計算機アーキテクチャ　128－18
ハイパフォーマンス　70－18
コンピューティング
（１９９８．３．６）

軽量メモリベース通信のためのネットワークインタフェース

周東 福強† 山本 淳二‡ 西 宏章† 天野 英晴† 工藤 知宏‡

†慶應義塾大学 理工学部

‡新情報処理開発機構

本稿では，軽量なメモリベース通信を支援するネットワークインタフェースの設計と予備的な評価について述べる．ネットワークインタフェースは，高速なノード間通信をマルチユーザ・マルチタスク システムで実現するために，異機種接続・ゼロコピー通信・アドレス変換機構・プロテクションの機能を提供する．また，分散環境における並列処理を実現するために分散共有メモリやメッセージパッシングをサポートする通信プリミティブ処理機能を提供する．シミュレーションを用いた検証から，このネットワークインタフェースは最大 124MBytes/sec のバンド幅を実現することが可能である．

Network Interface for Memory-based Light-weight Communication

Fukukyoh Sudoh† Junji Yamamoto‡ Hiroaki Nishi† Hideharu Amano†
Tomohiro Kudoh‡

†Keio University
‡Real World Computing Partnership

This paper presents the design and performance evaluation of the network interface which supports memory-based light-weight communication. Network interface provides following functions: heterogeneous interconnection, zero-copy protocol, address translation mechanism for global address space, and protection among different processes. These functions are provided to execute the communication primitives which supports applications with distributed shared memory and message passing. Through simulation, it is verified that the network interface can deliver the maximum bandwidth of 124MBytes/sec.

1 はじめに

現在，我々は高速 LAN 上の分散環境において，ユーザのニーズに合った最適な計算パワーをネットワーク上の計算機群から引き出すシームレス並列分散システムの研究を行っている [1].

分散環境における並列処理では，ノード間通信の性能が全体の性能を左右する要素になる．TCP/IP ベースの通信では，ソフトウェアオーバヘッドが非常に大きく，ハードウェアのピーク性能により近づいた通信機構を実現することは不可能である．よって，メモリベースのユーザレベル通信と統合されたネットワークインタフェースを実現することが重要になる．また，分散共有メモリやメッセージパッシングといったアプリケーションを開発する上で，使いやすい上位レベル API(Application Program Interface) を提供することも重要である．

本稿では，軽量なメモリベース通信を支援するネットワークインタフェースの設計と性能評価に関して述べる．このネットワークインタフェースは，高速なノード間通信・マルチタスクでのプロセス間のプロテクション・他ノードの仮想アドレス空間への透過的なアクセスなどの機能を提供する．

2 MLC-1 の概要

現在，シームレス並列分散システムの開発の一環として，Memory-based Light-weight Communication mechanism -1 の開発を行っている [2]. MLC-1 は，シームレス並列分散システムを実現するテストベッドである．構成ノードには市販の PC を使用し，ネットワークルータおよびネットワークインタフェースは MLC-1 専用に開発する．通信媒体には，ギガ bps クラスの通信性能を持つ光インタコネクションと Gigabit Ethernet を用い，ネットワークルータ⇔ネットワークインタフェース間の高いバンド幅を実現する．以下に，MLC-1 を構成する OS と通信アーキテクチャについてその特徴を示す．

2.1 オペレーティングシステム

MLC-1 では，グローバルな OS を実現する上で必要なソフトウェアレベルの機能を，デバイスドライバを用いて提供する．これにより，既存の OS を利用し，カーネルに改変を施すことなくシステムを実現することが可能になる．MLC-1 では，OS に Linux を用いる．Linux は，x86・680x0・Alpha・Sparc 等のアーキテクチャをサポートするマルチユーザ/マルチタスクの OS である [3].

