シーケンシャルPROLOGマシンPEKの アーキテクチャとソフトウェアシステム

田村直之、和田耕一、松田秀雄、小畑正貴、
金田悠紀夫、前川禎男　（神戸大工学部）

1. はじめに

本稿では、我々が現在設計・開発中のPrologマシンPEKのアーキテクチャおよびソフトウェアシステムについて報告する。

PEKはPrologのプログラムを高速に実行するためのアーキテクチャの研究を目的としたProlog専用マシンである。

シーケンサおよびALUにはビットスライスLSI（AMD社のAm2909A、Am2903A）を使用し、インタプリタのマイクロプログラム化を行なう。また特に次の3点

(1) multi environment の実現
(2) unification 処理
(3) backtracking処理

を中心にハードウェア化を進め、Prologプログラムの高速実行を目指している。

2. 設計方針

PEKはPrologのプログラムを高速に実行するためのアーキテクチャの研究を目的としたProlog専用マシンである。マシンの規模は研究室レベルで実現できる程度におさえ、そのかわりに早期完成をめざすことにした。Prologの言語仕様としては、Edinburgh 版と同程度のものを考えている[1]。その他の点については以下のような基本方針をとった。

- マイクロプログラム制御方式
- シーケンシャル実行
- インタプリタ ＋ コンパイラ
- Structure Sharing

Prologコンパイラは最終的にはマイクロコードにまで翻訳する予定である。

Structure Sharing か Copying かについては各種の議論が報告されているが[2]、PEKではStructure Sharing方式を採用し、オーバーヘッドが予想される部分については専用ハードウェアにより高速化することにした。

以上のような基本方針のもとで、ハードウェア化が有効だと思われる次の3点を中心に設計をすすめた。

- multi environmentの実現
- unification 処理
- backtracking処理

2.1 multi environment の実現

- Edinburgh 版Prologでのローカルスタックに相当するメモリを用意し、ベースレジスタ +0 ～+255 の範囲を1マイクロ命令でアクセスできるようにする。

2.2 unification 処理

- データバスを14（frame 部）＋20（term部）の計34ビット巾とし、molecule単位でのデータ処理を可能にする。なおterm部は4ビットのtag部と16ビットのvalue部から成る。
- unification 時に2つのstructure（skelton）を並列して読みこめるように、アドレスレジスタとデータレジスタを2つづつ用意する。また、連続データの高速readを可能にするため、データレジスタからデータを読みこむと自動的に次アドレスのデータreadを開始するようにしておく。
- moleculeを格納するための2つのレジスタおよびマッチング用の回路を

設ける。マッチング回路は2つのmoleculeのterm部の比較結果に応じて16通りのmulti way jumpを行なうためのものである。

- 変数セルのアドレス計算を自動的に行なうためのハードウェアを付加する。すなわち2つのmolecule格納用レジスタのどちらか一方のframe部14ビットとterm部の下位8ビット（変数のインデックス値）の合計をポインタとして、グローバルスタックをアクセスできるようにする。どちらをポインタとして選択するかはマイクロ命令中の1ビットで指定する。

2.3 backtracking処理

- トレイル作業を自動化する。すなわちグローバルスタック中の変数セルへ値を書きこんだとき、その変数セルへのポインタ値をトレイルスタックへ自動的にpushしておく。
- アンドゥ作業を自動化する。アンドゥ用のサブシーケンサを用意し、メインのシーケンサの動作と並列して、アンドゥ処理を行なえるようにする。

3. 全体構成

システムの全体構成を図1に示す。

PEKはホストプロセッサMC68000と共有メモリおよびコマンドレジスタを介して結合されている。ホストプロセッサはPEKの立上げを行ない、実行時には入出力をサポートする。またすべてのソフトウェアはホストプロセッサ上で開発する予定である。Prologプログラムの読みこみなども、できるかぎりホストプロセッサ上で行ない、ゴール文の実行時のみPEK側に制御が移るようにする。

