイテンシも削減しなければならない．

以上を踏まえ，本論文では Rearrangeable NoC（RNoC）アーキテクチャを提案する．RNoC には次の特徴がある．

(1) オンチップルータの機能をバッファと制御ロジックから成る単純なユニットに分解し，リピータバッファの

† 慶應義塾大学大学院 理工学研究科
Graduate School of Science and Technology, Keio University
†† 国立情報学研究所
National Institute of Informatics
††† 東京大学大学院情報理工学系研究科
Graduate School of Information Science and Technology, The University of Tokyo

☆ 例えば，メモリリードのために 1-flit のリード要求（アドレス）がメモリタイルへ送信された後，多量のデータがメモリから返信される．

置き換えとしてリンク上に分散配置する．
(2) 分解された各ユニット（バッファと制御ロジック）は複数ネットワーク間で共有できるようにする．
(3) また，ネットワーク上の各ユニットを必要に応じてバイパスできるようにする．

1つ目の特徴によって，今後顕在化するであろう配線遅延の影響を軽減でき，高い動作周波数を狙えるようになる．また，2つ目の特徴によってルータのバッファ量を削減できる．これは，同一リンク上のすべてのネットワークのバッファが一度に全部埋まることは考えにくいため，バッファを複数ネットワーク間で共有することでトータルのバッファ量を削減できるためである．次に3つ目の特徴によって，パケットの転送サイクル数（通信遅延）とフリット転送エネルギーを削減できる．これは，リンク上のユニットをバイパスできればバッファリング回数を減らせるためである．ただし，分解されたユニットをバイパスすると，ラッチされない長い配線が形成されて配線遅延が増加してしまう．したがって，要求される動作周波数を満足できるように，かつ，ネットワークとしての動作に支障の無いユニット選んでバイパスする必要がある．なお，使われないユニットをバイパスすることで使われないバッファが生じたら，他のネットワークが利用するバッファを増量する．もしくは，不要なユニットへの電力供給を止めることでリーク電力を抑えることも考えられる．

以上を実現するため，本論文では，オンチップルータの機能をどのような単位に分解して，リンク上に分散配置するか，また，分解されたユニットをどのように利用するか議論する．まず，2章で既存のルータアーキテクチャを説明し，3章と4章で Rearrangeable NoC アーキテクチャを提案する．そして5章で予備評価と考察を示し，6章にて本論文をまとめる．

2. 既存のルータアーキテクチャ

まず一般的なルータの構造を説明し，リンク上に配置されたリピータをバッファとして活用するための技術を紹介する．

2.1 Wormhole（WH）ルータ

WH スイッチングは柔軟性が高く，NoC を含め幅広い用途で利用されている．**図 2** に典型的な WH ルータを示す．ルータに入力されたパケットは入力チャネルでバッファリングされ，その間に出力ポートの計算（RC），出力仮想チャネルの割り当て（VA），出力チャネルのアービトレーション（SA）を行い，スイッチ上をパケットが転送される（ST）．

一般的な WH ルータの遅延モデルを以下に示す．図 2 に示す3サイクルルータの場合，RC，VSA，ST ステージの遅延のうち最も大きい値がルータの最大遅延 T_{WH} となる．

$$T_{WH} = \max(T_{RC}, T_{VSA}, T_{ST}) \quad (1)$$

各ステージの遅延は以下のように表記できる．

$$\begin{aligned} T_{RC} &= T_{link} + \max(T_{fifo\_wr}, T_{rc}) \\ T_{VSA} &= \max(T_{va}, T_{sa}) \\ T_{ST} &= T_{fifo\_rd} + T_{cb,n} \end{aligned} \quad (2)$$

ただし，T_{link} をリンクの配線遅延とする．T_{fifo_wr}，T_{fifo_rd}，T_{rc}，T_{va}，T_{sa}，$T_{cb,n}$ を，それぞれ，FIFO バッファ書き込み，FIFO バッファ読み込み，経路計算，出力仮想チャネルの割り当て，出力物理チャネルのアービトレーション，$n \times n$ クロスバスイッチを通過するためのゲート遅延とする．

