関数型言語向き計算機アーキテクチャの一方式†

飯田三郎†† 楠菊信††
下村和弘††* 木全俊秀††**

近年，種々の計算機アーキテクチャの研究が，並列処理・分散処理の観点からあるいはプログラミング言語の観点から行われている．本論文ではプログラミング言語 Lisp を対象とした関数型言語向き計算機アーキテクチャの一方式を提案し，その実験結果について報告する．このアーキテクチャは，(1)関数に適用する引数の並列評価機構，(2)データ駆動方式による引数の関数への適用，(3)条件式の逐次評価機構，に特徴づけられる．ハードウェアは，プログラム制御方式によらずに，ワイヤード・ロジックにより構成された演算モジュール群を二つのバスに結合し構成されている．実験の結果，計算機システムの代表的なベンチマーク・プログラムである tarai [8; 4; 0] 関数を 376 ms という相当の高速度で計算することを確認した．

1. まえがき

現代の科学技術の発展の原動力となっているノイマン型計算機は，計算機アーキテクチャの中核に位置しているが，その計算機構はプログラム・カウンタによる逐次制御方式に特徴づけられ，計算速度はハードウェア素子により定められる．したがってハードウェア素子の速度を越えた高速性を求めるためには，何らかの並列処理技術の導入が不可欠である．また，ノイマン型計算機の他の特徴である汎用性は，巨大なシステム・ソフトウェアを必要としソフトウェア危機の一要因になっているとも考えられる．一方，近年の計算機システムにおけるハードウェア・コストに対するソフトウェア・コストの著しい増加は，生産性の高いソフトウェア・システムを求める傾向となり，ひいてはプログラミング言語の観点から計算機アーキテクチャを模索する研究へと発展しつつある．

並列処理性を自然な形で反映するアーキテクチャとしてデータフロー・モデル[1]あるいはリダクション・モデル[2]による非ノイマン型計算機が提案され，長谷川ら[3]はデータフロー・モデルの概念をリスト処理に適用しても多大な並列性の存在することを示した．また，小長谷ら[4]は並列評価・遅延評価可能な Lisp を対象としたリダクション計算機を提案している．この論文では Lisp を対象とした関数型言語向き計算機アーキテクチャの一方式を提案し，本方式に基づいたハードウェアの製作及びこれによる実験結果について報告する．

このアーキテクチャは，ソース・プログラムを機械命令列に変換した後，この機械命令列を実行するコンパイル方式に基づき，(1)関数に適用する引数の並列評価，(2)データ駆動方式による引数の関数への適用，(3)条件式の逐次評価，という特徴を有している．このことにより，データフロー・モデルの適用により生ずる不要な計算が除去される．また，小長谷らは，条件式をリデュース・オペレータにより評価しているが，ここでは条件式は，機械命令により逐次に評価される．

ハードウェアの構成は単純であり，プログラム制御方式によらずに，ワイヤード・ロジックによる演算モジュール群を二つのバスに結合することによって構成されている．したがって，ハードウェアレベルにおいても非プログラムカウンタ方式による並列処理が実現されている．

2. アーキテクチャの主要点

以下に本システムのアーキテクチャ上の主要点について述べる．

(1) 利用者の定義した関数は，コンパイラによりシステム固有の機械命令列に変換され，記憶装置内に格納される．これらの関数の評価は，引数を該当位置に埋め込んだ機械命令列を生成することにより行われる．この機械命令列は，関数本体のコピーに対応している．このコピー方式は，色付トークン方式[5]に比し，ハードウェアのリソース量の増加を招くが，制御

† A Computer Architecture for Functional Languages by Saburou Iida, Kikunobu Kusunoki, Kazuhiro Shimomura and Toshihide Kimata (Information and Computer Sciences, Faculty of Engineering, Toyohashi University of Technology).
†† 豊橋技術科学大学工学部情報工学系
* 現在 (株)日立製作所 Hitachi Ltd.
** 現在 中部電力(株) Chubu Electric Power Inc.

方式は単純化される.

(2) このコピーに際し,条件式を用いずに関数の合成により定義された関数(以後,合成関数と呼ぶ),例えば[6]

third [x]=car [cdr [cdr [x]]]

等の関数に対しては,定義された全体に対応するコピーが生成される.この機械命令列は,計算過程を木構造で表現した場合,葉を部分木に置き換えることに対応している.すなわち関数の評価は,部分木を生成することと等価であり,この部分木は計算過程の木に接続される.

(3) 関数に適用する引数は,最内側からかつ並列に評価されるが,この評価規則は部分木内に保持される.

(4) 引数が並列に評価された場合,引数の関数への適用は,すべての引数の評価がそろった時点で関数を起動するデータ駆動方式により行う.

(5) 条件式を用いて定義された関数,例えば[6]

ff [x]=[atom [x]→x ; T→ff [car [x]]]

等の関数の評価に対しては,まず最初の条件式述部に対応する機械命令列のコピーが生成され,この述部の評価結果である真偽に応じ,真の場合は対応する値部のコピーが,偽の場合は次の述部に対応するコピーが生成される.すなわち,値部あるいは次の述部は逐次に評価される.この逐次評価を実現するためには,二つの機能が必要である.第1は,条件式を用いて定義された関数に適用する引数を保持すること,すなわち,条件式の評価環境を保持することである.第2は,述部及び値部を関数とみなし,これらの関数間の対応をとることである.この機能は,条件式内に現れるすべての述部に後述する二つの機械命令を付加することにより達成される.

