DIMMnet-2 ネットワークインタフェースコントローラの設計と実装

北村 聡[†] 濱田芳博[††] 宮部保雄[†]
伊澤 徹[†] 宮代具隆[†] 田邊 昇[†††]
中條拓伯[††] 天野英晴[†]

DIMMnet は Personal Computer（PC）の DIMM スロットに装着するタイプの PC クラスタ向けインターコネクト用ネットワークインタフェースである．メモリバスは一般的なネットワークインタフェースが装着される PCI バスよりもアクセスレイテンシが小さく，ここにネットワークインタフェースを装着することで，きわめて低いレイテンシで通信を行うことが可能である．我々はコントローラ部に FPGA を用いた DIMMnet-2 のコントローラ部の設計，および実装を行い，データ転送時のレイテンシを測定した．その結果，8 Byte データ送信時のレイテンシが 0.570 μs と，商用の PC クラスタ向けインターコネクトである QsNET II の 43.8%程度に抑えられることが示された．

Design and Implementation of Network Interface Controller on DIMMnet-2

Akira Kitamura,[†] Yoshihiro Hamada,[††] Yasuo Miyabe,[†]
Tetsu Izawa,[†] Tomotaka Miyasiro,[†] Noboru Tanabe,[†††]
Hironori Nakajo[††] and Hideharu Amano[†]

DIMMnet is a network interface for PC cluster interconnect, with uses the DIMM slot on a common PC. Compared with PCI bus used for traditional network interfaces, the access latency of memory bus can be much improved. DIMMnet-2 network interface is the second generation network interface for memory bus which can use DDR-SDRAM slot. In order to cope with high clock frequency of DDR-SDRAM slot, the indirect accessing method is implemented. Although the current board is a prototype using an FPGA, the latency for 8 Byte data transfer is only 0.570 μsec that is 43.8% of that in the high performance commercial interconnect QsNET II.

1. はじめに

Personal Computer（PC）の価格対性能比の著しい向上により，これらをノードとして用いた PC クラスタは実用的な計算資源として企業・大学等で広く用いられるようになった．PC クラスタの多くはノード PC を Myrinet [1] や QsNET [2]，InfiniBand [3] 等の専用の高速なインターコネクトにより相互接続することで高い性能を実現する．通常，これらインターコネクトのネットワークインタフェースは PC における標準的な I/O バスである PCI バスに接続されているが，近年のホスト CPU，メモリ，インターコネクトの性能の向上にともない，ホスト CPU から PCI バスへのアクセスレイテンシがシステムの性能に大きな影響を与えつつある．たとえば，QsNET II の場合，8 Byte のデータをリモートノードに転送する処理を完了するのに約 1.3 μs 要するが，このうち，ホストのチップセットの遅延が約 1 μs と 3/4 以上を占めている[4]．このことは，メッセージサイズが小さい場合の通信性能にホストの遅延が大きく影響することを意味し，同期処理のようなメッセージサイズの小さい通信を多くともなう集団通信において性能低下の原因となる．

そこで，我々はホスト CPU から，PCI バスよりも低レイテンシでアクセス可能な DIMM スロットに着目し，ここにネットワークインタフェースを装着することでホストの遅延を抑える手法を提唱している．この手法に基づき，DIMM スロット装着型のネットワークインタフェースである DIMMnet の開発を行い，す

† 慶應義塾大学
Keio University
†† 東京農工大学
Tokyo University of Agriculture and Technology
††† 株式会社東芝研究開発センター
Corporate Research and Development Center, Toshiba Corporation

でに SDR-SDRAM に接続する DIMMnet-1 [5] の稼働に成功している．

本論文では，DIMMnet-1 の経験に基づき，新たに開発した DIMMnet-2 [6] のネットワークインタフェースコントローラの設計と実装について述べ，最も基本的な転送性能の評価によってメモリスロット装着型ネットワークインタフェースの有効性を示す．

以下，2 章で DIMMnet の概要と DIMMnet-2 について述べる．3 章でネットワークインタフェースコントローラ部の設計と実装について，4 章ではその評価について，それぞれ示す．5 章で関連研究について触れ，最後に 6 章で本論文についてまとめる．

2. メモリスロット装着型ネットワークインタフェース DIMMnet

DIMMnet は DIMM スロットに装着するタイプの，PC クラスタ向けネットワークインタフェースである．PC クラスタ向けインターコネクトのネットワークインタフェースは，汎用 I/O バスである PCI バスに装着されるのが一般的である．しかしながら，メモリバスに比べて PCI バスへのホスト CPU からのアクセスレイテンシは大きい．これは，PC のマザーボードに搭載されるチップセットの構成が主な原因である．**図 1** に PentiumIII 以降の Intel 系 CPU を搭載した一般的な PC, サーバ機のチップセットの構成を示す．図 1 に示されるように，チップセットは CPU やメインメモリ，これらに次いで高い性能が求められるグラフィックスデバイスが接続される Memory Control Hub と，ハードディスクや USB 等の各種 I/O インタフェースが接続される I/O Control Hub から構成される．PCI バスや非グラフィックスデバイス向けの PCI-Express は I/O Control Hub に接続される．サーバ向けチップセットでは I/O Control Hub に相当するチップは I/O に応じて別々のチップを用いる例も見られるが，

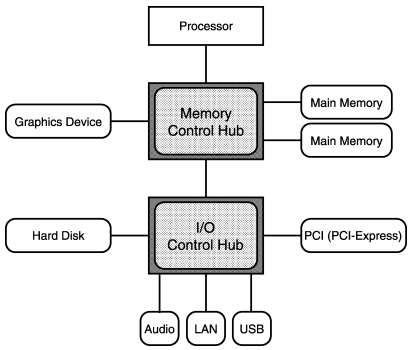

図 1 汎用 PC のチップセットの構成
Fig. 1 Structure of chipsets of commercial PC.

CPU との間に Memory Control Hub が存在する点は同様である．そのため，ホスト CPU から PCI バス上のデバイスに対してアクセスするには 2 つのチップを経由することになり，メインメモリに対するアクセスよりもレイテンシが大きくなる．たとえば，ホスト CPU から PCI バスに装着されたネットワークインタフェースをポーリングした場合，μs オーダのレイテンシを要する．

また，サーバ向けのシステムにはバスクロックが 133 MHz で動作する PCI-X バスが搭載される例が多く見られるが，現在，汎用 PC において主流となっている I/O バスは，いまだバスクロックが 33 MHz の PCI バスである．対して，メモリバスは現在主流の DDR-SDRAM バスで 100〜200 MHz のバスクロックであり，汎用 PC ではメモリバスと PCI バスとの間で，アクセスレイテンシの差がより大きくなる．さらに，メモリバスのバンド幅は近年のメモリクロックの著しい向上から，PCI バスに代わる高速 I/O である PCI-Express の x8 の規格に匹敵する性能に達しているため，ここにネットワークインタフェースを装着し，PC クラスタを構築することで，PCI バスや PCI-X バスに装着される PC クラスタ向けインターコネクトを用いた PC クラスタよりも高性能なシステムを構築できると予想される．

メモリスロットを用いることの利点は，システムの構築コストの観点からもあげられる．一般に，高性能な PC クラスタを構築するには 64 bit/133 MHz の PCI-X バスを搭載したサーバ機を用いるが，このようなサーバ機はノード単価が汎用 PC に比べて高価であるため，システム構築のコストが上昇する．一方でメモリスロットは汎用 PC に標準的に搭載されていることから，PCI-X バスを搭載したサーバ機を用いずに安価な PC をノードとして用いた場合でも高性能な PC クラスタを低コストで構築可能であると考えられる．

