計算機アーキテクチャ 152－12
（２００３．３．１０）

FPGA 実装型光通信ネットワークインタフェースによるソフトウェア分散共有メモリシステムの実装と評価

石井雅明 † 齋藤 彰一 †† 上原 哲太郎 ††
國枝 義敏 †† 中條 拓伯 †

本稿では，RHiNET/NI-NPC ネットワークインタフェースによるソフトウェア分散共有メモリシステムの実装とその評価について述べる．我々は FPGA 実装型光通信ネットワークインタフェース RHiNET-2/NI0 を利用し，ユーザレベルゼロコピー通信およびページキャッシング機構を備えたネットワークインタフェースである RHiNET/NI-NPC を実装した．また，ソフトウェア分散共有メモリシステム *Fagus* を RHiNET/NI-NPC の環境に移植し RHiNET/NI-NPC の性能を評価した．その結果，ゼロコピー通信においてはリモートメモリライトで約 48MB/sec，リモートメモリリードで約 45MB/sec のバンド幅が得られた．また，RHiNET/NI-NPC を用いた *Fagus* において LU 分解プログラムを実行したところ，受信メッセージを待つ処理などで用いられているポーリング処理が主な原因となり，現状では 100BASE の Ethernet を用いた *Fagus* にくらべプログラム実行時間の増加がみられた．

Implementation and Evaluation of a Software Distributed Shared Memory System by a Optical Network Interface with FPGA

Masaaki ISHII ,† Shoichi SAITO ,†† Tetsutaro UEHARA ,††
Yoshitoshi KUNIEDA †† and Hironori NAKAJO †

This paper describes an implementation and evaluation of software distributed shared memory system with RHiNET/NI-NPC network interface. We implement RHiNET/NI-NPC network interface for software distributed shared memory system with RHiNET-2/NI0 optical network interface which has FPGA. RHiNET/NI-NPC provides a user-level zero-copy communication and a page caching mechanism. Also, we implement *Fagus* software distributed shared memory system on RHiNET/NI-NPC and evaluat its performance. Bandwidth of zero-copy communication is up to 48MB/sec for remote memory write and 45MB/sec for remote memory read. We execut a LU decomposition program on *Fagus* with RHiNET/NI-NPC. Its execution time is longer than that on *Fagus* with 100BASE Ethernet, through polling used by waiting processing for a receiving message.

1. はじめに

近年，低コストで導入できる並列計算機システムとして，多数のパーソナルコンピュータ (PC) をネットワークで接続して構築する PC クラスタが注目されている．この PC クラスは，各 PC に計算処理を割り振り並列に処理させることで，高い計算処理能力を得ようとするものである．

この PC クラスタ上で共有メモリによる並列プログラミングを提供するシステムとして，ソフトウェア的に分散共有メモリの仕組みを実現するソフトウェア分散共有メモリ (ソフトウェア DSM) システムがある．ソフトウェア DSM システムでは，逐次プログラミングとの親和性が高い共有メモリによる並列プログラミングが可能なため，メッセージパッシングシステムにくらべ比較的容易に並列プログラミングを行える利点がある．その反面，分散共有メモリの実現に必要となる処理が並列処理のオーバヘッドとなる．

高性能で実用的なソフトウェア DSM システムを実現するため，新たなメモリコンシステンシモデルに基づくシステムや，オペレーティングシステムまたはコンパイラに改良を加えたシステムなど様々なソフトウェア DSM システムが提案・実装されている．その一方で，ソフトウェア DSM の性能を向上させるためにはこれらシステムを構成するソフトウェアを改良す

† 東京農工大学大学院工学研究科
Graduate School of Technology, Tokyo University of Agriculture and Technology
†† 和歌山大学システム工学部
Faculty of Systems Engineering, Wakayama University