2.2 ネットワーク

MLC-1 のネットワークは，最大 1024 ノードを接続することが可能であり，トポロジは接続するノード数により，自由に構成することができる．ネットワークを構築するには，現在開発中のルータを複数組合せ，ルータボックスを構成し，これらを相互に接続する．ルータは，単体で 6 台のノードを接続することができ，point-to-point の送信やマルチキャストを自らが保持するルーティングテーブルを用いて行う．パケット転送は Virtual Cut Through 方式 [4] で行い，最大 4096 バイトのデータを単一パケットで送信する．このため，ルータチップ外部にメモリを用意し，パケットを直ちに転送できない場合は，外部メモリに退避する．

2.3 ネットワークインタフェース

ネットワークインタフェースはネットワークルータとともに，MLC-1 における高速かつ効率的なノード間通信を実現する通信ハードウェアである．ネットワークインタフェースはノード間通信に必要な処理をハードウェアで実行することにより，オーバヘッドを低減する．ネットワークインタフェースは，MLC-1 に特化した通信アーキテクチャを持ち，システムの全体的な性能を左右する高スループット/低レイテンシのノード間通信を実現する．

MLC-1 を実現する上で，ネットワークインタフェースには，3 つの機能が求められる．まず，異機種計算機を相互接続する機能である．シームレス並列分散システムでは異機種計算機をサポートしなければならないため，この機能を提供する．次に，高速かつ効率の良いノード間通信を実現するメモリベース通信である．メモリベース通信のためには，(1) 低レイテンシ/高バンド幅を実現するゼロコピー通信，(2) 透過的なグローバルアドレス空間へのアクセスを提供するアドレッシング機構，(3) プロセス間の不当干渉を防ぐプロテクションの 3 つの機能が必要になる．そして最後に，並列アプリケーションを効率良く実行するために必要な通信プリミティブの処理機能である．

3 異機種接続

プロセッサアーキテクチャ・ノード構成・ノード性能の異種性をサポートするためには，異機種計算機を相互接続する必要がある．より性能の高い通信アーキテクチャを求めるならば，メモリバスを用いるのが望ましい．しかし，メモリバスの仕様は計算機のアーキテクチャに依存するため，すべての計算機をサポートするためには，各仕様に合わせたネットワークインタフェースを設計する必要がある．これに対して，I/O バスは一般的に標準化されているため，多くの異機種計算機をひとつのネットワークインタフェースで同時にサポートすることが可能である．また，I/O バスのバンド幅の向上も著しく，大容量のデータ転送を伴うアプリケーションにも適用可能になってきている．

特に PCI バスは，現在生産されている多くの PC/WS に標準的な I/O バスとして採用されている [5]. バンド幅も充分に広く，効率的なノード間通信と異機種接続の点において，MLC-1 に適していると言える．よって，ネットワークインタフェースは PCI バスを介した異機種計算機の相互接続を実現する．

4 メモリベース通信

並列処理の最大の特徴はノード間の通信と同期であり，効率の良い並列実行環境のためには，高速なユーザレベル通信が必要である [6]. また，高速なノード間通信を実現するには，データをあるノードの仮想アドレス空間から別のノードの仮想アドレス空間に直接転送するメモリベース通信が有効である．

4.1 ゼロコピー通信

Myrinet[7] や ATM[8] に代表される高速な結合網が研究・開発されている．しかし，実質的な通信性能は，通信プロトコルの処理にかかるオーバヘッドがボトルネックになり，常にハードウェアの能力を充分に引き出すのは困難である．更に，システムコールを用いた通信では，カーネル内でのデータのコピーが発生し，通信のバンド幅がメモリコピーのバンド幅に支配されてしまう．そこで，ネットワークインタフェースはデータを主記憶から直接転送するユーザレベルのゼロコピー通信を実現することにより，メモリコピーバンド幅に制限されない，高速なノード間通信を提供することが可能になる．

