PACS-CS のための Ethernet を用いた高性能通信機構の設計

住元真司[†] 久門耕一[†] 朴泰祐[††]
佐藤三久[††] 宇川彰[††]

本論文では、PACS-CS システムのための Ethernet を用いた高性能通信機構 PM/Ethernet-HXB の設計について述べる。PACS-CS の計算ノードは Gigabit Ethernet を計算ネットワークとして 6 系統備えており、これを 3 次元に 2 系統ずつ用いた Hyper Crossbar 結合を用いた通信を行なう。PM/Ethernet-HXB の設計においては、ノード間の直接通信の他、中継ノードを経由した間接通信も並行して行うため、複数 Ethernet を用いた通信処理コストを極限まで下げることが課題である。この課題を解決するため、異なるノードの通信バッファ間での Zero-Copy 通信を実現した他、複数ネットワークからのパケット処理を低コストで行う軽量通信プロトコルを開発した。

PM/Ethernet-HXB を実装し、評価した結果、Gigabit Ethernet を 6 系統を用いた 1 方向の低レベルの通信バンド幅で 735MB/s と物理バンド幅性能の 97.9%、8 系統で 979MB/s を実現している。また、Gigabit Ethernet 2 系統を用いたルーティング性能についても、ルーティング 2 段で 237MB/s を実現しており、高い通信性能を実現している。

A Design of High Performance Communication Facility Using Ethernet for the PACS-CS system

Shinji Sumimoto,[†] Kouichi Kumon,[†] Taisuke Boku,[††]
Mitsuhisa Sato[††] and Akira Ukawa [††]

This paper discusses a design of high performance communication facility called PM/Ethernet-HXB using Ethernet for the PACS-CS system. The PACS-CS computing node has six Gigabit Ethernet interfaces for computation network, and is connected with the other nodes using three-dimensional hyper Crossbar connection. Two Gigabit Ethernet interfaces are used for each connection. In the PM/Ethernet-HXB design, communication protocol overhead must be minimized on multiple Ethernet devices, because the PACS-CS requires not only direct communications between nodes but also in-direct communication using routing nodes. To minimize the communication protocol overhead, Zero-copy communication between communication buffers of nodes is used, and a light weight communication protocol has been developed. The protocol can handle multiple packets from multiple networks.

We have implemented the PM/Ethernet-HXB on Linux, and evaluated its communication performance. The PM level communication bandwidth are 735 MB/s(97.9%) using six Gigabit Ethernet network, 979 MB/s using eight Gigabit Ethernet network. The PM level communication bandwidth with two routing nodes is 237 MB/s using two Gigabit Ethernet networks. These results show that PM/Ethernet-HXB realizes high communication performance.

1. はじめに

PACS-CS システム [1](以下、PACS-CS) は、筑波大学で開発進行中の PC クラスタシステムで、3 次元の Hyper Crossbar ネットワークを持つ。PACS-CS は、高い通信性能を安価に実現するために、計算用ネットワークに 6 系統（3 次元各方向に 2 系統）の Gigabit Ethernet ネットワークを採用している。

本論文では、PACS-CS の持つ 3 次元 Hyper Crossbar ネットワーク上で高い通信性能を実現するための通信機構である PM/Ethernet-HXB の設計について述べる。PM/Ethernet-HXB の目標は市販の Ethernet を用いてクラスタ専用インターコネクトに匹敵する通信性能を実現することにある。これを実現するためノード間の通信バッファ間での Zero-Copy 通信と複数ネットワークからのパケット処理を低コストで行う軽量通信プロトコルを開発した。

PM/Ethernet-HXB を実装し、評価した結果、Gigabit Ethernet を 6 系統を用いた 1 方向の低レベル

† 富士通研究所
FUJITSU LABORATRIES
†† 筑波大学
University of Tsukuba

の通信バンド幅で 735MB/s と物理バンド幅性能の 97.9%、8 系統で 979MB/s を実現している。また、Gigabit Ethernet 2 系統を用いたルーティング性能も、ルーティング 2 段で 237MB/s を実現しており、高い通信性能を実現している。