DIMMnet はメモリスロットを介してノード間を接続することにより，PCI-X バスのような高速 I/O を用いた PC クラスタ向けインターコネクトに比べ，低コストで高性能なシステムを構築することを目的とする．

2.1 DIMMnet-2

DIMMnet-2 は DIMM スロット装着型ネットワークインタフェースの 2 世代目として，東京農工大学，および新情報処理開発機構によって開発された DIMMnet-1 [5] の経験に基づいて開発が行われている．DIMMnet-2 は既存の PC クラスタ向けインターコネクトよりも低レイテンシな通信を実現することによって，PC ク

ラスタの性能向上を目指す．これは，文献 7) において，64 bit/66 MHz の PCI バスと 64 bit/133 MHz の PCI-X バスにそれぞれ InfiniBand のネットワークインタフェースを装着した小規模な PC クラスタ上で NAS Parallel Benchmarks による性能比較が行われており，PCI バスと PCI-X バスのバンド幅の差ほどの性能差が見られなかったことから，今後はバンド幅の向上のみでは PC クラスタのシステム全体の性能向上は難しく，また，1 章で述べたとおり，ホスト PC のチップセットのレイテンシがメッセージサイズの小さい通信時に大きく影響することから，通信レイテンシの削減が必要であると考えられるためである．

DIMMnet-1 は SDR-SDRAM メモリバスに対応しており，メモリバスインタフェース部のスイッチの切替えにより，DIMMnet-1 上に搭載されたメモリを直接 PC からアクセスする構成になっていた．しかし，この構成ではメモリバスのインタフェースの遅延の問題から高クロック化が難しく，DDR-SDRAM メモリバスには対応することができないことが判明している．また，DIMMnet-1 で採用したネットワークスイッチは独自設計であったため，量産ができず，高価であった．

DIMMnet-2 では，これらの問題点を解決するために，PC から DIMMnet-2 ボード上のメモリのアクセスをメモリバスとのインタフェース内のバッファを介した間接転送にする．これにより，DDR-SDRAM メモリバスとの接続を可能にした．

間接転送方式を採用することで，行列へのストライドアクセス等，レイテンシの大きくなりやすい複雑なアクセスをインタフェース上のハードウェアで制御し，高速化することが可能となる[8]．これにより，DIMMnet-2 はネットワークを介した並列処理を行わない場合でも，アプリケーションの高速化が可能であるため，高機能なメモリモジュールとしての利用が可能である．

また，ネットワークスイッチ等のネットワークインタフェース以外のコンポーネントには標準ネットワークである InfiniBand を用いることで，専用のスイッチを用いた場合よりもシステム構築のコストを低く抑える．

2.2 本研究の目的

本研究では DIMMnet-2 のネットワークインタフェースコントローラの設計と実装を行い，コントローラ部に FPGA を搭載した DIMMnet-2 試作基板（2.2.1 項）を用いた検証，評価等を通してコントローラの ASIC 化の足がかりとすることを目的とする．

DIMMnet-2 ではホストから DIMMnet-2 上の SO-DIMM に対して，コントローラを介した間接アクセスを行うため，DIMMnet-1 とは大幅に構造が異なる．したがって，ネットワークインタフェースコントローラを一から設計し直す必要がある．

また，メモリスロット装着型ネットワークインタフェースを用いて安価に PC クラスタを構築するためには，ネットワークインタフェースの製造コストを下げるためにネットワークインタフェースコントローラを ASIC として実装する必要がある．

そのため，DIMMnet-2 試作基板上の FPGA に，設計したコントローラを実装し，機能検証，論理検証，および性能評価を行う．

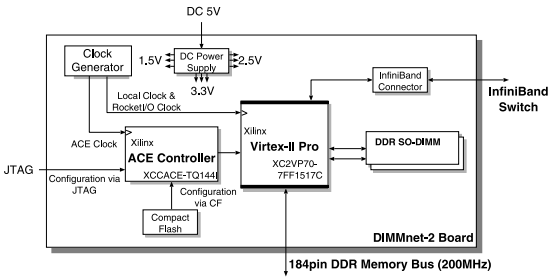

図 2 DIMMnet-2 試作基板の構成図
Fig. 2 Structure of DIMMnet-2 prototype.

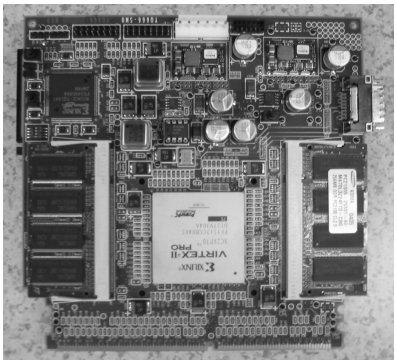

図 3 DIMMnet-2 試作基板
Fig. 3 Picture of DIMMnet-2 prototype.

2.2.1 DIMMnet-2 試作基板

DIMMnet-2 試作基板は，DIMMnet-2 ネットワークインタフェースコントローラの ASIC 化のための機能検証，論理検証を目的とした，DIMMnet-2 のプロトタイプである．コントローラ部に FPGA を搭載しており，これにネットワークインタフェースコントローラの実装を行う．FPGA を用いるため，設計や実装の変更が容易であり，様々な検証の結果を反映させていくことが可能である．FPGA 上には高速インターコネクトの IP が搭載されており，InfiniBand へ

の接続が容易に実現可能である．

図 2 に試作基板の構成を，**図 3** に試作基板の概観をそれぞれ示す．

本試作基板では，FPGA に Xilinx 社の Virtex-II Pro XC2VP70-7FF1517C を用いている．このチップは InfiniBand，Fibre Channel や Gigabit Ethernet に対応した高速シリアル I/O インタフェースである RocketIO トランシーバを搭載しており，これを利用して InfiniBand スイッチ（4X: 10 Gbps）に接続する．

DIMMnet-2 試作基板はノート PC 用の汎用メモリである 200 pin DDR SO-DIMM を 2 枚搭載する．これは通信時のバッファのほか，ホスト PC のデータ記憶領域として使用する．現在，256 MByte の SO-DIMM を 2 枚搭載しているが，将来的には SO-DIMM の 1 枚あたりの容量や枚数を増加させることによって，ホスト PC のメモリスロット 1 本あたりに搭載可能な最大メモリ容量より多くの記憶領域を設け，大規模な分散共有メモリシステムを構築することを視野に入れている．

なお，試作基板は FPGA を用いているため，高い動作周波数での稼働が困難である．現在，FPGA を 100 MHz で動作させることで PC1600 の規格での動作に対応している．

3. DIMMnet-2 ネットワークインタフェースコントローラ

本章では，設計を行った DIMMnet-2 のネットワークインタフェースコントローラについて述べる．

3.1 ホスト PC からの間接アクセス

DIMMnet-2 のネットワークインタフェースコントローラの特徴は，インタフェース上の SO-DIMM に対する間接アクセスである．メモリバスインタフェースに密接して設けたコントローラ内部の複数のバッファやレジスタにアクセスすることで，コントローラに対してコマンドの発行やデータの読み書きを行い，間接的に SO-DIMM にアクセスする．