るだけでなく，ハードウェアであるネットワークインタフェースの改良も進めていく必要があるといえる．

そこで我々は，ソフトウェア DSM システム向けのネットワークインタフェースである RHiNET/NI-NPC (RHiNET Network Interface with Network Page Cache) を実装した．RHiNET/NI-NPC は，ユーザレベルで実装されるソフトウェア DSM システムでの利用を想定し，ユーザレベルゼロコピー通信をハードウェアでサポートしている．また，ページの送信を高速化するため Network Page Cache と呼ばれるページキャッシング機構を備えている．

RHiNET/NI-NPC は，技術研究組合 新情報処理開発機構 並列分散アーキテクチャつくば研究室にて開発された RHiNET-2/NI0[1] 上に実装されている．RHiNET-2/NI0 は，FPGA を搭載した光通信ネットワークインタフェースであり，FPGA のコンフィギュレーションを作成することにより様々な通信プロトコルを実装することができる．

また，本研究では和歌山大学で研究されているソフトウェア DSM システムである *Fagus*[2] を RHiNET/NI-NPC 上へ移植し，RHiNET/NI-NPC と *Fagus* から構成されるソフトウェア DSM システムを構築した．

本稿では，最初に RHiNET/NI-NPC の実装と RHiNET/NI-NPC 上への *Fagus* の移植について述べる．次に，RHiNET/NI-NPC の基本性能，RHiNET/NI-NPC と *Fagus* によって構築されるソフトウェア DSM システムについて，評価方法および評価結果を述べ評価結果を検討する．最後にまとめと今後の課題を述べる．

2. RHiNET/NI-NPC の実装

RHiNET/NI-NPC は，ユーザレベルで実装されるソフトウェア DSM システムでの利用を想定したネットワークインタフェースであり，以下のような特徴を持つ．

- **ユーザレベルゼロコピー通信**
  ユーザレベル通信は，ユーザプロセスがシステムコールを利用することなく，ネットワークインタフェースに直接アクセスし通信を起動する方式である．ゼロコピー通信は，通信バッファなどを介することなく，ネットワークインタフェースが DMA によりユーザプロセス空間との間で直接データ転送を行う方式である．
  これらユーザレベル通信とゼロコピー通信を併用することにより，ユーザレベルで実装されるソフトウェア DSM システムは高速に通信を行うことができる．
- **ページキャッシング機構**
  RHiNET/NI-NPC は，ページの送信を高速化するため Network Page Cache と呼ばれるページキャッシング機構を備えており，ネットワークへ送信されるページをキャッシングしておくことが可能である．
  Network Page Cache にキャッシングされているページは，ネットワークインタフェースから直接ネットワークへ送信することができる．ホストの主記憶からページを転送する必要がなくなるので，ページ送信の高速化が期待できる．

以下，RHiNET/NI-NPC の実装について詳細を述べる．

2.1 アドレス変換

RHiNET/NI-NPC はユーザレベルゼロコピー通信を行うため，仮想アドレスを物理アドレスへ変換する機構を備えている．

また，RHiNET/NI-NPC におけるリモートメモリ領域のアドレス指定は，RHiNET[1] のグローバルアドレス ID (GID) とオフセットによって指定する方法を用いた．このため，GID を仮想アドレスへ変換する機構を備えている．

仮想/物理アドレス変換

ユーザレベルゼロコピー通信では，ユーザプロセスからネットワークインタフェースへ通知されるアドレスは仮想アドレスとなる．一方，DMA では一般に物理アドレスを利用するので，仮想アドレスを物理アドレスへ変換する必要がある．このため，仮想アドレスを物理アドレスへ変換する機構として，仮想アドレスと物理アドレスとの対応づけを保持するページテーブルを実装した．

GID/仮想アドレス変換

RHiNET では，通信対象のメモリ領域にグローバルアドレス ID (GID) と呼ばれる固有の値を割り当てることにより，すべてのノードに共通する大域的なアドレス空間を作り出している．リモートメモリのアドレスはすべて，GID とそれに対応付けられている仮想アドレスからのバイトオフセットの組み合わせで指定される．このため，GID を仮想アドレスへ変換する機構として，GID と仮想アドレスとの対応づけを保持する GID テーブルを実装した．

