

解説

応用指向計算機の構成技術*

飯塚 肇**

1. ま え が き

情報処理技術の発展につれて、計算機の応用分野は著しく拡大し、近年の計算機には単なる「加減乗除の数値計算を高速に行う機械」ではなく、非常に広い範囲の情報処理に効率よく適用できることが求められている。第3世代以来続いてきたいわゆるはん用計算機は大多数を占める平均的応用分野に対しては秀れた処理能力を有するが、新しいやや特殊な応用分野においても、常に効率よく利用され得るとは限らない。また、能力的にははん用計算機では不足な新しい応用分野も拡大しつつある。

一方、近年ハードウェアの価格は著しく低下したものの素子自身の速度は限界に近づきつつあるというのが一般的見方である。しからば、次の世代の計算機では安くなったハードウェアを多量に用いた並列処理によって、処理量を高める方式を基本に考えねばならないであろう。しかし、この並列処理では応用側の処理構造がマシン側の並列構造に一致していなければ効果が上がらないこともよく知られている。このハードウェア構造と問題構造の適合性は、何も並列処理に限らず、一般的に成立することであって、効率のよい処理、あるいは高い処理能力を達成するための極めて重要な要因である¹⁾。

応用指向計算機システムははん用形のような平均的適応度を得るような設計ではなく、特定の応用、あるいは類似の応用分野に対して、高い適応性、したがって、高い処理効率をあげることが意図して設計されたシステムである。

このような特定の応用への適応性を高めた応用指向計算機では、応用が明確に定義されていれば、その設計そのものはそんなに難しくないのであるが、実際には2つの大きな問題がある。一つはある応用に指向さ

せると他の応用への適応性が悪くなって、適用できる問題の範囲が著しく狭くなることであり、もう一つはその結果、同品種の生産数が少なくなって生じるコスト高である。

すなわち、たとえ、どれほど応用との適応性がよいといっても、一つの応用ごとに新しく全ハードウェアを設計するのであれば応用指向計算機の現実性はない。したがって、応用指向計算機がその現実的価値を認められるためには、最適とはいえなくても相当適応性のよい計算機を低コストで構成するための技術が確立されなければならない。

さて、これまでに開発されたこのような応用指向計算機の構成技術は技法的に次の3つのタイプに分類される。

- ① インタフェースの統一されたモジュールを組み合わせるもの。
- ② マイクロプログラムの(動的)変更によるもの。
- ③ 応用の特性を表わす情報をセットアップするものの。

次章からは、これらの各技法について、現状や問題を説明するが、各技法を論じる前に、ここで、応用への適応のレベルについて考察しておこう。

一般のはん用計算機であっても、特定の応用に対するソフトウェアをのせた状態では、ユーザから見る限り、その応用に指向した計算機になっているわけである。これに対し、いわゆる応用指向計算機はその適応化がよりハードウェアに近いレベルで行われるのが特徴であるから、応用指向計算機の構成技術を考えるために適応のレベルを整理するのは無意味なことではあるまい。

さて、表-1(次頁参照)は計算機の構造をレベル分けしたものである。応用指向計算機とはこの図のレベル6以下のハードウェアの処理構造において、応用への適応を効率的に行っている計算機であるということが出来る。しかしながら、レベル0~2のいわゆる部品レベルで特定の応用へ指向させる方法はその応用が

* Technologies for application oriented computer design by Hajime IIZUKA (Electronic Computer Division, Electrotechnical Laboratory)

** 電子技術総合研究所電子計算機部

表-1 計算機構造のレベルと適応化技法

レベル	名称	レベル	適応技法
0	部	電源, 回路	
1	品	ゲート, ロジック	
2	ハードウェア	集積回路 (SSI)	
3		レジスタ, ALU (ボード)	RT モジュール, PLA
4		マイクロプロセッサアーキテクチャ	レジデュアル制御, デスクリプタ, 語長アライメント
5		マイクロプロセッサ結合構造	PMS モジュール
6	ファームウェア	マイクロプログラム	マイクロプログラミング, ダイナミックチューニング
7	ソフトウェア	ターゲットシステム	
7.0		機械語	
7.5		目的向き中間言語	
8		ソフトウェアシステム	
9		ユーザ(応用プログラム)	目的向き高級言語

よほど大量に存在しなければ、コスト的に成立しない。したがって、実際上の応用適応技法はレベル 3~6 に対するものということになるが、具体的には表-1 の最右欄に示したようなものがある。これら各技法の特徴は次章以下に詳しく述べるが、全体的に言えばレベルが下るほど効率の点では好ましいが、動的変異性や、コストにおいては逆にレベルが高いほどすぐれている。したがって、これらの技法をいかにうまく組み合わせていくかが、応用指向計算機設計上の重要なポイントとなる。