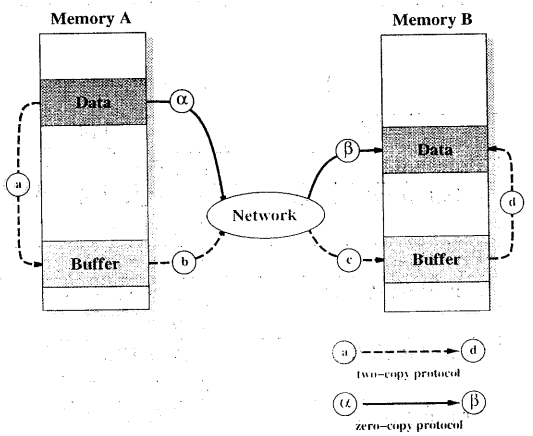

図 1: ゼロコピー通信

コピーを必要とするノード間通信では，送信するデータを一旦送信バッファにコピーしてからネットワークに送り出している．受信側も，データを一旦受信バッファで受け取り，送信先のアドレスに書き込む．よって，送信側・受信側の主記憶内でデータコピーが合計 2 回発生する．これに対して，ゼロコピー通信は，送信するデータを I/O バスを介した DMA 転送により直接ネットワークに送り出してしまう．受信側も，ネットワークから到着するデータを DMA 転送で I/O バスから主記憶に直接書き込む．図 1にゼロコピー通信のデータパスを示す．ゼロコピー通信は，主記憶中のメモリコピーバンド幅に制限されない，I/O バスのピーク性能に匹敵するバンド幅を実現することが可能である．

4.2 アドレス変換機構

仮想アドレスから仮想アドレスへの DMA 転送では，グローバルアドレス空間への透過的なアクセスが必要になる．ネットワークインタフェースは，仮想・物理・ネットワーク仮想の 3 種類のアドレスを取り扱い，ノード間とノード内のメモリアクセスを円滑に行なう．ネットワークインタフェースは処理内容に応じて，適切なアドレス変換をページ単位で行なう．アドレス変換は，(1) 仮想アドレス→物理アドレス，(2) 仮想アドレス→ネットワーク仮想アドレス，(3) ネットワーク仮想アドレス→仮想アドレスの 3 種類が存在し，ネットワークインタフェースは各アドレス変換にページテーブルをひとつ保持する．図 2にアドレスの相対的な関係を示す．

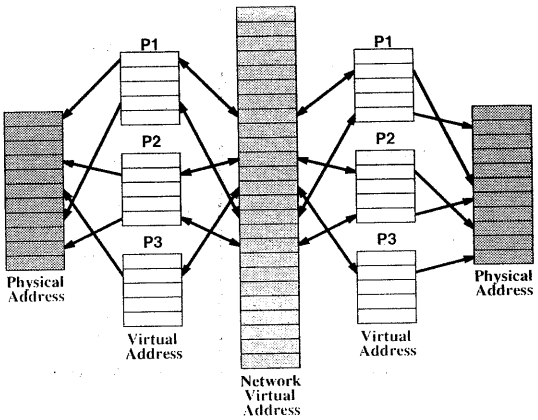

図 2: アドレス変換機構

- 仮想アドレス:
  プロセス内で使用するアドレスであり，通信時にネットワークインタフェースに通知されるアドレスはすべて仮想アドレスである
- 物理アドレス:
  主記憶をアクセスする際，ネットワークインタフェースは仮想アドレスを物理アドレスに変換する必要がある
- ネットワーク仮想アドレス:
  ノード間通信に用いるアドレスであり，送信側では仮想アドレスからネットワーク仮想アドレスへの，受信側では逆の変換を行なう必要がある (グローバルアドレス空間に相当する)

4.3 プロテクション

マルチユーザ・マルチジョブのシステムでは，各プロセスが使用する資源を保護する機能を提供しなければならない．プロセスが実行するユーザレベル通信も同様である．メモリベースのユーザレベル通信の処理は，ネットワークインタフェースが行なうため，ネットワークインタフェースは通信を起動した

プロセスがアクセスする領域に対して有効な権限を持っていることを確認する情報を持たなければならない．通常，ホストプロセッサでは，プロテクションを実現するに当たって，仮想記憶管理機構 (MMU) を利用する．ユーザプロセスは，MMU を介して割り当てられた物理ページにアクセスするため，MMU はジョブ間の不当干渉を排除することができる．