Z80はホストプロセッサの入出力装置としての役割りをはたす。

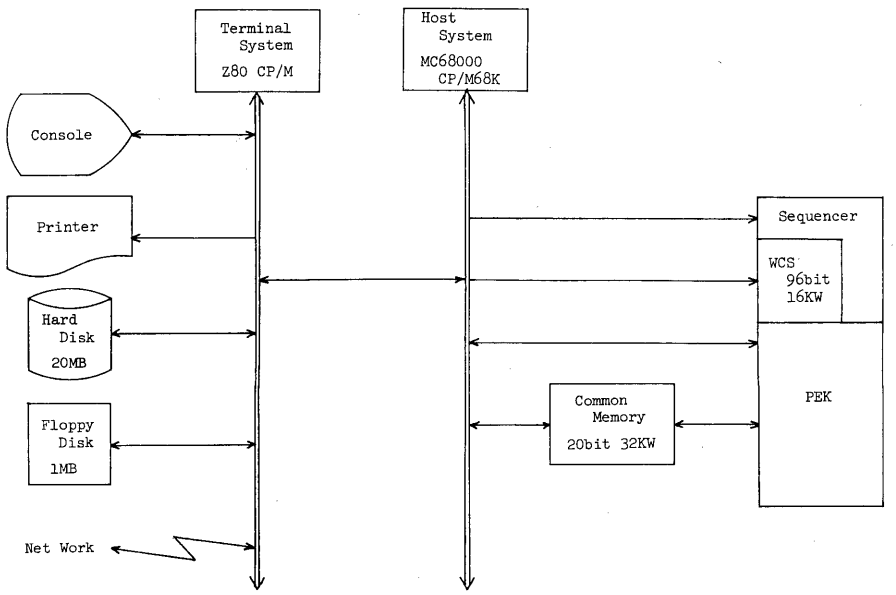
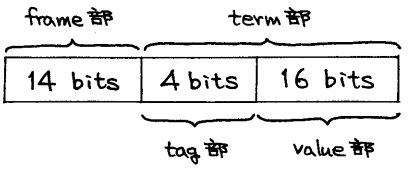


図1 システムの全体構成

4. PEKのハードウェア

PEKのハードウェア構成を図2に示す。

PEKではシーケンサとALUには市販のビットスライスLSI、AMD(Advanced Micro Devices)社のAm2909AとAm2903Aを採用した[3]。WCSは96ビット×16K語でアクセスタイムは55ナノ秒である。内部バスはRバス、Sバス、Yバスの3バス構成であり、すべて34ビット巾のデータを転送できる。これらのデータは14ビットのframe部と20ビットのterm部にわかれており、またterm部は4ビットのtag部と16ビットのvalue部から成っている。



ALUも同様に3つのフィールドにわかれており、演算結果のフラグやレジスタファイルへの書きこみはフィールドごとに独立している。

以下では、各ハードウェアモジュールの説明をおこなう。

4.1 CCU

シーケンサにはAMD社の4ビットスライスAm2909Aを4個使用している。WCSは96ビット×16K語、アクセスタイムは55ナノ秒である。また、マイクロ命令の実行サイクルは120～400ナノ秒となっている。

4.2 ALU

ALUにも同じくAMD社の4ビットスライスAm2903Aを9個使用している。これらは3つのフィールド、すなわち16ビットのframe部(上位2ビットは外部と接続されていない)および4ビットのtag部と16ビットのvalue部にわかれており、独立にフラグがセットされる。レジスタへの書きこみも独立して指定できる。また、レジスタファイルは16語を外付けして計32語に拡張してある。

4.3 内部バス

Rバス、Sバス、Yバスの3バス構成であり、すべて34ビット巾でmolecule単位のデータ転送を可能にしている。

4.4 バイパス

Rバイパス、Sバイパスの2つがあり、データをALUを通さずにYバスへ高速転送できる。

4.5 シフタ

左シフタと右シフタの2つがある。それぞれ、term部(下位20ビット)のデータをframe部(上位14ビット)へ、frame部のデータをterm部へシフトするのに用いる。