リンク長を L_{link} [mm] とするとき，unbuffered リンクの遅延 T_{link} は以下のように L_{link} の2乗に比例する．

$$T_{link} = 0.5(L_{link})^2 C_{wire} R_{wire} \quad (3)$$

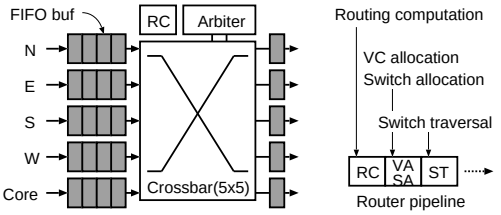


図 2 Wormhole（WH）ルータとそのパイプライン構造．

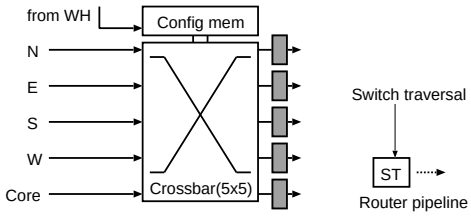


図 3 Circuit switch（CS）ルータとそのパイプライン構造．

ただし，C_{wire}，R_{wire} は配線容量 [F/mm]，配線抵抗 [Ohm/mm] とする．

配線遅延が大きいときは，一般的に，リンクに一定間隔でリピータバッファを挿入することで配線遅延を軽減するが，リピータの挿入によりリンクの消費エネルギーが最大 60% 増加する[5]．半導体技術の微細化に伴い，今後，T_{link} のウェイトがさらに大きくなることが予想される．実際，T_{link} のみに1サイクルを割り当てて Link traversal（LT）ステージとする実装も見られる[6),7]．一方，RNoC ではリンク上にルータ機能を分散配置することで配線遅延の影響を軽減する．

2.2 Circuit Switch（CS）ルータ

図 3 に典型的な CS ルータを示す．CS ネットワークでは，使用する経路上に制御パケットを流して帯域を予約してから，パケット転送を開始する．一度経路を setup してしまえば高い転送性能を実現できるが，経路を setup するためのオーバヘッドが大きいため，CS は多量のデータを一度に転送する場合に有効である．CS は bufferless flow control であり，WH と違い，パケットをルータ毎にバッファリングする必要はない．しかし，CS においても，配線遅延を抑えるため，経路上のルータ毎にバッファリングすることが一般的である．

CS ルータの最大遅延 T_{CS} は次式で表すことができる．

$$T_{CS} = T_{ST} = T_{link} + T_{cb,n} \quad (4)$$

2.3 バッファ機能を持つリピータ技術

Elastic interconnect[8] では，リンク上に，通常のリピータの代わりにデータラインの電圧値を保持できる特殊なリピータを持たせている．また，adaptive channel buffer[9] はリンク上に配置されるリピータ回路ではあるが，ネットワーク負荷が低いときはリピータ，高いときはバッファ用のストレージとして動作する．一方，本論文ではバッファ等のルータ機能をマルチステージ化し，リンク上に分散配置する．分散された各機能は，文献 8)，9) のように状況に応じてバイパスしたり，バッファ用のストレージとして使うことができる．

3. Rearrangeable NoC アーキテクチャ

3.1 RNoC の概要

RNoC では，オンチップルータの機能をバッファと制御ロジックから成る単純なユニットに分解し，リピータバッファの置き換えとして，リンク上に分散配置する．

通常のネットワークでは，**図 4** に示すように，バッファや制御ロジックなどのルータ機能はルータモジュールという「枠」の中に集中して実装される．しかし，今後ますます配線遅延

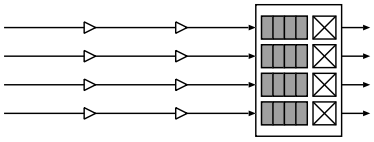


図 4 複数ネットワークのバッファ構成（オリジナル）.

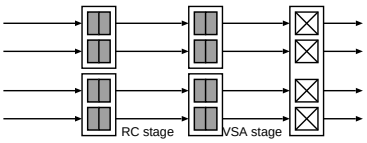


図 5 複数ネットワークのバッファ構成（分散型）.

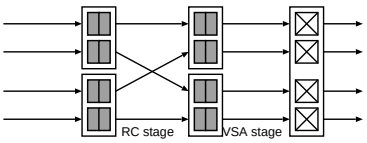


図 6 複数ネットワークのバッファ構成（分散型 + MIN 接続）.

の影響が大きくなるため，図 4 のように長い配線を残すより，**図 5** のようにルータ処理のためのバッファを分散配置してマルチステージ化したほうが配線遅延の点で有利となる．さらに各ステージ内でバッファを共有すればハードウェア量の削減も期待できる．この例では最初のステージで経路を計算し（RC），次ステージでアービトレーション（VA/SA）を行うため，パケット転送サイクル数は既存のルータと同じである．

図 6 のように複数ネットワーク間を MIN 状（この例では 2-ary 2-fly Butterfly）に接続することできる．ただし，以降では議論を単純にするため，図 5 の場合のみを考える．

図 7 と **図 8** に，通常の NoC と RNoC アーキテクチャのさらに詳細な図を示す.

