

RHiNETの高速通信ライブラリPMv2による評価

原 田 浩^{t1,t3} 山 本 淳二^{t1,t4} 土 屋 潤一朗^{t2}
渡 辺 幸之介^{t2} 天 野 英 晴^{t2}
工 藤 知 宏^{t1} 石 川 裕^{t1,t5}

我々は高速ネットワーク RHiNET 上にクラスタ向け高速通信ライブラリ PMv2 を移植し, RHiNET の性能評価を行った. その結果メッセージ通信においてラウンドトリップ 16 μ sec, 帯域 297MBytes/sec, ゼロコピー通信において書き込み, 読み込みそれぞれ 469MBytes/sec, 185MBytes/sec を記録した. また, ソフトウェア分散共有メモリシステムである SCASH 上で動作するラプラス変換プログラムを実行し, 4 台で 2.57 倍の台数効果が得られた.

Evaluation of RHiNET network using PMv2 communication library

HIROSHI HARADA^{t1,t3} JUNJI YAMAMOTO^{t1,t4}
JUNICHIRO TSUCHIYA^{t2} KOUNOSUKE WATANABE^{t2}
HIDEHARU AMANO^{t2} TOMOHIRO KUDOH^{t1}
and YUTAKA ISHIKAWA^{t1,t5}

We implemented high speed communication library PMv2 on RHiNET and evaluated its performance. PMv2 is a communication library for cluster computing, and RHiNET is a network for high performance computing which can connect personal computers. The round trip of a message was 16 μ sec and the message bandwidth was up to 297Mbytes/sec. Bandwidth of zero-copy communication was up to 469Mbytes/sec for remote write and 185Mbytes/sec for remote read. Laplace transform implemented on SCASH software distributed memory system yields 2.57 times performance using 4 personal computers.

1. はじめに

RHiNET はローカルエリアネットワークとシステムエリアネットワークの2つの長所を合わせ持つことを目標に研究開発が進められている高速ネットワークである. 我々は RHiNET 上にクラスタ計算機向け高速通信ライブラリ PMv2¹⁾ を移植し, RHiNET のクラスタ計算機のノード間通信ネットワークとしての性能を検証した. PMv2 は SCORE クラスタシステムで用いられているクラスタ向け高速通信ライブラリであり, Myrinet, Ethernet などの複数ネットワークへの対応, ゼロコピー通信²⁾ のサポートなどの特徴を持っている. 本稿では, 最初に RHiNET の特徴を紹介する. 次に, RHiNET 上への PMv2 の実装方法について説明する. RHiNET の評価方法と評価結果を述べ

た後, 評価結果に関して考察を行う. 最後に, まとめと今後の課題を述べる.

2. RHiNET

RHiNET は, ビル内やフロア内に分散して配置された計算機同士を接続して, 高性能並列処理環境を実現するために開発されたネットワークであり, そのハードウェアは計算機に装着されるネットワークインタフェース, ネットワークスイッチ, およびそれらを接続する光インタコネクションにより構成される.

RHiNET ネットワークは物理層でパケットを破棄することがないため, Myrinet³⁾ などの System Area Network(SAN) で広く用いられているユーザレベル通信, ゼロコピー通信を用いて, 低遅延・高バンド幅の通信を実現できる.

RHiNET のネットワークインタフェースには, Martini^{4), 5)} と呼ぶ専用のネットワークインタフェースコントローラが搭載され, ユーザレベル通信, ゼロコピー通信をハードウェアによってサポートする.

t1 技術研究組合 新情報処理開発機構

t2 慶応義塾大学

t3 現在: コンパックコンピュータ (株)

t4 現在: (株) 日立製作所

t5 現在: 東京大学大学院情報理工学系研究科

ユーザレベル通信はユーザプロセスが直接ネットワークインタフェース (NI) にアクセスして通信を起動する方式である。

また、ゼロコピー通信はユーザプロセスのアドレス空間上にあるデータを NI が直接アクセスしてネットワークへ送信したり、ネットワークから受信したデータを直接ユーザプロセスのアドレス空間上の受信領域へ書き込むことである。

