計算機アーキテクチャ　128－10
ハイパフォーマンス
コンピューティング　70－10
（１９９８．３．５）

「*PPRAM-Link* API on Linux」の開発，ならびに，そのPCクラスタ上での性能評価

目次勝彦‡　　大澤 拓†　　村上和彰†

†: 九州大学 大学院システム情報科学研究科 情報工学専攻

‡: 九州大学 工学部 情報工学科

〒816–8580 福岡県春日市春日公園 6–1

Phone: 092–583–7621
E-mail: ppram@c.csce.kyushu-u.ac.jp
http://kasuga.csce.kyushu-u.ac.jp/~ppram/

あらまし　*PPRAM-Link* とは，システム LSI(*PPRAM* チップ) 同士を相互接続するための標準高速 LSI 間通信インタフェース規格であり，標準化団体 *PPRAM* コンソーシアムにて，物理層，論理層，および，API(アプリケーション・プログラミング・インタフェース) の各仕様の策定が行われている．本稿では，標準的なハードウェア・プラットフォーム上でのソフトウェア開発環境の提供，ならびに，機能検証を目的として，Linux 上に *PPRAM-Link* API をユーザ・プログラムとして実装している．さらに，PVM を *PPRAM-Link* API 上に実装し，ハードウェア・プラットフォームとして PC クラスタを用い，*PPRAM-Link* API の基本通信性能，および，機能の充足度に関して評価を行っている．

キーワード　*PPRAM-Link*，API，PC クラスタ，標準高速 LSI 間通信インタフェース

Development of "*PPRAM-Link* API on Linux" and Performance Evaluation on a PC Cluster

Katsuhiko METSUGI　　Taku OHSAWA　　Kazuaki MURAKAMI

Department of Computer Science and Communication Engineering
Kyushu University
6–1 Kasuga-koen, Kasuga, Fukuoka 816–8580 Japan

Phone: [+81] (0)92–583–7621
E-mail: ppram@c.csce.kyushu-u.ac.jp
http://kasuga.csce.kyushu-u.ac.jp/~ppram/

Abstract　PPRAM-Link is a standard communication interface between system LSIs(PPRAM chips), and each specification of physical layer, logical layer, and API(application programming interface) of PPRAM-Link is planned at PPRAM consortium. In this paper, in order to offer a software development environment on a common hardware platform, and verify the function of PPRAM-Link API, PPRAM-Link API is implemented on Linux as a user program. In addition, the authors implement PVM on PPRAM-Link API, and evaluate communication performance and function sufficiency of PPRAM-Link API using the PC cluster as a hardware platform.

Keywords　PPRAM-Link, API, PC cluster, standard high-speed communication interface between LSIs

1 はじめに

PPRAM(*Parallel Processing Random Access Memory*) とは,

- メモリ/ロジック混載システム LSI を基本構成要素 (これを *PPRAM* チップと呼ぶ) として,
- それらを 1 個以上, 並列／分散に,
- 標準高速 LSI 間通信インタフェースで相互接続することで,

任意サイズ, 任意機能, 任意性能のコンピュータ/電子機器システムを構築しようという新しい「アーキテクチャ上の概念」である [1].

最後の「標準高速 LSI 間通信インタフェース」を定めることで, 異なるベンダーの *PPRAM* チップ同士の相互接続性, 相互運用性, および, その上でのソフトウエア可搬性を保証することが可能となる. 1997 年度から 3 年計画で現在, *PPRAM* コンソーシアム [8] において, 当該「標準高速 LSI 間通信インタフェース」の 1 規格として『*PPRAM-Link*』[2, 4] の開発が進められている.

標準化の対象となっているのは次の 3 つであり (図 1参照), 各々に対応して分科会が設置されている. それぞれ, 1998 年 3 月にはドラフト仕様 1.0 版を発行する予定である.

- 物理階層: チップ内外に関わらず, *PPRAM* ノード (定義は後述) 間をパラレル・リンク当り 1G バイト/秒以上, あるいは, シリアル・リンク当り 1G ビット/秒以上の高スループットで相互結合する通信媒体に関する電気的/ 機械的仕様を定める [5].
- 論理階層: *PPRAM* ノード間のトランザクション・プロトコル, トランスミッション・プロトコル, エラー検出プロトコル, 初期化プロトコル, 等を定める [6].
- API (Application Programming Interface): *PPRAM-Link* インタフェース (定義は後述) を I/O デバイスと見做してデバイス・ドライバ等の低レベル・ソフトウェアを開発する際に必要となる, ハードウェア独立な API を定める [7].