(6) 再帰関数は,通常1次元的データ構造であるスタックあるいは連想リストを用いて評価されるが,ここでは引数の並列評価のため多進木の制御が必要であり,Bobrow スタック[7]を簡略化して用いる.

(7) 計算過程の木における葉は,基本演算により消滅し上位の節が葉となる.基本演算としては,リスト処理のための基本関数及び算術比較演算関数をハードウェアで用意する.ストリング処理等の基本演算は,ハードウェアの拡張により可能である.

(8) リスト構造は一次元アドレスの記憶装置内に実現し,そのポインタをリスト処理の際の引数または値とする.これにより,リスト処理に伴う不要セルの回収機能が必要となる.このセルの回収は,計算の進行と並列に参照カウント法のアルゴリズム[8]に基づく並列ガーベジコレクタにより行う.

(9) ハードウェアについては,上述の関数評価のための種々の機能をハードウェア演算のレベルまで分割し,この演算モジュール群を二つのバスに結合する.バスがシステムの隘路となることが懸念されるが,構成は非常に簡素化される.演算モジュール群は,プログラム制御によらないワイヤード・ロジックで構成し,数段のパイプライン化を行いハードウェアレベルにおいても,非プログラムカウンタ方式による並列処理を実現する.

(10) システムにはフロント・エンド・プロセッサを接続し,この上のコンパイラにより機械命令を生成し,システム内にロードする.また,このフロント・エンド・プロセッサは Lisp における入出力関数の機能も果たす.

3. 機械命令

コンパイラにより生成される1機械命令は,次に述べる10のフィールド OP, OPR, P, OPC, F, U, V, S, D, L から構成される.一つのフィールドはそのフィールドの意味により,1ビットから数10ビットのビット長を有する.リスト処理関数 equal の場合の機械命令列の例を**図1**に示す.図中の＊印の行は注釈行である.

(1) 合成関数に対しては,合成回数+1の機械命令列が生成される.

(2) フィールド OP (OPeration code) は,関数名をコード化したものであり,関数名ごとに一意のコードがコンパイラにより付与される.

(3) フィールド OPR (OPeRand) は,OP で定められた関数に適用する引数または関数の評価結果である値を保持する.関数によりこのフィールドの個数は不定である.図1において OPR の # 印は,関数に引数を適用する際の実引数の埋め込み位置を示す.

(4) 関数が条件式を用いて定義された場合は,条件式の述部,値部ごとに機械命令列が生成されるが,述部の場合は特別に二つの機械命令が先頭に付加される.

(i) 一つはフィールド P (Predicate) に表示を有する機械命令である.この機械命令は,OP には関数名に付与したコードと同じコードを,OPR には条件式を用いて定義された関数に適用する引数を保持

```
(DEFUN EQUAL (X Y) (COND ((ATOM X) (EQ X Y))
                         ((ATOM Y) NIL)
                         ((EQUAL (CAR X) (CAR Y))
                          (EQUAL (CDR X) (CDR Y)))
                         (T NIL)))
```

P	OP	OPC	U	V	S	D	L	F	OPR-0	OPR-1
*(ATOM X)										
1	EQUAL	0	1	0	0				#OPR-0	#OPR-1
0	CONDX		1	0	1	1	F			
0	ATOM		0	0	1	0	OPR-0		#OPR-0	
	GC								#OPR-0(+1)	
*(EQ X Y)										
0	EQ		0	0	0				#OPR-0	#OPR-1
*(ATOM Y)										
1	EQUAL	2	1	0	0				#OPR-0	#OPR-1
0	CONDX		1	0	1	1	F			
0	ATOM		0	0	1	0	OPR-0		#OPR-1	
	GC								#OPR-1(+1)	
*NIL										
0	NOP		0	1	0				NIL	
	GC								#OPR-0(-1)	#OPR-1(-1)
*(EQUAL (CAR X) (CAR Y))										
1	EQUAL	4	1	0	0				#OPR-0	#OPR-1
0	CONDX		1	0	1	1	F			
0	EQUAL	0	2	0	1	1	OPR-0			
0	CAR		0	0	1	0	OPR-0		#OPR-0	
0	CAR		0	0	1	0	OPR-1		#OPR-1	
	GC								#OPR-0(+1)	#OPR-1(+1)
*(EQUAL (CDR X) (CDR Y))										
0	EQUAL	0	2	0	0					
0	CDR		0	0	1	0	OPR-0		#OPR-0	
0	CDR		0	0	1	0	OPR-1		#OPR-1	
*T/NIL										
0	NOP		0	1	0				NIL	
	GC								#OPR-0(-1)	#OPR-1(-1)

図 1 機械命令列
Fig. 1 Sequence of machine instructions.