本論文では、第 2 章で PACS-CS の概要と通信機構の課題を整理し、第 3 章で PM/Ethernet-HXB の設計について述べる。第 4 章で PM/Ethernet-HXB の実装、第 5 章で低レベルの通信性能を評価する。

2. PACS-CS の概要と通信機構の課題

2.1 PACS-CS の概要

PACS-CS[1] は、筑波大学で開発進行中の PC クラスタシステムで、3 次元の Hyper Crossbar ネットワークを持つ。システムの概要は次の通り。

計算ノードのプロセッサ: Intel IA32 互換プロセッサで 2.8GHz 以上

計算ノードのネットワーク: クラスタ用のネットワークとして Gigabit Ethernet 6 系統、I/O 用他にも Gigabit Ethernet を搭載

計算ノード間のネットワーク結合方式: 6 系統の Gigabit Ethernet は 2 系統を 1 組として 3 次元 Hyper Crossbar ネットワーク網を構成、JUMBO FRAME (9000 バイト) をサポート

計算ノード数: 2,560(16x16x10) 台を予定

ノード OS/クラスタミドルウェア: Linux / SCore クラスタシステムソフトウェア [2]

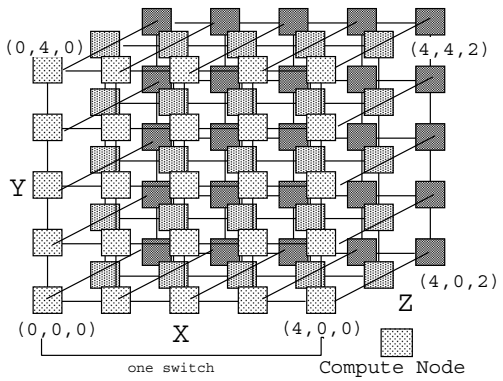

図 1 PACS-CS の持つ 3 次元 Hyper Crossbar 結合例 (5x5x3)

PACS-CS は、3 次元の Hyper Crossbar 結合を採用している (図 1)。図 1 において、各次元に連なる 1 本の線が 1 台のスイッチに相当し、この (X,Y,Z) の各次元につながるノードが 1 台のスイッチで結合される。

3 次元の Hyper Crossbar 結合におけるノード間通信は、例えば、図 1 中において、座標 (0,0,0) のノードから、座標 (1〜4,0,0), 座標 (0,1〜4,0), 座標 (0,0,1〜2) のノードへは Ethernet スイッチ経由で直接通信が可能であるが、例えば、座標 (4,0,2) や (4,4,2) のノードへは直接通信することができない。座標 (0,0,0) のノードから座標 (4,0,2) のノードへは最短 1 回中継ノード☆を経由した通信が必要である。同様に座標 (4,4,2) のノードでは最短 2 回中継ノード☆☆を経由した通信が必要である。

2.2 PACS-CS 上の通信機構の課題

第 2.1 節で述べた PACS-CS 上の通信機構の目標は、

- 複数の NIC を用いた高性能通信
- 次元間のルーティング処理 (最大 2 段)

を低通信処理オーバヘッドで実現することにある。

特に PACS-CS では、1 ノードにクラスタ用ネットワークとして、Gigabit Ethernet を 6 系統持っている。1 ノードが扱う総通信バンド幅は片方向 750MB/s(双方向 1.5GB/s) と InfiniBand や Myrinet などのクラスタ専用インターコネクトと同等レベルである。かつ、クラスタ専用インターコネクトが通信処理をネットワークインターフェイス (NIC) 上で処理するのと異なり、ホストプロセッサで処理することになる。

この 1 ノードが扱う総通信バンド幅をホストプロセッサ上で通信処理するためには、クラスタ専用インターコネクトで採用しているホストプロセッサによるコピー処理を排除し、かつ、複数の NIC からのパケットを無駄なく高速に処理することが必須となる。

