ICOT Technical Memorandum: TM-0224

TM-0224

AI ワークステーション PSI の アーキテクチャ

山本　明，江原輝文
(沖電気工業)

August, 1986

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

ＡＩワークステーションＰＳＩの アーキテクチャ

沖電気（株）総合システム研究所　コンピュータシステム研究部
アーキテクチャ研究室　　山本　明　　江原　輝文

１．　まえがき

最近、人工知能の分野では知識システムの研究、とりわけ知識表現及び知識利用の研究が盛んに行なわれている。これに伴い、知識システムの構成要素である知識ベースと推論機構をうまく表現できるプログラミング言語として述語論理に基づくプログラミング言語（論理型プログラミング言語）が注目されてきている。しかしこれまで、充分実用に耐えうる論理型プログラミング言語のためのプログラム開発環境が少なくその普及を妨げる要因になっており、このような状況を打開するため、実用規模のプログラムを高速に実行できる処理系及びマシンの開発が急務であった。

当社は、第５世代コンピュータ研究開発プロジェクトの発足１９８２年以来、（財）新世代コンピュータ技術開発機構を中心とした論理型プログラミング言語実行専用AIワークステーションPSI(Personal Sequential Inference Machine)の開発計画に参画して共同研究を続けて来ており、１９８４年にはハードウェアシステムの開発を完了した。

本稿では、PSIのアーキテクチャ、処理方式、ハードウェアのそれぞれの概要と特徴について述べ、アーキテクチャの評価のために行なった実行性能の測定結果を示す。

２．　開発目標

PSIは、論理型プログラミング言語によつて実用的なプログラムを開発でき、かつユーザに十分効率の良い実行環境を提供するために、以下の特徴を備えたＡＩワークステーションとして開発された。

１）実用的な論理空間と実行速度

論理型プログラミング言語で記述された実用規模のプログラムを実行するには、充分に高い処理速度と大きなメモリ空間が必要とされる。特に、大きなメモリ空

間が取れなければ実用規模のプログラムは実行できない。したがってPSIでは、物理空間として、代表的な論理型プログラミング言語の処理系であるDEC-2060上のDEC10-Prologより一桁多い１６Ｍワードを、又速度については少なくともDEC10-Prologと同等の３０KLIPS(Kilo Logical Inference Per Second)を有することを目標に設計した。

２）優れた操作性

AIワークステーションとして充分実用に耐えるためには、論理型プログラミング言語を高速に実行できると共に、使い易いプログラミング環境をユーザに提供する必要がある。したがって、PSIは高解像度のビットマップディスプレイを用いたマルチウィンドウシステムやポインティングデバイスを備えると共に、複数のPSIが資源の共通利用を図ることができるようにローカルエリアネットワークをサポートすることとした。

３）システム記述能力の導入

論理型プログラミング言語は記述力が高く、非決定的な処理やパターンマッチング処理などを記述するために向いている。さらにこれが、処理が決定的で処理効率が問題となるような分野でも充分使用できることが明らかになれば、その応用分野はさらに広がるものと思われる。そのため、PSIでは決定的な処理であるオペレーティングシステムを論理型プログラミング言語で記述でき、かつ充分な処理速度で実行できるようにした。

４）測定、評価サポート機能

PSIは論理型プログラミング言語処理のための多くの機能を持っているが、論理型プログラミング言語向きアーキテクチャは新しい技術分野であり、その妥当性について充分な評価を行なう必要がある。したがって、PSIではアーキテクチャの評価を簡単に行なうための計測機構を組込んだ。

３．　KL0(Kernel language Version 0)

ソフトウェアとハードウェアとのインタフェースとしてPSIでは言語KL0を設定した。KL0は、DEC10-Prologと同じ一階の述語論理に基づいた論理型プログラミング言語であり、言語レベルも同じである。しかし、オペレーティングシステムを容易に記述できるように、ハードウェア操作機能などDEC10-Prologには無い以

下に示す機能を持たせている。

1）豊富なデータタイプ

実用に充分耐えうるようにKL0には豊富なデータタイプが用意されている。それらにはシンボリックな定数を表わすためのデータタイプ（アトム）の他、整数、実数、ベクタ及びストリングを表わす各データタイプがある。

