計算機アーキテクチャ 134−11
（1999. 8. 3）

GigaE PM II: Gigabit Ethernetによる高速通信ライブラリの設計

住元真司† 堀敦史† 手塚宏史†
原田浩† 高橋俊行† 石川裕†

我々は、Gigabit Ethernetを用いたクラスタシステム上で並列アプリケーションを効率良く稼動するための高速通信ライブラリGigaE PMを開発している。GigaE PMでは、並列処理に必要な信頼性のある低遅延かつ高バンド幅な通信機能を提供すると共に、従来の通信プロトコルであるTCP/IPを支援している。次期GigaE PM（GigaE PM II）では、より多数のNICに対応し、かつ、ハードウェアの性能を最大限に発揮することができるよう設計している。

試作として、Packet Engines社G-NIC II NIC上で、GigaE PMプロトコルを実装した。Compaq社XP-1000での評価の結果、アプリケーションレベルで98.2 MB/sのバンド幅（1,468バイト時）、44.6 μsのラウンドトリップ遅延（8バイト時）を実現している。

GigaE PM II: Design of High Performance Comminication Library using Gigabit Ethernet

Shinji Sumimoto,† Atsushi Hori,† Hiroshi Tezuka,†
Hiroshi Harada,† Toshiyuki Takahashi† and Yutaka Ishikawa †

A high performance communication library, called GigaE PM, provides not only a reliable high bandwidth and low latency communication function but also supports an existing network protocols such as TCP/IP. The GigaE PM is only available on an Essential Communications gigabit ethernet card in which the PM protocol is handled by the onboard processor. The successor of the GigaE PM, GigaE PM II, has been designed to adapt it to other network cards. The PM protocol is handled in the kernel.

A prototype system has been implemented on Compaq XP-1000 using Packet Engines G-NIC II. The performance results show that 44.6 μs round trip time for an eight byte message and 98.2 MBytes/sec bandwidth for a 1,468 byte message are achieved.

1. はじめに

近年、Gigabit Ethernetが普及し始め、次世代のコモディティ・LANとなりつつある。より対線に対応した1000BASE/Tの標準化も終了し、低価格のGigabit Ethernet機器が広く出まわると予想され、Gigabit Ethernetを利用すれば、高性能なクラスタシステムがLAN環境上で安価に構築可能になる。

我々は、Gigabit Ethernet上で既存の通信プロトコルをサポートし、同時に、並列アプリケーションのための高バンド幅、低遅延通信を実現するGigaE PM通信ライブラリ[1)~3)]を開発している。GigaE PMはPM[4)~7)]と同一のAPIを提供しているため、PM上で開発されたアプリケーションを再コンパイルにより実行することができる。

しかし、現状のGigaE PMは、NIC上CPUのファームウェアを変更して実装する方式であり、NICにCPUを持たないGigabit Ethernet NIC、及び、CPUを持つNICでも、今後発売されると予想されるすべてのNICに対応することは困難である。

そこで、より多くのNICに対応でき、かつ、ハードウェアの性能を最大限に発揮可能な通信ライブラリGigaE PM IIの設計と試作を行なった。本稿ではGigaE PM IIの設計、プロトタイプ実装、評価について報告する。

2. PMとGigaE PM

PM[4)]は、Myrinet上で信頼性のある非同期メッセージパッシングとリモートメモリアクセスによるゼロコピー通信[6),7)]などをサポートしている低レベルの通信ライブラリで、NIC上のファームウェアとユーザレベルライブラリにより、カーネルオーバヘッドのない高

† 技術研究組合 新情報処理開発機構 つくば研究センタ
並列分散システムソフトウェアつくば研究室
Parallel & Distributed System Software Laboratory
TRC, Real World Computing Partnership
http://www.rwcp.or.jp

性能な通信機能を提供している。PMは、信頼性のあるコネクションレス通信を実現するチャネルと呼ぶ仮想ネットワークを提供しており、SunOS、NetBSD、Linuxなどに移植されている。

