ハイパフォーマンス
コンピューティング 90－6
（２００２．５．２７）

HYPHENクラスタにおける基本プリミティブの設計と実装

大西淑雅[†] 木原大悟[††,☆]
立川純[††] 有田五次郎[††]

我々は，HYPHEN クラスタの研究開発を行っている．HYPHEN クラスタは一般的なPC に，HyphenLink と呼ばれる専用ボードを追加することで，PC クラスタを構成する．HyphenLink には，分散共有メモリ (DSM) と並列処理を支援する機能を持っている．メモリとネットワークインターフェースを持つことで，HyphenLink 同士を接続し利用者にDSM を提供する．また，並列処理を支援するための FIFO キューを持つことで，PB タスクモデルと呼ばれる並列プログラムを動作させることができる．我々は，HyphenLinkのプロトタイプ実装として，基本プリミティブを設計した．本稿では，HYPHEN クラスタが必要とする基本プリミティブの設計と実装について報告する．

Design and Implementation of basic primitives on the HYPHEN Cluster

Yoshimasa Ohnishi,[†] Daigo Kihara,[††,☆] Jun Tachikawa[††]
and Itsujiro Arita[††]

We have been researching and developing a PC cluster computer called HYPHEN cluster. HYPHEN cluster composes some general PCs and extended boards which are called HyphenLinks. Our implementation of HyphenLink has 2 features: one is for Distributed Shared Memory (DSM) and the other is for Parallel Processing. For DSM, HyphenLink has a memory module and a network interface module. Each HyphenLink, which is added to PCs, can be connected to others, so it can provide memorys of HyperLinks on every PC as DSM to users. For parallel processing, HyphenLink has a FIFO queue. We have constructed a prototype of HyphenLink which consists of some primitives each of which can totally maintain functions as a PC cluster. In this report, we describe a design and implementation of primitives on the HyphenLink.

1. はじめに

計算機の低価格化に伴い，一般的なパーソナルコンピュータ (PC) を用いたPC クラスタが，並列処理のプラットフォームとして注目を集めている．PC クラスタにおいても，並列処理における一般的なプログラミングモデルとして，メッセージパッシングモデルと共有メモリモデルがあり，MPIやPVM に代表されるメッセージパッシングモデルに基づいたクラスタが主流である．一方，逐次プログラムからの並列化が容易に行えるなどの理由から，共有メモリモデルをサポートするクラスタに関する研究も盛んに行われている．一般にクラスタで実現される共有メモリは，各ノードに分散配置した分散共有メモリ (Distributed Shared Memory; DSM) の形態をとる．

従来，汎用のネットワークを用いて構成するDSM システムは，主としてソフトウェアによって実現され，その管理に伴う処理を計算機のプロセッサが行うため，共有メモリに対するメモリ操作によって発生するオーバヘッドが大きい．このオーバヘッドが，大規模なクラスタにおけるスケーラビリティ低下の大きな要因の１つである．

既存の高性能なPC クラスタでは，Myrinet やGigabit Ethernet といった高速なネットワークに対して，PM[1),2)] と呼ばれるクラスタ用プロトコルを採用している．これは，信頼性のある低遅延な通信を実現し，高性能なクラスタシステムの構築を可能にしている．また，IEEE1394を使ったPC クラスタ[3)] も提案されており，低コストで容易にクラスタを構成することができる．これらの研究は，まず高速なネットワーク基盤

† 九州工業大学 情報科学センター
Information Science Center, Kyushu Institute of Technology
†† 九州工業大学 情報工学部
Department of Artificial Intelligence, Kyushu Institute of Technology
☆ 現在，興安計装 (株)

を作り，その上でクラスタを構築するといったアプローチである．

我々は，共有メモリと高速な同期操作をハードウェアで提供することを念頭におき，スケーラビリティの高いPCクラスタを構築するアプローチをとる．具体的には，拡張I/Oバスとして一般的なPCIバスに接続可能なHyphenLink(ハードウェアDSM)を開発し，HYPHENクラスタと呼ばれるPCクラスタの実現を目指している．HyphenLinkは大容量のメモリとクラスタ用ネットワークインターフェースを有しており，計算機のプロセッサが担当していた処理の一部を代行することで，オーバヘッドの削減を狙っている．

