ICOT Technical Report: TR-0733

TR-0733

並列推論マシンPIMの アーキテクチャ

中島　克人、益田　嘉直、中島　浩（三菱）
近藤　誠一、武田　保孝、村澤　靖
小森　隆三

January, 1992

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191 ~ 5
Telex ICOT J32964

Institute for New Generation Computer Technology

並列推論マシン PIM のアーキテクチャ

中島克人* 益田嘉直** 中島 浩*** 近藤誠一+ 武田保孝* 村澤 靖* 小森隆三*

1. まえがき

複雑で大規模な問題をより簡単にプログラムし，より高速に実行するという，ユーザーのコンピュータに対する要求はとどまるところを知らない。これに対して，コンピュータメーカーは半導体技術の目覚ましい進歩を背景にメモリの大容量化やプロセッサの高性能化という形で速度向上への期待にこたえてきた。しかし，単一プロセッサに関しては，今までのような性能向上をこれまでのペースで続けることが難しくなってきている。

パイプライン，分岐予測などの従来からの大型計算機技術に加え，最近ではSuperscalar，VLIW などのように複数の命令を同時に実行するための様々なアーキテクチャ上の改良が施されているが，それらによる速度向上には限界が見えてきている。半導体の集積度を十分に生かす道は，設計工数の点から見ても，もはやマルチプロセッサ化以外になくなりつつある。一方，ユーザーの立場からは，マルチプロセッサ化はプログラムを更に難しくする。それぞれのプロセッサを有効に活用するための従来どおりのプログラミング技術に加え，複数のプロセッサを無駄なく働かせる，いわゆる負荷分散の技術が必要となるためである。

1982年から10年計画で始められた国家プロジェクト"第五世代コンピュータ"は上記の半導体技術の進歩を予測し，複雑な問題を高速に実行できる計算機の実現を，論理型言語と並列処理を結び付けることによって目指したものである。

三菱電機(株)はこのプロジェクトの推進母体である(財)新世代コンピュータ技術開発機構（ICOT）の委託を受け，この度プロジェクトの最終成果である並列推論マシン PIM[1] の開発を完了した（図1）。

この論文では，大規模知識情報処理マシンを指向した PIM のアーキテクチャとその上に実装した高レベル並列言語KL1について述べる。

2. システムの特長

三菱電機(株)が，ICOT からの委託で開発した PIM は，PIM／m と称され，プロトタイプとしてやはり1987年に受託開発したマルチ PSI[2][3] と同様，二次元格子状に多数の要素プロセッサ（PE）を接続した疎結合型マルチプロセッサである。PIM／m ではそれぞれの PE がローカルメモリを備えることから分散メモリ型とも称する。PIM／m の最大構成は256 PE である。

バス接続などで一つのメモリを複数 PE で共有する共有メモリ型のマルチプロセッサでは，各 PE にキャッシュメモリを配するとしても，メモリへのアクセス競合による性能低下のために，接続 PE 数はせいぜい8～16台程度が限界であると言われている。これに対して PIM／m のような分散メモリ型マルチプロセッサは，メモリへのアクセス競合が生じないことから，PE の台数に関する拡張性が高く，大規模並列マシンに適している。PIM／m で採用した二次元格子接続は PE 間の最大距離が PE 数の平方根に比例して増大するため，PE 数が大きくなるにつれて任意の PE 間の通信には配慮が必要となるが，実装が非常に容易であることから，1,000台規模程度までのシステムに適している。

分散メモリ型では PE 間の通信のオーバヘッドが大きいため，仕事の分散（負荷分散）が難しい。分散の単位（粒度という。）を制御し，負荷分散と PE 間通信オーバヘッドのバランス点を見いださなければならないからである。残念ながらこれをシステムで自動的に制御するような技術はまだ確立しておらず，ユーザーがそれぞれの問題に応じた負荷分散手法をソフトウェアで実装し，それをチューンアップしていかなくてはならない。そのために，並列プログラムの記述を容易にし，負荷分散の技術開発をサポートするための強力なプログラミング言語と充実したプログラミング環境が必要である。

PIM で採用している並列論理型言語 KL1（核言語第1版）は，①変数の値の待ち合わせによる同期機構によってバグの少ないコーディングができる，②宣言的な記述で小粒度