間接アクセス手法では，ホストから，直接ネットワークインタフェース上の SO-DIMM にアクセスできないものの，ローカルとリモートの双方のネットワークインタフェース上の SO-DIMM に対して統一した手段でアクセスすることが可能となる．

このためのバッファとして以下の 3 種類を用意する．

- Write Window：ホストからネットワークインタフェース上，あるいはネットワーク経由でリモートのネットワークインタフェース上の SO-DIMM にデータを送信するためのバッファ．ホストからは書込み専用である．
- Prefetch Window：ローカルのネットワークインタフェース上，あるいはネットワーク経由でリモートのネットワークインタフェース上の SO-DIMM から読み出したデータを格納するバッファ．ホストからは読み出し専用である．
- LLCM（Low Latency Common Memory）：ホストとコントローラの両者から直接読み書き可能な共有メモリ．パケット受信ステータスの読み書きやサイズの小さな転送におけるバッファ等，汎用的に用いるための領域である．

また，ホストからの要求発行，モード設定，ネットワークコントローラの状態を読み出すための制御用レジスタが用意され，これらへのアクセスによってホストはネットワークインタフェースを制御する．

これらのバッファ，および制御レジスタは，それぞれ異なる物理アドレスにマップされ，用途に応じて，Pentium Pro 以降の IA32 アーキテクチャのプロセッサで利用可能な MTRR（Memory Type Range Register）の設定を行うことによってアクセスの高速化を図る．

Write Window はホストから読み出しを行わないため，キャッシュを汚さずに高い書込みバンド幅を得られる Write Combining 属性とする．

Prefetch Window は SO-DIMM から読み出したデータが格納される．そのため，通常のメモリと同様に MTRR による設定を何も行わない場合，CPU のキャッシュ上のデータとの間で不整合が生じてしまう．一方で，Uncachable 属性に設定すると，読み出し時のバンド幅が大幅に低下してしまう[6]．また，Write Combining 属性は書込み時のバンド幅は高いものの，読み出し時のバンド幅向上には効果がない．Write Back 属性の場合，読み出したデータは CPU のキャッシュに格納されるため，キャッシュにヒットする限り，高い読み出しバンド幅を得ることができる．そこで Prefetch Window は Write Back 属性にし，プリフェッチ要求を発行する前に，キャッシュをライン単位で無効化する CLFLUSH 命令を用いてキャッシュのラインを無効化し，プリフェッチ完了後に PREFETCHNTA 命令を用いてプリフェッチしたデータをキャッシュに格納する．このようにすることで読み出し時のバンド幅を向上させることが可能となる[8]．

LLCM はホストから読み書き可能であるが，現状ではホストからはパケット受信ステータスの読み出しを行うのみであるため，Uncachable 属性とする．ただし，DIMMnet-2 に機能を追加し，LLCM に別の用

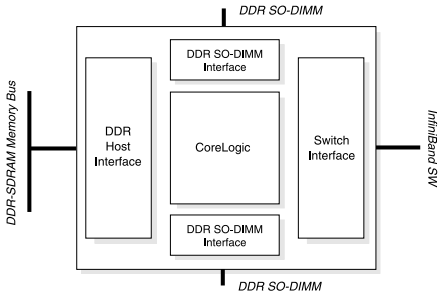

図 4 コントローラ部のブロック図
Fig. 4 Block diagram of network interface controller.

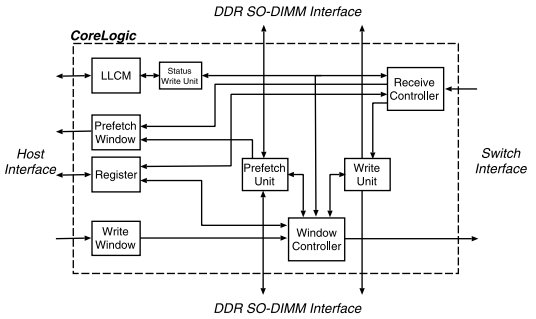

図 5 CoreLogic 部の構成
Fig. 5 Structure of CoreLogic.

途が追加された場合は，それに適した属性に変更する必要がある．

制御レジスタはホストからもコントローラからも読み書き可能であるが，内部で要求発行用レジスタ，モード設定系レジスタ，ステータス系レジスタに分かれており，レジスタによってアクセス権限は異なる．また，用途の性質上，ホストからレジスタに対しては 64 bit か 128 bit での読み書きしか行わなず，読み出しを行うレジスタはコントローラ側から書込みが行われるものがほとんどであるため，Uncachable 属性とする．

3.2 コントローラ部の構造

図 4 にコントローラ部のブロック図を示す．

コントローラ部は大きく以下の 4 つのブロックから構成される．

- DDR Host Interface 部：ホスト CPU とのトランザクションを処理するブロックである．ホスト側は DDR-SDRAM メモリバスのプロトコルに従って動作するため，ホスト CPU との間ではクロックの立ち上がりと立ち下がりの両エッジに対してそれぞれ 64 bit 幅のデータが転送される．この 64 bit 幅のデータを Host Interface 内でクロックの立ち上がりに対する 128 bit 幅の片エッジデータ転送に変換する．
- DDR SO-DIMM Interface 部：基板上の DDR SO-DIMM へのアクセスを制御するブロックである．CoreLogic 部（後述）からの 128 bit 幅，100 MHz 片エッジのデータ転送を SO-DIMM の 64 bit 幅，100 MHz 両エッジ転送に変換する．また，この逆の変換も行う．
- Switch Interface（SWIF）部：InfiniBand スイッチとのインタフェースとなるブロックであり，End-to-End の再送機構を持つ．
- CoreLogic 部：ネットワークインタフェースの制御部である．送信パケットの生成や受信パケットの解析といった通信処理や，SO-DIMM Interface 部へのアクセス要求の制御を行う．

なお，試作基板では，DIMMnet-1 を InfiniBand に接続するルータである bDais [9] の設計を SWIF 部に流用することによってコントローラの開発期間の短縮を図る．

以降では CoreLogic 部について重点的に述べる．

3.2.1 CoreLogic 部の構成

図 5 に CoreLigic 部の構成を示す．

DIMMnet-2 の CoreLogic は Write Window，Prefetch Window，LLCM，制御用レジスタ等のホストとのインタフェース，および以下の制御部から構成される．

- Prefetch Unit：SO-DIMM からの読み出し制御部
- Write Unit：SO-DIMM への書込み制御部
- Receive Controller：受信パケット処理部
- Window Controller：送信パケットのヘッダ生成等の送信処理部
- Status Write Unit：パケット受信ステータス書込み制御部

Myrinet や QsNET ではネットワークインタフェースコントローラに RISC プロセッサを内蔵し，上位の通信ライブラリを RISC プロセッサ上で実行するが，DIMMnet-2 では低レイテンシな通信を実現するためにこのような RISC プロセッサを用いずに，種々の通信処理をハードワイヤードで実装する．RISC プロセッサを搭載しないため，例外処理やコントローラの制御を代替の手段で提供する必要がある．DIMMnet-2 では制御用レジスタを介して，ホスト CPU が例外処理や制御を行うことで実現する．

また，ホストから SO-DIMM に対して間接アクセス方式を採用したことにより，SO-DIMM の領域はホストの MMU（Memory Management Unit）の物理–仮想アドレスの管理対象とはならない．したがって，SO-DIMM の領域にアクセスする際には，ホスト