筆者らは, 上記 3 番目の API について, それを Linux 上に実装し仕様の検証を行っている. 本論文では, その結果ならびに PC クラスタ (64 台構成) 上での性能評価結果について報告する.

まず, 2章で *PPRAM-Link* の概要を述べ, 3章で *PPRAM-Link* API について説明する. 4章で *PPRAM-Link* API on Linux の実装方法について述べ, 5章でその評価を行う。6章で本稿のまとめとする.

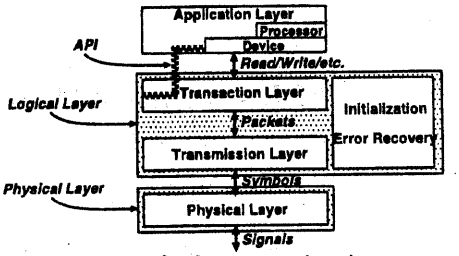

図 1: プロトコル・スタック

2 *PPRAM-Link* の概要

2.1 *PPRAM* ノード, *PPRAM* チップ, *PPRAM* ベース・システム

PPRAM ノードとは, システムの基本構成単位であり, 図 2に示すように以下のものから成る.

- 0 バイト以上のメモリ
- 0 個以上のプロセッサ/ロジック
- 1 個のネットワーク・インタフェース (以下 *PPRAM-Link* インタフェース)

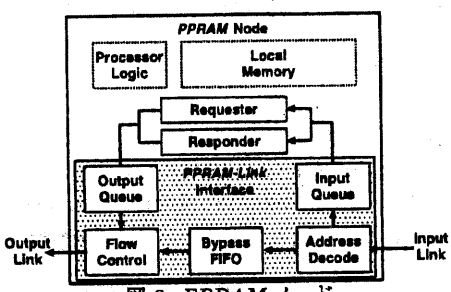

図 2: *PPRAM* ノード

1 個の *PPRAM* チップは, 1 個以上の *PPRAM* ノードを含む. 2 個以上の *PPRAM* ノードを含む場合, それらはチップ内で *PPRAM-Link* で相互結合される.

PPRAM ベース・システムは, 1 個以上の *PPRAM* チップから構成される. 2 個以上の *PPRAM* チップから成る場合, それらは *PPRAM-Link* で相互結合される. 図 3に *PPRAM* ベース・システムの構成例を示す.

PPRAM-Link とは上述の通り, チップ内外を問わず *PPRAM* ノードを相互接続し, 各ノードに対して他ノードとの間の通信を可能とならしめる通信インタフェースのことである.

PPRAM-Link インタフェースとは, *PPRAM* ノードにあって *PPRAM-Link* との間のインタフェースをつかさどるロジックである. 図 2の下半分の部分がこれに相当する.

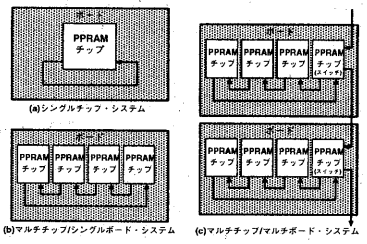

図 3: *PPRAM* ベース・システムの構成例

2.2 論理階層

2 個の *PPRAM* ノード間のデータ交換をトランザクションと呼ぶ.1 個のトランザクションは, 図 4に示すように, 一般に次の 2 個のサブアクションから成る（スプリット・トランザクション）.

1. **要求サブアクション** (request subaction): 要求側ノード (requester) から応答側ノード (responder) へ要求を送る.
2. **応答サブアクション** (response subaction): 応答側ノードから要求側ノードへ応答を返す.

さらに, 1 個のサブアクションは, 一般に次の 2 個のパケットから構成される.