このユーザレベル通信とゼロコピー通信を併用することで、通常の通信に必要なシステムコールやユーザ空間とカーネル空間の間で行われるコピーなどを省けるため、低レイテンシで広バンド幅な通信が可能である。

ユーザレベル・ゼロコピー通信では、ユーザプロセスが送信データや受信バッファのアドレスを直接 NI に伝え、NI はデータやバッファに対して DMA でアクセスする。ユーザプロセスが使用するアドレスは全て仮想アドレスであるが、一般に DMA 時に必要なアドレスは実アドレスである。そのため、NI は仮想アドレスから実アドレスへ変換する機構を備える必要がある。

また、プロセス間の不当な干渉を防ぐために通信の保護機構が必要である。ユーザレベル通信では通信に関わる情報がユーザプロセスから直接 NI に渡されるため、NI がその情報の正当性を判断する仕組みが必要である。

Martini はユーザレベル・ゼロコピーによるリモートメモリへのライト機構 (PUSH プリミティブ) とリード機構 (PULL プリミティブ) などの単純だが高速性が要求される通信をハードウェアで処理し、それ以外の複雑な操作が必要な通信や生起率の低いイベントに対してはコアプロセッサを用いてソフトウェア処理することで、高速性と柔軟性の両方を実現する。

PUSH/PULL プリミティブをユーザレベルでこれを実現するために、Martini は内部に独自の TLB を持つ。また、PUSH/PULL プリミティブではデータ転送に DMA を用いているが、DMA 転送は少量のデータ転送時にはオーバーヘッドが大きい。そこで Martini は少量のデータをより低いレイテンシで転送するためにホストプロセッサからの PIO により送信を行う Block On The Fly (BOTF) 機構と Atomic On The Fly (AOTF) 機構を備える。これらもハードウェアにより処理される。

通信プリミティブ自体は、window と呼ばれる Martini 上のメモリ領域に対して必要な情報を書き込むことで起動する。window はホストプロセッサの仮想記憶機構によってプロセスに割り付けられており、割り付けられたプロセスのみがプリミティブを発行できる。Martini 内にも管理テーブルがあり、発行されたプリミティブの内容が正当かどうかを判定する。これによりプロセス間の保護を実現している。

PUSH プリミティブが起動すると、まず PUSH パケット発行側で Martini 内の TLB を参照し、転送するメモリ領域の仮想アドレスを物理アドレスに変換する。次にその DMA 要求を発行して、物理アドレスの領域からネットワークに対してデータを DMA 転送する。

PUSH パケット受信側では、Martini 内の TLB を参照して、まず転送先の領域の仮想アドレスを求め、次にそのアドレスを物理アドレスに変換し、DMA 転送によってデータを書き込む。

DMA 終了後、指定があれば、PUSH パケット受信側のネットワークインタフェースによって PUSH プリミティブ完了を示すパケットが PUSH パケット発行側のホストへ転送され、PUSH プリミティブを起動したプロセスのメモリ領域の指定アドレスにフラグがセットされる。これにより PUSH プリミティブ完了をプロセスが検知する。

一方、PULL プリミティブが起動すると、PULL パケット発行側で受信領域や要求データ等の情報を含んだパケットが生成され、ネットワークへ送出される。

PULL パケット受信側では、Martini 内の TLB を参照により転送領域の仮想アドレスを取得し、さらに物理アドレスに変換する。次に DMA 要求を発行し、このアドレスから、データを PULLED パケットとしてネットワークへ DMA 転送する。

PULL パケット発行側では、PULLED パケットに書かれた受信領域を物理アドレスに変換し、そこへ DMA 転送で PULLED パケットのボディを書き込む。DMA 転送完了後は、PULL プリミティブを起動したプロセスのメモリ領域の指定アドレスに、PULL プリミティブ完了を示すフラグがセットされる。これにより PULL プリミティブの完了をプロセスが検知する。

しかし、現在運用している第二次試作 Martini チップでは、PULL プリミティブを処理する回路に問題点があり、PUSH プリミティブと同時に使用すると誤動作する。このため、ハードウェア処理の PULL はアプリケーションでは使用することができず、オンチッププロセッサを用いてハードウェア処理の一部エミュレーションする事で実装している。