からコントローラに対して SO-DIMM の物理アドレスを直接指定するため，コントローラ内部にアドレス変換用の TLB を内蔵する必要がなく，コントローラの構造を簡潔にすることが可能となった．

Prefetch Unit や Write Unit は SO-DIMM に対して，単純な連続した領域に対する読み書き以外に，等間隔アクセスやリストアクセスといった，不連続な領域に対する読み書きをサポートする[8]．DDR-SDRAM は連続領域に対するアクセス時にバースト長に応じた高いバンド幅を得ることができる構造になっているが，一方で，不連続なデータに対するアクセスに対しては不必要なデータを多く読み出してしまうため，読み出したデータに対する有効なデータの割合が低く，この点で，実バンド幅は低下する．そこで，このような機能をコントローラに持たせることにより，ホスト CPU に対して不連続なアクセスに対しても有効なデータの割合を高めることで実バンド幅を向上させる．これにより，CPU のキャッシュヒット率やメモリバスの利用効率を向上させることが可能となる．このように，DIMMnet-2 のネットワークインタフェースコントローラはネットワークコントローラとしての機能だけでなく，メモリコントローラとしての機能もあわせ持つ．

これらのモジュールを FPGA 独自の機能を極力用いずに実装する．Virtex-II Pro には PowerPC をはじめとする Xilinx 社が提供する様々な IP を搭載することが可能であるが，ASIC 化の際にはこれらの IP をそのまま使用することは困難であるため，要求処理用の FIFO や SWIF 部の RocketIO 等，必要最小限の IP の使用にとどめている．

試作基板では LLCM は 64 KByte，Prefetch Window，Write Window は Window1 枚あたり 512 Byte とし，Prefetch Window を 8 枚，Write Window を 4 枚搭載する．LLCM は汎用的に用いるために容量を大きめに確保し，Prefetch Window，Write Window は限定的な利用法で用いるために小容量の Window を複数枚搭載するという構成になっている．試作基板ではネットワークインタフェースを同時に利用可能なプロセスの数は 2 プロセスであり，ネットワークインタフェース上の資源は各プロセスに均等に割り当てられる．

3.3 DIMMnet-2 におけるデータ転送処理

DIMMnet-2 でのデータ転送処理は以下のように大別される．矢印はデータの転送方向を示す．なお，リモートとの通信においては，通信を起動した側をローカル側とする．

(1) Local (Host) – Local (DIMMnet-2)
  (a) Write Window → SO-DIMM
  (b) Prefetch Window ← SO-DIMM
(2) Local (Host) – Remote (Host/DIMMnet-2)
  (a) Write Window → Prefetch Window/SO-DIMM
  (b) Prefetch Window ← SO-DIMM
(3) Local (DIMMnet-2) – Remote (Host/DIMMnet-2)
  (a) SO-DIMM → Prefetch Window
  (b) SO-DIMM ↔ SO-DIMM

(1) の処理はローカルの DIMMnet-2 上の SO-DIMM に対するアクセスであり，連続したアドレスからの読み書き以外に，等間隔アクセスやリストアクセスといった不連続アクセスに対応する．

(2)–(a) の処理は Write Window に書き込んだデータをそのままパケットとしてネットワークに送出する処理であり，Block On The Fly（BOTF）通信[10] と呼ばれる．BOTF では，ホストからパケットヘッダを含むパケットイメージ全体を Write Window に書き込む．パケットイメージのパケットヘッダ部分を書き換えることで，任意のパケットを発行することが可能である．また，コントローラからデータを読み出す場合，Write Window は SO-DIMM よりも低レイテンシで読み出し可能であるため，SO-DIMM からデータを転送するよりも低レイテンシでのパケット発行が可能となる．ただし，1 回の BOTF では，ヘッダを含む最大転送サイズは Write Window の Window1 枚分の大きさ（512 Byte）に制限される．(2)–(b) の処理はリモートの SO-DIMM のデータを読み出し，ローカルの Prefetch Window に格納する処理である．(1)–(b) と同様の不連続アクセスによる読み出しも可能である．

(3) の処理はローカルの DIMMnet-2 上の SO-DIMM とリモートの DIMMnet-2 上の SO-DIMM または Prefetch Window との間のデータ転送である．転送するデータをローカルの SO-DIMM から読み出す際，またはリモートの SO-DIMM から読み出す際には (1)–(b) と同様の不連続アクセスによる読み出しも可能である．

DIMMnet-2 ではこれらの処理を基本通信命令としてハードウェアで実装し，ホストから 64 bit または 128 bit 長の命令を要求発行用レジスタに対して書き込むことで要求を発行する．(2)–(a) 以外の処理は 128 bit で発行し，(2)–(a) の BOTF に関してはパケット構築に必要な情報が Write Window に書き込まれているため，レジスタに対しては 64 bit の書込み

を行うことで要求発行が可能である．

リモートとの通信は同一の PGID（Process Group ID）を持つプロセスどうしでのみ行うことができる．PGID は同一の並列処理に参加しているプロセス群に固有の ID である．並列プロセスを起動させる最初の時点で，並列プロセスに対して DIMMnet-2 上の資源を割り当てるのと同時に，特権プロセスがプロセス ID と PGID を組にしてコントローラ上のレジスタに設定する．このレジスタ上の PGID は特権プロセス以外のプロセスからは操作ができないようになっている．パケットを構築する際，コントローラ側でレジスタに設定された PGID をヘッダに付加することで異なる PGID のプロセスを宛先に指定できないようにする．これにより，プロセスグループ間での通信が禁止されるため，他プロセスグループに属するプロセスの SO-DIMM 領域に対するアクセスを防ぐことが可能となる．

4. 評　　価

本章では，DIMMnet-2 における最も典型的な低遅延通信手法である BOTF のレイテンシを試作基板を用いて測定し，基本通信性能を示す．

BOTF 時にホスト側から Write Window に書き込むパケットイメージのフォーマットを**図 6** に示す．1 ライン 64 bit であるが，これに Window Controller でパケットの先頭ライン，最終ライン，それ以外のラインを識別するために 2 bit の識別子を付加し，計 66 bit のデータとして SWIF に転送する．このデータを SWIF で InfiniBand のパケットにカプセル化し，ネットワークにパケットを送出する．受信側の SWIF では受信したパケットから DIMMnet-2 のパケットを取り出し，Receive Controller にデータの受信通知を行う．

Header 部はパケットの種別や転送サイズによって 2 ラインもしくは 3 ラインとなる．1st Header 部にはパケットサイズ，要求する処理の内容，送信先の PGID 等のプロセス識別子が記述され，2nd Header 部にはデータの格納先アドレス等が記述される．

3 ラインまたは 4 ライン目から最後のラインまでが転送するデータ本体となる．BOTF の場合，Data 部の最大転送サイズは Header が 2 ラインの場合で 512 Byte − 8 Byte×2（Header 部）= 496 Byte となる．

4.1 BOTF によるパケットの送信処理

BOTF の処理の流れは以下のようになる．

(1) ホストから Write Window に転送するデータを書き込む．

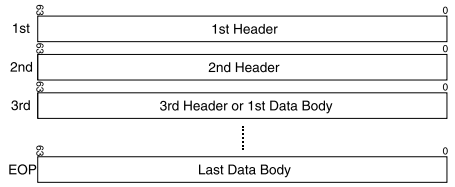

図 6 パケットフォーマット
Fig. 6 Packet format.