1. **送出パケット** (send packet):
   - **要求送出パケット** (request-send packet): 要求側ノードから応答側ノードへ, 要求を送出するのに用いる.
   - **応答送出パケット** (response-send packet): 応答側ノードから要求側ノードへ, 応答を送出するのに用いる.
2. **受領パケット** (echo packet):
   - **要求受領パケット** (request-echo packet): 応答側ノードから要求側ノードへ, 要求を受領したことを通知するのに用いる.
   - **応答受領パケット** (response-echo packet): 要求側ノードから応答側ノードへ, 応答を受領したことを通知するのに用いる.

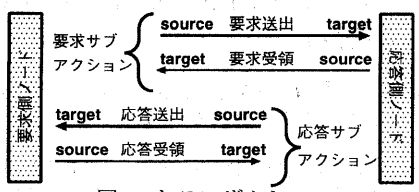

図 4: トランザクション

PPRAM-Link では, 以下のトランザクション・セットを定める.

- **読出し** (read): 指定されたアドレスから指定されたサイズ S ($1 \leq S \leq 16$, 64, 256 バイト) だけ読み出す.
- **書込み** (write): 指定されたアドレスに指定されたサイズ S ($1 \leq S \leq 16$, 64, 256 バイト) だけ書き込む.
- **選択ワード書込み** (writesw): 指定されたアドレスから始まる指定サイズ (64, 256 バイト) の範囲において, 選択されたワード (4 バイト) に対してのみ (最小 0 ワードから最大 16/64 ワードまでの任意数) 書込みを行う. ソフトウェア DSM (Distributed Shared Memory)[3] でキャッシュないしメモリのコヒーレンスを保証する際, 変更済みワードのライト・バックのために使用する.
- **移動** (move): 指定されたアドレスに指定されたサイズ S ($1 \leq S \leq 16$, 64, 256 バイト) だけ書き込む.「書込み」とは異なり, 応答サブアクションは伴わない.
- **ロック** (lock): 指定されたアドレスから指定されたサイズ S ($1 \leq S \leq 16$) だけ読み出すと同時に, それを用いて指定された不可分操作を施して同一ロケーションに書き込む.
- **イベント** (event): 受領パケットも応答サブアクションも伴わない特殊なトランザクション.
- **アクティブ・メッセージ** (actmes): 要求送出パケットを受信したノードで割込みを起こし, パケット・ヘッダ内に指定されたアドレスから割込みハンドラを起動する. 応答サブアクションは伴わない.
- **未使用** (unused): トランザクション層では何の動作も行わず, 上位層のアプリケーション層にパケット・ヘッダも含めてパケットをそのまま引き渡す.

3 *PPRAM-Link* API

3.1 概念

PPRAM-Link API とは, *PPRAM-Link* インタフェースを一種の I/O デバイスと見做してデバイス・ドライバ等の低レベル・ソフトウェアを開発する際のプログラミング・インタフェースを与えるものである. 図 5に, その位置付けを示す.

UNIX ネットワークとの対比で, *PPRAM-Link* API の概念を説明する. 標準的な UNIX ネットワークは図 6(a) に示すように, Ethernet を最下層としてその上に各種ソフトウェアが実装される. 一方, *PPRAM-Link* を用いて UNIX ネットワークを構築した場合は,

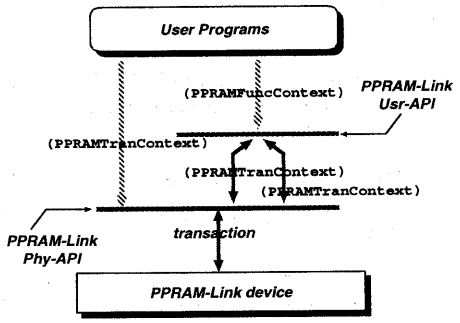

図 5: *PPRAM-Link* API の位置付け

図 6(b) のような階層構造となる．すなわち，Ethernet を代替した *PPRAM-Link* と上位ソフトウェアとの間に *PPRAM-Link* API が入り、*PPRAM-Link* デバイスに対してのプログラミング・インタフェースをソフトウェアに与えることになる．