3.2 RNoC ユニット

図 8 の RNoC の例では，WH ネットワーク 2 本，CS ネットワーク 1 本を持つ．各ルータの機能は **図 9(a)** から **図 9(c)** に示す 3 つのユニットに分解される．以下ではこの 3 種類のユニットについて説明する．

3.2.1 Post Xbar ユニット（ST/RC）

前段クロスバからの出力フリットを入力とする．前段クロスバからの出力のラッチ（ST ステージに相当）と，次段クロスバのための経路計算（RC ステージに相当）を行う（図 9(a)）．複数ネットワーク間で共有するバッファストレージ，WH ネットワーク向けの経路計算ユニット（RC）を持つ．

ST 用バッファ

前段クロスバからの出力をラッチしないと，クロスバをまたいだ長い配線が形成され，動作周波数に影響が出る恐れがある．その一方で，DVFS 等のために低速で動作している場合は，動作周波数さえ MET すれば ST 用のバッファをバイパスすることでもできる．ST 用バッファをバイパスするとパケットのバッファリング回数が減るため，通信サイクル数，および，消費エネルギーが削減される．その一方で，トラフィック負荷が高いときは積極的にバッファリング用のストレージとして使うことでスループット性能を改善できる．

RC ユニット（WH のみ）

次段クロスバのために，宛先アドレスからどの出力ポートを使うかを計算する．RC の結果は Pre Xbar におけるアービトレーション作業で使用される．RC は WH ネットワークのみ必要であり，CS ネットワークでは不要である．

3.2.2 Pre Xbar ユニット（VA/SA）

Post Xbar ユニットからの出力フリットと経路計算の結果

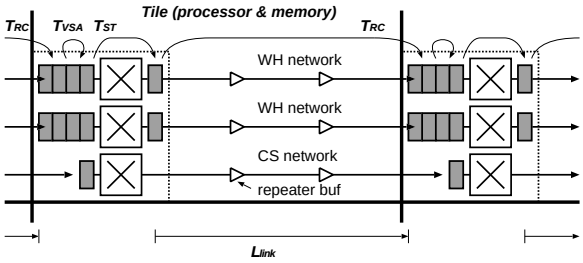


図 7 一般的な networks-on-a-chip の例（WH 2 本，CS 1 本）.

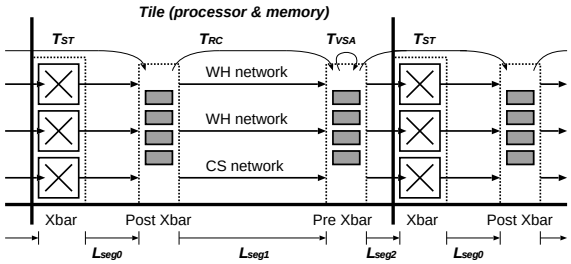


図 8 Rearrangeable networks-on-a-chip（RNoC）の例.

（出力ポートの情報）を入力とする．複数ネットワーク間で共有できるバッファを持ち，次段クロスバのアービトレーションと，アービトレーション中のバッファリングを行う（**図 9(b)**）．

VA/SA 用バッファ

出力仮想チャネルの割り当て（VA ステージ）と次段クロスバの割り当て（SA ステージ）を行うための要求を出す．VA と SA の割り当て完了を意味する grant 信号がアサートされるまで，パケットをバッファリングする．なお，CS ネットワークでは VA/SA 処理は不要であるため，単なるバッファとして利用される．バッファリングせずとも動作周波数を満足できる場合，本ユニットをバイパスすることで通信サイクル数および消費エネルギーを抑えることができる．

3.2.3 Xbar ユニット

Pre Xbar ユニットからの出力を入力とし，クロスバスイッチを介して指定された方向へパケットを転送する．Xbar ユニットの出力は次の Post Xbar ユニットに入力される．ネットワークの個数分のクロスバスイッチを持ち，クロスバごとに VA/SA アービタ（WH ネットワークの場合），または，コンフィグレーションメモリ（CS ネットワークの場合）を持つ．