3. PM/Ethernet-HXB の設計

PM/Ethernet-HXB は PACS-CS 上でクラスタインターコネクトに匹敵する通信性能の実現を目標としている通信機構である。PM/Ethernet-HXB の設計では、PM/Ethernet[3),4)] と同様に既存の通信プロトコルも同時に使えること、既存の OS の枠組への変更を最小限に抑えることを念頭に置くが、カーネルへの変更は行わない点が PM/Ethernet と異なる。

本章では、PM/Ethernet-HXB で採用している複数 NIC を用いた軽量通信プロトコルと通信バッファ間の Zero-Copy 機構の設計について述べる。

3.1 複数 NIC を用いる通信処理の要件

表 1 に、NIC 数毎の片方向通信を行った場合の各パケットサイズ毎の到着時間を示す。表 1 は同様に、1 パケットあたりの送信、受信の各々の処理を表記載の時間以内に処理できないと最大通信バンド幅が得られないことを意味している。

表 1 より、Giga x 6 で 750MB/s を実現するには、1 パケットあたり 10.9μs 以内に通信処理を行う必要がある。双方向通信の場合は、最大通信バンド幅を実現するために、半分の 5.5μs 以内に通信処理を行わなければならない。

☆ Ex 座標 (4,0,0) が中継ノード

☆☆ Ex 座標 (4,0,0) と座標 (4,0,2) が中継ノード

表 1 NIC 数とパケットサイズ毎の到着時間間隔 (μs)

	1500B	4096B	8192B
Giga x 1	12.0	32.8	65.5
Giga x 2	6.0	16.4	32.8
Giga x 4	3.0	8.2	16.4
Giga x 6	2.0	5.5	10.9
Giga x 8	1.5	4.1	8.2
Giga x 10	1.2	3.3	6.6

この 5.5μs が PM/Ethernet-HXB の実現目標になるが、この時間の中には Ethernet デバイスドライバ、Linux OS の処理など逐次的に実行されるものはすべて影響するため、通信処理は極限まで無駄を排除しなければならない。

3.2 複数 NIC を用いた軽量通信プロトコル

複数のネットワークにパケットを分散させた場合の通信において、最も問題となるのは、送信順にパケットが届かないため、受信側でパケットを送信順に並びかえる処理 (パケット Ordering) が必要な点である。

複数の Ethernet を用いた高性能通信機構として PM/Ethernet Network Trunking[5](以下、PM/Ethernet NT) がある。PM/Ethernet NT の実装では、パケット Ordering に skbuf の持っている構造体リンクを用いてパケットの順にキュー構造で実装し、このキューへのアクセスのために test and set 関数による排他制御を用いている。この排他制御は、1 つのパケットの出し入れに 2 回行われることになる。

しかし、この排他制御のコストを Linux 2.6.11 Xeon 2.8GHz の計算機で測定したところ、1 回あたり 0.05 μs 、2 回で 0.10 μs と 5.5μs の 2%の定常コストがかかることが分かった。このため、実装する軽量通信プロトコルの設計では、定常的に発生する排他制御処理を排除する。

パケット Ordering 処理において排他制御処理を排除するために、パケットの Sequence 番号の特性を利用した。Sequence 番号が同時に同じ番号のパケットが届かない性質を利用して、ノードの宛先毎に一定数の配列を設け、配列の値が 0 の場合は空き、0 以外の場合は受信済として排他制御を不要にしている。なお、0 以外の値としては skbuf を識別できる値を代入する。

この方式は、排他制御処理が不要な点において高速であるが、相手先毎に一定数の配列が必要であるため、メモリ資源を多く必要とする。

3.3 Zero-Copy 通信の設計

既存の Gigabit Ethernet NIC と Linux を用いた通信では、Linux の通信用のバッファである skbuf を用いた通信となる。Ethernet を用いた高性能通信において、送信時の Zero-Copy は PM/Ethernet[3),4)] で既に実現されているため、Zero-Copy 通信の実現上の課題は、受信時の Zero-Copy にある。ゆえに、本節では受信時の Zero-Copy の実現について議論する。