2）強力な実行順序制御機能

DEC10-Prologをプログラミング言語として実用的に使用できるものにしている機能としてカットがあるが、PSIではカットをさらに拡張した強力な遠隔カットの機能を備えている。その他にもユニフィケーションに失敗してバックトラックした際に前以て指定されていた処理を実行する機能（オンバックトラック機能）などいくつかの強力な実行制御機能がある。

3）オペレーティングシステムサポート機能

PSIではオペレーティングシステムも論理型プログラミング言語によって記述することを開発目標としたため、KL0にはオペレーティングシステムを記述するために必要となるプロセス管理及びメモリ管理などの機能が備えられている。その他、ハードウェア資源を操作する基本的な機能をも備えており、PSIの内部レジスタ、メモリの読出し／書込み、入出力装置の操作などを行なうことができる。

4．　アーキテクチャ

PSIでは論理型プログラミング言語で記述されたプログラムを効率良く実行するために以下のようなアーキテクチャを採用した。

1）ワード形式

各データのデータタイプを高速に検出することが論理型プログラミング言語を高速に処理するための重要な要素である。そのためには、豊富なデータタイプを表現できる充分な長さのタグ部を各データに付けることが望ましい。又、データ長としては、充分な大きさのアドレス情報や数値データを表現できるだけの長さが必要である。

そのほか、論理型プログラミング言語の処理では、不用となったメモリを回収してそれを再利用するガベージコレクションが必要になる。ガベージコレクションを効率良く行なうためには、専用のタグが必要である。これらの要求を満足さ

せるためPSIでは、１ワードの長さを８ビットのタグ部と３２ビットのデータ部から成る４０ビットとした。そして、８ビットのタグ部のうち２ビットをガベージコレクション専用のタグ部とし、残りの６ビットをデータ用タグ部とした。

データ用タグ部は、KL0で使用されるデータタイプを表現すると共に、PSI内部でコード情報や実行制御のためにも使用される。

２）機械語形式

機械語の決定は、マシン全体のアーキテクチャに大きな影響を及ぼす。論理型プログラミング言語では、メモリアクセス回数が多いこと、実行時に動的に決定される処理が多いことから、PSIでは実行時に全体の処理を見通しながら不要なメモリアクセスを減らすなどの最適化が図れるように、機械語のレベルを汎用計算機のLoad／Store命令のレベルより高く設定した。

PSIの機械語とKL0はほぼ１対１に対応しており、ワード列からなるテーブル形式（内部形式）で表現される。KL0で記述されたプログラムは、まずコンパイラにより内部形式に変換され、この内部形式がPSIのマイクロプログラムと専用のハードウェアにより解釈されながら実行される。

内部形式は、KL0で記述されたプログラム中で同一述語名でかつ同数の引数を持つクローズの集合を１つのテーブル（プロシジャテーブル）として表わしたものである（図－１）。プロシジャテーブルは、テーブルサイズ、ヘッド引数の個数等プロシジャ全体に関する情報を持つヘッド部と各クローズ単位の内部形式で構成される。各クローズの内部形式はさらにクローズのタイプ、引数個数等各クローズに関する情報が置かれるヘッド部と、ヘッド引数部及び各ボディゴールの内部形式で表わされる。

３）論理アドレス

PSIではプログラムの実行に必要な変数領域や実行制御情報を４つの独立したメモリアドレス空間を用いて管理すると共に、マルチプロセシング機能もサポートする。したがって、物理メモリを効率良く使用するため論理アドレスと物理アドレスを分離し、さらに物理メモリを１K語単位（ページ）に分割して管理するようにした。又、論理アドレスは２５６個の独立した論理アドレス空間（エリア）に分けて管理するようにした。

４）アドレス変換

プログラムが充分な物理アドレス空間を利用できるようにするためには、仮想記憶方式を採用することが望ましいが、仮想記憶方式を採用した際の有効なガベージコレクションの方式がまだ確立されていないため、PSIでは仮想記憶方式を採用していない。そのかわり、物理メモリの容量を最大１６M語まで実装できるようにした。