GigaE PM[1)~3)]は、PMのAPIをGigabit Ethernet上に実現したものである。NICとホスト間の情報交換コストを最小にして性能を引き出すために、CPUを持つNICを採用し、NICのファームウェアを変更している。また、プロテクション実現のためシステムコール経由でNIC上メモリへの更新を行なっている。GigaE PMは、Ethernetフレーム上にメッセージを構築して送受信を行ない、ルーティングは行なわないため、利用は裸のEthernetフレームが届く範囲に限られる。GigaE PMは現状Linux上で動作する。

3. 目標と課題

3.1 目　　標

GigaE PM IIの目標を以下にまとめる。なお、プラットフォームOSとしてLinuxを前提に考える。

- より多くのNICで利用可能にする。
- クラスタ通信のみでなく、既存のプロトコル通信をサポートすることにより、通常の分散アプリケーション利用と、並列アプリケーション実行を可能にする。
- クラスタ向け通信ライブラリとして高い通信性能を実現する。

3.2 課　　題

前節の目標を実現するための課題を以下にまとめる。

- **多くのNICに対応可能なアーキテクチャ：** 既存のデバイスドライバコードを利用し、デバイスドライバコードへの変更を最小にする。
- **既存の通信プロトコルとクラスタ通信の同時サポート：** GigaE PMはEthernetフレームタイプによりGigaE PM通信と既存のプロトコル通信を分類しているために、データリンク層に既存の通信プロトコルとGigaE PM用通信プロトコルの分配用のコードを挿入する必要がある。これをカーネルの変更を行なわない方法で実現する。
- **高バンド幅、低遅延クラスタ通信：** ホストとNIC間の情報交換コストを最小にする。

4. 設　　計

本章では、3章で述べた課題を解決するための以下の設計ポイントについて述べる。

- 多くのNICに対応可能なアーキテクチャ
- 既存の通信プロトコルとクラスタ通信の同時サポート
- 高バンド幅、低遅延クラスタ通信
  - ディスクリプタの配置
  - 割り込みオーバヘッドの回避
  - データコピーのオーバヘッド
  - PM ゼロコピー通信

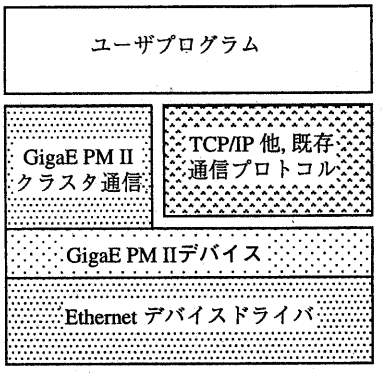

図1　GigaE PM IIのアーキテクチャ

4.1 多くのNICに対応可能なアーキテクチャ

GigaE PM IIでは、既存のデバイスドライバコードを再利用し、かつ、既存の通信プロトコルとクラスタ通信の同時サポートを行なうため、既存のEthernetデバイスドライバの上にGigaE PM IIデバイスを置き、既存の通信プロトコルに対してはEthernetデバイスドライバとして振舞い、かつ、クラスタ通信用のインターフェイスをユーザに提供するアーキテクチャとしている（図1）。カーネル変更は行なわずデバイスドライバレベルで実現する方針から、利用するEthernetデバイスドライバコードを変更している。利用するインターフェイスは、デバイスドライバとネットワークインターフェイス間の標準インターフェイスを使用することにより、多くのNICに対応が可能である。

4.2 既存の通信プロトコルとクラスタ通信の同時サポート

前節で述べたアーキテクチャとすることで、GigaE PM IIデバイスで、受信時にはEthernetフレームの分配、送信時にはEthernetデバイスドライバへの転送制御を行なうことにより既存の通信プロトコルとクラスタ通信の同時サポートを実現する。

Ethernetは、ハードウェアレベルで信頼性のあるメッセージ転送を提供していないので、メッセージの到着と順序性を保証するためのクラスタ通信プロトコルとして、GigaE PM[1)~3)]と同様、**STOP and GO**方式による受信バッファフロー制御を備えた**GO back N**方式を採用している。