本論文では，まず，現在研究開発を進めているHYPHENクラスタの概要について述べる．次に，HyphenLinkの機能設計について，DSM管理とPBタスク管理に関係する基本プリミティブを中心に報告する．最後に，プリミティブの実装について述べ，まとめとする．

2. HYPHENクラスタの概要

HYPHENクラスタは，HYPHENアーキテクチャ[4)~9)]をベースに設計した分散共有メモリ型クラスタ計算機である．一般的なPC(CPUとしてはIntel x86系)とローカルOSとしてLinuxを使用し，コモディティなI/Oバスに接続可能な専用のボード(HyphenLink)を追加することで**図1**に示すノードを構成する．

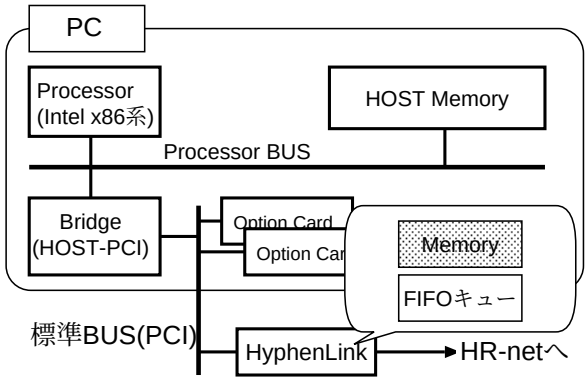

図1 HYPHENクラスタのノード構成例

各ノードは，HyphenLinkを経由してクラスタ専用ネットワーク(HR-net)[10)]で接続する．HyphenLinkは，DSMを構成するためのメモリを持ち，他のHyphenLinkと協調してハードウェアDSMを実現する．なお，HR-netは，プログラムの局所性を利用して近接した分散共有メモリへのアクセスを効率化し，スケーラビリティを確保しようとする階層構造のルーチングネットワークである．

HYPHENクラスタにおける並列プログラムは，依存関係のない処理単位であるPBタスクの集合として捉え，複数のPBタスクのプロセッサ割り当てを明示的に記述することで行われる．このプロセッサ割り当ては，システムの提供するParallel Branch(PB), EXchange Task(EXT)の2つのプリミティブによって記述される．

PBプリミティブは，PBタスクの実行を要求するノード番号と実行するPBタスク自身を引数として，PBタスクの起動要求を発行する操作である．この起動要求は各ノードが持つFIFOキューに一時格納され，実際の処理開始はEXTプリミティブが呼ばれるまで遅延される．EXTプリミティブは，自ノードのFIFOキューにPBタスクの起動要求があれば，それを取り出し実行に移し，なければFIFOキューへの投入を待つ．但し，この並列プログラムをうまく実行するためには，PBタスクの命令流の最後には，EXTプリミティブを必ず記述し，次のPBタスクへの切り替え要求を発行しなければならない．このような実行モデルをPBタスクモデルと呼び，HYPHENクラスタのソフトウェアモデルである．

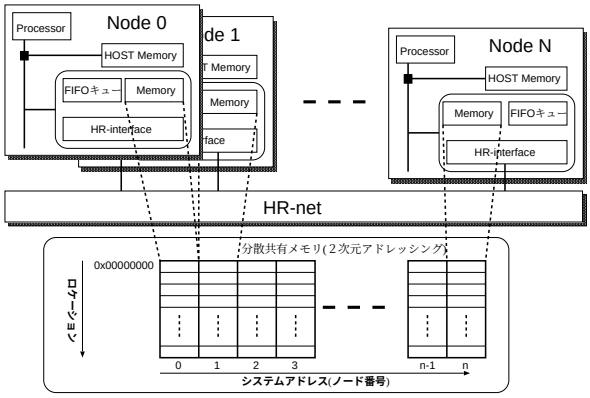

図2 HYPHENクラスタのシステムモデル

2.1 システムモデル

図2にHYPHENクラスタのシステムモデルを示す．システムが使用するメモリは，個々のHyphenLink上のメモリ(Memory)を接続してDSMを構成し，すべてのノードからアクセス可能である．よって，プログラムが使用するメモリオブジェクト(データ)は，このDSM上にすべて配置される．しかし，プログラムコード自身は，ローカルOSであるLinuxの管理下で，ノードの基となる計算機の物理メモリ(HOST Memory)に配置される．DSMのアドレス構成は，ノードを特定するシステムアドレスとHyphenLink上のメモリのロケーションを使って2

