

# Myrinet を用いた高速データ通信機能の設計と実現

中島 耕太†      下崎 誠 ††      谷口 秀夫 ††

† 九州大学大学院システム情報科学府

†† 九州大学大学院システム情報科学研究科

‡ 九州大学大学院システム情報科学研究院

近年、通信路のデータ転送速度が高速化し、計算機の内部バスの転送速度に近付いてきた。このため、計算機内部でのデータ転送処理速度が計算機間のデータ転送処理速度に大きな影響を与えるようになってきた。そこで、計算機内部でのデータ転送の主体、DMA (Direct Memory Access) 転送用バッファの取り扱い、転送終了の検知法について検討し、高速データ通信機能を設計した。通信路として、Myrinet を用い、Myrinet 制御部を *Tender* オペレーティングシステムに実装し、評価を行った。本稿では、高速データ通信機能の設計について述べ、実装した Myrinet 制御部の評価結果を報告する。

## Design and Implementation of High-Speed Data Transfer Mechanism on Myrinet

Kohta NAKASHIMA, Makoto SHIMOSAKI and Hideo TANIGUCHI

Graduate School of Information Science and Electrical Engineering, Kyushu University

As recently speed of data transfer increase on communication line, and it approach speed of internal bus in computer. And performance of data transfer in computer influence speed of data transfer between computers. Thus, we examine about subject of data transfer in computer, treatment of buffer for DMA transfer and detection of end of transfer. We use Myrinet as communication line, and implement Myrinet controller on *Tender*. In this paper, we report design of mechanism of high-speed data transfer and evaluation of performance of Myrinet controller.

### 1 はじめに

1990 年ごろより、プロセッサ性能の向上と低価格化、および通信路の高速化を背景として、PC/WS クラスタの研究や開発が行われるようになってきた。PC/WS クラスタ環境では、計算機間のデータの授受が頻繁に行われる。従って、各計算機間の通信性能は、全体の性能に大きな影響を与える。

計算機間でデータを送受信するためには、送信側計算機の主記憶から通信を制御する部分 (以降、NIC と呼ぶ) へデータを転送し、NIC から通信路へデータを送信する。そのデータを受信側計算機の NIC が受信し、主記憶へ転送する。従来の通信路のデータ転送速度は、計算機の内部バスの転送速度に比べ、非常に低速であった。このため、計算機内部でのデ

ータ転送処理速度が計算機間のデータ転送処理速度に与える影響は小さかった。ところが、通信路のデータ転送速度が高速化し、計算機の内部バスの転送速度に近付いてきた。このため、計算機内部でのデータ転送処理速度が、計算機間のデータ転送処理速度に大きな影響を与えるようになってきた。従って、計算機内部でのデータ転送速度の向上を図る必要がある。

そこで、我々は、計算機内部のデータ転送の主体、DMA 転送用バッファの取り扱い、転送終了の検知法について検討し、高速データ通信機能を設計した。通信路として 1.28Gbps の通信性能を持つ Myrinet<sup>[1]</sup> を用いた。また、Myrinet 制御部を我々が開発している *Tender* オペレーティングシステム<sup>[2]</sup> に実装し、評価を行った。

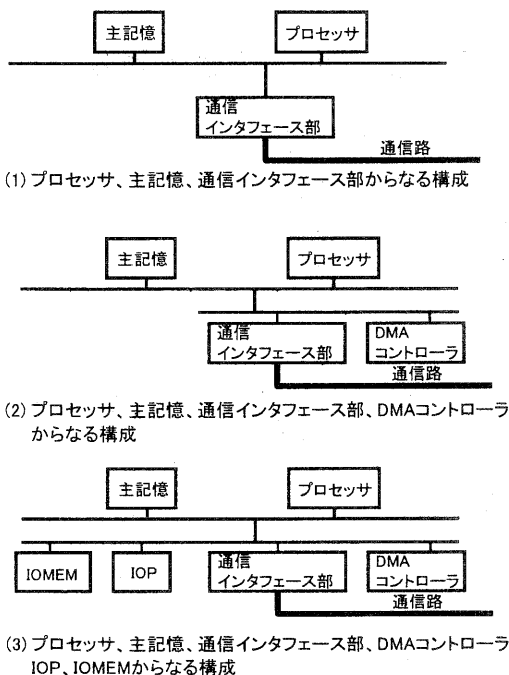


図 1 通信機構の形態

本稿では、高速データ通信機能の設計について述べ、実現した Myrinet 制御部の評価結果を報告する。

## 2 計算機間通信と通信路

### 2.1 通信機構

通信を制御する機構は、プロセッサ、主記憶、および通信インタフェース部の関係、さらに通信制御専用の DMA コントローラ、プロセッサ (IOP)、およびメモリ (IOMEM) の有無により、大きく 3 つに分類できる。各形態を図 1 に示し、以下に説明する。