パケット受信時には、Linux の枠組を使う限り skbuf に最初にパケットが格納される。このパケットを、ユーザプロセスがコピー無しに参照可能とするためには、受信パケットが格納された skbuf をユーザプロセスの仮想アドレス空間に map すれば参照可能になる。

通常、受信パケットを格納する skbuf は Ethernet デバイスドライバで受信ハードウェアに割り当てられ、パケット受信処理後に Linux カーネルに返還され再利用される。返還時には別目的で利用される可能性があるため、受信後にユーザプロセスに map し、通信処理後には、map を解除しなければならない。

しかし、論文 [6)] によると、この map に要するコストは 4KB ページあたり 0.064 μs で 9000 バイトの JUMBO FRAME の map に 0.192μs、map 解除にも同じコストが必要で、計 0.384μs かかる計算になりコストが非常に大きい。

この問題を解決するため、Ethernet デバイスドライバが受信用に割り当てる skbuf はすべて再利用し、この受信用の skbuf のすべてをユーザプロセスに予め map しておく方式を採用することにした (**受信 skbuf pre-map 方式**)。この受信 skbuf pre-map 方式では、予め受信用の skbuf に ID を与えておき、ユーザプロセスではその ID に応じたアドレスに該当の skbuf を map しておく。

メッセージ受信時、デバイスドライバの受信処理で skbuf から ID を解釈し、その ID をユーザプロセスに通知する。ユーザプロセスはその ID から ID に応じたアドレスを参照することにより、受信パケットをコピーすること無く参照可能になる。

この方式は、通信プロトコル処理を大きく削減することが可能であるが、全体の skbuf を予め map するため、skbuf の量に応じたユーザプロセスの仮想アドレス空間が必要である。

3.4 ルーティング処理

ルーティングアルゴリズムは CP-PACS のアルゴリズムを採用している [7),8)]。このルーティングアルゴリズムは、各軸の転送順を例えば、最初に X 軸方向、次に Y 軸、最後に Z 軸と一定にする方式である。

この処理を高速に実行するために、各計算ノードを 3 次元座標で管理すると共に高速にルーティング処理を実施するために、受信用の skbuf をそのまま送信用の skbuf として再利用し、ヘッダだけを変更して転送する方式とする。

4. PM/Ethernet-HXB の実装

4.1 全体の構成

図 2 に、PM/Ethernet-HXB のソフトウェア構成を示す。PM/Ethernet-HXB は、SCore の通信機構である PMv2[9),10)] の一つの通信デバイスとして実装されている。OS は Linux である。

PM/Ethernet-HXB は、PMv2 の API を実装して

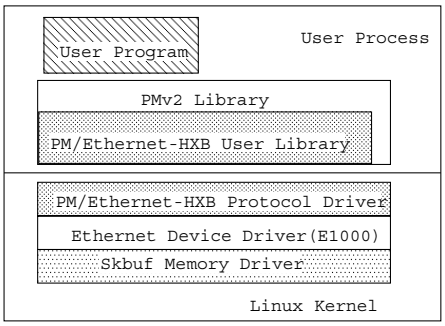

図 2 PM/Ethernet-HXB の構成

いる PM/Ethernet-HXB ユーザライブラリと 3 つのデバイスドライバで構成されている。各デバイスドライバの概要は次のとおりである。

Skbuf Memory デバイスドライバ： skbuf を扱うデバイスドライバである。デバイスドライバロード時に一定量の skbuf を確保し、再利用する。

改造 Ethernet デバイスドライバ： E1000 のデバイスドライバを改造して利用している。改造箇所は、関数置き換え用ヘッダの include 文挿入と、デバイスドライバ名の変更のみ。変更内容は、Skbuf Memory デバイスドライバを使ったバッファ割り当て関数名とメッセージ処理関数 (netif_rx など) の関数名の置き換えである。