し，条件式の評価環境を規定する．また，フィールドOPC (OPeration code for Conditional expression) には，0から始まる偶数の番号を順次割り当て，述部及び値部の関数としての対応関係を規定する．

(ii) 他は，OP をあらかじめ定められた関数名CONDX とした機械命令であり．条件式述部の評価結果である真偽に対応した1ビットの情報を生成する．

条件式内の条件式に対しては，コンパイラは未使用のコードを OP に割当て，上述の規則を再度適用する．

(5) フィールド F (Flag) は，条件式述部の評価結果である真偽に対応する1ビットの情報，あるいは数値演算におけるオーバフロー等の情報を機械命令間で受け渡すために用いられる．

(6) フィールド U (Undefined number) は，この機械命令で使用するOPR 及びFの未評価数を示す．

(7) フィールド V (Value) は，この機械命令の OPR−0 に関数の評価結果である値を保持することを示す．

(8) フィールド S (Subtree) は，この機械命令が関数本体のコピーにより生成された部分木であることを示し，次に述べるフィールドD及びLが部分木の構造を保持していることを示す．

(9) フィールド D (Destination) は，この機械命令の実行後，評価結果である値を引数またはフラッグとして使用する後方の機械命令を指すポインタであり，(自命令の位置)−(後方命令の位置)−(自命令と後方命令間で U=0 の命令数) で定められる．フィールド L (Location) は，値を結合する OPR の位置またはFを示す．

(10) OP が GC である機械命令は，後述するガーベジコレクタ用の命令であり，関数本体のコピーによる引数の埋め込み数の増減が OPR の括弧内で示される．

図1において，第4番目の条件式述部Tを評価せず，対応する値部を評価する最適化が行われている．

4. ハードウェア構成

ハードウェア構成を**図2**に示す．構成は，これまでに述べた関数評価のための種々の機能をハードウェア演算のレベルにまで分割し，この演算モジュール群を二つのバス，A-バス及び D-バスに接続する方式に基づいている．A-バス及び D-バスのビット幅はそれぞれ128ビットであり，機械命令はパケットとしてこのバスを用いて転送される．図中の矢印はパケットの転送方向を示す．パケットの形式は，**図3**に示すとおり機械命令の形式に準じるが，ハードウェアの規模を抑えるため不定数のフィールド OPR の個数は3までとし，それを越えた場合は，リストを構成するセル領域を用い引数の受け渡しを行う．パケットがフィールドCNTRL (CoNTRoL) に表示を有する場合，このパケ

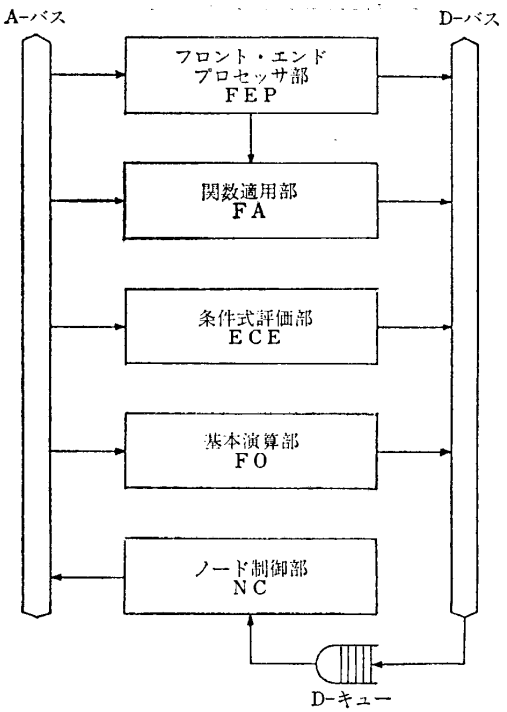


図 2 ハードウェア構成
Fig. 2 Configuration of the hardware.

ットはハードウェアの初期化・診断等の制御パケットであることを示すが，説明は省略する．演算モジュール群は，関数適用部 FA (Function Applier)，条件式評価部 ECE (Evaluater of Conditional Expression)，基本演算部 FO (Fundamental Operator) 及びノード制御部 NC (Node Controller) に大別される．フロント・エンド・プロセッサ部 FEP (Front End Processor) にはパソコンが充てられている．A-バスからパケットを入力する各モジュールは，パケットのフィールドP及び OP 上のデータをデコードし，これが自己の処理すべきパケットであれば自己モジュール内に取り込む．その後，取り込んだパケットのデータをもとに，指示された機能を実行し，D-バスの使用権を確保したのち，処理結果を D-バス上へパケットとして送出する．一方，NC は，D-キューよりパケットを入力し，処理結果を A-バスへ送出する．以下各部

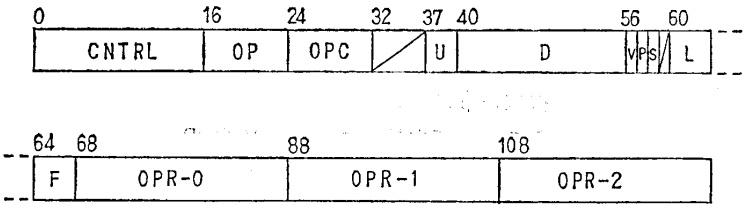


図 3 パケットの形式
Fig. 3 Format of a packet.