オンチッププロセッサは R3000 と命令互換の 32bit RISC であり、例外処理や PUSH/PULL プリミティブ以外の通信機構のネットワークインタフェース単体での実行に利用される。ハードウェア制御コアと並列に動作することが可能な上、ハードウェア制御コアの一部モジュールをステートレベルまで詳細に制御することができる。

なお、本報告の評価では AOTF, BOTF 通信機構は用いていない。

RHiNET のスイッチは 4 種類の実装・設計が行われている。本報告の評価ではこのうち RHiNET-2/SW⁶⁾ を用いた。RHiNET-2/SW に接続されるネットワーク

インタフェースを RHiNET-2/NI と呼ぶ。RHiNET-2/SW, RHiNET-2/NI は 各方向 8Gbps の帯域を持つ双方向光インタコネクションをリンクに用いている。リンク長はそれぞれ最大 100m であり、自由なトポロジのネットワークを用いることができる。

3. PMv2 の実装

PMv2 は、クラスタ計算機向けに低遅延、高帯域幅を目指して開発された通信ライブラリであり、これまでに Myrinet, Ethernet などのネットワーク上に実装されている。我々は、RHiNET をクラスタ計算機用ネットワークとして評価するため、RHiNET 上に PMv2 を移植した。

PMv2 は通常のメッセージ通信の他に、ゼロコピー通信機能を提供している。PMv2 では、遠隔メモリ書き込みと遠隔メモリ読み込みの2つが規定されている。

以下、RHiNET 上でのメッセージ通信とゼロコピー通信機構の実装について述べる。

3.1 メッセージ通信

RHiNET は通信機構として、PUSH により遠隔メモリ書き込みを、PULL により遠隔メモリ読みだしをサポートしている。通常のパケット通信機構はオンチッププロセッサのプログラムにより実装することを予定しているが、まだ実装が完了していない。また、オンチッププロセッサの代行による PULL の実装は高い性能は得られない。そのため、PMv2 のメッセージ通信機構は全て PUSH を用いて以下のように実装した。

- メッセージバッファ
各ノードがそれぞれ、全ノード分の送受信バッファを準備する。送受信バッファは PMv2 の初期化時にあらかじめピンダウンされる。
 - 送信処理
メッセージの送信ノードは、送信バッファの内容を PUSH によって送信先ノードの受信バッファに転送し、さらに、メッセージの大きさに応じて受信ノードの受信バッファポインタを進める。
 - 受信処理
受信ノードは、各ノードの受信バッファポインタを調べて、メッセージの到着を検知する。メッセージバッファが開放された場合、受信バッファの開放ポインタを進め、PUSH を用いて、開放ポインタを送信ノードにも転送する。開放ポインタを転送することにより、送信側は、送信先の受信バッファのオーバーランを防ぐことができる。
- しかし、以上の実装では以下のような問題点がある。

(1) 送受信バッファの増大

全ノードの送受信バッファを準備する必要があるため、送受信バッファの全サイズはノード数

に比例する。そのため、ノード数が大きな大規模クラスタへの適用が困難である。

(2) バッファポインタの不整合

バッファポインタを送信側、受信側でそれぞれ管理しているため、なんらかの原因で、バッファポインタに不整合が起きた場合、メッセージ通信が止まってしまうだけでなく、再開させることができない。

3.2 ゼロコピー通信

PMv2 では、遠隔メモリ書き込みと遠隔メモリ読み込みの2つのゼロコピー通信をサポートしている。2つのゼロコピー通信は共に、あらかじめピンダウンしたユーザ空間内の領域に対して、直接メモリの内容を読み書きするものである。RHiNET では、ユーザ空間中のメモリのピンダウン及び、PUSH, PULL 機能がサポートされている。

そこで、遠隔メモリ書き込みは PUSH を、遠隔メモリ読みだしは PULL を用いて、PMv2 のゼロコピー通信を実装した。ゼロコピー通信は基本的に RHiNET の NI によって処理されるため、ホスト側プロセッサのオーバーヘッドはほとんどない。