メッセージ処理ドライバ： 送受信処理、ルーティング処理などの処理を実装している。

この 3 つのデバイスドライバの実装において、Linux カーネルへの変更をしない実装となっている。また、PM/Ethernet で採用したハードウェア割込み遅延を削減する Interrupt Reaping 機構は実装していない。

第 3.2 節で述べた、宛先毎のメッセージ受信のための配列は現状、1 ノードあたり、int タイプの配列を 32 割り当てている。このため、2,560 ノードで、320KB 消費している。また、第 3.3 節で述べた、Skbuf Memory ドライバで確保する skbuf の量は 12KB のサイズを 4,096 個（48 MB）確保している。2 つのメモリ使用量については、適切なサイズに調整する予定である。

5. 基本通信性能評価

本章では、PM/Ethernet-HXB の低レベルの基本通信性能を評価する。評価項目は、1 次元における PM レベルの通信バンド幅性能と通信遅延を測定し、PM/Ethernet NT、クラスタインターコネクトと比較する。また、多次元における PM レベルの通信バンド幅性能と通信遅延を測定評価する。クラスタインターコネクトとの比較結果は、論文 [11] でのデータを用いている。測定環境を表 2 と表 3 に示す。なお、Intel 社 Dual E1000 NIC x 4 の搭載は PCI-X 100MHz のバス 4 本にそれぞれ 1 枚ずつ搭載している。また、評価は各ノードあたりのプロセッサを 1 にして実施した。

表 2 PM/Ethernet NT, PM/Ethernet-HXB の評価環境

	GC LE 自作クラスタ環境
ノード計算機	Self made DUAL Xeon 2.8GHz 搭載 (ServerWorks GC LE chipset, 1GB DDR SDRAM, 64 bit 133MHz PCI-X Bus)
Ethernet (1Gbps)	Intel 社 Dual E1000 NIC x 4 Fujitsu 144 port GigE Switch
ホスト OS	Fedora Core 3 (2.6.11 kernel), SCore5.8.2

表 3 論文 [11] の評価環境

	富士通 PRIMERGY RX200 クラスタ環境
ノード計算機	RX200 DUAL Xeon 3.06GHz 搭載 (Intel E7501 chipset, 2GB DDR SDRAM, 64 bit 133MHz PCI-X Bus)
Myrinet (2Gbps)	Myricom 社 M3F-PCIXD PCI-X 133MHz Myricom 社 M3F-E128
InfiniBand (8Gbps)	IB HCA(PCI-X 133MHz) InfiniCom 社 32 ポート Switch
ホスト OS	Redhat 8.0 Linux (2.4.21 kernel), SCore5.6

5.1 1 次元での低レベル通信性能

PM レベルの転送バンド幅とラウンドトリップ (RTT) 時間の測定結果を図 3 と表 4, 5 に示す。図中、Gig x N の表記は Gigabit Ethernet を N 本束ねた場合の結果を示している。

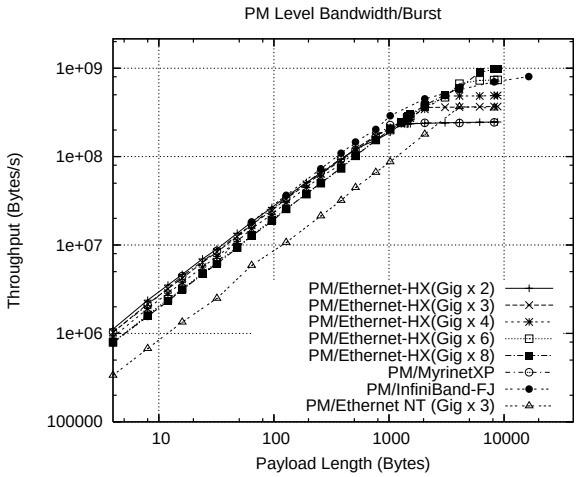

図 3 PM レベルの転送バンド幅性能 (メッセージ通信)