4.3 ディスクリプタの配置

GigaE PMでは、ホストとNIC間の情報交換コストを最小にするため、ディスクリプタの配置をNICメモリ上においた。しかし、GigaE PM IIでは、ディスクリプタの制御はNICのハードウェアの作りとデバイスドライバコードに依存する。

4.4 割り込みオーバヘッドの回避

割り込みオーバヘッドを回避するため、GigaE PM II では受信メッセージがない場合に、Ethernet デバイスドライバの割り込みハンドラを呼び出す方式を採用している。こうすることで、受信済みで割り込み待ちのメッセージに対して割り込みを待つことなくメッセージを処理することができる。

この方式は、処理実行中に割り込みによる同処理の再実行を防ぐため、割り込み禁止で走らせる必要がある。このため、割り込み禁止時間が長くなると、他の処理に影響が出る場合があり、実行間隔が短くなり過ぎないように適切に制御する必要がある。本方式は、対象となるデバイスドライバが割り込みレベルを他のデバイスと共有可能なように適切に書かれていれば動作すると考えられるが、デバイスにより動作しない場合も有り得る。

以下に、本方式の処理手順を示す。

```
// メッセージ受信処理
ret = gigepm_get_message(channel, result);
if (ret == NO_MESSAGE) {
    disable_interrupt();     // 割り込み禁止
    do_interrupt_handler(); // 割り込み処理
    enable_interrupt();      // 割り込み許可
    ret = gigepm_get_message(channel, result);
}
```

4.5 データコピーのオーバヘッド

GigaE PM II では、既存のデバイスドライバコードを利用する方針としているため、メッセージデータのハンドリングは Linux の *sk_buff* 構造体を利用する必要がある。しかし、現状の *sk_buff* 構造体処理をそのまま踏襲すると、メッセージの送受信時にそれぞれ CPU によるメッセージのコピーが発生するために、通信性能がホスト CPU とメモリ間のバンド幅で押さえられてしまう場合がある。

GigaE PM II では、送信時のメッセージバッファは予め割り当てておき、ユーザプログラムはこのメッセージバッファ上でメッセージを作成し、GigaE PM II のデバイスドライバで *sk_buff* 構造体を作成し、*sk_buff* 構造体のデータへのポインタを送信メッセージバッファへのポインタに切替えることで、送信時のデータコピーを回避している。

しかし、受信時は、受信してみないと GigaE PM II 用のメッセージかどうか、また、GigaE PM II 用のメッセージでも、どのチャネルへのメッセージかどうか判断できない。このため、受信メッセージは *sk_buff* 構造体内のデータバッファに転送した後、振り分けるようにしている。クラスタ通信においてメッセージを GigaE PM II のチャネルに振り分ける方式は、受信メッセージバッファにコピーする方式と、*sk_buff* 構造体内のデータバッファを Read-only でマップする方法がある。

どちらの方式が良いかを調べるために、Compaq 社 XP-1000 を用いて、それぞれのコストを測定した。コピーコストは、4MB のキャッシュの効果がなくなるよう 20MB の領域に対して連続コピーした場合の測定値である。map は linux の map 関数（*remap_page_range()*）における 1 ページ（8KB）の map 時間を測定した。表 1に結果を示す。結果より、メッセージサイズが 8 バイトの時は、copy の方がコストが少なくて済むが、1,536 バイトの場合は map する方がコストが低い。

表 1　Copy と map のコスト

	コスト μs
8 バイト copy	0.05
1,536 バイト copy	4.61
1 ページ map	2.18

4.6 PM ゼロコピー通信

GigE PM II では、受信してみないと GigaE PM II 用のメッセージかどうかがわからないので、GigaE PM II では、ゼロコピー通信は実現されない。

5. 実　　装

実装に用いた Gigabit Ethernet NIC は Packet Engines 社の G-NIC II である。G-NIC II を選択した理由は数種の Gigabit Ethernet NIC を評価した結果、G-NIC II が linux 上で動作し、かつ、もっとも性能が良かったからである [8]。