2.2 通信プリミティブ

RHiNET/NI-NPC の通信は，ユーザプロセスによって起動される．RHiNET/NI-NPC は，通信プリミティブと呼ばれる基本的な通信命令を備えており，ユーザプロセスは RHiNET/NI-NPC に対し通信プリミティブを発行することにより通信を起動する．

RHiNET/NI-NPC には，以下の 4 種類の通信プリミティブが実装されている．

- **Push**
  Push プリミティブは，ローカルメモリのデータを GID とオフセットで指定されるリモートメモリ領域へ書き込む通信プリミティブである．

- **Pull**
  Pull プリミティブは，GID とオフセットで指定されるリモートメモリ領域のデータをローカルメモリに書き込む通信プリミティブである．
- **Push Page**
  Push Page プリミティブは，ページ単位で Push を行う通信プリミティブである．Push Page プリミティブでは，送信と同時に Network Page Cache にページをキャッシングする，あるいは Network Page Cache から直接ページを送信することが可能である．
- **Send**
  Send プリミティブはメッセージパッシング通信を行う通信プリミティブであり，ローカルメモリのデータをリモートノードの受信バッファへ送信する．受信バッファは予めユーザプロセス空間上に確保しておき，RHiNET/NI-NPC にバッファのアドレスとサイズを登録しておく．

2.3 Network Page Cache

RHiNET/NI-NPC は，Push Page プリミティブによってネットワークへ送信されるページをキャッシングする Network Page Cache と呼ばれるキャッシュメモリを備えている．Network Page Cache は複数のキャッシュラインから構成されており，キャッシュラインのサイズは 4K バイトである．キャッシュラインの総数は，RHiNET/NI-NPC に搭載される SDRAM の容量によって異なってくるが，現在は 16M バイトの SDRAM が搭載されているのでキャッシュラインの総数は 4096 個となっている．

Network Page Cache はユーザプロセスによって制御される．ユーザプロセスは，Push Page プリミティブをオプション付きで発行することにより Network Page Cache を制御する．

- **Page Caching オプション**
  指定されたキャッシュラインにページが送信と同時にキャッシングされる
- **Send from Cache オプション**
  指定されたキャッシュラインからページが読み出され直接ネットワークへ送信される

3. RHiNET/NI-NPC への *Fagus* の移植

本研究ではソフトウェア DSM システムとして，和歌山大学にて研究されている *Fagus*[2] を利用した．*Fagus* は，cc-COMA[3](compiler controlled-Cache Only Memory Archtecture) と呼ばれるモデルに基づくコンパイラが生成する並列プログラムの実行時環境として実装されているが，*Fagus* 自身はコンパイラに依存しないソフトウェア DSM システムであるので，RHiNET/NI-NPC の環境にて動作させることは可能である．

Fagus は，UDP/IP を用いた Ethernet 環境において動作するシステムであるので，RHiNET/NI-NPC 環境にて動作させるにあたっては必要な変更を行った．*Fagus* に施した主な変更点は以下の 2 点である．

(1) *libComm* (*Fagus* の通信ライブラリ) の送受信処理で使用されている通信インタフェースの変更
(2) RHiNET/NI-NPC の Push Page プリミティブおよび Network Page Cache を利用した *libFagus* (*Fagus* のキャッシュ制御のユーザインタフェースを提供するライブラリ) のキャッシュ更新処理の高速化

以下では，これらの変更点について述べる．

3.1 *libComm* への変更点

libComm は，UDP/IP による通信を行うためソケットの通信インタフェースを利用している．このため，*libComm* が利用しているソケットの通信インタフェースを，RHiNET/NI-NPC の通信ライブラリである *libRHiNET* の通信インタフェースに置き換えた (表 1)．これら *libRHiNET* の通信インタフェースは，RHiNET/NI-NPC の Send プリミティブを使用して実装されている．

表 1 ソケットの通信インタフェースに換わる *libRHiNET* の通信インタフェース

ソケット	*libRHiNET*
sendto()	rhinet_sendto()
sendmsg()	rhinet_sendmsg()
recvfrom()	rhinet_recvfrom()