表 4 の結果より、PM/Ethernet-HXB は、Gigabit Ethernet 2 本時に 246 MB/s, 6 本時に 737 MB/s(物理リンク性能の 97.9%), 8 本時に 979 MB/s の転送性能を実現している。8 本時の性能は、PM/InfiniBand-FJ に比べても 22%高い。

また、図 3 と表 4 の結果より、Gigabit Ethernet 3 本時の PM/Ethernet NT との最大通信バンド幅はほ

表 4 PM レベルの最大転送バンド幅

	転送バンド幅
PM/Ethernet-HXB (Gigx2)	246 MB/s
PM/Ethernet-HXB (Gigx3)	368 MB/s
PM/Ethernet-HXB (Gigx4)	491 MB/s
PM/Ethernet-HXB (Gigx6)	737 MB/s
PM/Ethernet-HXB (Gigx8)	979 MB/s
PM/Ethernet NT (Gigx3)	366 MB/s
PM/MyrinetXP	244 MB/s
PM/InfiniBand-FJ	803 MB/s

とんど違いはないが、メッセージサイズ 2KB あたりまでのバンド幅は 2 倍以上高く、PM/Myrinet XP, PM/InfiniBand-FJ の通信性能に匹敵する通信性能を実現している。

表 5 PM レベルのラウンドトリップ (RTT) 時間

	RTT	Ratio
PM/MyrinetXP	8.3 μs	100%
PM/InfiniBand-FJ	15.6 μs	188%
PM/Ethernet NT	29.6 μs	357%
PM/Ethernet-HXB	30.1 μs	362%

注) Switch の 1 段あたりの往復遅延 6.0 μs を含む

表 5 に PM レベルのラウンドトリップ時間を示す。PM/Ethernet-HXB のラウンドトリップ時間については、PM/Myrinet XP に比べても 3.6 倍遅く、PM/Ethernet NT に比べても 0.5 μs 遅い結果となっている。Interrupt Reaping 機構は実装していないにも関わらず 0.5 μs の差ですんでいるのは、Linux 2.6 カーネルの持つポーリング機構によるものである。

5.2 多次元での低レベル通信性能

表 6 に多次元通信環境における PM レベルの通信バンドとラウンドトリップ時間 (RTT) を示す。

表 6 PM レベルの多次元環境の通信バンド幅性能と RTT

	Giga x 2 BW	Giga x 4 BW	Giga x 2 RTT
1 次元	247 MB/s	491 MB/s	30.2 μs
2 次元	241 MB/s	384 MB/s	58.1 μs
3 次元	237 MB/s	-	86.4 μs

注) RTT は Switch の 1 段あたり往復遅延 6.0 μs を含む
1 次元で 1 段、2 次元で 2 段、3 次元で 3 段の往復遅延が増加

表 6 の結果より、Gigabit Ethernet 2 本であれば 3 次元接続で 237 MB/s と良好な通信性能を実現している。しかし、Gigabit Ethernet 4 本では、2 次元接続で通信バンド幅性能が 3 本分の性能と劣化している。また、通信遅延については、ルーティングが 1 段増える毎に、28 μs(片道 14 μs) の増加となっている。

5.3 考　　察

第 5.1 節の結果で、低レベルの通信バンド幅性能でクラスタインターコネクトに匹敵する通信性能を実現することができた。本節では、第 5.1 の結果を用いて議論する。図 4 に、図 3 の結果から 1 パケットあたりの処理コストを算出した結果を示す。