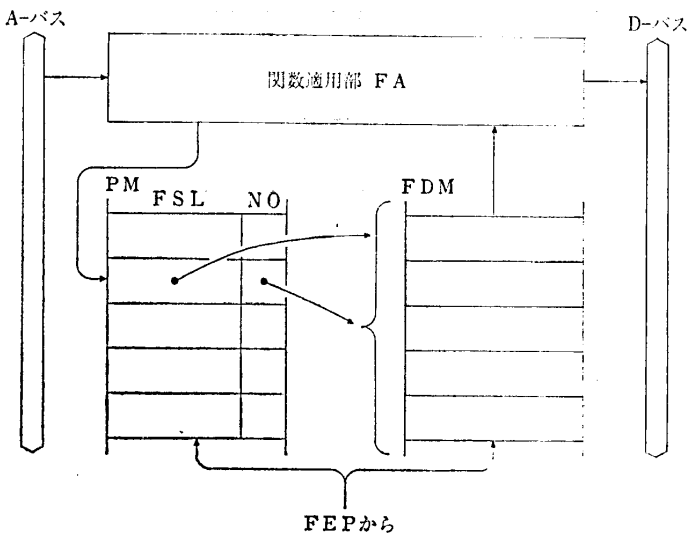


図 4 関数適用部
Fig. 4 Function applier.

の機能について述べる．

4.1 関数適用部

関数適用部 FA は，図4に示すように，本体及びポインタ・メモリ PM (Pointer Memory) と関数定義メモリ FDM (Function Definition Memory) より構成され，A-バスから関数の引数を受け取り，これを機械命令内のフィールド OPR に埋め込んだ後，D-バスへ送出する．PM，FDM ともに一次元アドレスのメモリである．

（1） コンパイラの生成した機械命令列は，計算に先立ち，関数ごとに FDM 上の連続した領域に，パケットの形式に合わせて格納される．この際，関数名に与えられたコードを OP とし，合成関数に対しては OPC＝0 とする．また，関数が条件式を用いて定義された場合は，条件式内の述部−1，値部−1，述部−2，値部−2,…に，0から始まる自然数を割り振り，これを OPC とする．この OP と OPC の連結を PM のアドレスとして，FDM 上の機械命令列の先頭格納位置 FSL (First Stored Location) 及び機械命令の数 NO (No. of instructions) を PM に記録する．

（2） FA は入力したパケットの OP と OPC を連結し，これを PM のアドレスとして内容を読み出し，FDM 上の機械命令列の格納位置を求める．

（3） 次に，FA は FDM から機械命令列を順次読み出し，D-バスへパケットとして送出する．この際，出力するパケットの先頭パケットのフィールドD及

びLには，入力したパケットのD及びLの内容をコピーする．すなわち，計算過程の木における葉を部分木に置き換える際，木の接続関係は部分木の根により保持される．また，出力するそれぞれのパケットのフィールドOPRには，入力したパケットのOPRを所定の位置に埋め込む．

（4） 出力するパケット数は，PM上に記録されている機械命令数により定められる．

4.2 条件式評価部

条件式評価部ECEは，二つのモジュールCONDX及びCONDにより構成され条件式の逐次評価を行う．

（1） CONDXは，条件式述部の評価結果を入力パケットのOPR−0から受け取り，真偽に応じ出力パケットのフィールドOPR−0に0（真の場合）または1（偽の場合）の値を設定する．

（2） CONDXの出力パケットのフィールドD及びLには，4.1節(3)で述べた理由により入力パケットのフィールドD及びLの内容をコピーするとともに，フィールドVに標示をたてる．また出力パケットのOPR−1には，入力パケットのOPR−0をコピーする．これは，後述するガーベジ・コレクタがセルへのポインタ数の増減を知るためである．

（3） CONDはA-バス上のパケットのフィールドPに標示がある場合，このパケットを入力する．このパケットのフィールドFには，CONDXがOPR−0に定めた条件式述部の評価結果が，後述のノード制御部により設定されている．またOPRには，条件式を評価する際に必要なすべての実引数が保持されている．CONDは入力パケットのOPC及びFを用いてOPC＋F＋1の演算を行い，これを出力パケットのフィールドOPCに設定する．これにより，条件式述部の真偽に応じ，対応する値部または次の述部が評価可能となる．

（4） CONDの出力するパケットのフィールドPには標示をたてず，フィールドOP及びOPRには，入力したパケットの対応するフィールドの内容をコピーする．

4.3 基本演算部

基本演算部FOの演算モジュール群は，リスト操作のための基本関数（CAR，CDR，CONS，ATOM及びEQ），加減演算（ADD，SUB）及び数値の大小比較演算（COMP）より構成されている（**図5**）．