4.6 コマンドレジスタ

入力用(ICR)と出力用(OCR)の2つがあり、ホストプロセッサとのコマンドのやりとりに使用する。

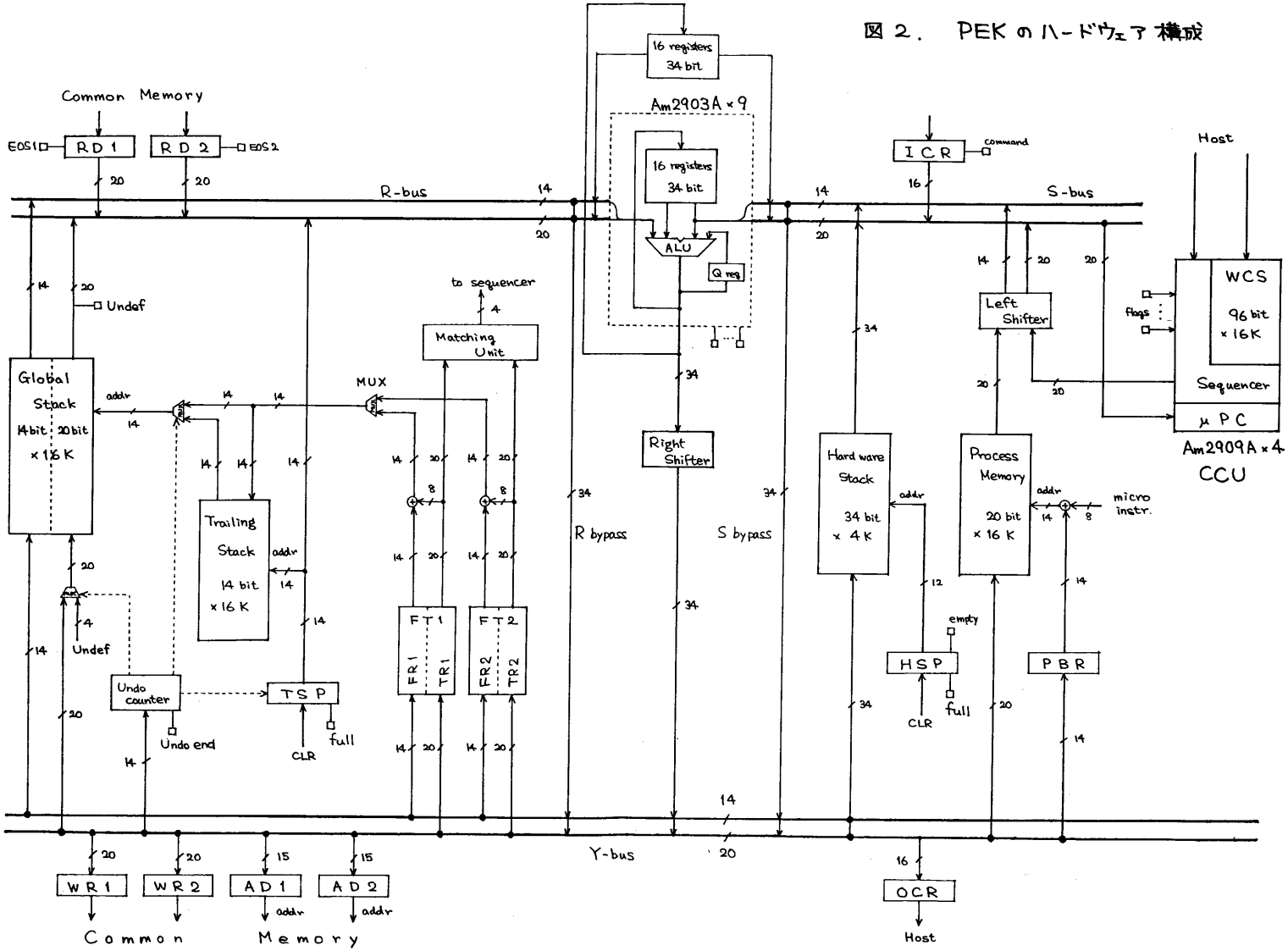
4.7 共有メモリ

ホストプロセッサとの共有メモリ(20ビット×32K語)で、構造体データ(skelton)の格納やホストプロセッサとの通信用に用いる。PEKからは2つのアドレスレジスタAD1、AD2および読みこみレジスタRD1、RD2、書きこみレジスタWR1、WR2を通してアクセスする。連続したアドレスのデータの読みこみを高速にできるように、RD1あるいはRD2からデータを読みこむと自動的にAD1あるいはAD2をインクリメントして次アドレスのデータの読みこみを開始する。

4.8 プロセスメモリ

20ビット×16K語のメモリであり、プロセスの管理情報およびローカル変数の格納に用いる。PBRという専用のベースレジスタを持っており、

図２．　PEK のハードウェア構成



PBR +0～+255の範囲を1マイクロ命令でアクセスできる。

4.9 ハードウェアスタック

34ビット×4K語のスタックメモリであり、ユニフィケーション処理などで局所的に使用する。

4.10 マッチング回路

2つのmolecule格納用レジスタFT1、FT2のterm部(TR1とTR2)中のtag部とvalue部の比較結果により16通りのmulti way jumpを行うための回路である。