- (1) プロセッサ、主記憶および通信インタフェース部で構成され、プロセッサが通信インタフェースの制御およびデータの転送を行う形態
- (2) プロセッサ、主記憶、DMA コントローラおよび通信インタフェース部で構成され、プロセッサが DMA コントローラと通信インタフェース部の制御を行い、DMA コントローラがデータを転送する形態
- (3) プロセッサ、主記憶、IOP、IOMEM、DMA コントローラおよび通信インタフェース部で構成され、プロセッサは IOP を制御し、IOP が通信を制御する形態

これらの特徴を比較する。形態 (1) は、プロセッサが直接通信インタフェースを制御し、データの転送を行うため、通信の応答時間が短い。しかし、通信処理の際のプロセッサへの負荷が大きい。また、プロセッサで通信プロトコル処理を行う必要がある。形態 (2) は、形態 (1) と比べ、データ転送処理を DMA コントローラに任せることができるため、データ転送処理の間、プロセッサはデータ転送処理から解放される。そのため、通信のスループットが大きい。また、プロセッサはデータ転送処理を行わないため、プロセッサの負荷は小さい。しかし、形式 (1) に比べ通信の反応時間がやや遅くなる。また、形態 (1) と同様にプロセッサで通信プロトコル処理を行う必要がある。形態 (3) は、通信インタフェース部を IOP が制御するため、3 形態のなかで最も通信のスループットが大きい。さらに、プロセッサは、通信インタフェース部の制御から解放されるため、プロセッサの負荷も小さい。また、IOP が通信プロトコル処理を行うので、複雑なプロトコル処理に対応できる。しかし、通信の応答時間が最も遅くなる。

形態 (3) は、プロセッサと IOP の機能分担により、さらに様々な形態に分類出来る。データの転送主体に着目すると、主記憶から IOMEM への転送と IOMEM から主記憶への転送の各々について、プロセッサ、IOP、DMA コントローラの 3 通り、計 6 通りがある。また、プロセッサと IOP の通信方式に着目すると、プロセッサが IOP の状態を検知する方法として、割り込みとポーリングに 2 通りがある。通信路の高速化が進んだため、通信のスループットを向上させるために、DMA コントローラによる転送方式が多く用いられている。利用した Myrinet の NIC は、形態 (3) である。

### 2.2 Myrinet-NIC

Myrinet 通信制御機能を実現する通信インタフェースボード (以降、Myrinet-NIC と名付ける。) は、LANai chip と呼ばれる制御用 LSI と記憶部である SRAM からなる。Myrinet-NIC の概要を図 2 に示す。LANai chip は、DMA コントローラ、パケットインタフェース部、IOP、Myrinet インタフェース部、及びバスからなる。このうち、DMA コントローラは、主記憶と Myrinet-NIC 上の SRAM との間のデータ転送を制御する。パケットインタフェース部は、SRAM から通信路、通信路から SRAM へのデータ転送を制御する。IOP は、SRAM 上に格納されたプログラ

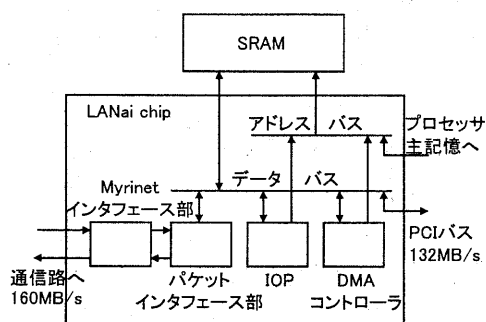


図2 Myrinet-NICの概要

ム (MCP:Myrinet Control Program、以降 MCP と呼ぶ) を実行し、DMA コントローラとパケットインタフェース部を制御する。また、SRAM は Myrinet-NIC 上の記憶部であるが、IOP の主記憶としての役割とデータ転送の際のデータバッファとしての役割を果たす。

Myrinet による通信では、次の手順でデータの送受信する。データを送信する際は、主記憶上のデータを Myrinet-NIC 上の SRAM へ転送し、SRAM から通信路へデータを転送する。受信する際は、通信路から SRAM へデータを転送し、SRAM から主記憶へデータを転送する。