論理アドレスから物理アドレスへの変換はページマップベースメモリとページマップメモリと呼ぶ2種類のテーブルによって管理される。ページマップベースメモリには各エリアに対応したページマップベースのアドレスが、又ページマップメモリには各エリアの論理ページ番号に対応した物理ページアドレスが格納されている。高速なアドレス変換を実現するためページマップベースメモリ及びページマップメモリを高速メモリで実現し、アドレス変換を１マシンサイクルで実行できるようにした。

5）プロセス管理

オペレーティングシステムを記述する上で、マルチプロセシング機能による並列処理は不可欠である。PSIでは、論理型プログラミング言語でオペレーティングシステムを記述するために論理型プログラミング言語の環境でのマルチプロセシング機能を実現した。

プロセスの切り替えでは、現在実行中のプロセスが後で再実行可能なように実行途中の各種の情報を保管する必要がある。このため、PSIではプロセスの状態などハードウェアとソフトウェアとのインタフェース情報（プロセスコントロールブロック）を格納するエリアと、内部制御レジスタの値等ハードウェアのみで使用する情報（拡張プロセスコントロールブロック）を格納するエリアを用意した。プロセス切り替え時のコントロールブロックと拡張コントロールブロックの格納と取り出しはPSIのマイクロプログラムによって高速に実行される。

PSIで同時かつ独立に実行できるプロセスの数は最大６３個である。

6）割込み処理

PSIではプロセスを起動する手段の１つとして、割込機能をサポートしている。割込みの検出は、プロセススイッチの際に格納すべき情報が最も少なくなるような処理の切れ目で行われている。しかし、ユニフィケーション等処理に要する時間が非常に長くなる可能性があるものについては、処理の途中であっても緊急度

の高い割込み要求を受付けるようにした。

5．　処理方式

論理型プログラミング言語KL0の基本的な実行処理はDEC10-Prologと同様に、クローズのコール、呼び出し元クローズへのリターン、ユニフィケーション及びユニフィケーション失敗時のバックトラックである。このほかKL0には強化された実行順序制御機能やオペレーティングシステムのサポート機能がある。これらはマイクロプログラムで記述されたインタプリタ（マイクロプログラムインタプリタ）で効率良く実現されている。

PSIのマイクロプログラムインタプリタには以下の特徴がある。

1）スタック方式

各クローズで使用される変数セルはクローズが呼び出された時点で生成され、そのクローズの全ての処理が完了し呼び出し元のクローズに戻る際に解放される。このような処理にはスタックを使用するのが効果的である。したがって、PSIは4本のスタックを使用して処理を実現する方式を採用した。これら4本のスタックをそれぞれコントロールスタック、ローカルスタック、グローバルスタック、トレイルスタックと呼ぶ。

コントロールスタックは、プログラムの実行順序を制御するために使用される。ローカル及びグローバルスタックはプログラム中の変数に対応する変数セルを動的に格納するために使用される。変数にはローカル変数とグローバル変数の2種類があり、グローバル変数は構造体の引数として使用されている変数であり、その他の変数をローカル変数と呼ぶ。ローカル変数、グローバル変数はそれぞれローカル、グローバルスタックに格納される。トレイルスタックは、ユニフィケーションにより値がバインドされた変数セルのセルアドレスを格納するために使用される。これらのセルアドレスはユニフィケーションの失敗によりバックトラックが発生した場合、バインドされた値をリセットするために使われる。

なお命令コードはヒープ領域と呼ばれるコードエリアに置かれる。

2）引数コピー方式

ユニフィケーションは、呼び出し元クローズのボディーゴールの引数の値と呼び出されたクローズのヘッド引数の値との間で行なわれる。PSIでは、ユニフィ

ケーションに先だって呼び出し元の引数の値をあらかじめ専用のバッファにコピーしておき（引数コピー）、実際のユニフィケーションは呼び出された側のヘッド引数の値とこのバッファ上の値を使って行われる。ユニフィケーション終了後、必要ならこのバッファの値はメモリに格納される。

この方式は、バッファへの格納および読み出しに多少のオーバヘッドはあるが、ユニフィケーションに失敗し他のユニフィケーションを行う場合、呼び出し元の引数を新たに引数コピーする必要はなく、以前の引数コピーで生成された値をそのまま使用できるため、処理の高速化が図れるという利点がある。