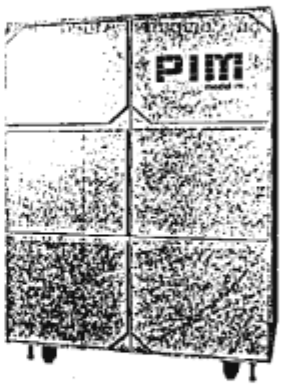


図1．PIM／m の筐体外観（32PE 内蔵）
（最大構成は8筐体256PE）

74(682) *三菱電機(株)情報電子研究所 ***同研究所（工博）
**同コンピュータ製作所 +(財)新世代コンピュータ技術開発機構

の並列計算を自然に記述でき，③仕事をどれだけまとめて実際の負荷分散の単位にするかは，プログラムのロジックを変更することなく後から指定できる，という特長をもつ。このため，ユーザープログラムのデバッグや性能改良，さらには新しい負荷分散手法の開発に大変適している。PIM／m ではこのKL 1を高速に実行するハードウェア及びファームウェアを備えている。

以下では，PIM／m のシステム構成，ハードウェア，言語とその実行方式などについて順に説明する。

3. ハードウェアアーキテクチャ

3.1 システム構成

最大構成はマルチ PSI では64 PE であったが，PIM／m では16×16，すなわち256 PE からなり，これらは1きょう（筐）体に32 PE ずつ（図1），合計8筐体に収められる。PE

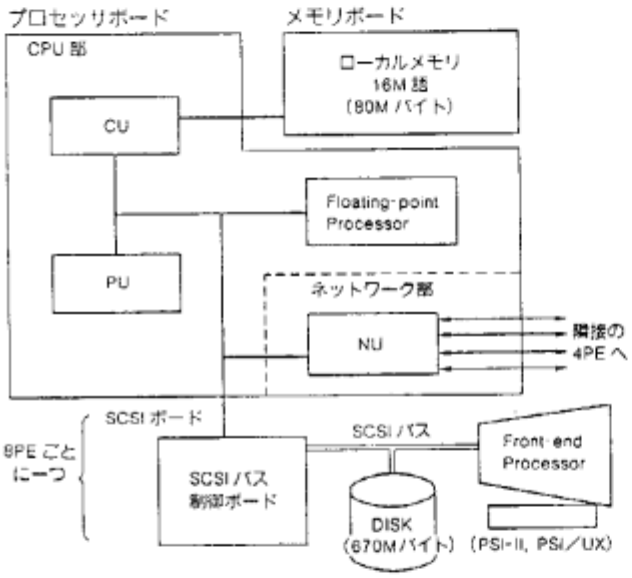


図2．要素プロセッサのブロック図

表1．要素プロセッサ（PE）諸元

		PIM／m	マルチ PSI
プロセッサボード		1枚	4枚
	サイクル時間	65ns	200ns
	使用 LSI	CMOS セルベース 0.8／1.0μm 3種	CMOS ゲートアレー 8K／20K ゲート 9種
	マイクロ命令	64ビット水平型	53ビット水平型
	パイプライン	5段	なし
	キャッシュメモリ	データ 4K 語×1 命令 1K 語×1	データ，命令共通 4K 語×1
	ネットワークバンド幅	10ビット／チャネル（ch） 3.84M バイト／秒・ch	10ビット／チャネル（ch） 5M バイト／秒・ch
メモリボード		1枚	4枚
	メモリ容量	80M バイト（16M 語）	80M バイト（16M 語）
	使用素子	4 M ビット DRAM	1 M ビット DRAM

1台当たりの性能はマルチ PSI の2－3倍となる。PE はメモリとプロセッサの2枚のボードから構成される（図2）。メモリボードは4 M ビット DRAM を使用することによって80 M バイトの容量を実現している。プロセッサボードは CPU 部とネットワーク部からなり，PU (Processing Unit)，CU (Cache Unit)，NU (Network Unit) の三つの CMOS セルベース LSI，及び周辺回路から構成される。8 PE ごとに一つの PE には入出力用 SCSI バスの制御ボードを通して670 M バイトの内蔵磁気ディスクやフロントエンドプロセッサ (FEP) などが接続される。FEP には "MELCOM PSI-II" 若しくは "MELCOM PSI／UX"[4][5] が用いられる。これらは大容量データベースや高度なマンマシンインタフェースを必要とする応用システムに活用される。