次元空間で表すことができる．システムのメモリ空間は，ノード数に比例して柔軟に拡張できる．なお，HOST Memoryは，Linux自身が用いると共にDSMのキャッシュとして取り扱う．

2.2 PBタスクモデル

HYPHENクラスタは，FIFOキューを用いることで自然に同期が取れるPBタスクモデルを採用する．FIFOキューに対する操作は，PBプリミティブがキューへの書き込み，EXTプリミティブがキューからの読み出しとなっている．

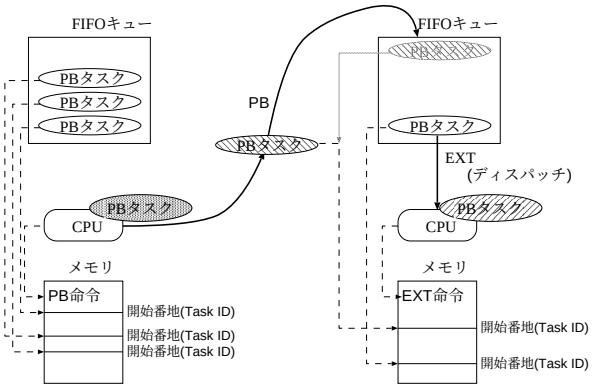

図3 PB，EXTプリミティブ

2.2.1 PBプリミティブとEXTプリミティブ

図3にPBプリミティブとEXTプリミティブの内容を示す．PBプリミティブは，PBタスクが静的に割り付けられた各ノードのFIFOキューに，開始すべきPBタスクの開始番地(Task ID)を登録する操作である．PBプリミティブは非同期に実行され，他ノードのFIFOキューを操作しつつ，自ノードの処理も並列に処理をすることができる．

EXTプリミティブは，PBタスクの最後に必ず実行される操作であり，PBプリミティブによって登録されたPBタスクを，自ノードのFIFOキュー先頭から取り出し制御を移す．もしこの時，PBタスクが登録されていなければ，PBタスクはプロセッサに割り当てられず，プロセッサはアイドル状態になる．そしてPBタスクが登録されるのを待つ．

2.2.2 PBタスクの動作

データを分散配置して並列に処理を行い，最後に集計するといった簡単な並列プログラムを例にとると，並列プログラムは**図4**に示すような形となる．PBタスクAが各ノードに配置されたPBタスクB,C,D(並列処理を行う部分)をPBプリミティブによって登録する．各ノードではEXTプリミティブによって，PBタスクB,C,Dに制御が移りデータを処理する．PBタスクBはデータ処理が終了した時点で，PBプリミティブによってノード1に配置されたPBタスクS1を登録する．S1は同期のためのPBタスクであるので，PBプリミティブによってノード2に配置されたPBタスクS2を登録するのみである．

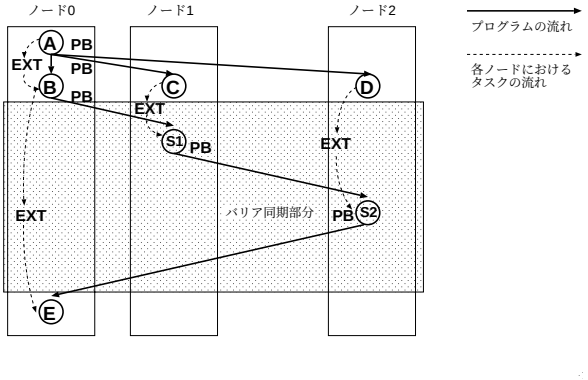

図4 PBタスクモデルにおけるバリア同期

もしPBタスクCが終了していなければ，PBタスクCの最後に記述しているEXTプリミティブが実行されないため，PBタスクS1に制御は移動しない．PBタスクCがEXTプリミティブを実行した時点で，PBタスクS1に実行が移る．同様に，ノード2においてもPBタスクDが終了すれば，PBタスクS2に制御が移動する．