3.2 *libFagus* への変更点

libFagus は，キャッシュの更新処理においてページの送信を行っている．通常キャッシュの更新処理にともなうページの送信では，同期変数に関連づけられている共有メモリ領域中のすべてのページが送信される．このページ送信処理を，RHiNET/NI-NPC の Push Page プリミティブによってページを送信するように変更した．

RHiNET/NI-NPC と *Fagus* によって構築したソフトウェア DSM システムの全体構成を図 1 に示す．

4. 評価と検討

RHiNET/NI-NPC を搭載した 4 台の PC から構成される小規模なクラスタシステムを構築し，通信プリミティブの実効的な通信性能，キャッシュ更新やバリア同期など *Fagus* の基本的な性能，および *Fagus* 上で動作するアプリケーションの実行性能を評価した．

4.1 評 価 環 境

評価に用いた PC クラスタの主な仕様は以下のとおりである．

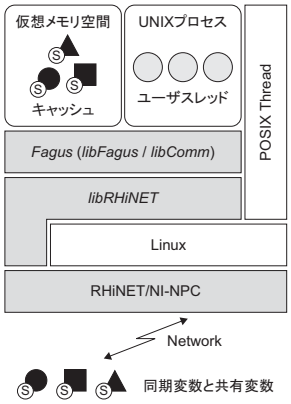

図 1 RHiNET/NI-NPC と *Fagus* によるソフトウェア DSM システムの全体構成

- PC
  - CPU　Intel PentiumIII 800EBMHz
  - Chipset　Intel 815EG
  - Memory　512MB
  - PCI　32bit/33MHz
  - OS　Linux 2.2.22
- RHiNET
  - NIC　RHiNET-2/NI0
  - Switch　RHiNET-2/SW
- Ethernet
  - NIC　Intel PRO/100S Desktop Adaptor
  - Switch　PLANEX COMMUNICATIONS FX-16NH

4.2 通信プリミティブ

通信プリミティブの評価として，通信の遅延時間とバンド幅を測定した．Push，Push Page，および Send プリミティブの評価では，2 ノード間でのデータの往復時間を測定し通信の遅延時間とバンド幅を求めた．さらに，Push Page プリミティブにおいては

(1) 通常の送信 (Normal)
(2) Network Page Cache に書き込みながら送信 (Page Caching)
(3) Network Page Cache から送信 (Send from Cache)

の 3 通りの送信方法について測定した．Pull プリミティブの評価では，プリミティブを発行してから要求したデータが到着するまでの時間を通信の遅延時間とし，測定された遅延時間からバンド幅を求めた．測定結果から得られた，各通信プリミティブの遅延時間とバンド幅を図 2 に示す．

Pull プリミティブでは，リモートノードへデータの転送を要求する処理が必要となるため，他の通信プリミティブにくらべ通信の遅延時間は若干増加する．

Send プリミティブの遅延時間は，Push プリミティブと比較して若干小さくなっている．Send プリミティブでは GID を仮想アドレスへ変換する処理が必要なく，また，プリミティブ発行に必要となるパラメータ