要素プロセッサの諸元をマルチ PSI のそれと比較して，表1に示す。

3.2 CPU 部[6]

CPU は論理型言語専用に開発された2種類のLSI（PU，CU）から構成され（注1），5段のパイプライン，水平型マイクロプログラム制御，タグアーキテクチャ（8ビットタグ＋32ビットデータの40ビット／語），及びデータと命令にそれぞれのキャッシュメモリを配したハーバードアーキテクチャが特長となっている。KL 1言語の実行速度は後述の append プログラムで530 K LIPS (Kilo Logical Inference Per Second) に達する。

5段のパイプライン（図3）の各ステージの機能は以下のとおりである。

D：命令キャッシュから読み出された機械語命令（後述）のデコードを行う。すなわち，機械語命令コードからオペランドのタイプ（レジスタ，即値，メモリアクセス）やオペランドに対する処理を決定する。

A：メモリアクセスオペランドのアドレス計算を行う。次命令若しくは分岐先命令のフェッチも行う。なお，分岐予測が失敗した場合や予測が不可能な場合には，Eステージの指示によって再フェッチが行われる。

R：Aステージで計算したアドレスに従って，必要であれば主記憶上のオペランドを読み出す。

S：Eステージでのマイクロ命令の読出し，オペランドのセットアップ，及びデレファレンス機能がある。デレファレンスとは変数の参照ポインタの連鎖をたどることで，このステージには，Rステージで読み出したデータが更に他の変数セルへの参照ポインタである場合に，この連鎖の末端まで自動的にたどり続ける機能が備えられている。

（注1） PU及びCUは国からの成果移転を受け，当社のAIワークステーション "MELCOM PSI／UX" の推論部にも使用されている。

E：機械命令の主な処理を64ビット幅のマイクロプログラムで実行する。マイクロ命令は32K語の高速のマイクロ制御メモリに格納され，毎サイクル読み出されて実行される。図4はマイクロ命令フォーマットである。条件分岐，内部レジスタ間データ転送，レジスタ間演算，キャッシュアクセス，条件フラグ設定，内部カウンタ操作などが1命令で同時に行える。特に，タグ値などの判定から分岐完了までが2サイクルで行えるため，KL1プログラムの実行のように動的な判定が頻繁な処理には大変有効である。

3.3 ネットワーク制御部

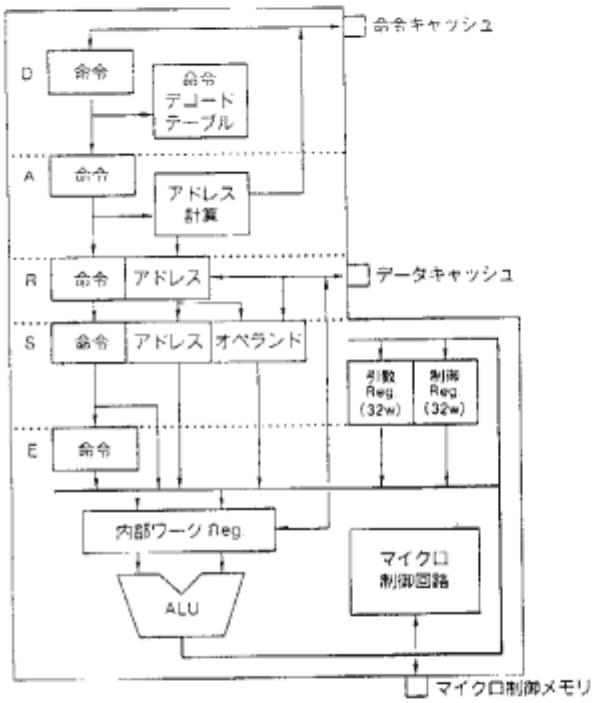
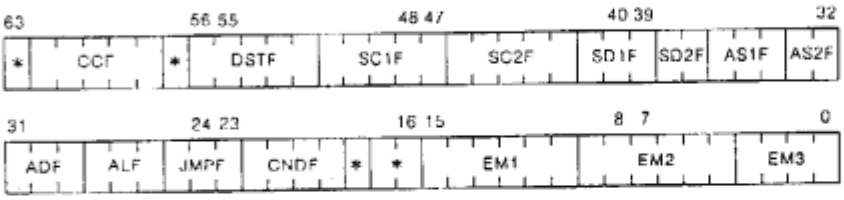


図3．要素プロセッサの5段パイプライン



CCF<5>：キャッシュ／メモリ制御
DSTF<5>：引数／制御レジスタ セット指定
SC1F<5>：引数／制御レジスタ 選択指定(1)
SC2F<5>：引数／制御レジスタ 選択指定(2)
SD1F<3>：内部ワークレジスタ セット指定(1)
SD2F<2>：内部ワークレジスタ セット指定(2)