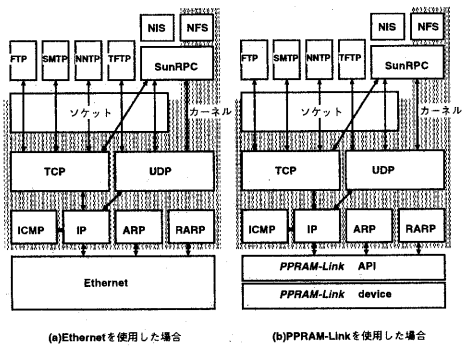

図 6: UNIX ネットワークとの対比

2.2節で述べたように，*PPRAM-Link* 論理階層自身はトランザクション・セットのみを規定しており，上位アプリケーション層との間のインタフェースについては何も規定していない．したがって，*PPRAM-Link* を具現化した *PPRAM-Link* インタフェースに関しても，その対 OS および対プロセッサ・インタフェースについて何も規定されない．すなわち，完全にインプリメンテーション依存である．

PPRAM-Link API はこのようなインプリメンテーションの相違を隠蔽かつ吸収し，上位ソフトウェアに対して統一された操作性を提供する．

3.2 ファンクション・タイプ

PPRAM-Link API では，以下の関数群を定義する（† はオプショナル）[7].

- Bit-of-data transfer functions：
  固定バイト長データの Read/Write/Lock 操作．Lock ではデータに対する不可分操作を行なう．
- Bunch-of-data transfer functions：
  指定サイズ長の一連のメモリ・ブロックに対するノンブロッキング（非封鎖型）Read/Write.
- Stride-data transfer functions：†
  指定サイズ長の一連のメモリ・ブロックに対する Gather/Scatter（1 より大きい定数ストライドを用いたノンブロッキング Read/Write）.
- Message passing functions：
  Send/Recive 型のメッセージ・パッシング操作.
- Active message function†：
  アクティブ・メッセージ.
- Raw packet-level functions：
  生のパケットを直接リンクに放出する操作.
- Miscellaneous functions：
  初期化，終了操作，等.
- DSM support functions†：
  ソフトウェア DSM（分散共有メモリ）に係わる操作.

4 *PPRAM-Link* API on Linux の実装

PPRAM ベース・システム以外の標準的なハードウェア・プラットフォーム上で *PPRAM-Link* API を用いたソフトウェア開発を可能とするために（すなわち，クロス開発環境を提供する目的で），Linux[10] 上に *PPRAM-Link* API を実装した（これを以降「*PPRAM-Link* API on Linux」と呼ぶ）.

Linux の可搬性を殺さないよう，これを一切改造することなく，すべて標準で用意されているツールと標準的なハードウェアのみを用いて，ユーザ・プログラム・レベルで *PPRAM-Link* API を実装することにした．結果として，Ethernet 上の Linux 通信インタフェースであるソケットを一種の *PPRAM-Link* デバイスと見なして，これに対する *PPRAM-Link* API を C 言語を用いて実装した．

図 7に，実装した *PPRAM-Link* API の Linux における位置付けを示す．

5 *PPRAM-Link* API on Linux の評価

実装した「*PPRAM-Link* API on Linux」に対して，以下の 2 点から評価を行う．

- 基本通信性能:4章で述べた通り「*PPRAM-Link* API on Linux」自身はクロス開発環境を提供することを目的としており，その上でアプリケーション・プログラムを本格的に動作させること

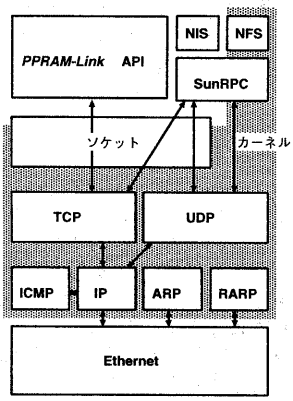

図 7: Linux への *PPRAM-Link* API の実装

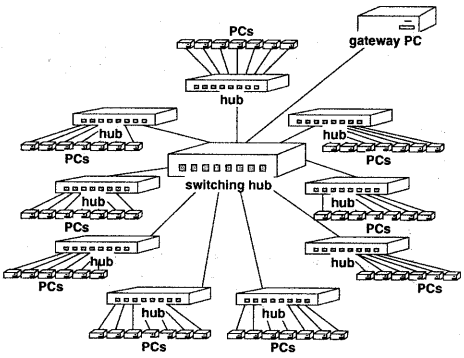

図 8: PC クラスタの構成

を目的にはしていない．したがって，実用に耐えうる通信性能を提供する必要性は必ずしもない．しかしながら，API 仕様上の問題点および実装上の問題点を洗い出すことは必須項目であり，これを目的に後述する PC クラスタを用いて基本的な通信性能を測定した．