3）構造体共有方式

構造体の計算機内部での扱いには構造の形を共有する共有方式と構造全体をそのままコピーして使うコピー方式がある。共有方式は構造をコピーしないため構造体全体を扱う場合にメモリアクセスの回数が少なくてすむが、各構造体要素へのアクセスでは一旦構造の形をアクセスする必要がある。一方、コピー方式は共有方式と全く対照的な特性を持っており、構造体要素へのアクセスでは要素に直接アクセスできるが、構造全体を扱う場合にコピーのために多くのメモリアクセスが必要である。論理型プログラミング言語では構造体の扱いが容易であり、大きな構造体の利用度が比較的高いと予想され、この場合構造全体をコピーする必要の無い共有方式の方がコピー方式より有利であると予想されることから、PSIでは共有方式を採用した。

4）最適化技法

マイクロプログラムインタプリタは、KL0の内部形式中の情報に基き、実行状態を動的に判断しながら最適化を図る。この最適化にはテイルリカージョンの最適化とクローズインデクシングがある。

a）テイルリカージョンの最適化

クローズの最後の述語呼び出しで、そのクローズが他の選択肢をもたず、処理が決定的に終わる場合、このクローズの変数領域は保存の必要がない。PSIでは引数コピー方式を採用しているので、コピー終了後呼び出し元クローズのために必要な変数領域を解放することができる。この最適化は、引数コピー方式と一体となって不要な変数セル領域によるスタックの消費を極力押さえる効果がある。

b）クローズインデクシング

クローズインデクシングは、ある述語呼び出しに対して対応するクローズが多数ある場合何らかのハッシュ値を基にして目的のクローズを高速に呼び出す方法である。PSIでは、クローズインデクシング用の情報がコンパイラにより生成されると共に、ハッシュ値を計算する機械語が計算機に組込みの述語の形で内部形式の中に挿入される。マイクロプログラムインタプリタはこれらの情報にしたがってヘッド引数の値からハッシュ値を計算して目的のクローズを求める。

5）その他

論理型プログラミング言語KL0のように高機能な言語でオペレーティングシステムを全て記述するよりも、プロセス管理、メモリ管理といった処理効率が問題となる部分は一部をハードウェアでサポートしたほうが効率がよい。したがって、PSIでは割り込み検出とハンドラの起動、プロセスの切り替え、物理メモリの動的な割り付けおよびガベージコレクション等はマイクロプログラムインタプリタと専用のハードウェアを使って実現した。

6．　ハードウェア構成

PSIは中央処理部と入出力部から構成されており、中央処理部はさらに、データ処理部、メモリ部、入出力バス制御部及びコンソールインタフェース部から構成されている。（図－2）

1）データ処理部

論理型プログラミング言語の処理では演算よりも単純な比較が多いこと、多くの実行環境を管理するための処理が多いことが特徴である。このことからデータ処理部を、簡単なALUとレジスタファイルを2本のソースバスと1本のデスティネーションバスで結合した簡単な構成とした。又、タグの処理が論理型プログラミング言語の実行速度に大きく影響することから、データバスとは別に、タグ値の比較、タグ値による多方向分岐が高速に行なえるようなハードウェア構成としている。データ処理部での演算及びタグに関する処理は全て1マシンサイクル(200ns)で完了される。

さらに、実行環境の管理を高速に行なうため管理情報をデータ処理部内のレジスタファイルに保持しておくことにし、そのために1K語という充分大きなレジスタファイルを持った。その他、効率の良いメモリアクセスが行えるようにする

ため、データレジスタ、アドレスレジスタと呼ばれる個別レジスタを2組用意し、メモリ部とのインタフェースレジスタとしても汎用レジスタとしても使用できる構成にしている。また、PSIはマイクロプログラム制御のマシンであり、６４ビット×１６Ｋワードの書き換え可能な制御記憶を持っている。マイクロ命令は3オペランド形式の垂直型を採用しており、１つのマイクロ命令を実行中に次のマイクロ命令を読み出す単純な方式とした。