5.1 実装の概要

以下に、GigaE PM II の実装の概要について述べる。

(1) 送信は Ethernet デバイスドライバの device 構造体中の送信関数（*hard_start_xmit()*）の置き換え、受信は Ethernet ドライバ中の受信関数（*netif_rx()*）関数の置き換えで実装。

(2) 送受信用メッセージバッファは、GigaE PM と同様デバイス初期化時に予め割り当てられ、Pin-down されている。ユーザはこの領域を mmap して用いる。

(3) 現状、受信メッセージの map 機能は実装されていないため、受信メッセージは受信メッセージバッファにコピーされる。

(4) ソースは、既存のデバイスドライバコードと GigaE PM II のコードから構成され、コンパイル時にリンクして GigaE PM II のデバイスドライバモジュールを構築する。

(5) クラスタ通信プロトコル処理は GigaE PM と同じ。

5.2 既存デバイスドライバへの変更

本節では、GigaE PM II の実装の詳細を述べ、利用する Ethernet NIC のドライバへの変更が少ないこ

と、及び、他社のNICでも同様に実現可能なことを示す。以下、変更に関連するLinuxの構造体と関数について説明した後、既存のデバイスドライバへの変更について述べる。

変更に関連するLinuxの構造体と関数：

- **device構造体：** Linuxのネットワークインターフェイス関連の構造体のうち、デバイス依存の情報はdevice構造体にある。この構造体の中に、デバイス依存のメッセージ送信関数へのポインタがある。
- **request_irq() 関数：** Linuxのデバイスドライバでデバイスの割り込み処理に使う関数を登録するために使う関数。この関数の引数より、実行すべき割り込み処理関数へのポインタを獲得できる。
- **netif_rx() 関数：** Linuxのネットワークを扱うデバイスドライバでメッセージ受信時に上のプロトコルスタック処理にメッセージを引き渡すために使う関数。
- **cleanup_module() 関数：** Linuxのダイナミックロード対応のモジュールでモジュールのunload時に呼び出される関数。

既存のデバイスドライバへの変更：

- **初期化コードの変更：**
  (1) *gigaepm_init()* の挿入：利用するNICデバイスのprobeが成功し、NICデバイスの初期化が終了したところに挿入する。引数は、device構造体へのポインタである。*gigaepm_init()* では、device構造体中のメンバである *hard_start_xmit* をGigaE PM IIの作業領域に記憶した後、GigaE PM II用の送信関数に置き換える。
  (2) *request_irq()* 関数の置き換え：*request_irq()* 関数を、*gigepm_request_irq()* に置き換える。引数は、*request_irq()* と同じ。*gigepm_request_irq()* では、内部で *request_irq()* を呼び出した後、割り込み処理関数へのポインタをGigaE PM IIの作業領域に記憶する。
- **デバイスunload用コードへの変更：** ダイナミックロードモジュール対応のため、*cleanup_module()* に *gigaepm_cleanup()* を挿入する。引数は、device構造体へのポインタである。*gigaepm_cleanup()* では、システム資源を解放する。
- **メッセージ受信コードへの変更：** *netif_rx* 関数をGigaE PM II用の *gigaepm_netif_rx()* に置き換える。引数は、*netif_rx()* と同じ。*gigaepm_netif_rx()* では、Ethernetフレームをチェックして、GigaE PM II用のフレームの場合はクラスタ通信プロトコル処理を行ない。それ以外のフレームの場合は *netif_rx()* を呼び出す。

以上のように、既存のデバイスドライバコードへの変更は4箇所、4行（プロトタイプ宣言を含めると8行）であり、G-NIC II以外のデバイスドライバに対しても容易に適用可能である。なお、変更のうち、関数名の置き換えは、コンパイル時のオプションで実現できるので実際の変更はさらに少なくて済む。