（1） これらのモジュールはいずれも，入力したパケットのOPRを実引数として定められた演算を行った後，値を出力パケットのOPR−0に設定する．

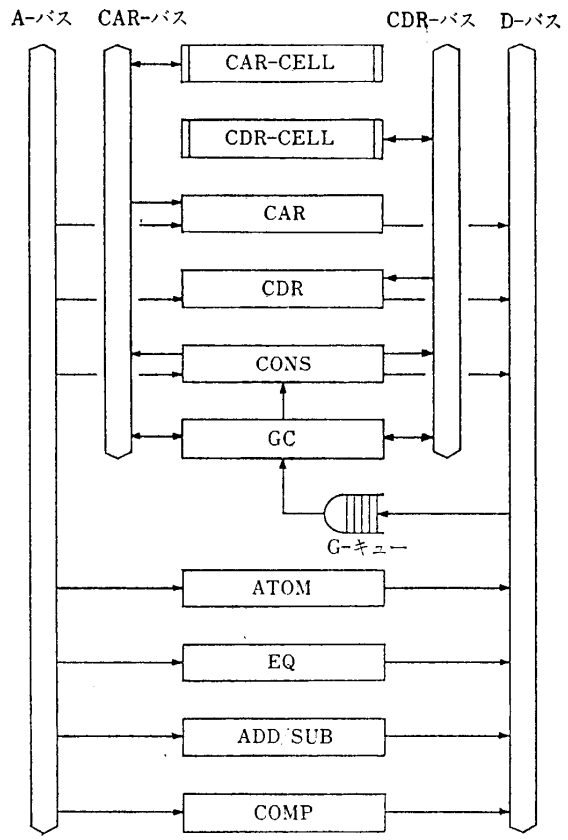


図5 基本演算部
Fig. 5 Fundamental operator.

（2） 出力パケットのフィールドD及びLには，4.1節(3)で述べた理由により入力パケットのフィールドD及びLの内容をコピーするとともに，フィールドVに標示をたてる．また4.2節(2)で述べた理由により，出力パケットのOPR−i (i=1, 2) には入力パケットのOPR−i (i=0, 1) をコピーする．

（3） リスト処理の際に生じる不要セルの回収は，ガーベジ・コレクタGC (Garbage Collector) により行われる．GCは，参照カウント法に基づき，参照カウンタの増減を行い，不要セルを回収するとともに基本関数CONSのセル使用要求に対しフリー・セルの使用許可を与える．参照カウンタの増減を行うために必要なすべての情報は，3章(10)，4.2節(2)及び4.3節(2)で述べたことにより，D-バス上のパケットから収集できる．GCは図5に示すように，G-キュー及び処理部から構成される．G-キューは，D-バス上に流れるパケット数と処理部の処理パケット数との整合をはかるバッファとして用いられ，D-バス上のパケットのフィールドOP及びOPRの内容が書き込

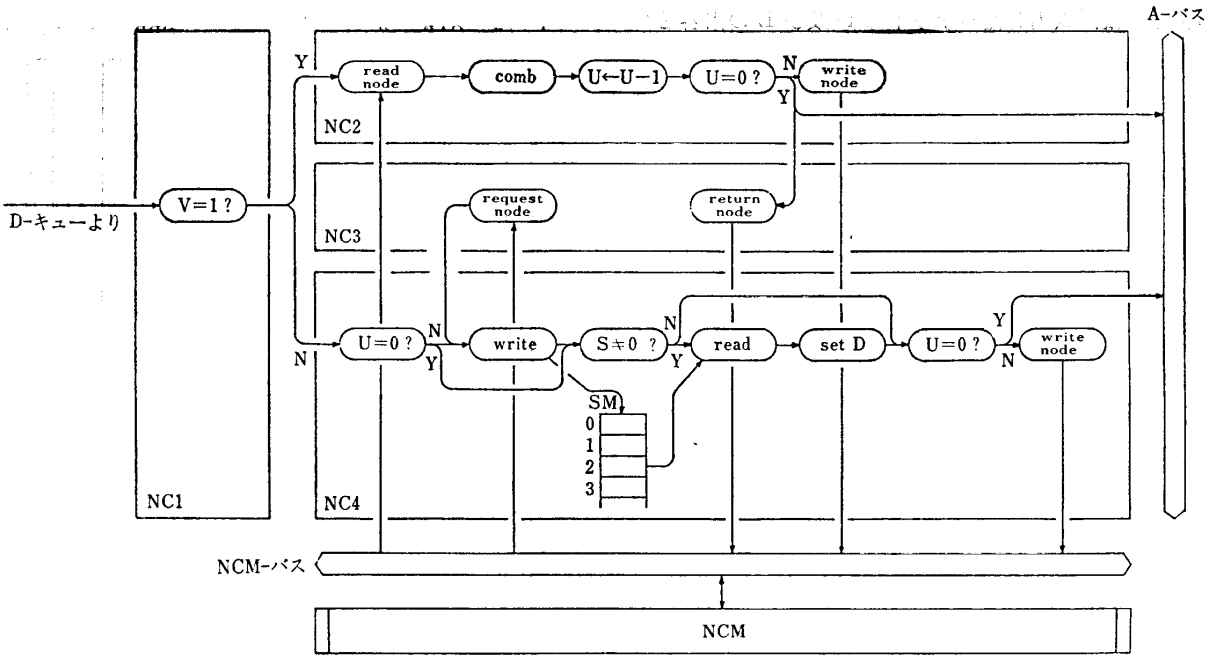


図 6 ノード制御部
Fig. 6 Node controller.