２）メモリ部

メモリ部は、キャッシュメモリ部、アドレス変換部、主記憶部から構成されている。論理型プログラミング言語の処理系の特徴の１つはメモリアクセスの頻度が高いことである。したがって、処理を高速化するには実際にメモリアクセスを高速にするか、データ処理部から見た時の見かけ上のメモリアクセスを高速にする必要がある。PSIでは、見かけ上のメモリアクセスを高速化するため、キャッシュメモリを採用すると共に、データ処理部はメモリ部と独立に処理を行なえる構成とした。

又、PSIで採用した論理型プログラミング言語の処理方式では、主記憶をスタックとして使い、実行環境を連続したアドレスへ書き込むことが多い。したがって、PSIではキャッシュメモリと主記憶との間のデータ転送幅を広くし、キャッシュメモリの制御方式としてストアイン方式を採用した。この方式は、主記憶へ書き込むべきデータをキャッシュメモリの中に溜めておき、後に別アドレスのデータによってキャッシュメモリ内のそのデータが書き換えられる時はじめて主記憶へ書き込む方式であり、連続したメモリアドレスへの書き込みが多い処理の高速化に有効である。その他、高速に処理を行うためキャッシュメモリを論理アドレスで管理するようにした。

アドレス変換部では、論理アドレスから物理アドレスへの変換を行なう。アドレス変換に必要なページマップベースメモリとページマップメモリはそれぞれ１５ビット×２５６語と１５ビット×３２Ｋ語で構成されている。物理メモリを効率的に使用するため、物理メモリの論理アドレスへの割当てはプログラム実行時に動的に行われる。

主記憶は、１語４０ビットで最大１６Ｍ語が実装可能であり、１ビットエラーの訂正と２ビットエラーの検出機構をもつ。

3）入出力バス制御部

PSIでは、市販の各種入出力装置を接続することも考慮し、入出力バスとして、IEEE-796バスを採用した。IEEE-796バスでは入出力装置アドレス空間はメモリアドレス空間と独立に割り付けられているが、PSIではプログラミングの容易さのために、ユーザインタフェースとしてメモリアドレス空間の一部に入出力装置のアドレス空間を割り当てた論理的な入出力バスアドレス空間を設定した。入出力バスの論理的アドレス空間は１６Ｍバイト（２４ビット）であり、そのうちの６４Ｋバイトを入出力装置のアドレス空間とした。

4）入出力部

PSIは、入出力装置として１２８０×８６４ピクセルの高解像度ビットマップディスプレイ、キーボード、ポインティングデバイス、２７２Ｍバイトのウィンチェスタ型固定ディスク装置、８インチのフレキシブルディスク装置、プリンタ及びバスメモリを備えている。

又、PSIはPSI相互間及び異機種コンピュータとの通信が可能なワークステーションとして設計されており、ローカルエリアネットワークはもちろんのこと、ローカルエリアネットワーク同士を接続するブリッジ、及びＤＤＸに接続するゲートウェイがサポートされている。

5）コンソールインタフェース部

保守、デバック、システム立上げのためにコンソールプロセッサが接続可能であるほか、ホストコンピュータで開発されたファームウェアやソフトウェアをダウンロードするためのインタフェースが備えられている。

7．　実行性能

論理型プログラミング言語そのものが新しく、現実に実用に耐えうる処理系が少ないこともあって、論理型プログラミング言語のどの要因がプログラムの実行にどのように影響するかということについては良く判っておらず、性能評価のための方法も確立していない。PSIの性能評価では、Prologコンテストの評価プログラムの中から選んだサンプルプログラムについて実行性能を測定し、DEC10-Prologの性能と比較した。

PSIの実行性能はPSI内部の１msecのタイマを利用して測定した。又、DEC10-Pr

ologの実行性能は、DEC10-Prologの組込述語であるSystatを使いPSIと全く同じ被測定プログラムを使って行なった。被測定プログラムはファストコードにコンパイルされ、DEC-2060上で実行された。

この結果、表－１からわかるようにnaive reverseのようなコンパイラによる最適化が図り易い簡単なプログラムでは、DEC10-Prologのコンパイラ方式の方が速いという結果が出たが、全体としてPSIはDEC10-Prologとほぼ同等の性能をもつと言える。プログラムが、ほとんどリスト処理として記述されており、性能評価のための充分なサンプルプログラムとはいえないが一応の目安となるであろう。