4. 評価

RHiNET 上に実装した PMv2 通信ライブラリを用いて、以下の3つについて性能を測定する。

4.1 メッセージ通信

通信機構の基本評価として、メッセージ通信の遅延時間とバンド幅を測定する。メッセージ通信の遅延時間として2つのノード間でのメッセージの往復時間を測定する。実際の測定はメッセージを 100 万回往復させた時間を測定し、1 回あたりの往復時間を求める。メッセージサイズは 4Bytes から、8KBytes まで測定する。

メッセージ通信のバンド幅として、2ノード間で同じ大きさのメッセージをバースト的に 100 万回送信する時間を測定し、1 秒当たりの転送バイト数を求める。メッセージサイズは 4Bytes から、8KBytes まで測定する。

4.2 ゼロコピー通信

次にゼロコピー通信機構の評価として、遠隔メモリ書き込みと読みだしそれぞれのバンド幅を測定する。2ノード間で同じ大きさのメモリ領域を 100 万回書き込み及び読み込みを行う時間を測定し、1 秒当たりの転送バイト数を求める。

4.3 アプリケーションによる評価

最後に、実際のアプリケーションを用いた評価を行う。アプリケーションとしてはソフトウェア分散共有メモリシステムである SCASH⁷⁾ 上で動作する Laplace 変換のプログラム LAPLACE を採用することにした。SCASH は、PMv2 を用いているだけでなく、バリア

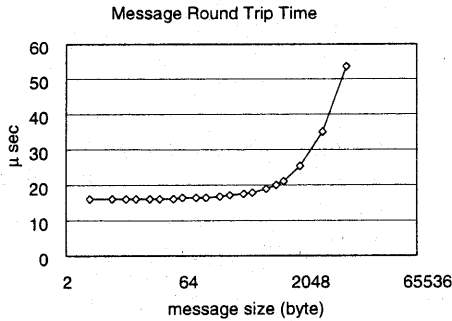


図1 メッセージ遅延
Fig. 1 Message Latency

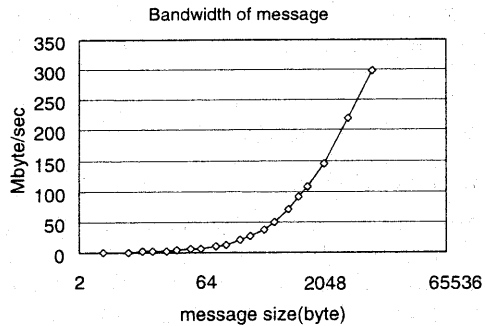


図2 メッセージバンド幅
Fig. 2 Message Bandwidth

同期等でメッセージ通信を用いる一方、ノード間のページコピーに遠隔メモリ読み込みを採用しており、メッセージ通信、ゼロコピー通信の2種類の通信機構を使用しているため評価に適していると考えた。ノード数1,2,4の場合のLAPLACEの実行時間を測定する。評価に用いる問題サイズを表1に示す。

表1 LAPLACE 問題サイズ

Maytrix	2048x2048
Iteration	50

4.4 評価環境

最後に評価に用いたPCクラスタの主な仕様を表2に示す。

表2 PCクラスタの主な仕様

# of Node	4
CPU	Intel PentiumIII 933MHz
Chipset	Serverworks Serverset LE
Memory	512MBytes/Node
Network Interface	RHiNET-2/NI
SWITCH	RHiNET-2/SW
PCI	64bit 66MHz
Node OS	Linux 2.2