Myrinet による通信を行うには、2つのソフトウェアが必要となる。一つは、SRAM 上に格納される MCP である。ここに記述された制御命令により、IOP が DMA コントローラ、パケットインタフェース部を制御する。もう一つは、主記憶上に格納される Myrinet 制御部である。ここに記述された制御命令により、プロセッサは、IOP を制御し、DMA コントローラ、パケットインタフェース部の管理を IOP に委託する。

### 3 Myrinet 制御部

#### 3.1 設計

##### 3.1.1 転送主体

主記憶と SRAM の間のデータ転送を行う主体としては、プロセッサと Myrinet-NIC 上の DMA コントローラが考えられる。両者の比較を表1に示す。

プロセッサによる転送では、主記憶へのアクセスにおいて、仮想アドレスで指定するデータ領域にアクセスする際、メモリ管理ユニットの働きや OS の要求時ページング機能の働きにより、任意の仮想アドレスのデータにアクセスする事が容易である。このため、任意の仮想アドレスのデータの転送が容易

である。一方、プロセッサによる SRAM へのアクセスは低速であるため、転送処理は DMA コントローラによる転送より低速である。また、転送処理の間、プロセッサを独占する。

DMA コントローラによる転送では、DMA コントローラによる主記憶、SRAM へのアクセスは比較的高速であるため、転送速度はプロセッサによる転送より高速である。また、転送処理の間、プロセッサを独占しない。一方、主記憶にアクセスする際、実アドレスでアクセスするため、仮想アドレスを実アドレスに変換するための処理が必要である。また、この領域が主記憶上に存在しない場合、外部記憶装置から主記憶へデータを読み込む必要がある。このように任意の仮想アドレスのデータ転送処理が複雑である。

以上に述べたように両者に得失があるものの、転送処理性能の観点から、DMA コントローラによる転送方式を用いることにした。

##### 3.1.2 DMA 転送用バッファの扱い

DMA コントローラを用いる方法には、DMA 転送用バッファの使用/不使用により、2つの転送方式がある。両者の比較を表2に示す。

DMA 転送用バッファを使用する方式について述べる。この方式では、システム起動時に主記憶に常駐する DMA 転送用のバッファを用意する。また、DMA 転送用バッファの実アドレスは Myrinet-NIC にあらかじめ通知しておく。データを転送する処理では、DMA 転送用バッファに転送データを複写して、DMA 転送用バッファから SRAM へ転送する。従って、データ複写のための処理時間が必要となる。この場合は、転送の度に Myrinet-NIC に転送データの先頭実アドレスを通知する必要はなく、転送データが主記憶に常駐する必要もない。

DMA 転送用バッファを使用しない方式について述べる。この方式では、データ転送時に、転送データの先頭実アドレスを Myrinet-NIC に通知する。そして、転送データ領域から直接 SRAM へ転送データを転送する。また、転送データは、主記憶に常駐する必要がある。この場合は転送データの主記憶内での複写処理は必要ない。

両者を比較すると、DMA 転送用バッファを使用する方式では、データ転送時のデータの複写処理が通信性能の大きなオーバーヘッドになる。Pentium II 450MHz の PC での実測では、4KB の主記憶内の複写処理時間は、26.3 $\mu$ s であった。一方、DMA 転送

表 1 転送処理の主体の比較

| 転送処理の主体    | 長所                                                        | 短所                                                         |
|------------|-----------------------------------------------------------|------------------------------------------------------------|
| プロセッサ      | (1) 主記憶上の任意の仮想アドレスに存在するデータの転送が可能                          | (1) データ転送中にプロセッサを独占<br><br>(2) 主記憶へのアクセスに比べ、SRAM へのアクセスは低速 |
| DMA コントローラ | (1) データ転送処理中にプロセッサを非独占<br><br>(2) 主記憶と SRAM の両方に高速にアクセス可能 | (1) 任意の仮想アドレスに存在するデータの転送が困難                                |

表 2 DMA 転送用バッファの使用/不使用の比較

| DMA 転送用バッファ | 長所                                                                | 短所                                                                |
|-------------|-------------------------------------------------------------------|-------------------------------------------------------------------|
| 使用          | (1) 転送データの主記憶への常駐が不要<br>(2) 転送データ領域の実アドレスを Myrinet-NIC へ通知する処理が不要 | (1) データ複写処理が必要                                                    |
| 不使用         | (1) データ複写処理が不要                                                    | (1) 転送データの主記憶への常駐が必要<br>(2) 転送データ領域の実アドレスを Myrinet-NIC へ通知する処理が必要 |