VA/SA アービタ（WH のみ）

Pre Xbar ユニットが指定した出力ポート（port）と要求信号（req）を受け，それに応じて出力仮想チャネル，および，クロスバの出力ポートを割り当てる．

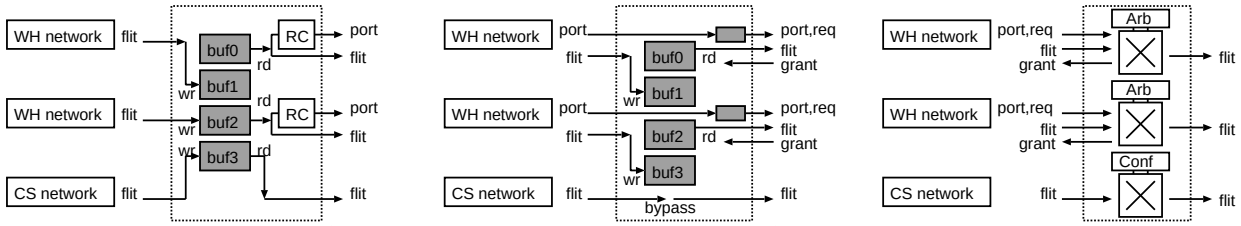
クロスバの構成データメモリ（CS のみ）

CS ネットワークではアービタの代わりにクロスバスイッチの設定を保持する構成データメモリを持つ．構成データメモリは，別の WH ネットワーク，もしくは，CS を制御するための軽量な専用ネットワークを用いて設定する．

3.2.4 その他の拡張

半導体技術の微細化や動作電圧の低減に伴ってソフトエラーに対するエラーマージンが小さくなってきているため，今後，通信中のエラーを検出，もしくは，訂正する機能が必要になる可能性がある．RNoC ではオプションとして，以下のような ECC 機能を提供するユニットをネットワーク中に配置することも考えられる．

- エラー検出符号（e.g., CRC8 checker）
- エラー訂正符号（e.g., Hamming encoder/decoder）



(a) Post Xbar ユニット（ST/RC）. (b) Pre Xbar ユニット（VA/SA）. (c) Xbar ユニット.

図 9 Rearrangeable NoC ユニットの例（WH ネットワーク 2 本，CS ネットワーク 1 本）.

4. Rearrangeable NoC の設計

まず，RNoC における WH ルータおよび CS ルータの遅延モデルを示す．そして，これらをもとに各ユニットのレイアウト方法，および，各ユニットの利用（コンフィグレーション）方法を議論する．

4.1 RNoC WH ルータの遅延モデル

図 8 に示した RNoC の WH ルータは 図 2 と同様に RC，VSA，ST の 3 つのステージから構成される．したがって，各ステージの遅延のうち最も大きい値が最大遅延 T_{WH} となる．

図 8 に示すように，Xbar から Post Xbar，Post Xbar から Pre Xbar，Pre Xbar から Xbar の配線長をそれぞれ L_{seg0}，L_{seg1}，L_{seg2} とする．また，それぞれの配線遅延を T_{seg0}，T_{seg1}，T_{seg2} とするとき，各ステージの遅延は以下のように表記できる（他の変数の意味は 2.1 節で説明したとおり）.

$$\begin{aligned} T_{RC} &= T_{seg1} + \max(T_{rc}, T_{fifo\_wr}) \\ T_{VSA} &= \max(T_{va}, T_{sa}) + 2T_{seg2} \\ T_{ST} &= T_{fifo\_rd} + T_{seg2} + T_{cb,n} + T_{seg0} \end{aligned} \tag{5}$$

つまり，RNoC では L_{link} を 3 つのセグメントに分割することで，T_{link} を各パイプラインステージに振り分けている．配線遅延は配線長の 2 乗に比例して増加するので，このように分割することで高い最高動作周波数を狙える．

逆に DVFS 等により，要求される動作周波数が非常に低くなっている場合，Post Xbar ユニットでのバッファリングをバイパスすることもできる．このときの最大遅延は $T_{WH} = \max(T_{VSA}, T_{ST} + T_{RC})$ となる．

4.2 RNoC CS ルータの遅延モデル

CS ルータの場合は，どの RNoC ユニットでバッファリングするか，バイパスするかによって遅延モデルが変化する．

RNoC ユニットを 1 個飛ばしとするとき（ある Post Xbar ユニットをバイパスする場合，または，ある Pre Xbar ユニットをバイパスする場合）の CS ルータの最大遅延 T_{CS} は次式で計算できる．