2.3 システムソフトウェア

HYPHENクラスタのシステムソフトウェアは，HyphenLinkが提供するDSM管理やPBタスク管理といった機能を支援するものである．

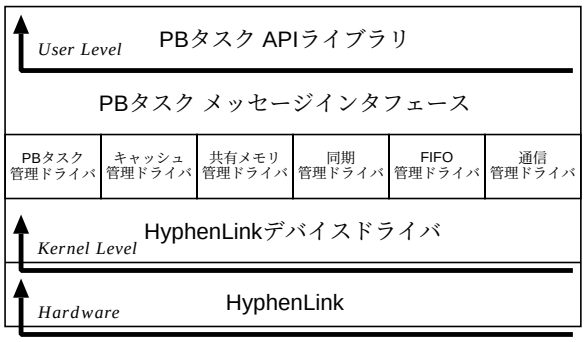

図5 設計中のシステムソフトウェア

図5にシステムソフトウェアのイメージを示す．PBタスクAPIライブラリはシステムに対するユーザレベルAPIであり，PBタスクのコンパイル時に静的リンクされる．これらのシステムソフトウェアの設計は，基本プリミティブの設計と並行して行なっている．

3. HyphenLinkの機能設計

3.1 DSM管理

HyphenLink上のメモリ(図1中のMemory)には，DSMのための領域とDSM管理用の領域などが展開される．HyphenLinkはHYPHENクラスタにおけるメモリ管理とメモリ操作の機能を持ち，DSMからキャッシュメモリ(図1中のHOST Memory)へデータをキャッシュするCACHEプリミティブと，キャッシュメモリからDSMへデータを書き戻すWrite Back(WB)プリミティブがある．プロセッサからみたDSM空間は，基本的にShared Virtual Memory(SVM)方式に準拠するため，x86系プロセッサでのページサイズ(4KB)を管理単位とする．なお，これらの操作は，SVMでのページミス時のページフォルトハンドラによって使用される機能である．

3.1.1 CACHEプリミティブ

図6にCACHEプリミティブの動作を示す．ここで，HyphenLinkは他ノードのページにアクセス要求が来たとしても，要求先のプロセッサへの介入が発生しないことがわかる．これは，外部からのページ要求への対応をHyphenLinkのみによって行ない，プロセッサへの介入を極力回避する設計となっている．

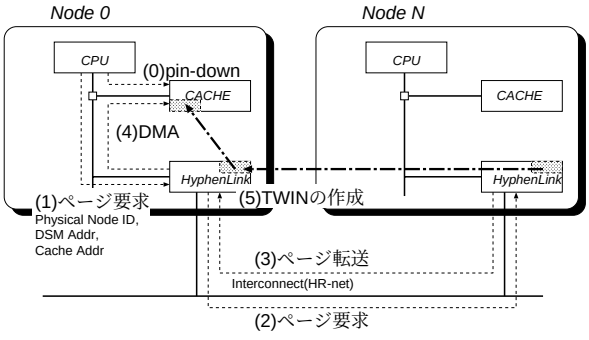

図6 CACHEプリミティブの動作

(1)プロセッサから，キャッシュすべきページの存在するHyphenLinkの物理ノード番号(Physical Node ID)と，そのHyphenLinkの持つDSMの物理ページ先頭アドレス(DSM Addr)及びキャッシュメモリの物理ページ先頭アドレス(Cache Addr)を受け取る．

(2)物理ノード番号が自ノードの場合は，自ノードのDSMからCache Addrで示すキャッシュメモリへページサイズ分DMA転送して終了となる．他ノードの場合は，対応するHyphenLinkにキャッシュすべきDSMのページ先頭アドレスを渡すため，ネットワークを経由してページ転送要求を送信する．

(3)ページ転送要求を受信したHyphenLinkは，指定されたDSMのページを，ネットワークを経由して要求元HyphenLinkへ転送する．

(4)要求元HyphenLinkはそのページデータを受け取ると，キャッシュメモリへページサイズ分DMA転送する．

(5)キャッシュしたページサイズ分のデータを，WBプリミティブで使用するTWIN領域にコピーしておき，必要なテーブルを更新する．

3.1.2 WBプリミティブ

図7にWBプリミティブの動作を示す．ここでも，HyphenLinkは他ノードのページにアクセス要求が来たとしても，要求先のプロセッサへの介入が発生しないことがわかる．