4.11 グローバルスタック

34ビット×16K語のメモリであり、グローバル変数の値の格納に用いる。FT1のframe部(FR1)14ビットとterm部(TR1)の下位8ビット(変数のインデックス値)との合計、あるいはFT2でのFR2とTR2の下位8ビットの合計のどちらかをアドレスとしてアクセスされる。どちらを選択するかはマイクロ命令中のMUXのフィールドで自由に指定できる。また、現在選択している変数セルの内容がundefかどうか示すためのフラグが用意されている。

4.12 トレイルスタック

14ビット×16K語のスタックメモリで、代入のあったグローバル変数のセルアドレスを保存しておくために用いる。この作業はハードウェアにより自動的に行われる。すなわち、グローバルスタックへの書きこみ時には、そのセルアドレスが自動的にトレイルスタックへpushされる。もちろん、トレイルスタックへのpushなしにグローバルスタックへ書きこむことも可能である(マイクロ命令中での指定による)。

4.13 アンドゥ回路

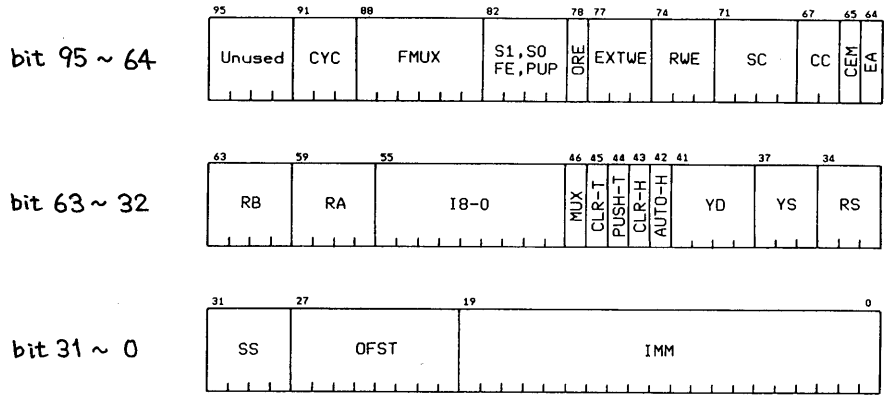
アンドゥ動作、すなわちグローバル変数の値をundef値に戻す動作を、指定された回数だけ自動的に繰り返し行なうための回路である。これはundo counterへ回数を書きこむことにより起動される。あとはメインのシーケンサの動作とは完全に並行に、サブシーケンサがトレイルスタックからpopしたグローバルスタックのアドレスにundef値を書きこむ動作を繰り返す。アンドゥ動作の終了はフラグにより判断できる。

5. マイクロ命令

図3にマイクロ命令のフォーマットと各フィールドの名称を示す。

WCSは96ビット×16K語であるが、最上位の4ビットはハードウェアのデバッグあるいは評価用に使用するので、実際の命令部分は92ビットである。

ビット91～78の14ビットはシーケンサ関係のフィールドであり、特にビット78のOREがmulti way jump用のものである。ビット77～47の31ビットはALU関係のフィールドとなっている。ビット46のMUXは2つのmolecule格納用レジスタFT1、FT2のどちらによるものをグローバルスタックのセルアドレスとして使うかを指定するためのフィールドである。ビット45～42の4ビットはそれぞれトレイルスタックおよびハードウェアスタックのクリアあるいは自動push/popモードへの切換用のフィールドである。ビット41～28は3つのバスのソースあるいはデスティネーションを示すためのものであるが、Yバスのソース指定にはバイパスのコントロールや右シフタのコントロールも含まれている。また、Sバスのソース指定についても左シフタのコントロールが含まれる。ビット27～20の8ビットはプロセスメモリのベースレジスタPBRに対するオフセット(+0～+255)用のフィールド、ビット19～0の20ビットは左シフタを通してSバスに出すイミディエイト値を示すためのものである。