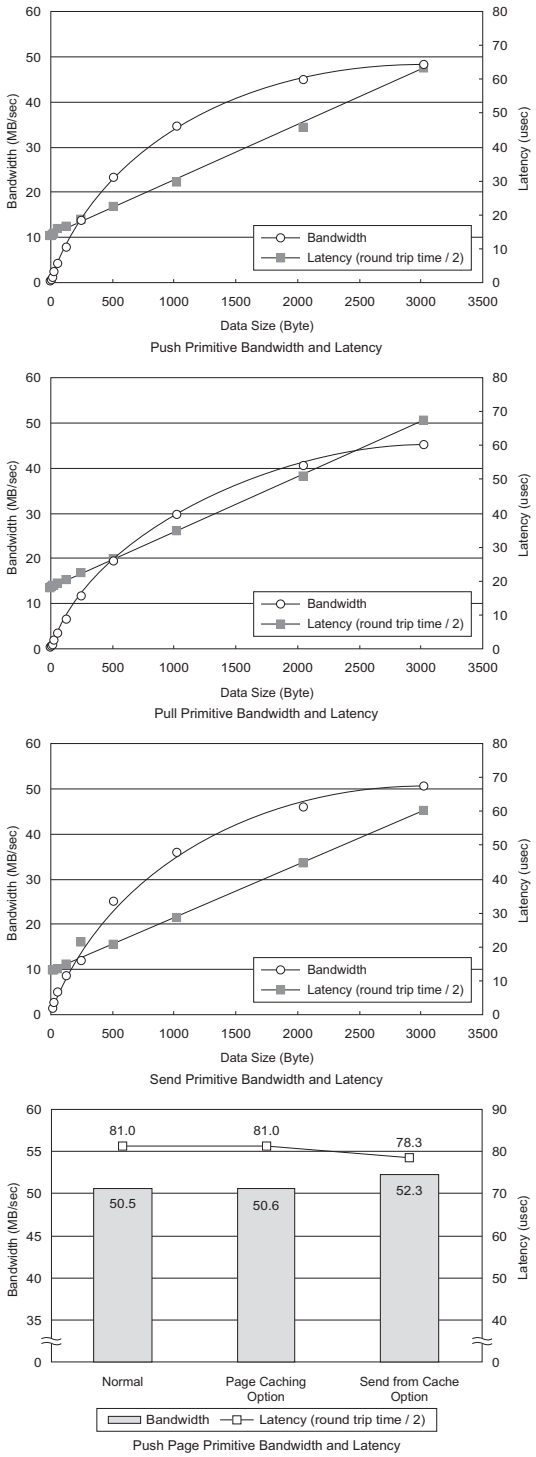

図 2 通信プリミティブのバンド幅と遅延時間

が他の通信プリミティブより少ないためユーザプロセスによる RHiNET/NI-NPC へのアクセス時間も短くなる．これらの要因のため，Send プリミティブの遅延時間が小さくなったと考えられる．

Push Page プリミティブでは，Network Page Cache から送信 (Send from Cache) することにより，通常の送信方法 (Normal) と比較して約 4%の性能向上が得られている．これは，Network Page Cache から直接ネットワークへ送信する場合，DMA 転送のセットアップ処理や DMA 転送の開始待ちが不要となり，ページ送信に要する時間が短くなるためである．また，RHiNET/NI-NPC では送信と Network Page Cache への書き込みを同時に行うので，通信の遅延時間を増加させずにページをキャッシングすることが可能である (図 2 の Normal と Page Caching を参照)．

4.3 *Fagus* の基本性能

RHiNET/NI-NPC による *Fagus* の基本性能の評価として，共有変数へのアクセスにともなうキャッシュ制御性能，およびバリア同期のスループットを評価した．本節以降は，RHiNET/NI-NPC による *Fagus* を用いた評価環境を *Fagus*/RHiNET と表記し，100BASE の Ethernet による *Fagus* を用いた評価環境を *Fagus*/Ethernet と表記する．

キャッシュ制御性能

キャッシュ制御性能の評価では，クラスタを構成するノードを共有変数への書き込みを実行する 1 台のサーバと，読み出しを実行する複数台のクライアントに分け，書き込みで実行されるキャッシュの無効化処理，および読み出しで実行されるキャッシュの更新処理に要する時間をそれぞれ測定した．共有変数のサイズは 4 バイトとし，サーバを含めノード数を 2 台から 4 台まで変化させて測定を行った．測定結果から得られた，キャッシュの無効化時間を図 3 に，キャッシュの更新時間を図 4 にそれぞれ示す．図中には参考値として，100BASE の Ethernet におけるキャッシュの無効化時間，およびキャッシュの更新時間を示しておく．