- API としての機能の充足度：3章で述べた通り，*PPRAM-Link* API はデバイス・ドライバ等の低レベル・ソフトウェアを対象にした API である．その API としての機能が充足しているかは重要な評価項目である．本論文では，低レベル・ソフトウェアの 1 例として PVM (Parallel Virtual Machine)[11] を取り上げ評価を行った．

5.1 基本通信性能

表 1および図 8に示す PC クラスタを用いて，「*PPRAM-Link* API on Linux」の基本通信性能を測定した．

表 1: PC クラスタ緒元

PC	FMV-6200D7×64 台
CPU	Pentium Pro 200MHz
1 次キャッシュ	命令:8KB データ:8KB
2 次キャッシュ	256KB
主記憶	32MB EDO
ハード・ディスク	2.5GB
ネットワーク	100BASE-TX
スイッチング・ハブ	FUJITSU SH2500×1 台
ハブ	FUJITSU LH1100×9 台

図 9および図 10に，2 ノード間でのデータ送受信に係わる通信性能を転送データ・サイズを変更させながら測定した結果を示す．図 9はレイテンシを，図 10はバンド巾をそれぞれ表している．両図とも比較のために，「*PPRAM-Link* API on Linux」実装のベースとなっているソケットの生の通信性能も示している．

「*PPRAM-Link* API on Linux」は内部でソケットを呼び出しているため，純粋にソケットのみで通信を行なう場合に比べてかなりの遅延が生じることが図 9からわかる．

また，ソケットのみで通信を行なった際には，データサイズが大きくなるほどバンド巾が向上するのに対して，「*PPRAM-Link* API on Linux」で通信を行なった際にはデータサイズが大きくなってもそれほどバンド巾が向上しないことが図 10からわかる．これは，ソケットでは 1 回の通信で転送できるデータ・サイズが最大 4G バイトの可変長であり，データ・サイズが大きくなっても通信の開始/終了に要するオーバーヘッドが一定なのに対し，「*PPRAM-Link* API on Linux」では 1 度に 256 バイトしかデータを転送できないため，データサイズが大きくなると 256 バイト毎にオーバーヘッドがかかるためである．

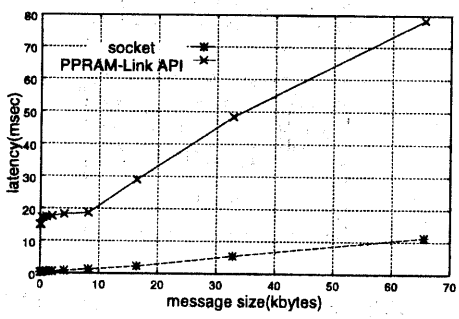

図 9: 基本通信性能 (レイテンシ)

5.2 機能の充足度

PPRAM-Link API の機能の充足度を検証する目

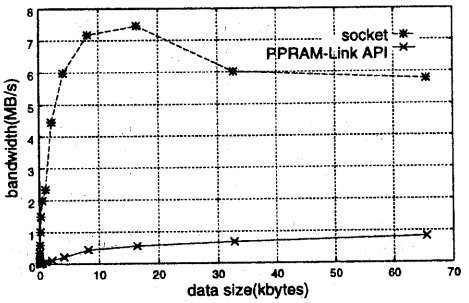

図 10: 基本通信性能 (バンド巾)

的で,「*PPRAM-Link* API on Linux」上に PVM を実装した．その構成を図 11に示す．

PPRAM-Link API は *PPRAM-Link* のトランザクションと等価であるために，最大データサイズが 256 バイトに制限される．OS やデバイス・ドライバを記述する際には良いが，PVM のような上位のアプリケーションを記述するにはパケット分割等の手間がかかり，使いやすいものではない．

また，Linux のように通信手段がメッセージパッシングのみのシステムにおいては，メッセージパッシング特有の使用上の制約が起こり，read/write を基本とする *PPRAM-Link* API を実装するには適していない．