表 1 レイテンシの測定環境
Table 1 Measure environment of latency.

CPU	Pentium4 2.6 GHz (2,625.918 MHz)
Chipset	VIA VT8751A
Memory	PC-1600 DDR-SDRAM 512 MByte ×1
	DIMMnet-2 ×1
OS	RedHat8 (Kernel 2.4.27, single user mode)
gcc	3.3.5 (no compile option)

(2) ホストから要求発行用レジスタに BOTF の要求を発行する．

(3) 要求を Window Controller に転送し，Window Controller を起動する．

(4) Window Controller が Write Window からデータを読み出す．

(5) (4) と同時に，Window Controller が読み出したデータに 2 bit 付加し，SWIF に転送する．

(6) SWIF は転送されたデータを InfiniBand パケットにカプセル化し，ネットワークに送出する．

これらの処理のうち，(1)〜(5) の処理に要するレイテンシをヘッダを 2 ライン（16 Byte），転送データサイズを 8〜496 Byte の範囲で 8 Byte ずつ変化させ，**表 1** に示される環境で測定した．

(6) の処理は bDais を用いた評価から得られた値を用いている．SWIF のレイテンシは転送サイズに依存性があり，サイズの大きな転送であるほど，レイテンシが大きくなる．8 Byte 送信時の SWIF 部のレイテンシは 0.188 μs であり，512 Byte 送信時のレイテンシは 0.676 μs である．

(1)〜(5) と (6) とでレイテンシの測定を分離している理由として，コントローラのみのレイテンシを測定する場合，Window Controller の処理が完了したことを検知することはステータスレジスタを参照することで可能であるが，SWIF の送信側の処理が完了したことを検知するのが困難であるためである．また，SWIF 部のレイテンシは bDais の評価から得られるため，ホスト〜CoreLogic までと SWIF 部で分離して評価を行った．

そのため，評価用にコントローラの構造を変更し，

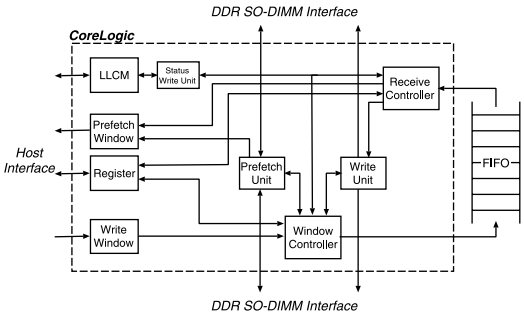

図 7 パケット受信処理評価用コントローラ
Fig. 7 Controller for evaluation of packet receiving.

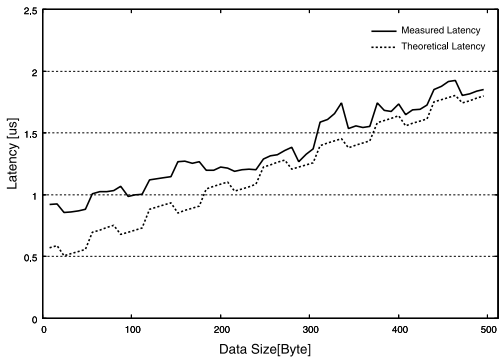

図 8 BOTF のレイテンシ
Fig. 8 Latency of BOTF.

SWIF 部に SWIF を模した FIFO を挿入したコントローラを用いて評価を行った．評価用コントローラのブロック図を**図 7** に示す．Window Controller から送出されたパケットは FIFO を通じて Receive Controller で処理される．このコントローラでパケットの受信処理の評価も行うため，Window Controller が転送するパケットすべてを書き込むまで Receive Controller が FIFO からデータを読み出さないように FIFO の信号線を制御し，送信側と受信側の処理が重ならないようにしている．

BOTF のレイテンシの測定結果を**図 8** に示す．

図 8 の Measured Latency が実機上で測定された値（以下，実測値）であり，Theoretical Latency が理論上の BOTF のレイテンシ（以下，理論値）である．理論値は，以下のように求めた値を足し合わせたものである．

- (1) の処理：SSE 命令によって Write Window にデータを書き込む処理を 100 万回実行し，その平均をとる．
- (2) の処理：MMX 命令によって要求発行用レジスタに要求を発行する処理を 100 万回実行し，その平均をとる．
- (3)〜(5) の処理：Verilog シミュレーションによって測定．
- (6) の処理：Measured Latency と同様に，bDais による測定値を用いる．

表 2 BOTF（8 Byte 送信時）のレイテンシの内訳
Table 2 Detail of the latency of BOTF (8 Byte send).

処理内容	処理時間 [μs]
ホストから Write Window へデータの書込み	0.201
ホストから Register へ BOTF 要求書込み	0.081
BOTF 要求発行	0.010
BOTF の要求処理	0.090
SWIF 通過遅延	0.188
合計	0.570

理論値と実測値とでは，全般的に理論値の方がレイテンシが小さいが，これは実機を用いた測定の際のステータスレジスタのポーリングが影響している．ステータスレジスタのポーリングに要するレイテンシを実機上で測定した結果，ポーリング 1 回につき 0.233 μs 程度要することが測定された．ポーリングレイテンシが最大になるのは，ステータスレジスタの値を読み出した直後に BOTF 処理が完了する場合であり，この場合のレイテンシは 0.350 μs 程度になると予想される．実測値と理論値のレイテンシの差は転送サイズが 152 Byte のときに最大であり，約 0.411 μs である．この値はポーリングレイテンシの予測最大値に近い値であり，ポーリング以外の処理に要するレイテンシは実測値と理論値でほぼ一致すると考えられる．実際に BOTF を何らかのアプリケーションで実行する場合には，送信側のステータス領域のポーリングは通信レイテンシに影響しないため，理論値に近いレイテンシで BOTF によるパケットの送出が可能であると思われる．

QsNET II でも BOTF と同様の機能が提供され，この機能を用いた場合の 8 Byte のデータ送信時のレイテンシは 1.3 μs である．対して，DIMMnet-2 では 8 Byte のデータ送信に要するレイテンシは実測値で 0.922 μs，理論値で 0.570 μs であり，BOTF が理論値に近いレイテンシで実行可能であるならば，QsNET II の約 43.8%のレイテンシとなり，きわめて低いレイテンシでデータ転送が可能である．8 Byte 転送時のレイテンシ（理論値）の内訳を**表 2** に，BOTF 要求に対する Window Controller の処理内容とそれに要するクロック数を**表 3** に示す．

レイテンシを低く抑えられた理由として，(1) PCI バスよりもアクセスレイテンシの小さいメモリバスを使用していること，(2) 基本通信命令をハードワイヤー

表 3 BOTF（8 Byte 送信時）実行時の Window Controller の処理の内訳
Table 3 Detail of transaction of BOTF (8 Byte send) at Window Controller.