まれる．処理部は，G-キューから順次パケットを取り出し，参照カウント法のアルゴリズムに基づき，参照カウンタの増減を行い不要セルを回収する．

4.4 ノード制御部

D-バスに送出されたパケットは，送出順にノード制御部 NC で処理されるが，関数適用部では，入力した1機械命令に対し複数個の機械命令を生成するため，システム内に滞留するパケットを一時格納しておくバッファが必要となる．このため，D-バスとノード制御部の間にモジュールD-キューを設ける．

ノード制御部は，関数の評価過程において生成あるいは消滅するパケットを多進木の節に対応させ，この多進木を制御する．すなわち関数の評価は，根から開始し，節の伸縮を繰り返し，根に戻った時点で終了する．**図6**に示すように，ノード制御部は演算モジュール NC 1〜4 及び NCM (Node Control Memory) から構成される．

（1） NCM は一次元アドレスのメモリであり，1語長は1パケット長に対応している．計算開始以前には NCM 内のすべての語はフリーノードとして線型リスト化され，計算開始後のこのノードの管理は NC 3によって行われる．すなわち NC3 はノードの使用要求 (request node) に対しては，このリストの先頭のノードのアドレスを要求元へ通知し，ノード使用後の返却 (return node) に対しては，このノードをフリーノードとして線型リストに繋ぐ．

（2） NC1 は D-キューより取り出したパケットのフィールドVを調べ（図6中のV=1?），Vに標示がある場合，すなわちこのパケットの OPR−0 に関数の評価結果の値が保持されている場合，このパケットを NC2 へ転送し，他の場合は NC4 へ転送する．

（3） NC2 は，パケットのフィールドDを NCM のアドレスとしてノードを読み出し (read node)，パケットのフィールドLの指示に従い，読み出したノードの OPR またはFに値を格納する．これにより値と関数の引数との結合が行われる (comb)．この後，ノードのフィールドUの値を1減じ (U←U−1) U の値を調べる (U=0?)．U≠0 のノードは引数中に未評価引数の存在を示しているので，NCM 上の読み出したノードの位置へ再度書き込む (write node)．U=0 のノードはすべての引数の評価の完了を示しているので，ノードを NC3 へ返却 (return node) したのち，このノードをパケットとしてA-バスへ送出する．

（4） NC4 は主として，FA により生成された部分木を NCM 上の木へ接続することを行う．

（i） 先ずパケットのフィールドUの値を調べ (U=0?)，Uが0でない場合は，未評価の OPR またはFが存在することを示しているので，このパケットは NCM 上にノードとして一時格納する必要がある．

このため格納するノードを確保する (request node). このノードは，FA により生成された部分木の前方命令で参照されるので，作業用メモリ SM (Scratchpad Memory) の 0～$n-1$ 番地の内容を 1～n 番地へ移した後，このノードのアドレスを 0 番地へ書き込む (write).

(ii) フィールドSに標示を有するパケット (S≠0?) のフィールドDは，関数の評価結果である値を実引数とするパケットを指しているので，Dを SM 上のアドレスとして SM の内容を読み出し (read)，NCM 上の実アドレスを求め，これをパケットのD部に設定する (set D).

(iii) この後Uの値を調べ (U=0?)，U=0 のパケットは A-バスへ送出し，それ以外のパケットは(i)で確保したノードの位置へ書き込む (write node).

4.5 フロント・エンド・プロセッサ部

フロント・エンド・プロセッサ部 FEP は，計算開始のパケット，すなわちフィールド OP に計算の入口となる関数名のコードを，フィールド OPR にこの関数に与える引数を設定し，D-バスへ送出する．計算終了時には，A-バスから評価結果を受け取る．また FEP はプログラムをコンパイルし，機械命令列を FDM 上へ格納する．

5. 実　　験

これまでに述べた本システムの構成法を検討し，ノイマン型計算機に代表される命令の逐次実行方式と本方式を比較し，またこの方式の構成上の隘路を調べるため，システムを試作し実験を行った．

5.1 ハードウェア

バスはバス・アビタにより管理され，バス上のデータはハンド・シェイク方式により転送される．このバスに接続される演算モジュールは**図7**に示すように，データのバッファ，処理部及び制御部より構成される．モジュールの構成は比較的簡単であり，例えば CAR 演算モジュールの場合，入力パケットの OPR−0 をセル・メモリのアドレスとしてメモリの内容を読み出し，これを出力パケットの OPR−0 へ設定する．

使用素子として論理素子に TTL-IC を，メモリ素子に DRAM 及び SRAM を用いた．図2の演算モジュール群の使用 IC は，メモリを除き関数適用部約 400 個，条件式評価部約 100 個，基本演算部約 600 個，ノード制御部約 700 個である．バスによるパケットの転送には約 250 ns の時間を必要とし，また DRAM のメモリ・アクセス・タイムは約 1 μs である．

5.2 動 作 性 能

システムの動作を確認するため，ここではベンチマーク・プログラムとして関数型プログラムの典型的な例である tarai 関数[9]

```
tarai[x; y; z]
  =[x>y→tarai[tarai[x−1; y; z];
              tarai[y−1; z; x];
              tarai[z−1; x; y]];
    T→y]
```

を用いた動作性能について報告する．この関数は，簡明なアルゴリズムの中に2重再帰を含み，システムの動作がよく抽出されるものと考えられる．この関数の評価には，関数適用部，条件式評価部，基本演算部(減算及び比較のみ) 及びノード制御部が使われる．また，バスは A-バス，D-バス及び NCM-バスが使われる．tarai 関数の評価の中心は，3引数の並列評価であるが，本システムのハードウェア構成では，関数適用部及び基本演算部の減算器等がそれぞれ1台であるため並列性は低減される．**表1**に tarai [8;4;0]