$$T_{CS}^{1hop} = T_{seg0} + T_{seg1} + T_{seg2} + T_{cb,n} \tag{6}$$

これは 1-cycle で L_{link} 分の距離を移動することに相当する．

次に，ある Pre Xbar ユニットとそれに続く Post Xbar ユニットをバイパスする場合を考える．このときの T_{CS} は，

$$T_{CS}^{2hop,\alpha} = T_{seg0} + 2T_{seg1} + T_{seg2} + T_{cb,n}. \tag{7}$$

一方，ある Post Xbar ユニットとそれに続く Pre Xbar ユニットをバイパスする場合は，

$$T_{CS}^{2hop,\beta} = 2T_{seg0} + T_{seg1} + 2T_{seg2} + 2T_{cb,n}. \tag{8}$$

以上より，RNoC ユニットを 2 個飛ばしするときの最大遅延は次のように表記できる．

$$T_{CS}^{2hop} = \max(T_{CS}^{2hop,\alpha}, T_{CS}^{2hop,\beta}) \tag{9}$$

これは 1-cycle で $3L_{link}/2$ 分を移動することに相当する．

同様の方法を用いれば，RNoC ユニットを 3 個飛ばし，4 個飛ばし，...，n 個飛ばしするときの T_{CS} も計算できる．

4.3 RNoC ユニットのレイアウト方法

ルータの最大動作周波数を向上させるには $T_{RC} = T_{VSA} = T_{ST}$ となるように Xbar，Post Xbar，Pre Xbar の各ユニットをレイアウトすれば良い．具体的には，以下の戦略で L_{seg0}，L_{seg1}，L_{seg2} を調節する．

- $\max(T_{RC}, T_{VSA}, T_{ST}) = T_{RC}$ なら L_{seg1} を短くする．
- $\max(T_{RC}, T_{VSA}, T_{ST}) = T_{VSA}$ なら L_{seg2} を短くする．
- $\max(T_{RC}, T_{VSA}, T_{ST}) = T_{ST}$ なら L_{seg1} を長くする．

既存の NoC では設計時に T_{link} を考慮して，各ステージの遅延を均等に揃えることは当然できない．一方，RNoC では 3 つのユニットの間隔を調整することで，各ステージの遅延を調節でき，最大動作周波数を高めることができる．

なお，$L_{seg0} \to 0$ かつ $L_{seg2} \to 0$ とするとき，通常のルータと同等のレイアウトとなる．これは配線遅延 T_{link} が十分に小さい場合に最適なレイアウトである．

4.4 コンフィグレーション方法

前節では，最大動作周波数を最適化するための RNoC のレイアウト方法について議論した．一方で，消費電力を削減するためにネットワークの動作周波数を低く抑えることもあり[10]，本研究が想定しているように複数系統のネットワークを持つようなシステムではその傾向が特に顕著である．

与えられた動作周波数を MET できる限り，RNoC ユニットをバイパスすることでバッファリング回数を減らすことができる．これによって，パケットの転送サイクル数を減らし，バッファリングに要すエネルギーを削減できる．このようなバイパス操作は，一度に多量のデータ転送が発生する CS ネットワークでとくに効果が大きい．

CS ネットワークの場合，許容される最大遅延が T のとき，バッファリング回数を $2/(n+1)$ に削減できる．ただし，n は $T_{CS}^{nhop} \le T < T_{CS}^{(n+1)hop}$ を満たす正の整数とする．

WH ネットワークでは効果はさほど期待できないが，$\max(T_{VSA}, T_{ST} + T_{RC}) < T$ なら Post Xbar ユニットでのバッファリングを省略できる．これにより，ルータのパイプライン段数は 3 段から 2 段に短縮され，通信遅延と消費エネルギーも減る．

5. 考　　察

本章では RNoC アーキテクチャを，最大動作周波数，フリット転送エネルギー，無負荷時レイテンシ，ハードウェア量について考察する．

5.1 評価パラメータ

予備評価の準備として，以降で用いるパラメータを示す．

RNoC アーキテクチャにおける動作周波数，および，消費エネルギーと通信サイクル数（RNoC ユニットのバイパス回数）は，使用するプロセスの配線遅延に強く依存する．まず，

表 1 オリジナル WH ルータのゲート遅延.

	delay [FO4s]	description
T_{rc}	11.3	routing computation
T_{vsa}	29.2	VC & switch allocation
T_{fifo_rd}	12.0	FIFO read
T_{fifo_wr}	21.2	FIFO write
T_{cb5}	18.5	5 × 5 crossbar

文献 5) に掲載されている 100nm，70nm，50nm，35nm プロセスの semi-global wire の配線抵抗と配線容量をもとに配線遅延を見積もった．**図 10** にリピータバッファを挿入しない unbuffered line の配線遅延 [FO4] を示す．

次に，ルータの遅延モデルで登場した各種パラメータ（T_{rc}，T_{vsa}，T_{fifo_rd}，T_{fifo_wr}，T_{cb5}）を求める．今回は，文献 10) で用いた WH ルータ回路を Synopsys Design Compiler を用いて 65nm プロセスで合成し，それぞれの遅延 [FO4] を見積もった．結果を **表 1** に示す．

5.2 最大動作周波数

オリジナルと RNoC の WH ルータの最大遅延を比較する．

表 1 より，オリジナルの WH ルータの最大遅延は次式で計算できる．

$$T_{WH}^{Orig} = \max\{(T_{link} + 21.2), 30.5\} \tag{10}$$

ただし，T_{link} を 1 本の unbuffered line としては配線遅延が大きくなり過ぎて RNoC と公平に比較できない．そこで，L_{link} を 3 つのセグメントに分割し，セグメント間にリピータバッファを挿入する場合を想定して T_{link} を計算した．