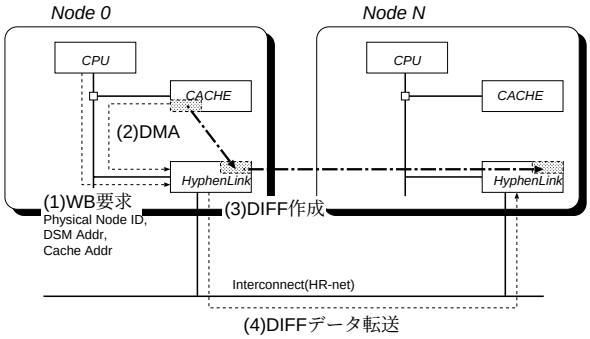

図7 WBプリミティブの動作

(1)プロセッサから，書き戻すべきページの先頭物理アドレス(Cache Addr)，書き戻し先HyphenLinkの物理ノード番号(Physical Node ID)とDSMの物理ページ先頭アドレス(DSM Addr)をHyphenLinkは受け取る．

(2)Cache Addrで示すキャッシュメモリからページサイズ分DMA転送する．

(3)CACHEプリミティブで作成したTWIN領域と，DMA転送されたデータとの差分を作成する．なお，書き戻し先の物理ノード番号が自ノードであれば，自ノードのDSMに対して差分を書き込む．

(4)HyphenLinkを接続しているネットワークを経由して，対応するHyphenLinkに書き戻し先DSMの物理ページ先頭アドレスと差分データと共に転送する．HyphenLinkはそれらのデータを受け取ると，指定されたDSMの領域へ差分を書き込む．

(5)最後に，自ノードのCACHEプリミティブが作成したTWIN領域およびキャッシュ領域を開放する．

3.2 PB タスク管理

HyphenLink は PB タスクモデルを実行するための FIFO キューを持ち，PB プリミティブと EXT プリミティブで FIFO キューを操作する．

3.2.1 PB プリミティブ

図 8 に PB プリミティブの動作を示す．なお，HyphenLink は論理ノードと物理ノードとの対応表を持っている．

(1) プロセッサから，PB すべき論理ノード番号 (Node ID)，対象 FIFO キューに投入すべき PB タスクの論理タスク番号 (Task ID) を渡す．

(2) 論理ノード番号から物理ノード番号に変換し，自ノードならば，FIFO キューへ PB タスクの Task ID を登録する．

(3) 他ノードならば，HyphenLink を接続しているネットワークを経由して，対象ノードの HyphenLink の持つ FIFO キューへ PB タスクの Task ID を登録する．

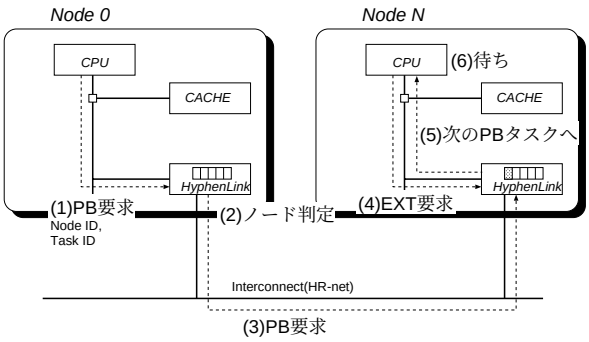

図 8 PB,EXT プリミティブの動作

3.2.2 EXT プリミティブ

図 8 に EXT プリミティブの動作を示す．なお，以下の番号は図 8 と対応している．

(4) プロセッサから，EXT 要求が来る．

(5) HyphenLink の持つ FIFO キューをチェックし，PB タスクの Task ID が登録されていれば，キューから取り出しプロセッサに実行させる．

(6) FIFO キューが空の場合には，プロセッサは実行すべき PB タスクがないので，FIFO キューの到着を待つ．

3.3 FIFO キュー管理

FIFO キューの深さや保持するデータ長は，HYPHEN クラスタの規模や FIFO キューのハードウェア規模とも密接に関わり持っており，現在検討中である．また，FIFO キューが溢れた場合の例外処理についても考慮する必要があるため，あわせて検討中である．