処理内容	クロック数
Window Controller を起動	0
Window Controller が要求を取得	+2
Window Controller が要求を識別	+1
BOTF 開始	+1
Write Window へデータの読み出し要求	+1
1st Header 取得・PGID フィールドの書き換え	+1
1st Header を SWIF に転送・2nd Header 取得	+1
2nd Header を SWIF に転送・1st Data 取得	+1
読み出したデータを EOP まで SWIF に転送	+1
合計	9

ドで実装し，コントローラ内部で TLB を用いたアドレス変換を必要としない構造にしたことによる通信処理に要するクロック数の削減，(3) DIMMnet-2 のパケットヘッダや BOTF 要求の発行に必要なコマンドの bit 数を低く抑えたことがあげられる．

今回の BOTF の評価は転送するデータが CPU のキャッシュに存在することを前提としたものである．キャッシュに存在しないデータを BOTF で転送する場合，データを Write Window に書き込む際のレイテンシがボトルネックとなり，性能は 1/2〜1/3 程度にまで落ちることが測定により明らかになっている．そのため，BOTF を低レイテンシで実行するには，転送するデータをあらかじめ PREFETCHNTA 命令等で CPU のキャッシュに入れておく必要がある．また，このような命令を使用しない場合でも，何らかの処理を行った結果をただちにネットワークに送出する場合等は，処理結果がキャッシュ上に存在するため，高いパフォーマンスを得ることが可能と予想される．

4.2 BOTF の連続発行

496 Byte のデータを転送する際の Window Controller での処理に要するクロック数は Verilog シミュレーションより 70 クロックと分かった．したがって，コントローラを 100 MHz で動作させた場合，BOTF で 496 Byte 転送した際に，Window Controller で 0.700 μs 要することになる．

また，ホストから 24〜512 Byte の範囲で Write Window にデータを書き込む際のレイテンシは，464 Byte 前後の場合が最大となり，約 0.409 μs であり，要求発行に要するレイテンシは約 0.081 μs であったことから，BOTF を Window Controller で処理している間にホストから 2 枚目の Write Window にデータを書き込み，次の BOTF 要求を発行することが可能である．

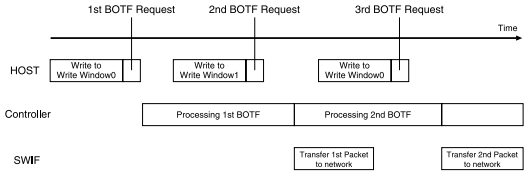

図 9 BOTF の連続発行
Fig. 9 Continuous issue of BOTF.

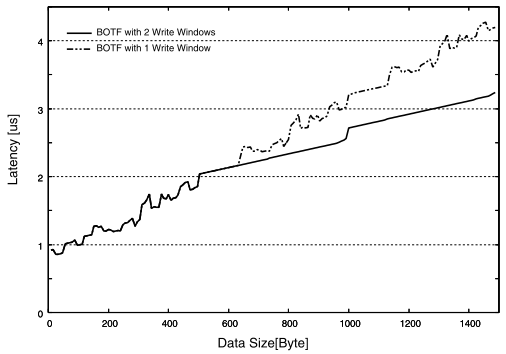

図 10 BOTF 連続発行時のレイテンシ
Fig. 10 Latency of BOTF with continuous issue.

この流れの概略を**図 9** に示す．

2 枚の Write Window を交互に使用して BOTF で 8〜1,488 Byte のデータを転送した場合のレイテンシを**図 10** に示す．図 10 は 8〜496 Byte までは 1 枚の Write Window を使用した場合の BOTF の評価値に SWIF のレイテンシを加えた値をそのまま用い，それ以降のサイズの場合は，各 BOTF のレイテンシを 1 回目の BOTF と同様であると仮定したグラフである．また，比較対象として Write Window を 1 枚のみ使用して BOTF を連続発行した際のレイテンシも同時に示す．

図 10 より，Write Window を使用する枚数によって，BOTF を連続発行した際の 2 回目以降の BOTF において顕著な差が見られることが分かる．Write Window を 2 枚使用することによって，2 回目以降の BOTF においてはホストから Write Window へのパケットイメージの書込み，要求発行レジスタへの要求発行，Window Controller での要求の識別に要するレイテンシが Window Controller における前回の BOTF の処理とオーバラップすることで隠蔽されるため，Window Controller を 1 枚のみ使用した場合と比較してレイテンシが低く抑えられている．しかしながら，2 回目以降の BOTF では，転送サイズによっては SWIF において前回の BOTF の SWIF の処理が完了していないことによる待ち状態が発生する．前回の BOTF で

512 Byte 転送した場合の SWIF の遅延は 0.676 μs であり，次の BOTF の (1)〜(5) の処理がこの時間以内に完了する場合は SWIF で待ち状態が発生する．1 枚の Write Window で BOTF を連続発行した際のグラフにおいて 504〜632 Byte，1,000〜1,128 Byte の範囲でグラフが直線になっているのはこの影響によるものである．

4.3 パケット受信処理

DIMMnet-2 では，パケットのデータ部をネットワークインタフェース上の SO-DIMM か，コントローラ内部の Prefetch Window のどちらかに受信することが可能である．どちらに受信するかは送信側で指定され，パケットヘッダに書き込まれる．

受信側の処理の流れを以下に示す．

(1) SWIF がパケットを受信し，Receive Controller にパケットが到着したことを通知する．
(2) Receive Controller が起動し，パケットヘッダを SWIF から読み出す．
(3) パケットヘッダを解析し，データ部を SWIF から読み出す．
(4) (3) で読み出したデータ部を最終ラインまで Prefetch Window または SO-DIMM に書き込む．
(5) 書込み完了後，Receive Controller は Status Write Unit に対し，完了通知を LLCM へ書き込むための要求を発行する．
(6) Status Write Unit が LLCM へ完了通知を書き込む．

図 7 に示されるコントローラに対して BOTF 要求を発行し，LLCM にパケット受信ステータスが書き込まれるまでのレイテンシを測定する．ホストからは LLCM をポーリングすることにより，パケットの受信を検知する．測定範囲は Write Window へのデータの書込みが完了し，要求発行レジスタに要求を発行する直前から，LLCM に受信ステータスが書き込まれたことを検知するまでである．この測定値から要求発行に要するレイテンシと CoreLogic 内部の送信側の BOTF 処理に要するレイテンシの値を引くことによって，CoreLogic の受信側のパケット処理に要するレイテンシを求める．この値に，bDais の受信側のレイテンシの値を加算し，SWIF を含めたパケット受信処理のレイテンシとした．受信先には Prefetch Window と SO-DIMM の両方を指定して評価を行った．

これらの結果を**図 11** に示す．

図 11 の Prefetch Window, SO-DIMM のグラフが実機上で測定された値（以下，実測値）であり，Theoretical Latency が Prefetch Window に受信した際の理論上のレイテンシ（以下，理論値）である．理論値はパケット受信処理を Verilog シミュレーションによって測定した値に，SWIF の受信側のレイテンシを足したものである．

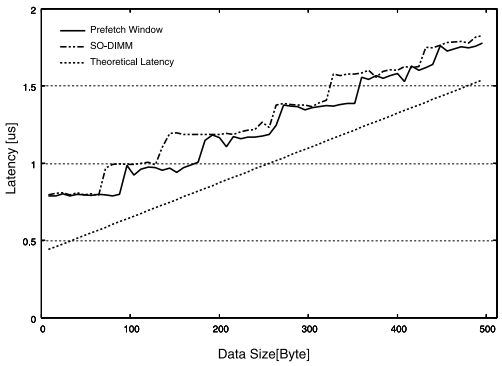

図 11 パケット受信時のレイテンシ
Fig. 11 Latency of packet receiving.

表 4 受信処理の処理時間の内訳
Table 4 Detail of the latency of receive.