用バッファを使用しない方式では、転送データの先頭実アドレスを Myrinet-NIC に通知する処理がオーバヘッドになる可能性がある。そこで、この処理の概要について説明する。

Myrinet-NIC では、SRAM を介して DMA コントローラに転送データの先頭実アドレスを通知する。この際、SRAM 上に転送データの先頭実アドレスを登録しておくテーブル（以降、このテーブルを通信パス表と呼ぶ）を設ける。プロセッサは、通信パス表に転送データの先頭実アドレスを通信パス表に登録し、IOP はこれを参照して DMA コントローラに実アドレスを設定する。

この処理を行う際、プロセッサは SRAM へアクセスする。プロセッサの SRAM へのアクセスは主記憶へのアクセスに比べ低速である。Pentium II 450MHz の PC での実測した結果、4B のデータを主記憶から SRAM への複写処理の時間は  $0.150\mu\text{s}$ 、SRAM から主記憶への複写処理の時間は  $0.540\mu\text{s}$  であった。したがって、なるべく通信パス表へのアクセス数を減らす必要がある。そこで、実アドレスの登録処理では、登録の際に通信パス表の参照は行わず、先頭実アドレスを書き込むのみの処理を行うこととした。これにより、実アドレスの登録処理は、Pentium II 450MHz の PC で  $0.579\mu\text{s}$  になった。

DMA 転送用バッファの使用/不使用を比較すると、転送処理の高速性という観点では不使用の場合の方が有効である。したがって、DMA 転送用バッファを使用しない転送方式を用いる事にした。

### 3.1.3 データ転送終了の検知法

データ転送終了の検知法には、ポーリング検知法と割り込み検知法の 2 つがある。両者の比較を表 3

に示す。

ポーリング検知法は、データ転送終了の際、IOP が SRAM にある転送終了フラグを変更し、その変更をプロセッサが監視し続けることで、転送終了を検知する方法である。ポーリング検知法では、データ転送終了後、データ転送終了の検知が高速に行える。一方、転送終了検知時は、プロセッサが転送終了フラグを監視し続けるため、他の処理に与える影響は大きい。また、ポーリングによるデータ転送終了検知を OS カーネル内に実装した場合、通信以外の処理が停止する可能性がある。さらに、プロセッサによる SRAM へのアクセスは、Myrinet-NIC の処理では、最優先に実行されるため、その間転送処理は停止する。

割り込み検知法は、データ転送終了の際、IOP がプロセッサに対して割り込みを発生させることによりデータ転送終了を検知する方法である。IOP がプロセッサに割り込みを発生させると、割り込み処理ルーチンが起動し、受信処理が行われる。また、転送待ちプロセスを起動する必要がある。これらの処理は処理時間がかかるため、データ転送転送終了後から、データ転送終了の検知が行われるのは、比較的低速である。しかし、転送待ちの間、プロセッサを独占しないため、他の処理に与える影響は小さい。

両者を比較すると、Myrinet 制御部は OS カーネル内に実装するため、ポーリング検知法では他の処理が停止する恐れがあり、好ましくない。よって、受信検知には割り込み検知法を用いることにした。

### 3.2 基本機能

基本機能として、以下の 3 つの機能が挙げられる。

表 3 転送終了検知方式の比較

| 転送終了検知方式 | 長所                 | 短所                |
|----------|--------------------|-------------------|
| ポーリング    | 処理速度は高速            | 転送終了待ちの間、プロセッサを使用 |
| 割り込み     | 転送終了待ちの間、プロセッサは不使用 | 処理速度は低速           |

| エントリ番号 | 実アドレステーブル   |
|--------|-------------|
| 0      | 0x012345000 |
| 1      | 0x023456000 |
| 2      | 0x0         |
| 3      | 0x0         |
| 4      | 0x0         |
| 5      | 0x034567000 |
| 6      | 0x0         |
| 7      | 0x0         |
| 8      | 0x045678000 |
| 9      | 0x0         |
| ⋮      | ⋮           |