4. プリミティブの実装

HyphenLink のハードウェアは，利用する計算機の内部構成に依存しない方式で用意することが望ましい．また，HYPHEN クラスタはスケーラビリティの高い PC クラスタの研究システムであるから，プリミティブ実装の変更や追加が可能でなければならない．そこで，HyphenLink のプロトタイプ実装として，再構成可能なデバイスである FPGA を搭載した汎用 PCI カードである SHOKE2000[11] を利用する．

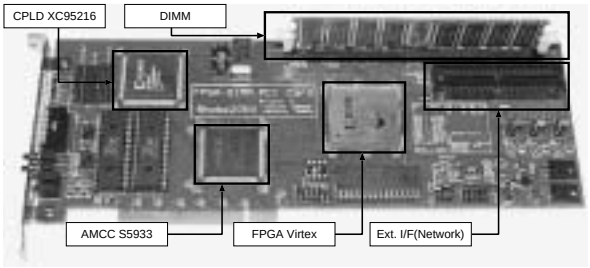

図 9 汎用 PCI カード SHOKE2000

SHOKE2000 を HyphenLink として機能させるために，FPGA 上に DIMM コントローラ，PCI コントローラへのインターフェースなどを組み込む．また，**図 9** 中の External Interface(Ext. I/F) を HR-net への接続として利用するためのネットワークインターフェース機能も組み込む必要がある．以下，機能ごとの実装方法について述べる．

4.1 基 本 実 装

HyphenLink 間の通信は HR-net を経由して行われるため，HR-net を構成するネットワークスイッチを含めた検討が必要である．HR-net を構成するスイッチとしては試作スイッチが存在するが，HyphenLink との接続性や HR-net 自身に関する検討がまだ十分でないため，今回は未実装である．

HyphenLink の基本プリミティブ群は，一般的によく用いられる，ゼロコピー通信を使って実装する．具体的には，HyphenLink の DMA 転送を用いて，プロセッサによるコピーオーバヘッドを削減する．また，HyphenLink の基本プリミティブ群の呼出には，HyphenLink のコマンドレジスタ群 (Primitive Controll Registers; PCR) をユーザ仮想空間にマップして[12]，直接起動できるようにする．これにより，システムコールによるオーバヘッドを削減することができる．なお，ユーザ用の PCR は 3 つであり，カーネル用は 1 つである．各 PCR には，**表 1** に

表 1 PCR の主な基本エントリ

PCR エントリ名	機能
CTRLFLAG	プリミティブを起動指定
NODEID	PB 用:論理ノード番号 (NodeID) を指定
TASKID	PB 用:起動するタスクのアドレスを指定
VIRTUALA	WB 用:仮想アドレスを指定
SELPAGE0	WB 用:ユーザによる WB のページ指定
SELPAGE1	WB 用:ユーザによる WB のページ指定
STATUSFLAG	プリミティブの実行状況を確認

示すようなエントリを持っている.

4.2 DSM 管理

DSM 管理の実装では，HyphenLink 上のメモリの物理容量の半分を DSM 自身として使用し，残りを TWIN 領域などに割り当てた．また，**表 2** に示すような DSM を管理するための DSM テーブルは，HyphenLink 上のメモリに OS の責任として作成する必要がある．ここで，Node はキャッシュすべき DSM の存在するノード番号であり，SM Addr は，Node で指定されるノード上にある DSM のページ番号であり，Cache Addr はキャッシュメモリのページ番号，TWIN Addr はノード上にある TWIN メモリのページ番号である.

表 2 DSM テーブルの例

Node	SM Addr	Cache Addr	TWIN Addr	Flags
0x00	0x000	0x012	0x001	0x001
0x01	0x000	0x013	0x002	0x001
:	:	:	:	:

WB プリミティブの実装では，プロセッサから通知された仮想アドレスとその時に使用された HyphenLink のコマンドレジスタ群の ID から DSM テーブルを一意に決定し，テーブルからキャッシュされているページをすべて検索して，テーブルからの検索結果を元に，差分生成と DSM への書き戻しを行う.

4.3 PB タスク管理と FIFO キュー

PB タスク管理の実装では，EXT プリミティブの実装方法にあわせて 2 種類用意した．1 つは，プロセッサから FIFO キューをポーリングしておき，PB タスクの到着を待つ方法である．もう 1 つは，HyphenLink 上の FIFO キューへの登録が発生した場合に，プロセッサに割り込みをかける方法がある．どちらの方法がすぐれているかは，実際に検証して決定する予定である.