AS1F<3>：ALU左入力レジスタ指定
AS2F<2>：ALU右入力レジスタ指定

ADF<3>：ALU出力レジスタ指定
ALF<3>：ALUオペレーション制御
JMPF<3>：分岐オペレーション制御
CNDF<4>：分岐条件制御

EM1<6>：即値／オペレーション修飾子(1)
EM2<6>：即値／オペレーション修飾子(2)
EM3<4>：即値／オペレーション修飾子(3)
＊<計5>：条件フラグ設定，カウンタ操作

図4．要素プロセッサのマイクロ命令フォーマット

PE間を接続するネットワーク制御回路(NWC)はNUを中心に構成される。NWCには隣接4方向のPE，及び自CPUのための5組の双方向通信チャネルが接続されており，PE間メッセージの送受信及び転送を制御する(図5)。各チャネル幅は10ビットで，9本のデータ線，1本のbusy線(逆方向)からなる。各チャネルのバンド幅は3.84Mバイト／秒である。

図6はメッセージパケットの形式を表す。データ線の上位の2ビットにより，メッセージパケットの先頭バイト(以下"ヘッダ"という。)と最後尾バイト(以下"テール"という。)が判別される。NWCはヘッダにあるあて先PE番号を読み取り，内部のルーチング用のテーブルを参照することによって転送すべき方向を決定する。

そして，その方向のチャネルが"busy"でなければそのチャネルにメッセージパケットを転送する。この時，そのチャネルはメッセージテールの転送完了まで保持される("wormhole"ルーチングという。)。あて先PE番号が自PEである場合はNWC内のリードバッファに取り込み，テールまでの取り込みが完了するとCPUに割込みを上げ，メッセージの解釈を要求する。CPUではKL1プログラムの処理の適当な区切りでこの割込みを処理する。CPUのメッセージ送信はNWCのライトバッファに一つの完全なメッセージを書き込み終わることにより，自動的に開始される。

このように，NWCはCPUから出入りするメッセージの操作を行うだけでなく，通り過ぎるメッセージの転送もCPUに負担をかけずに自動的に行うため，遠距離にあるPE間の通信を円滑に行うことができる。

4. 並列論理型言語KL1

大規模な知識情報処理システムを実現するには，プログラムの生産性や保守性が非常に重要となる。並列論理型言語KL1 (Kernel Language version 1)(図7)は，並列でかつ複雑な問題を効率良くプログラミングできることをねらった高レベル並列言語で，PIMの核言語(ハードウェアとソフトウェアとのインタフェース言語)としてICOTで開発された。並列論理型言語Flat GHC (Flat Guarded Horn Clauses)[7]にシステム記述やスケジューリング及び負荷分散指定のための機能を付加したものである。

KL1は並列言語として次の特長をもっている。

(1) 同期は，未束縛変数への値の代入を待ち合わせる機能で実現される。

(2) 通信は，変数値の読出し時に言語

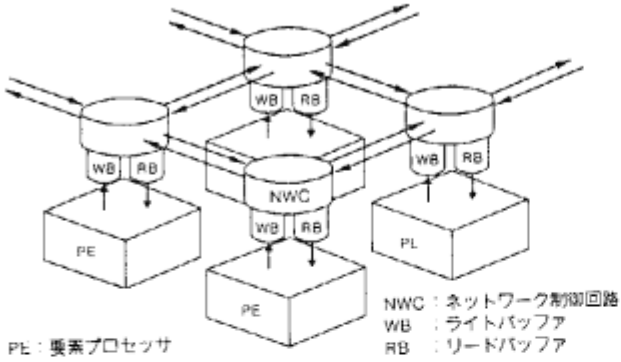


図5．ネットワーク制御回路(NWC)による
プロセッサ間結合

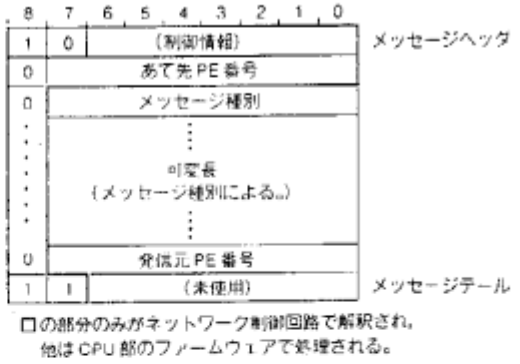


図6．メッセージパケットのフォーマット

機能として自動的に行われる。

(3) 論理変数の特長である単一代入規則により，いったん定まった変数値は変化しないことから，他PEへの値のコピーが許される。したがって，2回目以降の変数値の読出しにはPE間通信が省略できる。