6. 評　価

GigaE PM IIの評価として、TCP/IPとGigaE PM IIのpoint-to-point通信性能の比較を行なう。評価環境としては表2に述べる環境を利用し測定した。測定に利用したプログラムは、GigaE PM IIは独自作成のプログラム、TCP/IPはnetperfを利用した。測定は、G-NIC IIをCompaq社XP1000の64 bit PCIバスに搭載して測定した。なお、測定にはswitchは利用していない。

表2　測定環境

ハードウェア	Compaq社 XP1000 (Alpha 21264 500MHz, 21272 チップセット, 4MB キャッシュ, 256MB SDRAM メモリ)
NIC	Packet Engines社 G-NIC II 33 MHz clock, 64 bit PCI
ホスト OS	Redhat 5.2 Linux (2.2.7 kernel)

6.1 バンド幅

図2にアプリケーションレベルのバンド幅性能の測定結果を示す。TCP/IPの最大バンド幅が59.5MB/sに対し、GigaE PM IIでは、1,468バイトメッセージ時に98.2MB/sを実現している。

図2の128バイトメッセージより下でTCP/IPのバンド幅がGigaE PM IIに比べ良いのは、linuxのTCP/IPスタックが、nodelayオプションにおいて複数の送信要求を一つのIPパケットにまとめ上げているためである。これを確認するため、ベンチマークが発行するwriteシステムコールの回数とifconfigコマンドの情報から採取したデバイスから送信されたフレーム数の比較を示す。表3に結果を示す。8バイトメッセージでは平均20回のシステムコールがまとめられ一つのEthernetフレームで送信されていることがわかる。

6.2 遅　延

図3にアプリケーションレベルのラウンドトリップ遅延の測定結果を示す。TCP/IPは、8バイトメッセージ時にラウンドトリップ遅延が139.4μsに対し、GigaE PM IIでは、8バイトメッセージ時に44.6 μsを実現している。

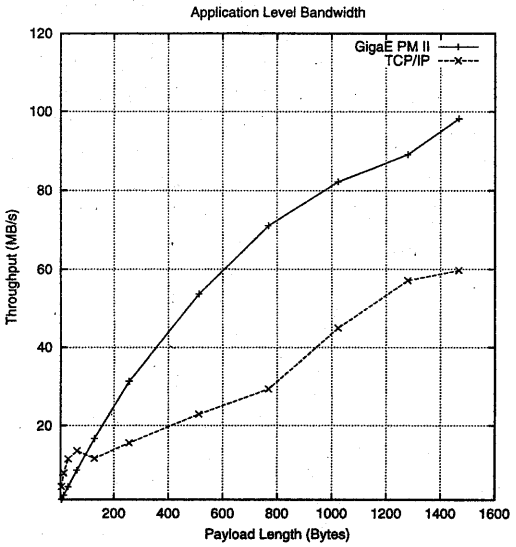

図2　バンド幅性能

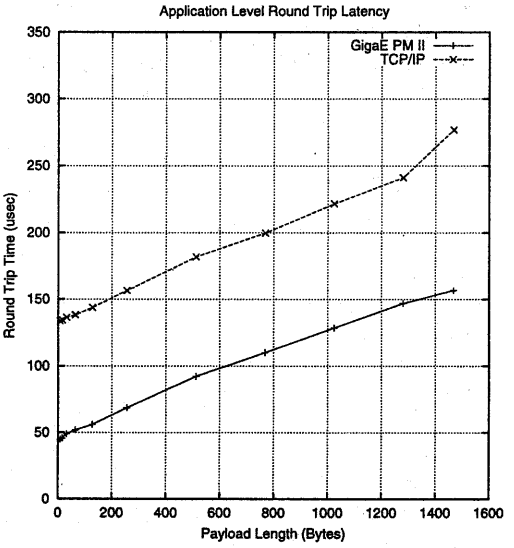

図3　ラウンドトリップ遅延

表3　TCP/IPでの送信フレーム数とシステムコール数

サイズ	送信フレーム数 (A)	システムコール数 (B)	比 (A)/(B)
8	246563	5025761	1/20.4
16	302983	4258025	1/14.1
32	392188	2912266	1/7.4
64	530531	1978826	1/3.7
128	609243	881241	1/1.5
256	566306	638908	1/1.1
512	458859	463168	1/1.0
768	407103	416269	1/1.0
1024	393911	394973	1/1.0