なお，HyphenLink 上の FIFO キューは，幅は 32bit，深さ 255 で構成し，動作の確認にとどめた.

5. ま と め

HYPHEN クラスタについて紹介し，ハードウェア DSM と PB タスクモデルを実行させるために必要な HyphenLink 機能について述べ，HyphenLink 上の基本プリミティブの設計と実装について報告した．基本プリミティブについては，個々の実装はほぼ完了しているが，すべての機能のフル (同時) 実装はまだ完了していない．よって，HyphenLink を早急に完成させ，各プリミティブの性能と評価を行う予定である.

参 考 文 献

1) 手塚 宏史, 堀 敦史,Francis O'Carroll, 原田 浩, 石川 裕: 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, Vol.97-ARC-125, pp167-172 (1997).
2) 住元 真司, 堀 敦史, 手塚 宏史, 原田 浩, 高橋 俊行, 石川 裕: GigaE PM II:Gigabit Ethernet による高速通信ライブラリの設計, 情報処理学会研究報告, Vol.99-ARC-134, pp61-66 (1999).
3) 兵頭和樹, 中山泰一: IEEE1394 を用いた PC クラスタシステム -通信機構の設計と評価, 情報処理学会論文誌, Vol.41, No.SIG8(HPS2), pp39-47 (2000).
4) 有田 五次郎: FIFO キューを同期手段とする並列プログラムについて (I) -待ちなし並列プログラム-, 情報処理学会論文誌, Vol.24, No.2, pp221-229 (1983).
5) 有田 五次郎, 荒木 啓二郎: FIFO キューを同期手段とする並列プログラムについて (II) -並列プログラムの待ちなし変換-, 情報処理学会論文誌, Vol.24, No.2, pp230-237 (1983).
6) 有田 五次郎, 末吉 俊則: FIFO キューを同期手段とする並列プログラムについて (III) -実行管理機構-, 情報処理学会論文誌, Vol.24, No.6, pp838-846 (1983).
7) 末吉 俊則, 最所 圭三, 有田 五次郎: 階層構造高多重並列計算機実験システム HYPHEN C-16 について, 情報処理学会論文誌, Vol.25, No.5, pp813-822 (1984).
8) 末吉 敏則, 有田 五次郎: 階層ルーチングバスについて, 電子通信学会論文誌, Vol.J67-D, No.11, pp1309-1316 (1984).
9) 末吉 敏則, 最所 圭三, 有田 五次郎: MIMD 型並列計算機における 2 進木構造アクセス機構の性能評価, 電子通信学会論文誌, Vol.J68-D, No.11, pp1309-1316 (1985).
10) 立川 純, 木原 大悟, 福澤 毅, 田中 康一朗, 大西 淑雅,Bernady O.Apduhan, 佐藤 寿倫, 有田 五次郎, クラスタ計算機 HYPHEN におけるメモリアクセス機構:HR-net, 並列シンポジウム JSPP2001 講演論文集, ポスタセッション, pp111 - 112 (2001).
11) 田中 康一朗, 有田 五次郎: SHOKE2000: PCI-Based FPGA Card の開発とその評価, 電子情報通信学会論文誌, Vol.J84-D-1, No.6, pp540-547 (2001).
12) 山本 淳二, 渡邊 幸之介, 土屋 潤一郎, 今城 英樹, 寺川 博昭, 西 宏章, 田邊 昇, 工藤 知宏, 天野 英晴: RHINET の概要と Martini の設計/実装, 情報処理学会研究報告, Vol.2001-ARC-144, pp37-42 (2001).
13) 西 宏章, 多昌 廣治, 稲沢 悟, 西村 信治, 工藤 知宏, 天野 英晴: RHiNET スイッチ RHiNET-2,3/SW, 情報処理学会研究報告, Vol.2001-ARC-144, pp55-60 (2001).
14) 田邊 昇, 濱田 芳博, 須田 均, 山本 淳二, 今城 英樹, 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 の通信性能評価, 情報処理学会研究報告, Vol.2001-ARC-145, pp51-56 (2001).