図 3 通信パス表への登録処理

- (1) 通信パス表への登録処理
- (2) 送信処理
- (3) 受信処理

通信パス表への登録機能の処理について述べる。この処理では、通信パス表を SRAM 上に用意し、転送データ領域の先頭実アドレスを登録する。送信側計算機は、転送データ領域を通信パス表に登録し、受信側計算機は、データ受信領域を通信パス表に登録する。登録の際、エントリ番号と転送領域を指定する。図 3 は、通信パス表の 0 番、1 番、5 番、8 番に転送領域が登録された様子を示している。転送領域の先頭実アドレスが 0x012345000 番地である領域は 0 番に登録されている。1 番、5 番、8 番についても同様に登録されている。また、通信パス表の値が 0x0 となるエントリは未登録のエントリである。処理は以下に行われる。まず、転送データ領域の先頭実アドレスを取得する。そして、エントリ番号で指定された通信パス表のエントリに実アドレスを書き込む。

送信機能の処理について述べる。送信処理では、通信パス表に登録された領域のデータを受信側計算機に送信する。受信側計算機では、あらかじめ受信領域を通信パス表に登録する必要がある。送信処理では、送信元の領域のエントリ番号、受信先の領域のエントリ番号、転送先計算機番号、転送データサイズを指定する。処理は以下に行われる。まず、送信元の領域のエントリ番号、受信先の領域のエントリ番号、転送先計算機番号、転送データサイズを

SRAM 上に設定する。そして、送信の契機を与える。ここまですがプロセッサの処理である。送信の契機が与えられると、DMA コントローラは、送信元の領域のデータを主記憶から SRAM に転送し、パケットインタフェース部が SRAM 上のデータを通信路へ転送する。この際、転送データサイズ、受信先の領域のエントリ番号も転送する。

受信機能の処理について述べる。受信処理では、受信したデータを通信パス表に登録された領域に転送する。処理は以下に行われる。パケットインタフェース部は通信路からのデータを SRAM 上にデータを受信する。受信したデータから受信先の領域のエントリ番号を取得し、DMA コントローラは転送データを通信パス表に登録された主記憶上の受信領域へ転送する。IOP は、転送が完了するとプロセッサに対し割り込みにより受信完了を通知する。プロセッサは、割り込みにより受信を検知し、受信先エントリ番号を SRAM 上より取得する。

## 4 性能評価

### 4.1 測定条件

Myrinet 制御部の評価を行うために以下の項目について測定を行った。

- (1) 基本性能
- (2) 複数データを連続転送した場合の性能

基本性能については、各データサイズにおけるデータ転送時間を測定し、基本性能を評価する。複数データを連続転送した場合の性能については、分散共有メモリ機構の実現の際には、ヘッダ情報と、本体データを連続転送する処理が行われるため、これらのデータを連続転送した場合の性能を評価する。

また、比較のため、RWCP(新情報技術開発機構)によって開発された Linux 上のユーザレベルの Myrinet 通信ライブラリである PM ライブラリ<sup>[3]</sup>を取り上げ、**Tender**での実現方式と比較する。

測定は、いずれも Myrinet-NIC が装着されたプロセッサ Pentium II(450MHz)、主記憶 128MB の PC 2 台で行った。尚、計算間は、Myrinet で接続されており、Myrinet-NIC は、PCI バスで装着されている。

**Tender**での性能測定は、OS 核内に Myrinet 制御部を実装し、カーネル内で測定を行った。PM ライブラリでの性能測定は、OS として、Linux 2.0.36

表 4 測定する転送方法

| 場合  | DMA 転送用バッファ | 受信検知                | 送信検知                | 通信回数 | 環境            |
|-----|-------------|---------------------|---------------------|------|---------------|
| (1) | 不使用         | 割り込み                | なし                  | 1 回  | <i>Tender</i> |
| (2) | 使用          | 割り込み                | なし                  | 1 回  | PM ライブラリ      |
| (3) | 使用          | 連続ポーリング             | なし                  | 1 回  | PM ライブラリ      |
| (4) | 使用          | 10 $\mu$ s 間隔でポーリング | なし                  | 1 回  | PM ライブラリ      |
| (5) | 不使用         | 連続ポーリング             | 10 $\mu$ s 間隔でポーリング | 2 回  | PM ライブラリ      |

を用い、Linux 上の PM ライブラリを用いたユーザプログラム内で測定を行った。尚、PM ライブラリでは、Myrinet-NIC 上のレジスタや、SRAM をユーザ空間にマッピングし、Myrinet-NIC の制御を直接行う。また、PM ライブラリでの測定は、シングルユーザモードで行った。全ての測定には、ハードウェアクロックカウンタを用い、2 箇所のカウンタ値の差分をクロック数で割ることにより、処理時間を算出した。