6.3　64 bit PCI vs 32 bit PCI

本節では、Compaq社XP-1000上において、G-NIC IIのPCI DMA転送を制御レジスタの設定により32 bit転送にした場合と64 bit転送の場合のGigaE PM IIのバンド幅性能を比較する。

図4に測定結果を示す。結果より、64 bit PCIの98.2 MB/sに比べ、32 bit PCIでは、75.3 MB/sのバンド幅性能にとどまる。なお、32 bit PCIにおけるTCP/IPのバンド幅性能は最大51.4 MB/sであった。

6.4　高速化対策の効果

本節では設計時に述べた高速化対策の効果を検証する。効果の検証のため、以下の4つの場合におけるGigaE PM IIのバンド幅とラウンドトリップ遅延性能を測定した。1,468バイト時のバンド幅と8バイト時のラウンドトリップ遅延の測定結果を表4に示す。

- デバイスドライバ：デバイスドライバ自体の通信性能
  - バンド幅：送信側で*sk_buff*割り当て、Ethernetフレーム作成と送信、受信側で*sk_buff*を解放するドライバを作成して測定
  - ラウンドトリップ：送信側では、*sk_buff*割り当て、Ethernetフレーム作成と送信、受信側ではフレーム受信時に、送信側に送り返すドライバを作成して測定
- 割り込み対策あり：受信メッセージがない場合にデバイスドライバの割り込みハンドラを呼び出した場合の通信性能

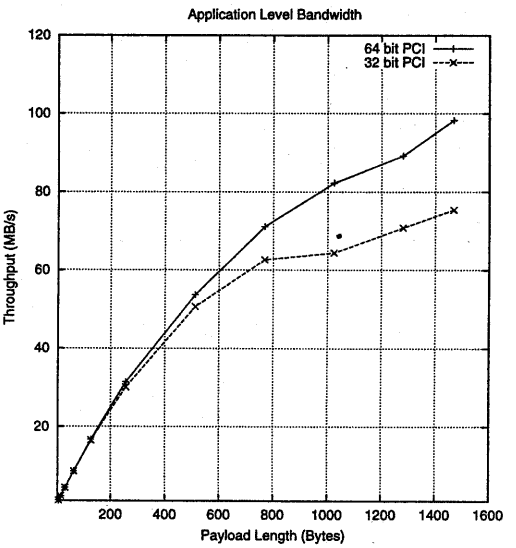

図4　バンド幅性能（64 bit PCI vs 32 bit PCI）

- 割り込み対策なし： すべての受信を割り込みを利用した場合の通信性能
- Copy なし：メッセージがない場合に割り込み対策を行ない、かつ、受信時に受信メッセージバッファへのデータコピーを行なわない場合の通信性能
  - 受信側のユーザプログラムには受信データが行き渡らないため、バンド幅性能のみ測定

表 4　高速化対策の効果

	バンド幅 (MB/s)	ラウンドトリップ (μs)
デバイスドライバ	112.6	40.7
割り込み対策あり	98.2	44.6
割り込み対策なし	96.2	49.8
Copy なし	98.5	-

表 4より、デバイスドライバレベルで 112.6MB/s であり、リンクの持つ 125MB/s のうち 90% のバンド幅性能である。高速化を行なった結果、98.2(MB/s)/112.6(MB/s)=87.2% のバンド幅性能、44.6(μs)/40.7(μs)=109.6% のラウンドトリップ遅延時間を実現している。

割り込み遅延対策ありの場合に割り込み遅延対策なしの場合に比べラウンドトリップ遅延で 5.2μs 短縮された。これは、割り込み遅延対策のコードにより、割り込みを待つことなくメッセージ受信が可能になった結果である。