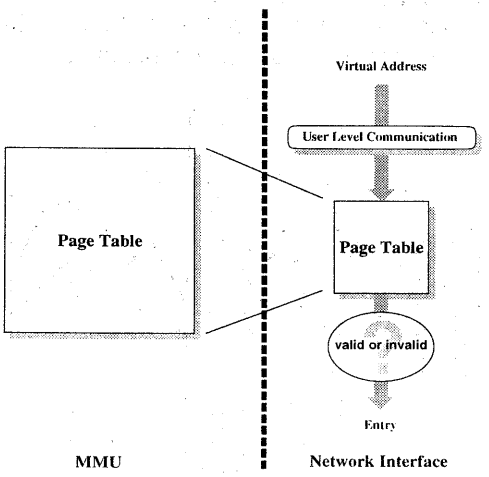

図 3: ページテーブル参照による資源保護

ネットワークインタフェースは，アドレス変換を行なうために既に 3 つのページテーブルを保持している．このページテーブルには，仮想アドレスと対応する物理・ネットワーク仮想アドレス等が保持される．ネットワークインタフェースはこの情報を用い，MMU がジョブ間の不当干渉を排除する方法と同様の方法で，プロテクションを実現する．

図 3にページテーブル参照によるプロテクションの実現方法を示す．ネットワークインタフェースは，ユーザレベル通信を処理するためにページテーブルを参照する．与えられた仮想アドレスに対応するエントリの有効性を確認し，エントリに含まれる物理アドレスを用いて主記憶にアクセスする．仮想アドレスから物理アドレスへの変換は，ノード内のプロテクションを目的とするのに対して，仮想アドレスからネットワーク仮想アドレスへの変換とその逆変換は，ノード間のプロテクションを目的とする．

また，ネットワークインタフェースがメモリアクセスを行なう際，該当する領域はピンダウンされている必要がある．ピンダウンされていない場合，ノード間通信を処理している途中でページのスワップアウトが発生し，異なるデータをアクセスしてしまうからである．そこで，ピンダウンすることにより，特定のデータ領域を主記憶に留めておくことができる．

5 通信プリミティブ

分散環境における並列処理を実現するために，ネットワークインタフェースは，表 1に示す 11 個の通信プリミティブを処理する．これらを用い，分散共有メモリモデルおよびメッセージパッシングモデルのアプリケーションを作成することができる．通信プリミティブ処理の基本方針は，有限資源を用いずに処理を行なうことである．これにより，ネットワークインタフェースを軽量なハードウェアで実現することができる．必要な資源はすべてホスト側で提供し，ネットワークインタフェースは通信プリミティブの処理に応じてデータ等を主記憶から取得する．

表 1: プリミティブ一覧

プリミティブ	説明
PrimPull	リモートリード
PrimPullWithTwin	TWIN 作成を伴う リモートリード
PrimPush	リモートライト
PrimPushRegions	領域指定による リモートライト
PrimPushDiff	差分抽出による リモートライト
PrimPPBCast	リモートライト (マルチキャスト)
PrimPPBCastRegions	領域指定による リモートライト (マルチキャスト)
PrimLock	ロック獲得
PrimUnlock	ロック解除
PrimBarrier	バリア同期
PrimBarrierWithPPBCast	バリア同期完了 時にマルチキャスト

ネットワークインタフェースは能動的に処理を開始することができないため，ユーザプロセスにより起動される．ユーザプロセスは，ネットワークインタフェース起動の前処理として，実行する通信プリミティブの詳細を記した管理構造体を主記憶中に作成する．管理構造体を作成した後，ユーザプロセスは，その構造体のアドレス (仮想) をネットワークインタフェースに通知する．ネットワークインタフェースは，通知された仮想アドレスに存在する管理構造体を主記憶から読み出し，プリミティブ処理を開始する．