８．　あとがき

本稿では、論理型プログラミング言語を効率良く実行するための専用マシンとして開発されたＡＩワークステーションPSIのアーキテクチャの特徴、処理方式の特徴、ハードウェアシステムの概要について述べた。又、最後にPSIで採用されたアーキテクチャの有効性を評価するために行なった実行性能の測定結果について述べた。

これまで実用のために充分な実行速度が得られる論理型プログラミング言語の処理系がほとんどなかったが、PSIは実用に耐えうるだけの充分な速度を持ち、かつ汎用計算機と比べても十分コスト／パフォーマンスの良いマシンと言える。現在、このPSI開発の経験を基に、アーキテクチャの見直しを行なうとともにCPU部をLSI化することにより、体積でPSIの４分の１、実効速度でPSIの約３倍を目標としたパーソナルＡＩワークステーションを開発中である。

論理型プログラミング言語は、推論、データベース、パターンマッチング、バックトラックなど、従来の言語にはない多くの機能を備えており、人工知能の分野で主要なプログラミング言語の一つとして用いられるようになるであろう。本PSIの開発が論理型プログラミング言語の記述力の高さを多くのユーザに示し、その普及に貢献できれば幸いである。

最後に、PSIの開発に当たり、ご指導いただいた（財）新世代コンピュータ技術開発機構第四研究室の研究員各位に厚く感謝の意を表す。

参考文献

1. Warren. D. H. D.
   Implementing Prolog -Compiling Predicate Logic Program Vol. 1. 2
   D. A. I. Reseach Report. No. 39-40. 1977. Depertment of Artificial Intelligence. Univ. of Edinburgh
2. Warren. D. H. D.
   An Improved Prolog Implementation which Optimises Tail Recurtion
   D. A. I. Reseach Paper. No. 141. 1980. Depertment of Artificial Intelligence. Univ. of Edinburgh
3. Bowen. D. L.
   DECsystem-10 PROLOG USER'S MANUAL
   1981. Univ. of Edinburgh
4. Yokota. M. et al.
   A Microprogrammed Interpreter for the Personal Sequential Inference Machine.
   Proc. of the Int. Conference on FGCS'84 1984
5. Taki. K. et al.
   Hardware Design and Implementation of the Personal Sequential Inference Machine (PSI).
   Proc. of the Int. Conference on FGCS'84 1984
6. 奥乃博
   第３回Lispコンテスト及び第１回Prologコンケスト課題案
   情報処理学会　記号処理研究会資料28-4 1984