#### 4.2 基本性能

*Tender* の Myrinet 制御部でのデータ転送処理時間について測定を行った。また、比較のため、PM ライブラリによる DMA 転送用バッファを用いる転送方式による転送処理時間、DMA 転送用バッファを用いない転送方式による転送処理時間の測定を行った。

以下の 5 つの場合についてデータ転送時間を測定した。5 つの場合を表 4 に示す。

以降、それぞれの場合について説明する。場合 (1) は、*Tender* に実装した Myrinet 制御部での測定である。転送方式は DMA 転送用バッファを用いない転送方式であり、受信の検知は割り込みによる方法である。尚、データ転送処理の際の通信回数は 1 回であり、送信終了は検知しない。場合 (2) は、Linux 上の PM ライブラリでの測定である。転送方式は DMA 転送用バッファを用いた転送方式である。受信の検知は割り込みによる方法である。尚、データ転送処理の際の通信回数は 1 回であり、送信終了は検知しない。場合 (3) は、Linux 上の PM ライブラリでの測定である。転送方式は DMA 転送用バッファを用いた転送方式である。受信の検知はポーリングによる方法であり、連続してポーリングを行う。尚、データ転送処理の際の通信回数は 1 回であり、送信終了は検知しない。場合 (4) は、Linux 上の PM ライブラリでの測定である。転送方式は DMA 転送用バッファを用いた転送方式である。受信の検知はポーリングによる方法であり、10 $\mu$ s の周期でポーリングを行う。尚、データ転送処理の際の通信回数は 1 回であり、送信終了は検知しない。場合 (5) は、Linux 上の PM ライブラリでの測定である。転送方式は DMA 転

送用バッファを用いない転送方式である。この場合は、受信終了の検知は、以下のように行う。送信側計算機が 10 $\mu$ s 周期のポーリングにより送信終了を検知する。そして、送信側計算機は転送終了を受信側計算機に通信により通知する。受信側計算機は、この通知を連続ポーリングにより検知する。この場合は、データの転送と転送終了通知の 2 回の通信がある。

それぞれの場合について以下に述べる方法でデータ転送時間を測定した。2 台の計算機 A、B の間でデータ転送を行う。計算機 A から計算機 B へデータを転送し、受信が完了したら、計算機 B から計算機 A へデータを転送する。この処理を 10000 回繰り返す。データを 2 台の計算機間で 10000 回往復させた。この処理の開始から終了までの処理時間を測定し、片道 1 回分にかかった処理時間を算出した。

*Tender* での Myrinet 制御部によるデータ転送時間および、PM ライブラリでの各場合のデータ転送時間を図 4 に示す。また、これらのグラフについて線形近似し、近似式を算出した。近似式を表 5 に示す。表中の数式の  $x$  は、転送データサイズ (Bytes) を表し、 $y$  は、転送時間 ( $\mu$ s) を表す。場合 (5) の転送時間については、8B、32B での転送時間が、グラフ上で直線から大きくはずれるため、この 2 つのデータを取り除いて近似した式を括弧付きで示している。

この結果を比較し、考察する。

割り込みによる受信検知と、通信による受信終了検知とを比較する。場合 (1) と場合 (5) では、各データサイズとも場合 (1) の方が 1.3~8.7 $\mu$ s 高速である。これについては以下のように説明できる。場合 (5) は、受信側が受信終了を検知するために、送信側の送信終了の通信による通知を受けなければならない。それに対し、場合 (1) では、受信終了時に割り込みで通知が行われる。これら処理を比較すると、受信完了を通信で通知する処理よりも割り込みによる受信終了通知の方が高速に行えるために、場合 (1) の方が高速であると考えられる。また、特にデータサイズが小さくなると、受信終了通知の通信によるオーバーヘッドが大きいとされる。尚、場合 (1) の割り込みによる受信検知では、受信割り込みが発生す

ると、即座に割り込み処理ルーチン内で相手側計算機への送信処理を行うので、ユーザレベルからみた割り込み処理より高速に処理を行うことができる。

また、ポーリングによる受信検知と、割り込みによる受信検知を比較する。場合(2)と場合(3)では、データサイズが1KB以下の場合については、場合(3)の方が高速である。これは、場合(2)の割り込みによる受信終了の検知と場合(3)のポーリングによる受信終了の検知を比較した場合、ポーリングによる検知の方が高速に行えるためだと考えられる。