6 ネットワークインタフェースの構成

ネットワークインタフェースの構成ブロックを図4に示す．PCIバスに接続される本ボードは以下のブロックにより構成される：(1)PCIバスインタフェースチップ，(2)アドレス変換用ページテーブル，(3)再送用パケットを保持するパケットバッファ，(4)光モジュール·Gigabit Ethernetへのインタフェースを提供する伝送モジュールインタフェース，(5)ネットワークから到着するパケットを一旦バッファリングするFIFO，(6)Multiple Writer Protocol[9]をハードウェアでサポートするTWINメモリ，(7)Multiple Writer Protocolをソフトウェアでサポートする Region Buffer，(8)すべてのブロックを制御するプリミティブ処理ユニット．プリミティブ処理ユニット以外は，すべて既製のチップ/メモリを使用している．プリミティブ処理ユニットは，AHDL(Altera Hardware Description Language)によりその機能を記述し，ALTERA社MAX+PLUS IIを用いて開発を行なった．ターゲットデバイスには，FLEX10K(EPF10K100GC503-3)を選択した．

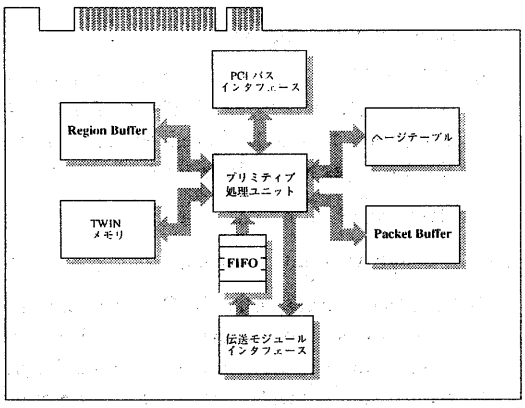

図4: ネットワークインタフェースの構成

7 性能評価

本研究で設計·実装したネットワークインタフェースの性能をMLC/Myrinetにおけるノード間通信の性能と比較し，その有効性を述べる．MLC/Myrinetは，既存の通信ハードウェアを用いて分散共有メモリをエミュレートするシステムである[10]．MLC-1との大きな違いは，ネットワークインタフェースにおける通信プリミティブの処理をLANaiと呼ばれるプロセッサにより行なっていることである．

MLC-1とMLC/Myrinetの基本的なデータ転送性能を示し，アプリケーションを実行した場合に得られる性能向上を予測する．

7.1 データ転送性能

基本的なデータ転送性能を測るために行なった，MLC-1とMLC/Myrinetにおける4 ~ 12Kバイトの PUSH プリミティブのバンド幅を図5に示す．8Kバイト以上の転送サイズでは，MLC-1·MLC/Myrinetともに PCI バスのピークバンド幅に近付いたスループットを実現していることが分かる．

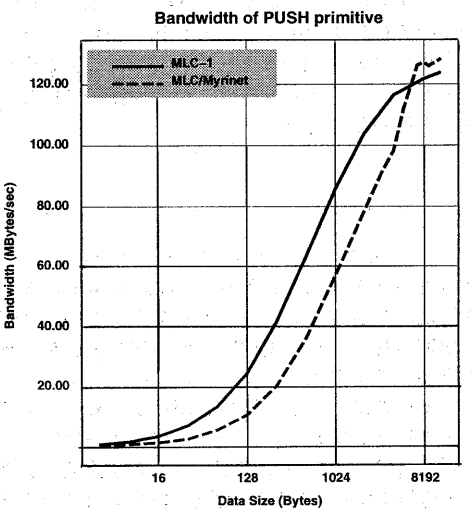

図5: PUSH バンド幅

7.2 性能比較

MLC/Myrinetにおける SPLASH2[11] の FFT の実行時間と通信回数を表2に示す．FFT では，1回の通信で256バイトを転送する．

表2: FFT 処理

PU 台数	実行時間 (msec)	通信回数
1	4.13	—
2	3.00	128
4	2.82	384
8	2.25	896

256バイトのデータ転送1回当り，MLC/Myrinetでは12.78μsec，MLC-1では6.15μsecの処理時間を要する．データ転送にかかるオーバヘッドのみに注目した場合，FFTでは，表3に示す実行時間の短縮を見込める．

表 3: 実行時間の比較 (単位: msec)