(4) 負荷分散は"プラグマ"と称するKL1述語呼出し時の付加情報で行われるため，プログラムの論理的な意味を変えずに負荷分散を指定・変更できる。

(5) メモリ領域の割付けや再利用も言語機能として自動的に行われる。これらにより，従来の手続き型言語に並列機構と同期機構を加えたようなものに比べ，同期のタイミングバグなどに煩わされることが非常に少なくなる。

5. KL1実行処理方式

KL1プログラムはPrologでのWAM[8]に相当する抽象機械語命令にコンパイルされ[9]，PEのマイクロプログラムによって直接解釈実行される。図8はリストの結合を行うKL1プログラムの抽象機械語命令へのコンパイル例である。switch系，wait系及びread系の命令が変数の待ち合わせと値のチェックを行い，get系及びwrite系の命令が主に未定義変数への値の代入を行う。put系命令は変数値を引数レジスタに用意する。

他PEへのKL1ゴールの送出にも専用機械語命令が用意されており，マイクロプログラムで解釈実行される。なお，送出先PEでのKL1ゴールの実行に必要なプログラムコード，データなどは必要になった時点で自動的にコピーが送られ，不要になったものは廃棄される。そして，空いたメモリ領域は再利用される。ゴール実行のための資源管理やゴールの終了報告なども言語表層以下の言語処理マイクロプログラムによってPE間通信を用いて行われる[10]。

以上のように，メモリ管理やプロセス管理のような従来のオペレーティングシステムの基本機能を並列に制御する部分が，言語機能の一部としてマイクロプログラムによって効率良く行われ，その上位に位置するPIMのオペレーティング

(述語goalの定義)

```
goal(X, Y) : −X > 0 | goal1(X, Y), goal2(Y, Z). ・・・クローズ1
goal(X, Y) : −X =< 0 | goal3(X, Y), goal4(Y, Z). ・・・クローズ2
```

ガード部　　ボディー部

並列論理型言語KL1によるプログラムは複数のクローズからなる述語定義の集まりからなる。|(bar)の左側の条件(ガード部)が最初に成立したクローズが選ばれ，他のクローズの実行は放棄される。値が決まっていない変数の条件を調べようとして，クローズが選択できなかった場合はその述語呼出しはサスペンドされる。選ばれたクローズのボディー部ゴール(goal1とgoal2，又はgoal3とgoal4)は並列に実行され，ゴール間の共有変数(上の例ではY)によって通信や同期が行われる。ゴールにはユーザー定義(他の述語を呼び出す。)，システム定義(算術演算を行うようなあらかじめシステムが提供している述語を呼び出す。)，及びボディーユニフィケーション("X=f(Y)"などのように表記し，変数への値の代入に用いられる。)がある。

図7．並列論理型言語KL1

システム(PIMOS[11]という。)やユーザープログラムへの負担が軽くなるよう考慮されている。

6. むすび

効率の良い並列実行のためにはどのようなタイプの問題に対しても，以下のようなことが必要となる。

(1) 並列度が高く，通信が1か所に集中しないように分散したアルゴリズム

(2) どのプロセッサもできるだけ遊ばせないようにする負荷分散

(3) (探索型の問題に対しては)無駄になってしまうような見込み計算を最低限で済ますスケジューリング

並列アルゴリズム，負荷分散などの問題のほかにも並列プログラミング言語の設計・改良，デバッグ環境の充実，故障に対する耐久性の増大など，並列処理技術の行く手には難しい課題が山積しており，並列ハードウェアの研究と足並みをそろえて一歩一歩進めていく必要がある。

現在PIM／m上では，KL1言語処理マイクロプログラム

```
append([X | L1], L2, L) :-
        true | L=[X | L0], append(L1, L2, L0).
append([], L2, L) :-
        true | L=L2.
```