同様に RNoC WH ルータの最大遅延は次式で計算できる．

$$T_{WH}^{RNoC} = \max\{\ (T_{seg1} + 21.2), (29.2 + 2T_{seg2}), \\ (30.5\ + T_{seg0} + T_{seg2})\} \tag{11}$$

RNoC では T_{WH}^{RNoC} が最小になるように L_{seg0}，L_{seg1}，L_{seg2} を調節することで配線遅延を 3 つのステージに分散できる．

図 10 の 50nm プロセスを想定して，リンク長を 0mm から 10mm に変化させたときの T_{WH}^{Orig} と T_{WH}^{RNoC} を**図 11** に示す．リンク長が 3mm を超えた付近から RNoC WH の最大遅延が通常の WH より小さくなり始め，リンク長 5mm では 19.5% の削減になっている．プロセスの微細化によって，クリティカルパスにおける配線遅延のウェイトが大きくなれば，さらに差が大きくなると考えられる．

5.3 最適なセグメント長

図 12 に T_{WH}^{RNoC} が最小となる L_{seg0}，L_{seg1}，L_{seg2} を示す．リンク長 1mm がときのように配線遅延が十分に小さい場合は $L_{seg0} \to 0$ かつ $L_{seg2} \to 0$ となり，オリジナルの WH ルータと同じレイアウトになる．一方，リンク長が長くなるにしたがい配線遅延のウェイトが大きくなり，L_{seg0} および L_{seg2} が伸び始める．オリジナルの WH ルータではリンク遅延を 1 つのステージ内に収めなければならないが，RNoC では配線遅延を 3 つのステージに分散できる．そのため，RNoC は 5.2 節で示したように，配線遅延が増大しても動作周波数を高く保つことができる．

5.4 ユニットのバイパス率

5.2 節とは逆に本節では動作周波数が低い場合を考える．ネットワークが低い周波数で動作するとき，CS ネットワークでは許容される最大遅延に応じていくつかの RNoC ユニットをバイパスできる．これによりパケット当たりのバッファリング回数が減り，消費エネルギーと通信サイクル数が減る．

ハイエンドプロセッサでは各パイプライン段の遅延は 15 FO4 以下に抑える必要がある．その一方で，数百 MHz 程度で動作する組込み用途もあるため，ここでは最大遅延 T が 40 FO4，60 FO4，80 FO4，100 FO4 の場合を考える．

ここでは，図 10 の 50nm プロセスを想定して，リンク長を 0mm から 10mm に変化させたとき，CS ネットワークで 1-cycle に移動できる距離（単位はホップ数 ☆）を計算した．

図 13 に結果を示す．当然，最大遅延 T の制約が緩いほうが 1-cycle で移動できる距離が長い．例えばリンク長が 3.0mm のとき，$T = 100$ では 1-cycle で 2.5 hop 分，$T = 80$ では 2.0 hop 分，$T = 60$ では 1.5 hop 分，$T = 40$ では 1.0 hop 分移動できることになる．これによって，どれだけ CS ネットワークの通信サイクル数とフリット転送エネルギーが減るかは 5.5 節と 5.6 節で議論する．

5.5 無負荷時レイテンシ

CS ネットワークの無負荷時レイテンシ D_{CS} [cycles] は次式で計算できる．

$$D_{CS} = D_{ST} \cdot h + L/\text{bandwidth} \tag{12}$$

ただし，D_{ST}，h，L は，それぞれ CS ルータのフリット転送サイクル数，平均ホップ数，パケット長とする．図 13 で示したように 1-cycle 当たりの移動距離が大きくなるにしたがい h の値が小さくなり，D_{CS} が削減される．

ここでは $D_{ST} = 1$，$L = 8$ として，k-ary 2-mesh で k を 4 から 11 に変えたときの D_{CS} を求めた．リンク長を 3.0mm としたときの結果を **図 14** に示す．通常の CS ネットワークの無負荷時レイテンシは $T = 40$ の場合と同じである．このように動作周波数に応じて RNoC ユニットを適切にバイパスすることで，$k = 10$ のとき最大 29.4 % の通信サイクル数を減らすことができた．

5.6 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，これに要す平均転送エネルギー E_{flit} は次式で計算できる．

$$E_{flit} = wh(E_{sw} + E_{link}) \tag{13}$$

ただし，w を 1-flit のビット幅，h を平均ホップ数，E_{sw} をルータが 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ルギーとする．