図 4 において Max と Min は，無効化時間が最も長かったノードと最も短かったノードをそれぞれ表す．サーバノードはキャッシュ更新要求を到着順に処理するので，要求の到着が早かったノードと要求の到着が遅れたノードとの間で無効化時間に差が生じる．

バリア同期のスループット

バリア同期のスループットの評価では，バリア同期によってキャッシュの転送を発生させバリア同期に要する時間を測定し，バリア同期によるキャッシュ転送のスループット (バリア同期のスループット) を求めた．ノード数は 2 台とし，バリア同期によって転送されるキャッシュのサイズを 1K バイトから 512K バイトまで 1K バイトづつ変化させ測定を行った．測定結果から得られた，バリア同期のスループットを図 5 に示す．

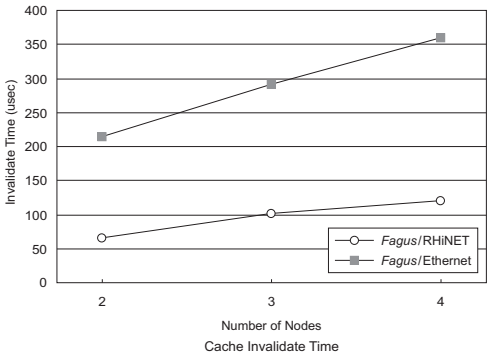

図 3 キャッシュ無効化時間

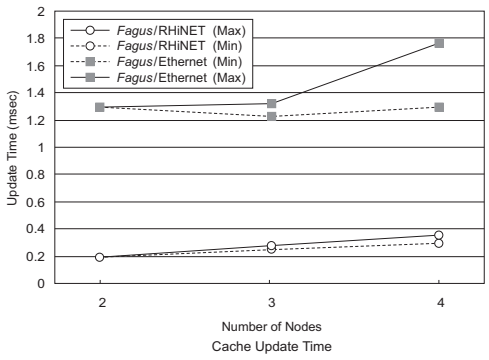

図 4 キャッシュ更新時間

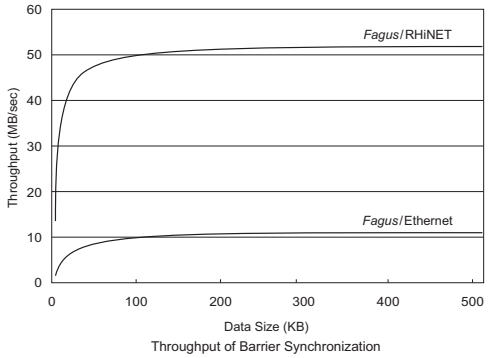

図 5 バリア同期のスループット

バリア同期のスループットは，*Fagus*/RHiNET では約 51MB/sec，*Fagus*/Ethernet では約 11MB/sec となっており，*Fagus* は RHiNET/NI-NPC および Ethernet のどちらの環境においても，ネットワークインタフェースのバンド幅をほぼ使いきっており良好な結果が得られている．

4.4 アプリケーションの実行性能

Fagus 上で動作するアプリケーションの実行性能の評価として，LU 分解 (軸選択なしの Do-little 法のアルゴリズムを並列化) プログラムの実行時間を測定した．ノード数を 1 台から 4 台まで変化させ，問題サイズが 2048 × 2048 の場合について実行時間の測定を行った．ノード数が 1 台のときは，*Fagus* を使用せず