また、ポーリング処理の有無によるデータ転送性能の比較を行う。場合(2)、(3)、(4)について近似式を比較すると、場合(2)、(4)より場合(3)の方がグラフの傾きが大きい。これについて以下のように説明出来る。場合(3)では、連続ポーリングを行うが、ポーリング処理の際、プロセッサによるSRAMの参照処理が頻繁に起こる。プロセッサによるSRAMの参照は、SRAMへアクセスの優先順位では最高位である。したがって、この間、DMAコントローラやパケットインタフェース部はSRAMにアクセスできないため、転送処理が中断されると考えられる。そのため、連続ポーリング処理を行うと、データ転送時間が長くなる。また、転送時間が長くなればなるほど、ポーリング回数が増加するので処理性能の影響が大きくなる。一方、場合(4)については、10 $\mu$ sの間隔をあけてポーリングするため、ポーリング処理による転送時間への影響は非常に小さい。そのため、場合(2)と場合(4)は、ほぼ傾きが等しくなると考えられる。

DMA転送用バッファへの複写処理について考察する。場合(1)、(5)と、場合(2)、(4)について、近似式を比較すると、場合(1)、(5)と、場合(2)、(4)では、場合(2)、(4)の方が傾きが大きい。これは、場合(2)、(4)は、主記憶内でデータを複写する必要があり、データ複写処理時間は、データサイズが大きくなればなるほど増加する。一方、場合(1)、(5)は、主記憶内のデータ複写処理が必要無いため、場合(2)、(4)に比べ、高速にデータ転送処理が行えると考えられる。

#### 4.3 複数データを連続転送した場合の性能

分散共有メモリ機構では、データの送受信の際、ヘッダ情報とデータを合わせて転送することが多い。そこで、**Tender**での転送方式の場合について、32Bのデータの送信終了後、256B、1KB、4KBのデータ

- (1)Tender,DMA用バッファ不使用,割り込みによる受信終了検知,送信終了検知なし
- (2)PM,DMA用バッファ使用,割り込みによる受信終了検知,送信検知なし
- (3)PM,DMA用バッファ使用,連続ポーリングによる受信終了検知,送信終了検知なし
- (4)PM,DMA用バッファ使用,10 $\mu$ s間隔のポーリングによる受信終了検知,送信終了検知なし
- (5)PM,DMA用バッファ不使用,ポーリングでの終了検知,10 $\mu$ s間隔ポーリングでの送信終了検知

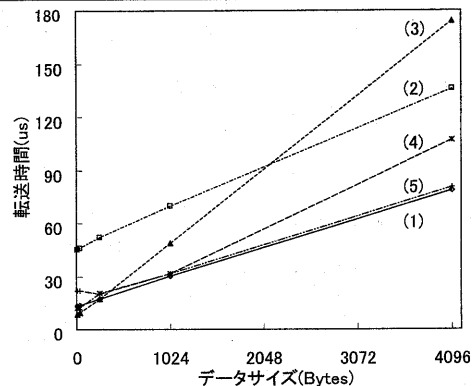


図4 各データサイズでの転送時間

表5 線形近似式

| データ系列      | 近似式                                        |
|------------|--------------------------------------------|
| 場合(1)の転送時間 | $y = 0.016x + 13$                          |
| 場合(2)の転送時間 | $y = 0.022x + 46$                          |
| 場合(3)の転送時間 | $y = 0.041x + 7.6$                         |
| 場合(4)の転送時間 | $y = 0.023x + 12$                          |
| 場合(5)の転送時間 | $y = 0.015x + 19$<br>( $y = 0.016x + 15$ ) |

を送信する時間を測定した。

測定は、以下のように行った。2台の計算機A、Bの間でデータ転送を行う。計算機Aから計算機Bへ32Bのデータを送信し、この転送終了を10 $\mu$ s間隔のポーリングにより検知する。その後、256B、1KB、4KBのデータを送信する。この送信が受信されたら、計算機Bは同様の送信処理を行い、データを計算機Aに転送する。この処理を10000回繰り返す。データを2台の計算機間で10000回往復させた。この処理の開始から終了までの処理時間を測定し、片道1回分にかかった処理時間を算出した。

この結果と図4で示したそれぞれの転送時間の和を図5に示す。連続して転送した場合の転送時間は、図4で示したそれぞれの転送時間の和より4.5~6.0 $\mu$ s小さい。これは、今回使用したMCPのデータ転送が、以下に説明するようにおこなわれるためであると考えられる。今回使用したMCPは、送信側計算機では、主記憶からSRAMへのDMA転送の開始直