ここでは $w = 64$，$E_{sw} = 0.12$ [pJ/bit]，$E_{link} = 0.15$ [pJ/bit/mm] として CS ネットワークの E_{flit} を計算した．RNoC の場合 1-cycle 当たりの移動距離が大きくなるにしたがい h の値が小さくなり，E_{flit} が削減される．結果を **図 15** に示す．通常の CS ネットワークの E_{flit} は $T = 40$ の場合と同じである．このように動作周波数に応じて RNoC ユニットを適切にバイパスすることで，$k = 10$ のとき最大 14.5% のフリット転送エネルギーを減らすことができた．

5.7 ハードウェア量

NoC 用のローコストなオンチップルータのバッファ量は 4-flit 程度であるが，これ以上バッファ量を削減すると通信遅延に影響が生じる．しかし，ネットワークのピークスループット時においても，トラフィックの空間的，あるいは時間的な局所性から各チャネルのバッファ利用率は 100% に近くなることはない．したがって，同一リンク上の複数ネットワーク間でバッファを共有すれば，その分バッファ量を減らすことができる．例えば，オンチップルータの機能を分解し，ネットワーク上に分散配置することで，ハードウェア量が 30% 増えたとしても，バッファを共有することでバッファ量を半分に減らすことができれば，十分に利得が出ると考えられる．

☆ 例えば，リンク長が 2mm のとき 1-cycle で 2-hop 移動できるなら，ラッチせずに 1-cycle で 4mm 移動できることになる．

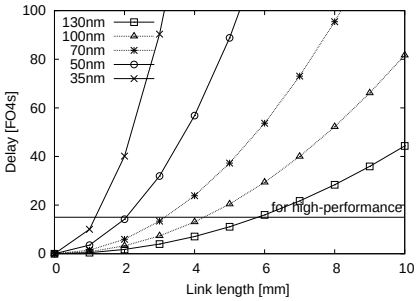


図 10 リピータ無し semi-global wire の配線遅延．配線抵抗，配線容量は文献 5) の値を使用．

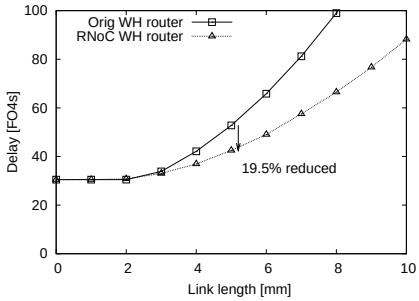


図 11 オリジナルと RNoC の WH ルータの遅延．テクノロジは文献 5) の 50nm プロセス．

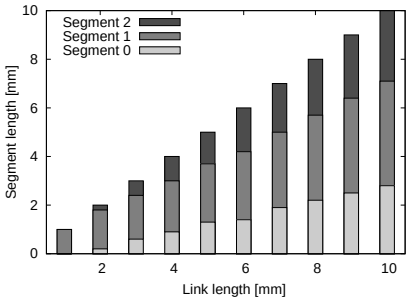


図 12 RNoC における最適なセグメント長．テクノロジは文献 5) の 50nm プロセス．

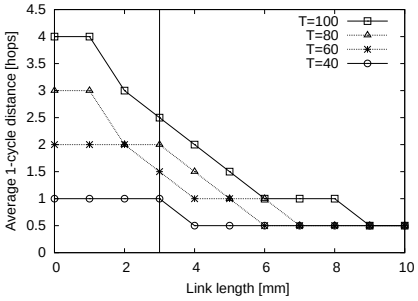


図 13 RNoC CS における平均 1-cycle 距離．テクノロジは文献 5) の 50nm プロセス．

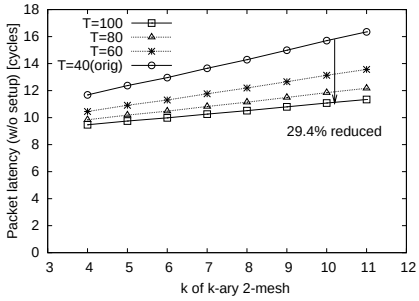


図 14 RNoC CS における無負荷時レイテンシ．リンク長 3.0 [mm]．パケット長 8-flit．

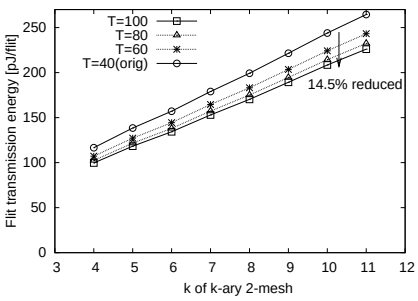


図 15 RNoC CS におけるフリット転送エネルギー．リンク長 3.0 [mm]．フリット幅 64 [bits]．

5.8 リーク電力

5.4 節で議論したように，動作周波数に応じて不要なユニットをバイパスできる．この分利用するバッファ量が減るため，power gating によって使われないバッファへの電力供給を止めてリーク電力を減らすことができる．また，5.7 節で考察したようにバッファの共有によってトータルのハードウェア量を削減できればその分，リーク電力も削減できる．