5. 評価結果と検討

5.1 メッセージ通信

測定結果から得られた、メッセージ通信の遅延とバンド幅をそれぞれ図1、図2に示す。

Martiniはハードウェアによるメッセージ通信はサポートしておらず、その実現法としては、大きく分けて

- Martiniのオンチッププロセッサのプログラムにより実装する。
- ホストプロセッサ上のプログラムにより実装する。

の2者がある。

今回の評価では後者の方法を用い、RHiNETのPUSH/PULLプリミティブを用いるホストプロセッサ上のプログラムによりメッセージ通信を実装した。

メッセージ通信における最小ラウンドトリップ時間は約16μ秒である。メッセージの送受信においては、送信側が2回、受信側が1回遠隔メモリ書き込みを実行するので、メッセージが2つのノード間を往復するには、2つのノードでそれぞれ3回ずつ遠隔メモリ書き込みを実行することになる。評価に用いたRHiNETでは、遠隔メモリ書き込みを一回実行することにNICの状態レジスタを確認する必要があるため、連続的に遠隔メモリ書き込みを発行することが出来ない。連続的に遠隔メモリ書き込みを発行することが出来れば、さらに遅延時間を短縮することが可能と考えられる。

次にバンド幅については、メッセージサイズ8Kバイトで300Mbytes/secに迫る性能を記録している。同じチップセットを持つPCクラスタ上でMyrinet-2000を用いた場合メッセージのバンド幅は約200MBytes/secである。これは、Myrinetのリンクのバンド幅が2Gbpsであり、これにより通信バンド幅が制限されるためと考えられる。これに対し、RHiNET-2のリンクのバンド幅は8Gbpsであるため、1対1の通信ではリンクにより通信バンド幅が制限されることはない。

5.2 ゼロコピー通信

測定結果から得られた、遠隔メモリ書き込みと遠隔メモリ読み込みのバンド幅をそれぞれ図3、4に示す。

遠隔メモリ書き込みは、最大470MBytes/secに迫る性能を示している。PCIバス(64Bit,66MHz)のバンド幅は最大528MBytes/secなので、PCIバスのバンド幅の約89%の性能が得られていることになる。

図4において、「hardware pull」は、全てハードウェア処理によるPULLを用いた場合、「firmware pull」はオンチッププロセッサが一部代行処理を行うPULLを用いた場合のバンド幅である。

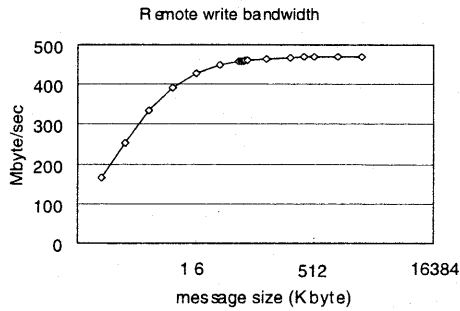


図3 遠隔書き込みのバンド幅
Fig. 3 Remote Write Bandwidth

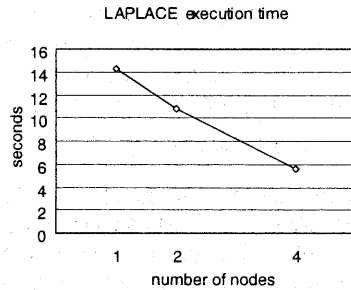


図5 LAPLACE 実行時間
図6 LAPLACE execution time

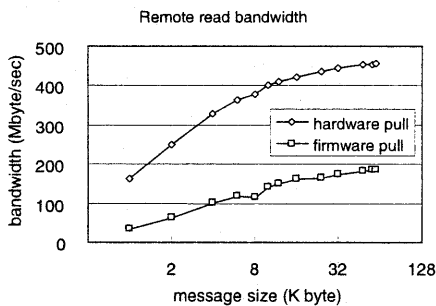


図4 遠隔読み込みのバンド幅
Fig. 4 Remote Read Bandwidth

評価に用いた第二次試作 Martini チップには、ハードウェア処理による PUSH と PULL を同時に行った場合に誤動作するという問題点があることが明らかになったため、アプリケーションで用いることが出来るのはファームウェア (オンチッププロセッサのプログラム) の一部代行制御を用いた PULL である。この場合、遠隔メモリ読み込みのバンド幅は、最大 185MBytes/sec 程度に留まっている。

ハードウェアにより遠隔メモリ読み出しを行った場合の性能 ("hardware pull") は PUSH 同様の高いバンド幅が得られている。現在、この問題点を改良した第三次試作 Martini チップを開発中であり、同チップではこの性能が得られると考えられる。