91 - 89	3	CYC	Cycle Length Control
88 - 83	6	FMUX	Flag Multiplexer Control
82 - 79	4	S1,S0,FE,PUP	Sequencer (Am2909A) Control
78	1	ORE	Or Enable Address Source
77 - 75	3	EXTWE	External Register Write Enable
74 - 72	3	RWE	Register Write Enable
71 - 68	4	SC	Shift Control
67 - 66	2	CC	Carry Control
65	1	CEM	
64	1	EA	ALU (Am2903A) EA
63 - 60	4	RB	ALU Register B
59 - 56	4	RA	ALU Register A
55 - 47	9	I8-0	ALU Function
46	1	MUX	Multiplex Global Stack Address Source
45	1	CLR-T	Clear Trailing Stack
44	1	PUSH-T	Auto Push Trailing Stack
43	1	CLR-H	Clear Hardware Stack
42	1	AUTO-H	Auto Push/Pop Hardware Stack
41 - 38	4	YD	Y-bus Destination
37 - 35	3	YS	Y-bus Source
34 - 32	3	RS	R-bus Source
31 - 28	4	SS	S-bus Source
27 - 20	8	OFST	Offset to Process Base Register
19 - 0	20	IMM	Immediate Data

図3.
マイクロ命令の
フォーマットと
各フィールドの
名称

6. PEKのソフトウェア

PEKで実現するPrologの言語仕様としては、Edinburgh 版Prologと同程度のものを考えているが、詳細については現在設計中である。ただし当初はグローバルスタックのガベージコレクションや、テイルリカージョン処理、コンパイラなどはインプリメントせず、できるだけ単純な構成の処理系を作成する予定である。また、述語の定義処理やパージングなどはできるかぎりホストプロセッサ側で実行し、ゴール文が入力された時のみPEK側に制御を移すことにする。これはPEK上でのソフトウェア量をへらすためである。

6.1 データ構造

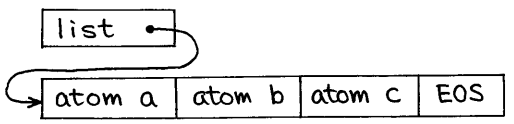
データは4ビットのtag部と16ビットのvalue部から成る。現在考えているtagは以下の9種である。

- integer
- literal atom
- undef
- global variable
- local variable
- void variable
- structure (term)
- structure (list)
- end of structure

Edinburgh 版では、listはconsをfunctorとするtermで表現されている

が、ＰＥＫではlistを別扱いし、メモリサイズの縮小および構造体へのアクセス回数の減少をはかる。すなわち、listを可変長のtermとみなし、listの終端にはＥＯＳ（end of structure）というtagのついたセルを１つ余分に付け加える。

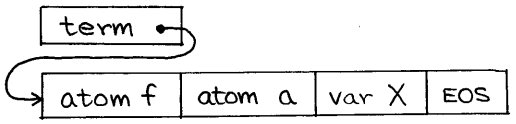
[a, b | c]



ＥＯＳかどうかは共有メモリのリードレジスタ（RD1、RD2）から実際に値を読み出さなくてもフラグ（EOS1、EOS2）で判断できるようになっている。

通常のtermについてもlistの場合と同様の構造とし、可変長にする。

f(a | X)



7. おわりに

本稿では、PrologマシンＰＥＫのアーキテクチャおよびソフトウェアシステムについて述べた。

ＰＥＫはPrologの高速実行に不可欠なunificationあるいはbacktrackingのハードウェアを備えている。またmulti environment用のプロセスメモリも、ソフトウェアの軽減および実行の高速化に役立つと思われる。

本マシンは、現在ハードウェアの製作中であり、ソフトウェアについても設計・開発を進めている。

参考文献

[1] D.H.D Warren : Implementing Prolog — compiling predicate logic programs. D.A.I Research Report No. 39, No. 40, 1977.

[2] 島津秀雄、小長谷明彦、中崎良成、梅村護：Shape Upにおけるユピイ方式とシェアリング方式の考察. 情報処理学会第26回全国大会 6D-9, 1983.

[3] The Am2900 Family Data Book. Advanced Micro Devices, Inc.

[4] 金田悠紀夫、和田耕一、田村直之、小畑正貴、松田秀雄、有尾隆一、小林久和、前川禎男：Prologマシンｐｅｋの全体構成. 情報処理学会第27回全国大会. 4P-6, 1983.（発表予定）

[5] 和田耕一、田村直之、金田悠紀夫、小畑正貴、松田秀雄、有尾隆一、小林久和、前川禎男：Prologマシンｐｅｋのハードウェア構成. 情報処理学会第27回全国大会. 4P-7, 1983.（発表予定）

[6] 田村直之、和田耕一、金田悠紀夫、小畑正貴、松田秀雄、有尾隆一、小林久和、前川禎男：Prologマシンｐｅｋのソフトウェア構成. 情報処理学会第27回全国大会. 4P-8, 1983.（発表予定）