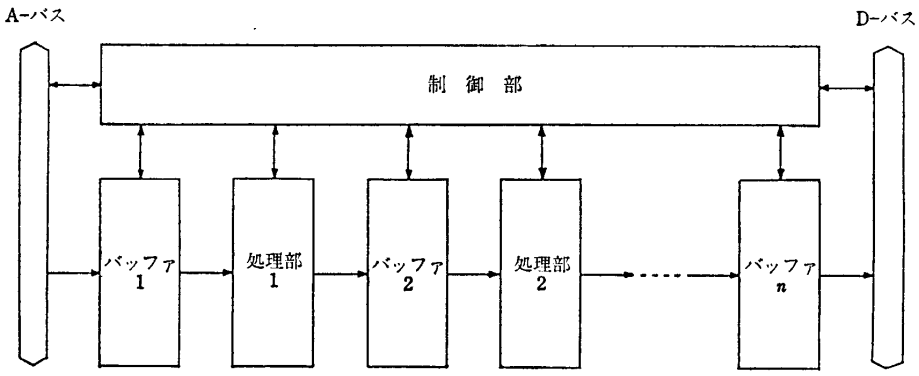


図7 モジュールの構成
Fig. 7 Configuration of a module.

表 1 tarai[8; 4; 0] の場合のリソースの使用状況
Table 1 Utilization of resources for tarai [8; 4; 0].

演算時間	376 ms
最大時のD-キューの深さ	484 パケット
A-バス・ビジー率	53%
D-バス・ビジー率	61%
NCM-バス・ビジー率	52%

に対するリソースの使用状況の測定結果を示す.

(1) tarai[8;4;0] の演算時間は 376 ms であり, これは大型汎用計算機上の Lisp コンパイラ・システムと同程度である[9]. 大型汎用計算機と本システムのバスの転送速度, メモリ・アクセス・タイム等を比較すると, この演算時間は, 本アーキテクチャの高速性を示していると考えられる.

(2) 演算中に D-キューに貯えられたパケットの最大時のパケット数は, 484パケットである. これは引数の並列評価により, パケット数が増殖されることを示している. パケット数は, tarai 関数の2重再帰の部分を呼び出したときに主として増殖される. したがってこの最大時のパケットがすべて2重再帰の部分で生成された機械命令と仮定しても, このシステムでは2重再帰の部分は7機械命令に相当するので, 484/7≃70 パケットは, 計算過程の葉に相当している. このことは, 並列処理により処理可能となる葉の莫大な増加を示唆している.

(3) A-バス, D-バス及び NCM-バスのビジー率は 50〜60% であり, バスはバランスがとれて動作していることを示している. さらにバスの動作からシステムの動作性能を抽出するため, A-バス, D-バス及び NCM-バス相互のビジーあるいはアイドル状態をを測定した. これらの状態の起きる時間の全演算時間に対する比率を**表 2** に示す. 表2より,

(i) A-バス及び D-バスがビジーで NCM-バスがアイドルである比率が最も高く 31.3% である.

(ii) 逆に NCM-バスがビジーで, A-バス及び D-バスがアイドルである比率が 20.9% と2番目に高い数値を示している.

(iii) 一方, A-バスあるいは D-バスが単独でビジーである比率は, それぞれ 2.9%, 4.8% と低い.

上記(i)〜(iii)のデータは, 動作性能上のハードウェア構成が, ノード制御部とそれ以外 (関数適用部, 条件式評価部, 基本演算部) とに分けられることを示唆している. 関数適用部, 条件式評価部及び基本演算部の中では, 関数適用部がメモリ・アクセスにより演算時間の大部分を占める. したがって, 動作性能上のハードウェア構成をノード制御部と関数適用部に大別すると, 上記(i)及び(ii)のデータは, パケットがノード制御部→関数適用部→ノード制御部→…という繰り返しで処理されていることを示している. この逐次的な動作は, ノード制御部及び関数適用部のメモリ・アクセスに起因している. したがって本システムでは, メモリ・アクセスの速度及び回数が動作性能を規定する重要な因子であると考えられる.

表 2 三つのバスのビジー(b)/アイドル(i)状態の比率
Table 2 Rate of busy (b)/idle (i) state of three buses.

状態			単位:%
A-バス	D-バス	NCM-バス	
b	b	b	12.8
b	b	i	31.3
b	i	b	6.4
b	i	i	2.9
i	b	b	12.3
i	b	i	4.8
i	i	b	20.9
i	i	i	8.6

6. むすび

本論文では, 関数型言語向き計算機アーキテクチャの一方式を提案した. このアーキテクチャは, 関数に適用する引数の並列評価, データ駆動方式による引数の関数への適用, 条件式の逐次評価という特徴を有している. また, この方式に基づくハードウェアを製作・実験することにより, このアーキテクチャの高速性, 並列性, ハードウェア・リソースの負荷分散等に関するいくつかのデータを提示した.

以下に本システムに関する今後の課題について述べる.