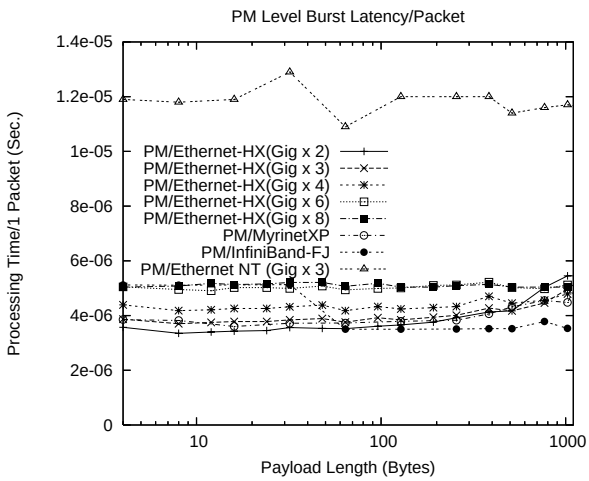

図 4 1 パケットあたりの処理コスト

図 4 より、4 バイトパケットで、最も処理コストが少ないのは PM/Ethernet-HXB の Gigx2 の 3.6 μs であり、次に PM/Myrinet XP の 3.8 μs である。また、PM/Ethernet-HXB の Gigx3 は 3.9 μs と PM/Ethernet NT Gigx3 の 11.9 μs に比べ 68%の処理時間を削減している。PM/InfiniBand-FJ では、32 バイトと 64 バイトパケットになるところで、処理コストが 3.5 μs に落ちている。これは、データの送信形式が Data on WQE による転送から通常の WQE による転送に変わるからである。

次に、PM/Ethernet-HXB の性能限界について考察する。Gigx8 の処理コストは 5.0 μs であり、NIC の数が増加する毎に処理コストが増えている。このコストを第 3.1 節のパケットサイズ毎の到着時間間隔と比較すると、第 3.1 節で述べた 5.5μs 以下の処理コストは満たしている。NIC 2 枚の増加で 1 μs 増えるとしても、6.0 μs になり、片方向であれば、NIC 10 枚の 1.25 GB/s までは処理可能である。1.25 GB/s は、10G Ethernet x 1 の物理リンク性能と同等であり、10G Ethernet でも通信処理可能であると言える。

6. 関 連 研 究

複数の NIC をサポートしたものに、PM/Ethernet Network Trunking[5]、PM/Ethernet-kRMA[12], Channel Bonding があるが、多次元の Hyper Crossbar を想定したルーティング機構は持っていない。Hyper Crossbar 結合を効率良く実現可能なものに VLAN TAG を用いた結合方式 [13] があるが、複数のネットワークを扱うには Channel Bonding との併用が必要である。

Hyper Crossbar 結合をハードウェアでサポートした専用ネットワークとして SCI[14], CP-PACS[8] などがあるが、PM/Ethernet-HXB は既存の Ethernet を用いソフトウェアのみで実現している点が異なる。

7. おわりに

本論文では、PACS-CS PC クラスタシステム向けの Ethernet を用いた高性能通信機構 PM/Ethernet-HXB の設計について述べた。PACS-CS の計算ノードは Gigabit Ethernet を計算ネットワークとして 6 系統備えており、これを 3 次元に 2 系統ずつ用いて Hyper Crossbar 結合を用いた通信を行なう。

PM/Ethernet-HXB の設計では、異なるノードの通信バッファ間での Zero-Copy 通信を実現した他、複数ネットワークからのパケット処理を低コストで行う軽量通信プロトコルを開発した。

PM/Ethernet-HXB を実装し、評価した結果、Gigabit Ethernet を 6 系統を用いた 1 方向の低レベルの通信バンド幅で 735MB/s と物理バンド幅性能の 97.9%、8 系統で 979MB/s を実現している。また、Gigabit Ethernet 2 系統を用いたルーティング性能についても、3 次元で 237MB/s の通信バンド幅性能を実現しており、高い通信性能を実現している。

今回は、低レベルの通信性能の評価に留まったが、PM/Ethernet-HXB では、ホストプロセッサで通信プロトコル処理を行うため、MPI レベルの通信となった場合に性能劣化が予想される。割込み処理の外乱などを抑える工夫により、更に通信プロトコルオーバヘッドを減らす必要がある。

今後は、定量的なオーバヘッドの解析、MPI レベルの通信性能評価、アプリケーションの性能評価を行ない、PACS-CS システム上で稼働させ、性能評価を行う予定である。