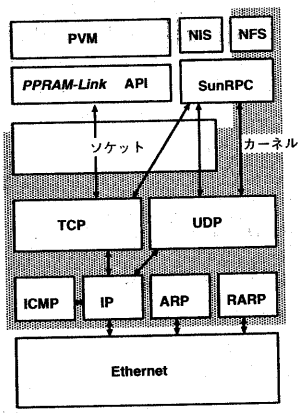

図 11: *PPRAM-Link* API on Linux 上への PVM の実装

6 おわりに

以上，*PPRAM-Link* API の概要を示し,「*PPRAM-Link* API on Linux」の実装および評価を行った．

その過程で *PPRAM-Link* API の現仕様にいくつか問題点を発見した．これら問題点は *PPRAM-Link* API 仕様にフィードバックして，今後当仕様の改良を重ねていく予定である．

謝辞

日頃から御討論頂く，九州大学 大学院システム情報科学研究科 安浦寛人 教授，岩井原瑞穂 助教授，*PPRAM* プロジェクトのメンバ諸氏，ならびに，安浦・村上・岩井原研究室の諸氏に感謝致します．また，*PPRAM* コンソーシアムの賛助会員，特別会員の皆様に感謝致します．特に，分科会の場で貴重なご意見を多数頂戴致しましたことに深謝致します．

今回の論文作成にあたって，特に御助力頂いた安浦・村上・岩井原研究室の宮嶋浩志 氏に心から感謝致します．

本研究は一部，文部省科学研究費補助金 基盤研究 (A)(2) 展開研究「メモリ/ ロジック混載技術に基づく大規模集積回路システム・アーキテクチャの研究開発」(課題番号:09358005) に依る．

参考文献

[1] 村上和彰，岩下茂信，宮嶋浩志，白川 暁，吉井 卓，“メモリ-マルチプロセッサー体型 ASSP (Application-Specific Standard Product) アーキテクチャ: *PPRAM*,” 信学技報，ICD96-13, CPSY96-13, FTS96-13, 1996 年 4 月．

[2] 村上和彰, 吉井卓, 岩下茂信, 宮嶋浩志, “*PPRAM* ベース・システム向け分散共有メモリ・システムの提案,” 情処研報, OS-73-2, 1996 年 8 月．

[3] 村上和彰, 岩下茂信, 宮嶋浩志, “メモリ-マルチプロセッサー体型 ASSP「*PPRAM*」用標準通信インターフェイス『*PPRAM-Link* Standard』Draft 0.0 の概要,” 情処研報, ARC-119-27, 1996 年 8 月．

[4] 山崎雅也, 橋本浩二, 沖野晃一, 村上和彰, “*PPRAM-Link* 論理階層仕様 (九大案 0.1 版) の概要,” 信学技報, ICD97-24, 1997 年 5 月．

[5] 九州大学 *PPRAM* プロジェクト・チーム, *PPRAM-Link* 物理階層仕様書 *(九大案 0.3 版)*, *PPRAM* コンソーシアム第 6 回物理階層分科会, 1998 年 2 月 12 日．

[6] 九州大学 *PPRAM* プロジェクト・チーム, *PPRAM-Link* 論理階層仕様書 *(九大案 0.4 版)*, *PPRAM* コンソーシアム第 6 回論理階層分科会, 1998 年 2 月 12 日．

[7] 九州大学 *PPRAM* プロジェクト・チーム, *PPRAM-Link API* 仕様書 *(九大案 0.4 版)*, *PPRAM* コンソーシアム第 6 回 API 分科会, 1998 年 2 月 13 日．

[8] *http://www.ppram.or.jp/*

[9] 山口和紀, 他, *The UNIX Super Text*, 技術評論社, 1992 年 12 月．

[10] *http://www.linux.org/*

[11] Geist,A.,Beguelin,A.,Dongarra,J.,Jiang,W., Manchek,R.,Sunderam,V., (村田英明 訳), *PVM3* ユーザーズガイド&リファレンスマニュアル日本語版, 1995 年 2 月．

[12] 梶崎浩嗣, PVM (Parallel Virtual Machine) FAQ, *http://csecw10.cs.nda.ac.jp/~kaji/reserch.html*, 1995 年 8 月．