処理内容	処理時間 [μs]
SWIF 通過遅延	0.312
受信処理	0.130
合計	0.442

グラフより，Prefetch Window に受信する際の実測値と理論値の差はほぼ 0.16〜0.35 μs の範囲に収まっている．これは LLCM のポーリングと送信側のレイテンシの影響によるものであると考えられる．LLCM のポーリングにはステータスレジスタのポーリングと同様に 1 回あたり約 0.233 μs 要することが測定されており，ポーリングとパケット受信処理完了のタイミングによって，約 0.117〜0.350 μs の範囲で変動すると思われる．そのため，この実測値と理論値の差は，ほとんどがポーリングによるものであると推測され，受信処理そのものに関してはホスト側はまったく影響しないことからも，受信処理はほぼ理論値に近いレイテンシで完了することができると考えられる．

8 Byte のデータを Prefetch Window に受信する際の理論上のレイテンシは 0.442 μs である．8 Byte 受信時のレイテンシの内訳を**表 4** に，受信側の処理内容とそれに要するクロック数を**表 5** に示す．

Prefetch Window は Prefetch Unit からも書込みが行われるため，アクセス制御のための処理が必要となる．また，ステータス書込みの処理のため，受信処理は送信処理よりも 4 クロック長くかかる．しかし，Receive Controller は Status Write Unit にステータ

表 5 受信処理（8 Byte 受信時）の処理の内訳
Table 5 Detail of transaction of receive 8 byte data.

処理内容	クロック数
SWIF にパケットが到着したことを検出	0
SWIF へデータの読み出し要求	+1
1st Header 受信	+1
2nd Header 受信・要求の識別	+1
Prefetch Window へ書込み開始	+1
Prefetch Window への書込み権要求	+1
Prefetch Window への書込み権取得	+1
SWIF へデータの読み出し要求	+1
Prefetch Window へ EOP までデータを書き込む	+1
Prefetch Window への書込み権解放	+1
Status Write Unit へステータス書込み要求	+1
Status Write Unit がステータスを書き込む	+3
合計	13

ス書込み要求を発行すると，ただちに次のパケットの受信処理に入ることが可能である．したがって，連続してパケットを受信した際には，パケットの受信処理と 1 つ前に受信したパケットのステータス書込みをオーバラップさせて動作させることができる．

Prefetch Window への受信と SO-DIMM への受信を比較すると，全般的に SO-DIMM への受信の方がレイテンシが大きい．これは，Prefetch Window にデータを書き込む際にはアドレスと同時に Write Enable とデータを転送するだけでデータの書込みを行えるのに対し，SO-DIMM への書込みには RAS 信号，CAS 信号を入力した後にデータを入力することによって書込みが行われるため，Prefetch Window に比べて書込みに要するレイテンシが大きいためである．また，SO-DIMM は DRAM であるため，リフレッシュが定期的に必要である．リフレッシュ中はデータの書込み等が行えないため，リフレッシュとデータの書込みが重なった場合は書込みはリフレッシュ完了まで待つことになる．したがって，SO-DIMM への書込みレイテンシは書き込むタイミングによって変動する．

今回の評価では LLCM をポーリングすることにより，パケットの受信を検知したが，LLCM をポーリングするのは CPU 資源の浪費であるため，パケットを受信した際には，DIMMnet-2 からホスト CPU に対して割込みをかけることでパケットの受信を通知する手法が望ましい．しかし，メモリバスには割込み線が存在しないため，メモリバスを利用してホスト CPU に割込みをかけることは不可能である．そこで，PCI バスに割込み専用のボードを装着し，DIMMnet-2 からそのボードに割込み線を接続することで，パケットの受信時に PCI バスから割込みをかけ，パケットの到着をホストに通知する手法を採用する予定である．

表 6 コントローラ部の配置配線結果
Table 6 Result of placement and route of network interface controller.

スライス数	14,202 (42%)
Block RAM	170 (51%)
動作周波数	98.261 MHz

合成ツール：Xilinx ISE 6.3i SP3
デバイス：XC2VP70-7FF1517C

4.4 合成結果

表 6 にネットワークインタフェースコントローラを Xilinx 社の提供する合成ツールである ISE（ISE6.3i SP3）を用いて配置配線を行った結果を示す．

回路規模は FPGA の回路規模の半分程度で済んでいるものの，論理合成の時点で動作周波数が PC1600 の規格で動作させるために必要な動作周波数である 100 MHz を下回ってしまっている．現状では配置配線後の動作に影響は出ていないが，ホストインタフェース部，SO-DIMM インタフェース部がクリティカルパスであることが分かっており，今後は，これらの論理を見直す必要がある．

また，Block RAM の使用量が大きく，170 個の Block RAM のうち，半数以上を Write Window，Prefetch Window，LLCM で消費していることが判明している．これらはホストから直接アクセスされることからホストインタフェースと接続されるモジュールである．そのため，FPGA 上のどの Block RAM を使用してもよいというわけではなく，ホストインタフェースの近辺に設けられている Block RAM を用いることが動作周波数の維持，向上という観点から望ましく，合成時にそのような制約を設けている．したがって，これらのモジュールに割り当てることが可能な Block RAM は限られており，Prefetch Window や Write Window の枚数，LLCM の容量等を再検討し，最適なサイズに変更する必要があると思われる．

このコントローラを ASIC 化する際には Virtex-II Pro に搭載された IP である RocketIO が使用できなくなるため，SWIF 部の一部の論理を InfiniBand の IP に変更する，またはコントローラとは別チップとして基板上に搭載する必要がある．そのため，ASIC 化時には回路規模は同程度または小さくなることが予想される．また，FPGA に搭載されている Block RAM は位置が固定されているため，配線が長くなる場合があるが，ASIC 化の際には RAM の位置もレイアウトの段階で変更可能なため，配線長の点では有利になると思われる．しかし，現状ですでに 200 pin DDR SO-DIMM インタフェースを 2 つ，184 pin DDR ホ

ストインタフェースを 1 つ持つため，I/O ピンの数は回路規模に比べて多いものになる．

ASIC 化することにより，基板面積は小さくなると予想される．これは，FPGA をコンフィギュレーションするための素子が必要なくなるためである．

ASIC 化したことにより動作周波数が向上し，PC1600 より高速な DDR の規格に対応可能になった場合，ホストからのアクセスレイテンシ，およびコントローラの処理速度が向上するため，さらに低レイテンシでの通信が可能となると予測される．表 2 から，8 Byte のデータ転送時においてホストから DIMMnet-2 に対するアクセスがレイテンシの半分程度を占めており，より高速な規格に対応することによるレイテンシの改善への影響は大きいものと予想される．

4.5 デュアルチャネルへの対応

BOTF とパケット受信処理の評価により，きわめて低いレイテンシでパケットの送受信が可能であることが示された．しかしながら，DIMMnet-2 を単純に DDR-SDRAM スロットに装着した場合，デュアルチャネルでの動作ができなくなり，ノードの性能が低下する．そのため，DIMMnet-2 をデュアルチャネルでの動作に対応させることは必須であり，現在，DIMMnet-2 を 2 枚装着する等，デュアルチャネル動作への対応の検討を進めている．

5. 関 連 研 究

PC クラスタに用いられるインターコネクトの中でも通信レイテンシが特に低いものとして，Quadrics 社の QsNET があげられる．

PCI-X に対応した QsNET II [4] QM-500 ネットワークインタフェースには，200 MHz で動作する Elan4 ネットワークプロセッサが搭載されている．Elan4 はサイズの小さいメッセージを低レイテンシで処理する STEN（Short Transaction Engine）を内蔵しており，STEN は BOTF と同様の機能を提供する．QsNET II は 8 Byte のデータ転送を 1.3 μs で完了することができ，このうち，Elan4 での処理によるレイテンシは約 0.240 μs である[11]．