2. モジュール構造

2.1 概 説

応用指向計算機のハードウェアによる実現を容易にする方法として、標準的機能をインタフェースの統一されたモジュールとして用意しておく方法がある。表-1 でもレベル 3 とレベル 5 でモジュール構造を用い得ることが示されているが、各論に入る前に、この方法の利害得失をまとめると次のようになる²⁾。

(1) モジュール構造の利点

- * システムの設計コストが小さくなる。
- * 設計時間が短い。
- * 後からの修正、拡張が容易。
- * 回路の電気的特性が表面に出ない。
- * システム構造を理解しやすい。
- * 故障位置の検出等、保守性がよい。

(2) モジュール構造の弱点

- * モジュール化のためのハードウェア・オーバーヘッドがある。
- * ハードウェアの利用効率が最大になるとは限ら

ず、最適システム設計ができない。

* モジュールのレベルをあげると低レベルでは容易な機能の実現が困難になる(透明性の減少)。

2.2 RT レベルモジュール

モジュールの機能レベルはその時点での半導体技術に大きく依存する。したがって、PMS レベルモジュールよりは低レベルの RT モジュールの方が早く開発された。現在でも、PMS レベルモジュールが研究段階であるのに対し、RT レベルモジュールは一応実用品が供給されている。

この RT レベルの RT とは Register Transfer の略であって、レジスタとそのデータの処理機能等を単位とするモジュールである。通常、データモジュール(レジスタ, ALU, ゲート等)と、制御モジュールとに分離され、前者は外部からの制御信号で動作し、命令を逐次実行する機能は含まれない。機能のレベルはほぼ MSI から小型 LSI で一般の集積回路としてモジュールセットとしてでなく、市販されているもの(例えば 4 ビット ALU SN 74181)もある。

2.2.1 マクロモジュール

さて、最初の RT レベルモジュールは 1967 年に Washington 大学で開発されたマクロモジュールである。マクロモジュールは表-2 に示されるような、インタフェースの統一されたデータモジュールと制御モジュールを用意し、これをバスケーブルで接続して希望の構造を持つ計算機を構成しようとするものである。原理的には非常に適応性の高い計算機を実現できるシステムと考えられ、応用指向計算機構成技法の先駆をなすものであったが、当時のハードウェア技術が不十分であったため、物理的大きさや、多量の接続ケーブル等の弱点があり、実用にはいたっていない。

2.2.2 R T M

次いで、1971 年には DEC 社から RTM という名称のモジュールセットが商品化された。RTM では新しいハードウェア技術と実用品という見地をふまえた

表-2 マクロモジュールの種類

関連する情報源	データ網	データ網, 制御網	制 御 網
モジュールの種類	・ ストージモジュール レジスタ メモリ ・ データ ブランチ	・ プロセッシングモジュール アダプタ シフタ ローダ ロジック ・ ディジションモジュール コンパ デコーダ	・ コントロールモジュール コール マージ ランデブ インタロック

設計が行われているので、マクロモジュールに比べ使いやすい、柔軟性は若干犠牲にされているが、かなりの数が中規模の特殊制御装置等に用いられている。

2.2.3 RT レベルモジュールによる応用指向効果

RT レベルモジュールについてはいくつかの実験結果が発表されている⁵⁾。応用指向計算機設計に対する RT レベルモジュールの能力を知るのによいデータであるから、ここで簡単に紹介しておこう。

- i) PDP-8 を RTM を用いて作成すると 55 個の制御モジュールと 10 個のデータモジュールが必要であるが、その設計はデバックまで含めて、8～10 人時間ですむ。もし、離散素子を用いるなら、6～7 人月、SSI 部品でも 2～3 人月かかるといわれるので、モジュール構造が設計時間短縮に如何に効果的であるかがわかるであろう。
- ii) しかしながら、よいことばかりではなく、この PDP-8 の場合、コストは 2 倍、速度は 40% しか得られない。もし、マクロモジュールを用いれば、最悪コストは 10 倍になることもあと予想される。これはモジュール化のためのハードウェアオーバーヘッドが大きい(RTM で 30%、マクロモジュール 70%) ためであるが、最近では LSI 技術が発展したのでこの点はかなり改善されていると考えられる。
- iii) 特殊応用に指向させた場合は表-3 に示すように大型機なみの能力を発揮する。この事実は RT レベルモジュールがはん用計算機では性能やコスト的には無効であるにしても、応用に対する適応性を与えることができるメリットは極めて大きく、応用指向計算機を作成する技術としては非常に効果的な技法であることを示している。