PU 台数	MLC-1	MLC/Myrinet
2	2.57568	3.00
4	2.18352	2.82
8	1.50744	2.25

MLC-1 は，それぞれ MLC/Myrinet の 85.86%(2 台)，77.43%(4 台)，67.00%(8 台) の実行時間で処理することが可能という計算になる．8 台では，通信回数の総数が 2 台で実行した場合の 7 倍になるため，全体の実行時間の多くをデータ転送のオーバヘッドが占めている．結果として，MLC-1 によるデータ転送オーバヘッドの改善は，通信回数が多い場合に大きく影響してくる．

8 まとめ

MLC-1 用ネットワークインタフェースを設計し，その性能を評価した．MLC-1 は分散環境における並列処理を目的としたシステムであるため，高速かつ効率の良いノード間通信が全体的な性能を左右する重要な要素になる．MLC-1 用ネットワークインタフェースは，以上の要件を満たすために，メモリベース通信を実現する．メモリベース通信は，ゼロコピー通信・グローバルアドレスへの透過的なアクセスを可能にするアドレス変換機構・プロセス間のプロテクションの機能を必要とする．本研究で設計したネットワークインタフェースは，これらの機能の他に，シームレスな環境を実現する上で必要な異機種接続性を提供する．また，分散共有メモリやメッセージパッシング型のアプリケーションを開発する際に，使い易いインタフェースを提供できるよう，PUSH や PULL 等の通信プリミティブを提供する．ネットワークインタフェースは通信プリミティブを処理する回路を中枢とし，アドレス変換や DMA 転送等を行なう．

MLC-1 用ネットワークインタフェースは，124M-Bytes/sec のバンド幅を提供し，プロセッサベースのネットワークインタフェースを用いた MLC/Myrinet と同等以上のスループットを実現している．また，FFT を 8 台で実行した場合，MLC/Myrinet の約 67% の処理時間で実行可能である．

謝辞

軽量メモリベース通信に関する議論を通じて貴重なご意見を頂いた，新情報処理開発機構 石川 裕氏，佐藤 三久氏に感謝致します．ならびに，SPLASH2 の実行結果を提供して頂いた 東京工科大学 上野 茂紀氏，NEC 情報システムズ 宮脇 達郎氏に感謝致します．

参考文献

[1] 石川 裕, 佐藤 三久, 工藤 知宏, 秋山 泰, 島田 潤一．分散環境におけるシームレス並列コンピューティングシステムの構想．*CPSY*, No. 51-63, pages 83-90, 1997.

[2] Tomohiro Kudoh, Junji Yamamoto, Yutaka Ishikawa, Mitsuhisa Sato, Fukukyoh Sudoh, Hideharu Amano. Memory based light weight communication architecture for local area distributed computing. *IWIA*, 1997.

[3] http://www.linux.org/

[4] Parviz Kermani and Leonard Kleinrock. Virtual Cut-Through: A New Computer Communication Switching Technique. *Computer Networks*, pages 267-286, Vol. 3, No. 4, 1979.

[5] http://www.pcisig.com/

[6] 松本 尚, 平木 敬. 汎用並列オペレーティングシステムにおける資源保護と仮想化. 情報処理学会研究報告 97-OS-75, 情報処理学会, Vol. 97, No. 56, pages 37-42, June 1997.

[7] http://www.myri.com/

[8] http://www.atmforum.com/

[9] Christiana Amza, Alan L. Cox, Sandhya Dwarkadas, and Willy Zwaenepoel. Software DSM Protocols that Adapt between Single Writer and Multiple Writer. In *Proceedings of the Third High Performance Computer Architecture*, pages 261-271, February 1997.

[10] 上野 茂紀, 山本 淳二, 宮脇 達郎, 工藤 知宏. 軽量メモリベース通信機構への SPLASH2 の移植と評価. *CPSY*, No. 101-111, pages 23-28, 1998.

[11] Steven Cameron Woo, Moriyoshi Ohara, Evan Torrie, Jaswinder Pal Singh, and Anoop Gupta. The SPLASH-2 Programs: Characterization and Methodological Considerations. *Proceedings of the 22nd Annual Intrenational Symposium on Computer Architecture*, pages 24-36, June 1995.