5.3 LAPLACE

測定結果から得られた、各ノード数ごとの実行時間を図5に示す。

2,4 ノードともシーケンシャル実行より高速に実行を終了し、台数効果が得られている。2台では1台の1.32倍、4台では2.57倍の台数効果が得られた。

LAPLACEに限らず SCASH 上のアプリケーション

ンでは、ページのリモートコピーに要する時間が大きく性能を左右する傾向がある。SCASH ではページのリモートコピーは、遠隔メモリ読みだしによって実装されている。現在はこの遠隔メモリ読み出しを firmware によりエミュレーションしているため性能が制限されている。LAPLACE では 4Kbytes のサイズの PULL を用いており、hardware による PULL を用いることが出来れば 328Mbytes/sec のバンド幅が得られるはずであるが、firmware による実装では 101Mbytes/sec しか得られていない。また、バリア同期は PUSH/PULL を用いたホストプロセッサ上のプログラムによって実現している。firmware によるバリア同期、メッセージ通信が実装され、Martini チップの改版によりハードウェアによる PULL が使えるようになればさらに性能の向上が期待できる。

6. まとめと課題

RHiNET 上にクラスタ計算機向け高速通信ライブラリ PMv2 を移植し、クラスタ計算機用ネットワークとしての RHiNET の性能検証を行った。その結果、メッセージ通信に関しては、ラウンドトリップタイム 16 μ sec、8KBytes のメッセージ転送で 297MBytes/sec のバンド幅を記録した。これは、ホストプロセッサ上のプログラムにより PUSH/PULL を用いて実装したメッセージ転送機能の性能である。今後、NI 上のプロセッサを用いてメッセージ転送機能を実装し評価を行う予定である。

ゼロコピー通信に関しては、書き込み、読み込みでそれぞれ、469MBytes/sec、185MBytes/sec のバンド幅を確保している。書き込みに関しては PCI バスにバンド幅の 92% を使用できているが、読み込みのバンド幅は、ハードウェアによる通信が利用できないため書き込みの 39.4% に留まっている。

また、LAPLACE の実行では、4台で 2.57 倍の台

数効果が得られた。今後、NIのハードウェアの不具合を解決するとともに、NI上のコントローラチップのプログラムを用いてメッセージ通信を実装し、さらに台数を増やして性能評価を行いたいと考えている。

参 考 文 献

- 1) Hiroshi Tezuka, Atsushi Hori, Yutaka Ishikawa, and Mitsuhsa Sato. PM: An Operating System Coordinated High Performance Communication Library. In *High-Performance Computing and Networking '97*, 1997.
- 2) 手塚 堀, O'Carroll, 原田, 石川. ピンダウン キャッシュを用いたユーザレベルゼロコピー通信. 情報処理学会研究報告. 情報処理学会, August 1997.
- 3) Myricom, Inc. <http://www.myri.com/>.
- 4) 山本淳二, 土屋潤一郎, 寺川博昭, 田邊昇, 渡邊幸之介, 今城英樹, 西宏章, 工藤知宏. RHiNETの概要とMartiniの設計/実装. In *SWoPP 2001*, ARC-144-7, pp. 37-42, Jul. 2001.
- 5) 渡邊幸之介, 山本淳二, 土屋潤一郎, 田邊昇, 西宏章, 今城英樹, 寺川博昭, 上嶋利明. RHiNET/MEMOnet ネットワークインタフェース用コントローラチップ Martini の予備評価. In *SWoPP 2001*, ARC-144-9, pp. 49-54, Jul. 2001.
- 6) 西宏章, 多昌廣治, 西村信治, 山本淳二, 工藤知宏, 天野英晴. LASN用 8 gbps/port 8times8 one-chip スイッチ: RHiNET-2/SW. In *JSP2000*, pp. 173-180, 5 2000.
- 7) 原田浩, 手塚宏史, 堀敦史, 住元真司, 高橋俊行, 石川裕. Myrinetを用いた分散共有メモリにおけるメモリバリアの実装と評価. 並列処理シンポジウム JSP'99, pp. 237-244. 情報処理学会, June 1999.