また、Copy なしの性能と Copy ありの性能の差は、0.3MB/s であった。測定に利用した Compaq 社 XP1000 では、メモリコピーバンド幅性能が 336MB/s☆と高いため、データ 1 回のコピーがバンド幅に及ぼす影響は少ないといえる。4.5節で、copy より map の方がコストが低いと述べたが、copy のオーバヘッド自体は実際のバンド幅への影響は少なかった。これは、CPU のコピーバンド幅性能が PCI DMA による受信メッセージ転送バンド幅性能（最大 266MB/s）に比べ高いためであると考えられる。

7. おわりに

本論文では、Gigabit Ethernet 上で並列計算向きの高速通信を実現する GigaE PM II の設計、プロトタイプ実装と基本通信性能評価について述べた。

GigaE PM II は、既存の Ethernet ドライバへの変更を最小限に抑えながら、ハードウェア性能を最大限に発揮するよう設計された。プロトタイプを Packet Engines 社の G-NIC II 上に実装し、Compaq 社 XP-1000 上での評価では、8 バイトメッセージ時に 44.6 μs のラウンドトリップタイム、1,468 バイトメッセージ時に 98.2MB/s のバンド幅を達成している。G-NIC II のデバイスドライバへのソースコード変更は 4 行（プロトタイプ宣言を含めると 8 行）であり、他の NIC にも容易に適用可能である。

GigaE PM II は、信頼のある、高バンド幅、低遅延通信を提供し、かつ、同時に TCP/IP などの既存の通信プロトコルもサポートしている。この機能を用いれば、並列アプリケーションと分散アプリケーションが共存した高性能なクラスタシステムが構築できる。

今後、実アプリケーションを利用したスケーラビリティの検証を行なう予定である。

☆ 20MB の領域に対して連続コピーした場合の測定値

参考文献

1) 住元真司, 石川裕, 堀敦史, 手塚宏史, 原田浩, 高橋俊行. Gigabit Ethernet を用いた高速通信ライブラリの設計. 情報処理学会研究報告 98-HPC-72 (SWoPP'98), pp. 109–114. 情報処理学会, August 1998.
2) 住元真司, 堀敦史, 手塚宏史, 原田浩, 高橋俊行, 石川裕. Gigabit Ethernet を用いた高速通信ライブラリの設計と評価. 並列処理シンポジウム JSPP'99, pp. 63–70. 情報処理学会, June 1999.
3) Shinji Sumimoto, Hiroshi Tezuka, Atsuhi Hori, Hiroshi Harada, Toshiyuki Takahashi, and Yutaka Ishikawa. The Design and Evaluation of High Performance Communication using a Gigabit Ethernet. In *International Conference on Supercomputing '99*, pp. 243–250. ACM SIGARCH, June 1999.
4) Hiroshi Tezuka, Atsushi Hori, Yutaka Ishikawa, and Mitsuhisa Sato. PM: An Operating System Coordinated High Performance Communication Library. In Peter Sloot Bob Hertzberger, editor, *High-Performance Computing and Networking*, Vol. 1225 of *Lecture Notes in Computer Science*, pp. 708–717. Springer-Verlag, April 1997.
5) 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.
6) Hiroshi Tezuka, Francis O'Carroll, Atsushi Hori, and Yutaka Ishikawa. Pin-down Cache: A Virtual Memory Management Technique for Zero-copy Communication. In *IPPS/SPDP'98*, pp. 308–314. IEEE, April 1998.
7) Hiroshi Tezuka, Francis O'Carroll, Atsushi Hori, and Yutaka Ishikawa. Pin-down Cache: A Virtual Memory Management Technique for Zero-copy Communication. Technical Report TR–97006, RWC, October 1997.
8) 住元真司, 堀敦史, 手塚宏史, 原田浩, 高橋俊行, 石川裕. Gigabit Ethernet NIC の性能評価. 情報処理学会研究報告 99-HPC-75 (HOKKE'99), pp. 49–54. 情報処理学会, March 1999.