2.2.4 その他の RT レベルモジュール

先に述べた 2 つのモジュールセットはいずれもやや古くなったが、RT レベルモジュールとしては代表的なものであって、その後開発されたものはほとんどこれらにもとづいている。なお、最新の LSI 技術を駆使したモジュールとしては各種のビットスライスマイクロプロセッサがあるが、特に Fairchild 社の macrologic は LSI 化 RT レベルモジュールセットの供給をねらったものとして注目される。

2.3 PMS レベルモジュール

PMS とは Processor, Memory, Switch の頭文字をとったもので、文字通りこのような計算機の要素を

表-3 RT レベルモジュールによる適応化効果

問 題	計 算 機	時 間 (μs)
(1) 行列の積	ミニコンピュータ	400
	CDC 7600	5
	マクロモジュール	35
(2) FFT	CDC 6600 マクロモジュール	同程度
(3) ECG プレプロセッサ	CDC 6600	7
	PDP-9	37
	マクロモジュール	3
	TTL による専用装置	1.5

モジュール化したものである。LSI、特に、高性能なマイクロプロセッサの開発によって、コンパクトにプロセッサやメモリをまとめることが可能になったので、このレベルのモジュールを用いたマルチプロセッサ構造の計算機の研究は最近、特に注目されている。

この PMS モジュールの考え方を積極的に打ち出し⁷⁾、最も熱心に研究を進めているのは CMU* の Fuller 等のグループであるが、それより以前に、ミニコンピュータに、このようなモジュール化を取り入れたものもある。

また、V. R. Lesser は応用環境の要求する PMS 構造 (仮想 PMS 構造) を現実の PMS 構造 (実 PMS 構造) にマッピングする適応性 (特に並列処理構造に関し) のよい計算機構造について研究している⁸⁾。

以下では、実際にプロトタイプを開発した、あるいは開発中のものについて、PMS モジュール構造計算機の例をいくつかあげておく。

2.3.1 S U E

Lockheed electronics 社の SUE はミニコンピュータのプロセッサやメモリをモジュール化し、これらを応用の必要にあわせて、バスで結合できるように実装方式を工夫したものである。SUE は LSI 化はされていないし、モジュール間の通信方式にも特別の工夫はなされていないが、初めての実用 PMS レベルモジュールセットとして大きな意義を持っている。

この SUE を用いて設計された応用指向システムとしては BBN が設計したスーパー IMP⁹⁾が有名である。スーパー IMP は ARPA 網の計算機間通信制御用のシステムであって、図-1 (次頁参照) のような構成を持っている。図からわかるようにこのシステムはプロセッサ 2 台とローカルメモリを結合したバスと、共用メモリ群のバスをスーパーモジュールとし、これを特殊なバスカブラで結合した構造になっている。しばしば用いているプログラムは各ローカルメモリにおき、共用メモ

* Carnegie-Mellon University

りのアクセスを減らす等、処理の特性を生かした方式が採用され、PMS レベルモジュールによる応用指向計算機の成功例といえる。

2.3.2 PMSモジュール間アドレッシング機構

PMS モジュールを用いた応用指向計算機は、一般に、マルチプロセッサ構造を取るから、各モジュール間の通信能力、通信の柔軟性は非常に重要である。CMU の Fuller 等は論理アドレスでモジュール間の通信を行う柔軟性の高い方法を提案している¹⁰⁾。後に述べる CM* や ACE ではこれを拡張した方法が採用されているので、ここで簡単に説明しておく。

この方法は各モジュールの持つローカルな(主記憶)アドレス空間と通信のために各モジュールが共通に用いるアドレス空間(バスアドレス: CMU, グローバルアドレス: ACE)を分離して、論理アドレスで通信ができるようにし、通信の柔軟性を高めたところに特長がある。

図-2 はこの方法におけるアドレス関係を示したものである。通信を希望するモジュールが自分のローカルアドレスを通信用共通アドレスに変換して、バス上に出すと、そのバスに接続された各モジュールはこのアドレスを監視していて、それが自分の関与すべきもの(どんなアドレスに回答するかは各モジュール内のテーブルできる。)なら応答し、通信アドレスを自分の持つローカルアドレスに再変換して、通信路を確立する。(Fuller の元の考え方ではこのローカルアドレスをまた、別の通信アドレスに変換し、他の通信バスへ出し、次々と目的地まで伝えることを認めているが、その場合はデッドロック等、若干の問題点も生じる。)この場合、通信アドレスはユニークなシステムアドレスとして用いられてもよいが、図のように複数のモジュールが応答できるように(図は PM₀ に対し、PM₁ と PM₂ が応答)すれば、1 対多の通信も容易に行える。