参考文献

1) 朴泰祐, 佐藤三久, 宇川彰. 計算科学のための超並列クラスタ PACS-CS の概要. 情報処理学会研究報告 05-HPC-103 (SWoPP'2005). 情報処理学会, August 2005.
2) SCore Cluster System Software: http://www.pccluster.org/.
3) 住元真司, 堀敦史, 手塚宏史, 原田浩, 高橋俊行, 石川裕. 既存ＯＳの枠組を用いたクラスタシステム向け高速通信機構の提案. 情報処理学会論文誌, 第 41 巻 第 6 号, pp. 1688–1696, June 2000.
4) Shinji SUMIMOTO, Hiroshi TEZUKA, Atsushi HORI, Hiroshi HARADA, Toshiyuki TAKAHASHI, and Yutaka ISHIKAWA. High Performance Communication using a Commodity Network for Cluster Systems. In *the Ninth International Symposium on High Performance Distributed Computing (HPDC-9)*, pp. 139–146. IEEE, August 2000.
5) 住元真司, 堀敦史, 原田浩, 石川裕. 複数 Ethernet を束ねる Network Trunking 機構の提案と 1,024 プロセッサ PC クラスタ上での性能評価. In *HPCS2002*. 情報処理学会, January 2002.
6) 住元真司, 佐藤充, 中島耕太, 久門耕一, 石川裕. 10Gb Ethernet を用いた高性能通信機構の設計. 情報処理学会研究報告 04-HPC-099 (SWoPP'2004). 情報処理学会, August 2004.
7) 朴泰祐, 板倉憲一, 曽根猛, 三島健, 中澤喜三郎, 中村宏. ハイパクロスバ・ネットワークにおける転送性能向上のための手法とその評価. 情報処理学会論文誌 Vol.36, No.7, pp. 1610–1618. 情報処理学会, 1995.
8) T. Boku, K. Itakura, H. Nakamura, and K. Nakazawa. CP-PACS: A massively parallel processor for large scale scientific calculations. In *International Conference on Supercomputing'97*, pp. 108–115. ACM, July 1997.
9) 住元真司, 堀敦史, 手塚宏史, 原田浩, 高橋俊行, 石川裕. 高速通信機構 PM2 の設計と評価. 情報処理学会論文誌, Vol.41 No. SIG 5 (HPS-1), pp. 80–90, August 2000.
10) Toshiyuki Takahashi, Shinji Sumimoto, Atsushi Hori, Hiroshi Harada, and Yutaka Ishikawa. PM2: A High Performance Communication Middleware for Heterogeneous Network Environments. In *Supercomputing 2000, IEEE and ACM SIGARCH, November, 2000, (Published by CD-ROM).*, November 2000.
11) 住元真司, 成瀬彰, 久門耕一, 細江広治, 清水俊幸. PM/InfiniBand-FJ: InfiniBand を用いた大規模 PC クラスタ向け高性能通信機構の設計. 情報処理学会論文誌：コンピューティングシステム Vol.45, No.SIG11 (ACS 7). 情報処理学会, October 2004.
12) Shinji Sumimoto and Kouichi Kumon. PM/Ehernet-kRMA: A High Performance Remote Memory Access Facility Using Multiple Gigabit Ethernet Cards. In *3rd International Symposium on Cluster Computing and the Grid*, pp. 326–334. IEEE, May 2003.
13) 工藤知宏, 松田元彦, 手塚宏史, 清水敏行, 児玉祐悦, 建部修見, 関口智嗣. VLAN を用いた複数パスを持つクラスタ向き L2 Ethernet ネットワーク. 情報処理学会論文誌：コンピューティングシステム Vol.45, No.SIG 6 (ACS 6), pp. 35–44. 情報処理学会, May 2004.
14) The Local Area Memory Port, Local Area MultiProcessor, Scalable Coherent Interface, and Serial Express Users, Developers, and Manufacturers Association: http://www.scizzl.com/.