6. まとめと今後の課題

RNoC アーキテクチャでは，オンチップルータの機能を Pre Xbar ユニット，Xbar ユニット，Post Xbar ユニットの 3 種類に分解し，リンク上に分散配置する．RNoC は既存の NoC と比べて以下の 4 点の利点がある．

最大動作周波数の向上： NoC のクリティカルパス遅延のうち，配線遅延の占めるウェイトが今後ますます大きくなる．RNoC では RNoC ユニットの位置を調節することでリンクの配線遅延を各パイプラインステージに振り分けることができる．各パイプラインステージの遅延が均等になるように RNoC ユニットをレイアウトすることで，リンク長が 5mm のとき最大動作周波数が 24.2% 向上した．

消費エネルギーの削減： 一方，低い周波数で NoC を動作させるとき，許容される最大遅延に応じていくつかの RNoC ユニットをバイパスする．これによりパケット当たりのバッファリング回数が減り，10-ary 2-mesh における CS のフリット転送エネルギーを最大 14.5% 削減できた．

通信サイクル数の削減： 同様に 10-ary 2-mesh における CS の無負荷時レイテンシを最大 29.4% 削減できた．

ハードウェア量の削減： RNoC では Pre Xbar ユニットと Post Xbar ユニットにバッファを持たせ，これらのバッファを同一リンク上の複数のネットワーク間で共有する．おのおののネットワークのバッファ利用率はそれほど高くないため，バッファを複数ネットワーク間で共有することでトータルのバッファ量を削減できる．

本論文では，図 5 のように複数ネットワーク間で行き来しない場合を考えたが，図 6 のように，複数ネットワーク間を MIN 接続することも考えている．例えば，途中まで CS ネットワークを使い，ブロッキングしたら WH に切り替えるというような使い方や，頻繁に使う経路を優先的に CS で流すような使い方が考えられる．また，複数ネットワークのつなぎかたとして 図 6 のような 2-ary 2-fly Butterfly の他に Clos 網などさまざまな結合方法が考えられる．ルータをリンク上にどのようにマルチステージ化するかは今後の検討課題とする．

参 考 文 献

1) Dally, W. J. and Towles, B.: Route Packets, Not Wires: On-Chip Interconnection Networks, *Proceedings of the Design Automation Conference*, pp. 684–689 (2001).
2) Benini, L. and Micheli, G.D.: *Networks on Chips: Technology And Tools*, Morgan Kaufmann (2006).
3) Taylor, M. B. et al.: The Raw Microprocessor: A Computational Fabric for Software Circuits and General Purpose Programs, *IEEE Micro*, Vol. 22, No. 2, pp. 25–35 (2002).
4) Wentzlaff, D. et al.: On-Chip Interconnection Architecture of the Tile Processor, *IEEE Micro*, Vol. 27, No. 5, pp. 15–31 (2007).
5) Ho, R., Mai, K.W. and Horowitz, M.A.: The Future of Wires, *Proceedings of the IEEE*, Vol. 89, No. 4, pp. 490–504 (2001).
6) Kumar, A. et al.: A 4.6Tbits/s 3.6GHz Single-cycle NoC Router with a Novel Switch Allocator in 65nm CMOS, *Proceedings of the International Conference on Computer Design (ICCD'07)*, pp. 63–70 (2007).
7) Jerger, N. E., Peh, L.-S. and Lipasti, M.: Circuit-Switched Coherence, *Proceedings of the International Symposium on Networks-on-Chip (NOCS'08)* , pp. 193–202 (2008).
8) Mizuno, M., Dally, W. J. and Onishi, H.: Elastic Interconnects: Repeater-Inserted Long. Wiring Capable of Compressing and. Decompressing Data, *Proceedings of the International Solid-State Circuits Conference (ISSCC'01)* (2001).
9) Kodi, A. K., Sarathy, A. and Louri, A.: Adaptive Channel Buffers in On-Chip Interconnection Networks — A Power and Performance Analysis, *IEEE Transactions on Computers*, Vol. 57, No. 9, pp. 1169–1181 (2008).
10) Matsutani, H., Koibuchi, M., Wang, D. and Amano, H.: Adding Slow-Silent Virtual Channels for Low-Power On-Chip Networks, *Proceedings of the International Symposium on Networks-on-Chip (NOCS'08)* , pp. 23–32 (2008).