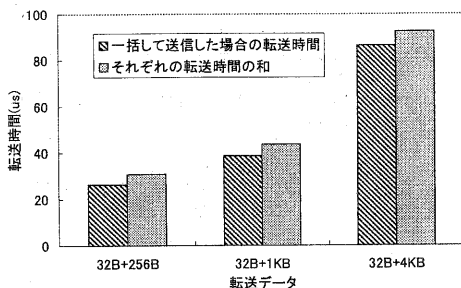


図 5 ヘッド情報とデータを連続転送したときの転送時間とそれぞれの転送時間の和

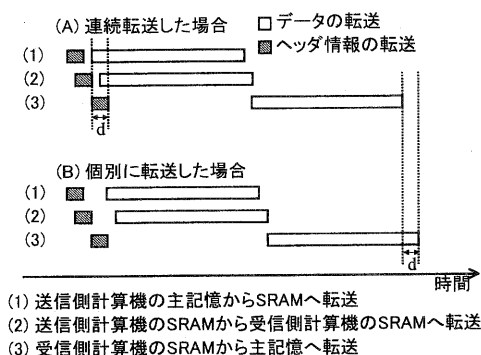


図 6 データ転送の様子

後に SRAM から通信路への転送処理を開始する。また、受信側計算機では、SRAM 上の受信領域を 2 つ用意し、受信データを主記憶へ DMA 転送している最中も受信を行う。このため、連続転送する際は、図 6 中の (A) で示されるようにデータ転送する。一方、それぞれの転送時間の和は、図 6 中の (B) のように個別にデータを送信した場合の転送時間と同じである。この場合、受信側の DMA 転送が完了してから次の送信処理を行うため、(A) の場合より図 6 中の d で示される時間だけ転送処理時間が長い。

## 5 おわりに

Myrinet を用いた高速データ通信機能を設計し、**Tender** オペレーティングシステムの Myrinet 制御部として実現し、評価を行った。

Myrinet 制御部の設計を行った。計算機内部のデータ転送処理を行う主体としては、プロセッサと DMA コントローラがあり、データ転送処理速度の観点から、DMA コントローラを用いることにした。DMA 転送用バッファの扱いについては、DMA 転送用バッ

ファを使用する転送方式と、使用しない転送方式があり、データ転送処理速度の観点から、DMA 転送用バッファを使用しない転送方式を用いることにした。転送終了検知方式については、割り込み方式とポーリング方式があり、転送以外の処理へ与える影響という観点から、割り込み方式を用いることにした。

また、Myrinet 制御部を実装し、評価した。基本性能として、割り込みによる受信終了検知と通信による受信終了検知を比較し、割り込みによる受信終了検知の方が受信検知にかかる時間が短いことを示した。ポーリングによる受信検知と割り込みによる受信検知では、ポーリングによる受信検知の方が受信検知にかかる時間が短いことを示した。また、ポーリング処理の有無によるデータ転送性能を考察し、ポーリング処理では、SRAM へのアクセスが頻繁におこるため、データ転送処理時間が増加することを示した。DMA 転送用バッファの使用の有無については、使用する場合は転送データの複写処理が必要となるため、データ転送処理時間が増加することを示した。複数データを連続転送した場合の性能については、個別に分けてデータ転送する場合よりも、複数データを連続転送した場合の方が 4.5~6.0 $\mu$ s データ転送処理時間が短くなることを示した。

今後の課題としては、**Tender** での Myrinet 制御部のデータ転送可能な空間の制限解除や、通信以外の処理が存在する場合の通信性能の評価がある。

## 参考文献

- [1] Nanette J. Boden, Danny Cohen, Robert E. Felderman, Alan E. Kulawik, Charles L. Seitz, Jakov N. Seizovic, and Wen-King Su: "Myrinet - A Gigabit-per-Second Local-Area Network", IEEE-Micro, Vol.15, No.1, February 1995, pp.29-36
- [2] 谷口秀夫: "分散指向永続オペレーティングシステム **Tender**", 情報学会コンピュータシステムシンポジウム, 論文集, Vol.95, No.7, pp.47-54(1995)
- [3] 手塚宏史, 堀 敦史, 石川 裕: "ワークステーションクラス通信ライブラリ PM の設計と実装", 並列処理シンポジウム JSPP'96, pp.41-48(1996)