(a) KL1 プログラム例

```
%%%3つの引数値が引数レジスタR1～R3にセットされて，app／3が呼出される
app／3 . try_me_else app／s                    %中断または失敗時の飛先を制御レジ
                                               %スタAPにセット
        switch_on_non_list      R1, app／n     %R1がListなら次，それ以外ならapp／n
                                               %ただし，未定義ならAPに分岐
        read_variable           R4             %入力ListのCAR要素をR4に取り出す
        read_variable           R5             %入力ListのCDR要素をR5に取り出す
        put_reused_structure    R1, R2, 2      %入力Listのcell回収が可能なら再利用
                                               %回収不可なら新しいList cellを割付
        write_value             R4             %出力ListのCARにR4を書き込む
        write_variable          R4             %新しい変数cellを割付け，それへのポ
                                               %インタを出力ListのCDRとR4に書込む
        get_bound_value         R3, R1         %R3とR1(出力List)をユニファイする
        put_value               R1, R5         %R1にR5をセット
        put_value               R3, R4         %R3にR4をセット
        execute                 app／3         %app／3を再帰呼出し

app／n : wait_nil               R1             %NILなら次，それ以外ならAPに分岐
        get_value               R2, R3         %R2とR3をユニファイ
        proceed                                %このクローズは終了し，他のゴール
                                               %を実行
app／s : suspend                app／3         %中断か失敗かの判定とそれぞれの処理
```

(b) KL1B 命令列へのコンパイル例

図8．リストを結合するKL1プログラムのコンパイル例

の評価・改良のほか，ユーザー管理・ファイル管理・資源管理などを行いプログラム環境を提供する並列オペレーティングシステムPIMOSの機能拡張などをICOTと協力して進めている。また，並列応用プログラムとしては，遺伝子情報処理の一環としてのたん(蛋)白質構造解析プログラム，判例を用いた法的推論システム，LSI-CADシステムの一環としての論理シミュレータ・セル配置プログラム・配線プログラム，囲碁システム，定理証明システムなどの開発が行われており，より実用的な並列処理システムに向けての研究が進められている。PIM／m上でのこれらの成果が将来の並列計算機の進歩への大きな足掛かりとなることを期待したい。

最後に，PIM／mの研究開発に際して御指導をいただいたICOT研究部内田俊一部長を始めとする関係各位に深く謝意を表す次第である。

参考文献

(1) Goto, A., Sato, M., Nakajima, K., Taki, K., Matsumoto, A. : Overview of the Parallel Inference Machine Architecture (PIM), Proc. of the International Conference on Fifth Generation Computer Systems, 208～229 (1988)

(2) Taki, K. : The Parallel Software Research and Development Tool : Multi-PSI System, Programming of Future Generation Computers, Elsevier Science Publishers B. V. (North-Holland) (1988)

(3) 益田嘉直，中川智明，岩山洋明，武田保孝，中島 浩，瀧 和男：㈱新世代コンピュータ技術開発機構向けマルチPSIシステム，三菱電機技報，62，No. 12，1100－1105 (1988)

(4) 湯浅維央，上田尚純，松本 明：三菱AIワークステーション《MELCOM PSI／UXシリーズ》の概要と特長，三菱電機技報，65，No. 6，594－601 (1991)

(5) 益田嘉直，田辺隆司，池田守宏，深沢 雄，岩山洋明，中島 浩：三菱AIワークステーション《MELCOM PSI／UXシリーズ》のハードウェアシステム，三菱電機技報，65，No. 6，602－607 (1991)

(6) Nakashima, H., Takeda, Y., Nakajima, K., Andou, H., Furutani, K. : A Pipelined Microprocessor for Logic Programming Languages, Proc. of the International Conference on Computer Design, 355－359 (1990)

(7) Ueda, K. : Guarded Horn Clauses : A Parallel Logic Programming Language with the Concept of a Guard, Technical Report TR-208, ICOT (1986)

(8) Warren, D. H. D. : An Abstract Prolog Instruction Set, Technical Note 309, Artificial Intelligence Center, SRI (1983)

(9) Kimura, Y., Chikayama, T. : An Abstract KL1 Machine and its Instruction Set, Proc. of the Symposium on Logic Programming, 468－477 (1987)

(10) Nakajima, K., Inamura, Y., Ichiyoshi, N., Rokusawa, K., Chikayama, T. : Distributed Implementation of KL1 on the Multi-PSI／V2, Proc. of the Sixth International Conference on Logic Programming, 436～451 (1989)

(11) Chikayama, T., Sato, H., Miyazaki, T. : Overview of the Parallel Inference Machine Operating System (PIMOS), Proc. of the International Conference on Fifth Generation Computer Systems, 230～251 (1988)