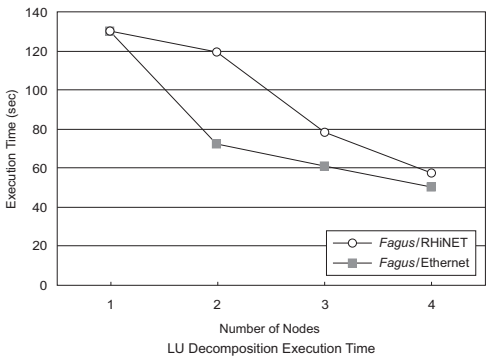

図 6 LU 分解プログラム実行時間

に実行時間の測定を行った．LU 分解プログラムの実行時間を図 6 に示す．

Fagus/RHiNET でのプログラム実行時間は *Fagus*/Ethernet と比較して，全体的にプログラム実行時間が増加している．プログラム実行時間が増加した原因を調査したところ，バリア同期に関連するメッセージの送受信処理がプログラム実行時間の多くを占めていることが判明した．ただし，直接的な原因はメッセージの受信やプリミティブの完了を検出するためのポーリング処理にあると推測される．ポーリング処理がスレッドのスケジューリングなどに影響を及ぼし，その結果バリア同期に関連するメッセージの送受信処理が多くの時間を消費していると考えられるからである．

RHiNET-2/NI0 では PCI コントローラの PCII1 に問題点が残されており，マスタとなってデータ転送を実行している途中でホスト CPU から読み出し要求が行われるとバスがロックしてしまう．割り込みを用いてメッセージの受信やプリミティブの完了を検出する実装を試みたが，デバイスドライバの割り込みハンドラによる割り込みステータスの読み出し処理によってバスのロックが発生するため，割り込みを用いて実装することはできなかった．このため，メッセージの受信やプリミティブの完了はポーリングによって検出するようにした．

ポーリング処理では，条件が成立していない場合は他のスレッドへプロセッサを明け渡すために sched_yield() を呼び出している．このため，sched_yield() が頻繁に呼び出されスレッドのスケジューリングに多くの時間が消費されることになり，プログラムの実行時間が増加したのではないかと考えられる．

5. まとめと今後の課題

FPGA 実装型光通信ネットワークインタフェース RHiNET-2/NI0 を利用し，ユーザレベルゼロコピー通信およびページキャッシング機構を備えたネットワークインタフェースである RHiNET/NI-NPC を実装した．また，ソフトウェア分散共有メモリシステム *Fagus* を RHiNET/NI-NPC の環境に移植し RHiNET/NI-NPC の性能を評価した．

その結果，ゼロコピー通信では，リモートメモリライトで約 48MB/sec，リモートメモリリードで約 45MB/sec のバンド幅が得られた．LU 分解プログラムの実行性能では，受信メッセージを待つ処理などで用いられているポーリング処理が主な原因となり，100BASE の Ethernet を用いた場合と比較してプログラム実行時間の増加がみられた．

今後は，PCII1 のバスがロックする問題点を避ける FPGA のコンフィギュレーションを作成し，メッセージの受信やプリミティブの完了を検出する処理をポーリングではなく割り込みを用いた実装にして，再度 *Fagus* 上で動作する並列アプリケーションの実行性能を評価したいと考えている．また，Network Page Cache の有効性を評価するために，並列アプリケーションやキャッシュ容量による Network Page Cache のキャッシュヒット率の変化などの評価を行っていきたいと考えている．

謝辞 RHiNET-2/NI0 を提供して頂き，また実装するにあたり多くの助言を頂いた新情報処理開発機構並列分散アーキテクチャつくば研究室の工藤知宏氏 (現在，産業技術総合研究所)，(株) シナジェテックの清水敏行氏に深く感謝致します．*Fagus* の研究と実装を担当されていた横手聡氏には，*Fagus* に関連する多くの助言を頂きました．

参 考 文 献

1) 大塚智宏，横山知典，土屋潤一郎，宮脇達朗，清水敏行，山本淳二，西宏章，工藤知宏，天野英晴：RHiNET ネットワークインタフェースプロトタイプの性能評価，情報処理学会研究報告，2001-ARC-143，pp.13-18 (2001)
2) 横手聡，齋藤彰一，上原哲太郎，國枝義敏：コンパイラによる制御が可能な DSM システム *Fagus* の実現，情報処理学会研究報告，2000-OS-85，pp.45-54 (2000)
3) Shoichi Saito, Tetsutaro Uehara, Kazuki Joe and Yoshitoshi Kunieda：cc-COMA: the compiler-controlled COMA as a framework for parallel computing, Proc. of IWIA98, IEEE Computer Society Press, pp.114-119 (1999)