(1) 通常の Lisp システムは, インタプリタのもとで関数の評価を行うが, 本システムはコンパイラにより機械命令列を生成し, この機械命令列を直接実行することにより, 関数の評価を可能としている. 一方, 本システムは, (i)関数に適用する引数を並列に評価することにより生じる副作用を伴う関数 (rplaca, rplacd 等), 及び(ii)引数を動的に束縛することにより生じる FUNARG 問題を伴う関数, を評価する機能を有していない. 今後, (i) に関しては, この機能を代替する並列処理アルゴリズム及びその実用性について検討し, (ii) に関しては, この機能を取り入れ

る方向で，本システムのコンパイル方式の中での引数の環境保存法について検討する，予定である．

（2） ハードウェアの構成については，今後次の項目に対して性能評価によるデータに基づき検討を加えるとともに，順次改良を行う予定である．(i)現システムでは，関数適用部あるいは基本演算部の演算器等は1台で構成されている．これら同一機能の演算モジュールの複数台設置を，ハードウェア・コストと演算時間とのトレード・オフ上の問題を考慮しつつ，検討する予定である．(ii)図6に示したようにノード制御部のパケット転送路は，基本的には2本であり，またメモリへのアクセスも頻繁に行われる．パケット転送路の本数の拡張，並びにメモリ・アクセス回数の減少，を図る方向でノード制御部における多進木の制御方法を検討することが望まれる．(iii)現在，関数定義メモリ上の機械命令は，パケットの形式に合わせて格納されている．この機械命令の格納法を検討し，関数定義部のメモリ・アクセス回数の減少を図ることが望まれる．(iv) 本システムは，A-バス及び D-バスを主とするバスの構成法を採用した．しかし，関数定義部で生成されたパケットは，ノード制御部の NC 4 で処理されるので，ノード制御部と NC 4 を直結することにより，システム全体としてのパケットの流れを分散させ性能向上を図り得るものと考えられる．

今回は紙数の制約上，本アーキテクチャを特徴づける動作特性は，代表的な基礎データにとどめた．実測及びシミュレーションによる網羅的な評価データは，別途一括報告する予定である．

謝辞 本研究に関し有益なご助言，ご激励をいただいた豊橋技術科学大学本多波雄教授，北橋忠宏教授に深謝する．

参 考 文 献

1) Dennis, J. B., Fosseen, J. B. and Linderman, J.: Data Flow Schemas, *Lecture Notes in Computer Science*, Vol. 5, pp. 187-216, Springer, New York (1972).
2) Berkling, K. J.: Reduction Languages for Reduction Machines, Proc. 2nd Symp. on Computer Architecture, pp. 133-140 (1975).
3) 長谷川隆三，三上博英，雨宮真人：リスト処理向きデータフローマシンアーキテクチャの評価，電子通信学会論文誌，Vol. J 67-D, No. 9, pp. 957-964 (1984).
4) 小長谷明彦，北森鉄治，山本昌弘，内藤祥雄：リダクションマシン "ARM" の基本アーキテクチャ，信学技報，EC82-28, pp. 69-78 (1982).
5) Arvind and Gostelow, K. P.: The U-Interpreter, *IEEE Comput.*, Vol. 15, No. 2, pp. 42-50 (1982).
6) McCarthy, J. et al.: *LISP 1.5 Programmer's Manual*, pp. 5-6, MIT Press, Cambridge (1962).
7) Bobrow, D. G. and Wegbreit, B.: A Model and Stack Implementation of Mulitiple Environment, *CACM*, Vol. 16, No. 10, pp. 591-603 (1973).
8) Knuth, D. E.: *The Art of Computer Programming*, Vol. 1, p. 634, Addison-Wesley, Reading (1978).
9) 竹内郁雄：第2回 Lisp コンテスト，情報処理，Vol. 20, No. 3, pp. 192-199 (1979).

（昭和60年 3 月 11 日受付）
（昭和60年11月 21 日採録）

飯田 三郎

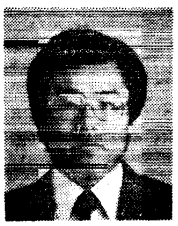
昭和41年東京大学理学部天文学科卒業．昭和46年名古屋大学大学院博士課程修了．理学博士．現在，豊橋技術科学大学情報工学系に勤務．計算機アーキテクチャの研究に従事．電子通信学会，物理学会各会員．

楠 菊信

昭和29年東京大学工学部電気工学科卒業．同年 NTT 電気通信研究所入所．以来，論理素子の研究，論理装置の設計，電子交換方式の開発，システム開発の研究などに従事．昭和55年から豊橋技術科学大学情報工学系教授．LSI 向き論理回路設計法，計算機アーキテクチャ，ネットワークアーキテクチャなどの研究に従事．工学博士．51年度電子通信学会論文賞受賞．著書「マイクロエレクトロニクス入門」，「通信情報ネットワーク工学」ほか．電子通信学会会員．

下村 和弘

昭和56年豊橋技術科学大学工学部情報工学科卒業．昭和58年同大学院修士課程修了．工学修士．現在，(株)日立製作所笠戸工場生産技術部に勤務．半導体製造装置の制御系の開発に従事．



木全 俊秀

昭和57年豊橋技術科学大学工学部情報工学科卒業．昭和59年同大学院修士課程修了．工学修士．現在，中部電力(株)平岡電力所に勤務．発電所自動制御装置の開発に従事．