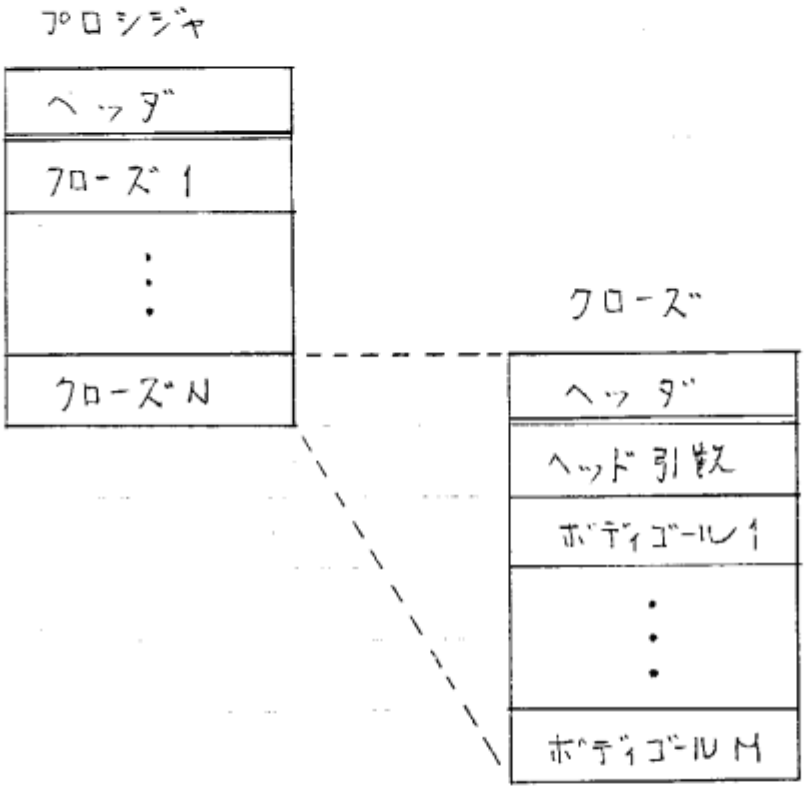

図-1　KLO 内部形式

KLO internal object form

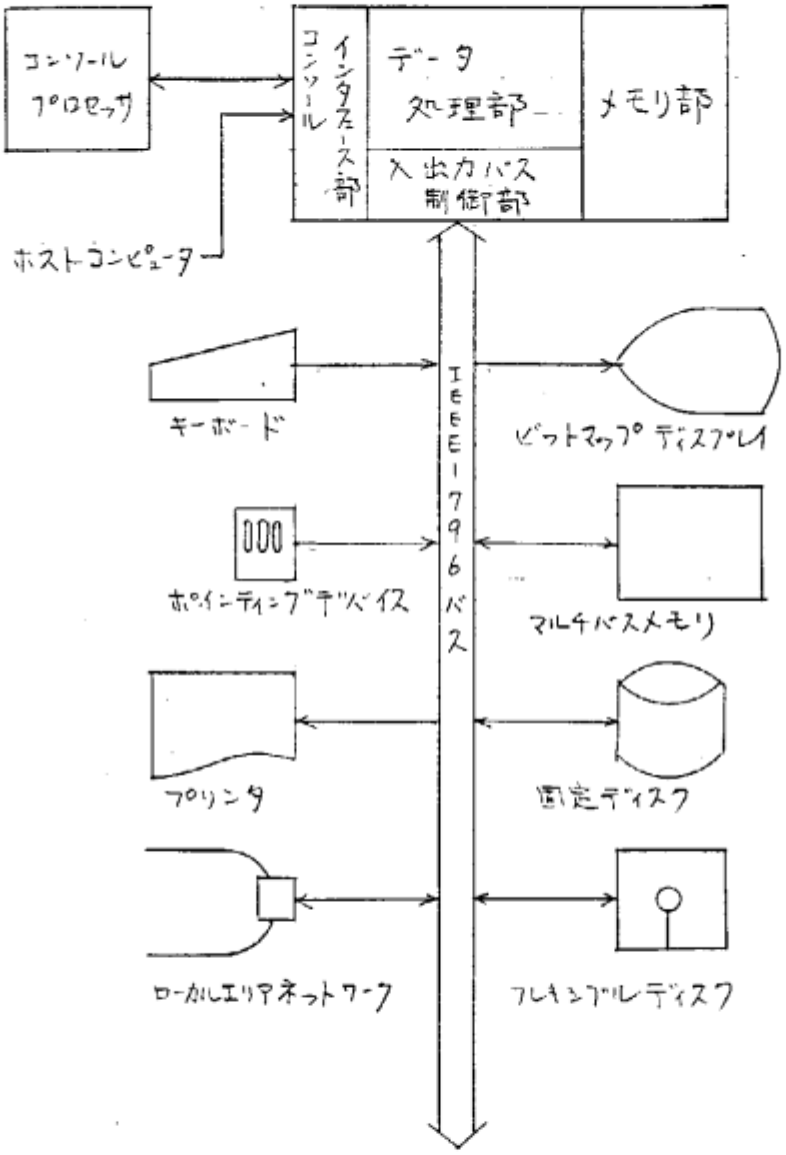

図-2 PSI システム構成

PSI System configuration

表-1 サンプルプログラムの実行時間
Execution times of sample programs

プログラム名	PSI(ms)	DEC(ms)	DEC/PSI
naive reverse (30要素)	13.6	9.5	0.70
quick sort (50要素)	15.2	14.6	0.96
tree traversing	51.7	61.1	1.18
lisp interpreter (tarai 3)	4024	4360	1.08
lisp interpreter (fibonacci 10)	369	407	1.09
lisp interpreter (naive reverse)	173	194	1.12
8 queens (全解)	1570	1580	1.01
reverse function	38.2	41.7	1.09
slow reverse (6要素)	99.4	89.0	0.90