一方で，DIMMnet-2 は 8 Byte データのホストからの送信を 0.570 μs で完了することができる．このうち，コントローラ部のレイテンシは 0.288 μs であり，Elan4 の半分の動作周波数でありながら，Elan4 の約 1.2 倍程度のレイテンシに抑えている．ホストのレイテンシも含めると，QsNET II の 43.8%程度のレイテンシに抑えられている．

6. まとめと今後の課題

本報告では PC クラスタ向けインターコネクトである DIMMnet-2 試作基板に搭載するネットワークインタフェースコントローラについて述べ，性能評価を行った．その結果，8 Byte データ送出のレイテンシが 0.570 μs と見積もられ，100 MHz という比較的低い動作周波数ながら，QsNET II の 43.8%程度のレイテンシで送信を行うことが可能であることが示された．

また，Xilinx 社の合成ツールを用いて回路規模を示し，それを元に ASIC 化を行った際の性能や回路規模等の予測を示した．ASIC 化により動作周波数が向上し，より高速な DDR の規格に対応することが可能となれば，より低レイテンシな通信が可能である．

今後は，ノード間をスイッチを介して接続し，ノード間通信の性能を測定していくと同時に，論理の見直しを行い，コントローラの動作周波数の向上を目指す．また，デュアルチャネルへの対応や DIMMnet-2 を用いることによる実アプリケーションへ与える効果を調べていく予定である．

謝辞 本研究は総務省戦略的情報通信研究開発推進制度の一環として行われたものである．DIMMnet-2 の開発に関する議論，開発にご参加いただいている（株）日立 IT の今城氏，岩田氏，上嶋氏，慶應義塾大学の西助手，渡邊氏，大塚氏に感謝いたします．

参 考 文 献

1) Boden N.J., Cohen, D., Felderman, R.E., Kulawik, A.E., Seitz, C.L., Seizovic, J.N. and Su, W.-K.: Myrinet — A gigabit per second local area network, *IEEE Micro*, Vol.15, No.1, pp.29–36 (1995).
2) Petrini, F., Fang, W.-C., Hoisie, A., Coll, S. and Frachtenberg, E.: The Quadrics Network: High-Performance Clustering Technology, *IEEE Micro*, Vol.22, No.1, pp.46–57 (2002).
3) InfiniBand Trade Association: http://www.infiniba-ndta.org/
4) Addison, D., Beecroft, J., Hewson, D., McLaren, M. and Roweth, D.: QsNet II: Performance Evaluation, http://www.quadrics.com/ (2003).
5) 田邊 昇，山本淳二，工藤知宏：メモリスロット搭載型ネットワークインタフェース DIMMnet-1 における細粒度通信機構，情報処理学会アーキテクチャ研究会，Vol.2000-ARC-137, pp.65–70 (2000).
6) 田邊 昇，濱田芳博，三橋彰浩，中條拓伯，天野

英晴：メモリスロット装着型ネットワークインタフェース DIMMnet-2 の構想，情報処理学会アーキテクチャ研究会，Vol.2003-ARC-152, pp.61–66 (2003).
7) Liu, J., Chandrasekaran, B., Wu, J., Jiang, W., Kini, S., Yu, W., Buntinas, D., Wyckoff, P. and Panda, D.K.: Performance Comparison of MPI Implementations over InfiniBand, Myrinet and Quadrics, *SC2003* (2003).
8) 田邊 昇，中武正繁，箱崎博孝，土肥康孝，中條拓伯，天野英晴：プリフェッチ機能付きメモリモジュールによる不連続アクセスの連続化，情報処理学会アーキテクチャ研究会，Vol.2004-ARC-157, pp.139–144 (2004).
9) 濱田芳博，西 宏章，田邊 昇，天野英晴，中條拓伯：bDais: DIMMnet-1/InfiniBand 間ルータの開発，先進的計算基盤システムシンポジウム SACSIS2004，Vol.2004，No.6，pp.133–134 (2004).
10) 田邊 昇，山本淳二，濱田芳博，中條拓伯，工藤知宏，天野英晴：DIMM スロット搭載型ネットワークインタフェース DIMMnet-1 とその高バンド幅通信機構 BOTF, 情報処理学会論文誌, Vol.43, No.04-005, pp.866–878 (2002).
11) Addison, D., Beecroft, J., Hewson, D., McLaren, M. and Petrini, F.: Quadrics QsNet II: A network for Supercomputing Applications, *Hot Chips 15*, Stanford University, Palo Alto, CA (2003). Available from http://www.c3.lanl.gov/~fabrizio/talks/hot03.ppt

(平成 17 年 1 月 24 日受付)
(平成 17 年 5 月 3 日採録)

北村 聡（学生会員）
2005 年慶應義塾大学大学院理工学研究科開放環境科学専攻前期博士課程修了．現在，同後期博士課程に在学．計算機アーキテクチャに関する研究に従事．

濱田 芳博
2001 年まで三菱製紙（株）に勤務．2003 年東京農工大学大学院工学研究科電子情報工学専攻前期課程修了．現在，同後期課程に在学．PC クラスタ向けネットワークインタフェースに関する研究に従事．

宮部 保雄
2005 年慶應義塾大学理工学部情報工学科卒業．現在，同大学大学院理工学研究科開放環境科学専攻前期博士課程に在学．計算機アーキテクチャに関する研究に従事．

伊澤 徹
2005 年慶應義塾大学理工学部情報工学科卒業．現在，同大学大学院理工学研究科開放環境科学専攻前期博士課程に在学．計算機アーキテクチャに関する研究に従事．

宮代 具隆
2005 年慶應義塾大学理工学部情報工学科卒業．現在，同大学大学院理工学研究科開放環境科学専攻前期博士課程に在学．計算機アーキテクチャに関する研究に従事．

田邊 昇（正会員）
1985 年横浜国立大学工学部卒業．1987 年同大学大学院工学研究科修了．同年（株）東芝に入社．1998 年より 2001 年まで新情報処理開発機構つくば研究センターに出向．並列処理，超並列計算機，ベクトルプロセッサ，PC クラスタ向けネットワークインタフェース，メモリアーキテクチャに関する研究に従事．現在，（株）東芝・研究開発センター勤務．工学博士．電子情報通信学会会員．

中條　拓伯（正会員）

1961 年生．1985 年神戸大学工学部電気工学科卒業．1987 年同大学大学院工学研究科修了．1989 年神戸大学工学部助手の後，1998 年より 1 年間 Illinois 大学 Urbana-Champaign 校 Center for Supercomputing Research and Development（CSRD）にて Visiting Research Assistant Professor を経て，現在東京農工大学大学院共生科学技術研究部助教授．プロセッサアーキテクチャ，並列処理，クラスタコンピューティング，高速ネットワークインタフェースに関する研究に従事．電子情報通信学会，IEEE CS 各会員．博士（工学）．

天野　英晴（正会員）

1986 年慶應義塾大学大学院理工学研究科電気工学専攻博士課程修了．現在，慶應義塾大学理工学部情報工学科教授．工学博士．計算機アーキテクチャの研究に従事．