要するに、この方法では論理的結合関係を物理的結合関係と相当程度切り離せることに意義がある。

2.3.3 CM*

CM* は Fuller 等が CMU で開発中のマイクロプロセッサを用いたモジュール構造計算機¹¹⁾で、前記通信機構(原理的に)を用い、応用の要求にあった柔軟な構造を取れるようにしたシステムである。

図-3(次頁参照)はその構造を示したものであるが、LSI-11 を用いて作られたコンピュータモジュール

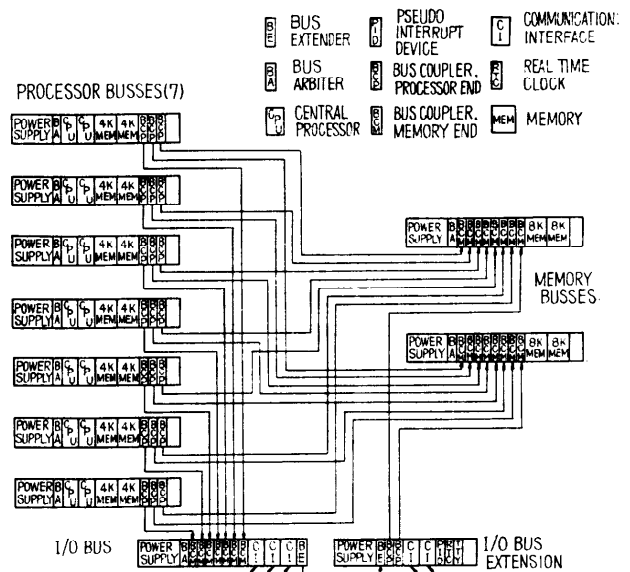
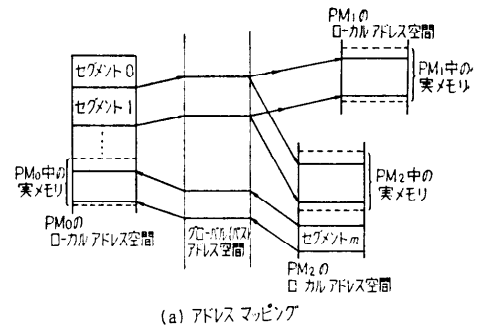
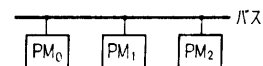


図-1 SUE によるスーパー IMP



(a) アドレスマッピング



(b) 結合構造

図-2 共通アドレス空間によるモジュール間通信

(CM)をスイッチとバスで接続した構造を持っている。各バスには Intel 3000 を用いたマッピングコントローラがあり、一本のバス上の CM 群(7個まで)をクラスタと呼んでいる。CM のプロセッサが発生したアドレスはセグメンテーション機構を用いて、自分のローカルメモリへのアクセスの他、他のクラスタへのアクセスにも用いられる。後者は前節の基本的方法に基づき、対応するクラスタまで中継され、他のモジュールのメモリを通信用にアクセスすることができる。

CM* の詳細は未発表な面も多く、実験システムであるから、にわかには評価しがたいが、少なくともモジ

ジュールの結合方式の点では最も進んだ試みであるといえよう。

2.3.4 その他のモジュール構造計算機

電子技術総合研究所の ACE システム¹²⁾は応用への適応機能を追求した実験システムで、モジュール間の通信方式は Fuller の基本構成にもとづいているが、スイッチを用いないバス構造のためのデッドロックがなく、また、同一バス上のモジュールに階層構造化性を付与できるように、新しくグローバルティ情報を各通信に加えている。また、ACE はプロセッサモジュール自身も市販品ではなく、適応機能を持つよう新設計されたものであるが、これについては後に述べる。

次に、大阪大学で開発中のモジュール構造システム¹³⁾はプロセッサのネスティング構造を可能にし、柔軟なシステム構造を取り得るよう工夫されている。

この他、Arnold 等も応用の要求に合わせて、システムを構成できるマルチマイクロプロセッサシステムを提案している¹⁴⁾が実現はされていない。

更に、通信回線レベルの接続において、自由な構成を取り得るよう工夫されたものとして、IBM の SNA¹⁵⁾が興味深い。しかし、SNA は通信回線レベルの接続を取り扱ったもので上記の各システムとはややレベルを異にするので、ここでは略す。

3. マイクロプログラミング

マイクロプログラミング方式の計算機では計算機ハードウェアの細部を直接プログラム制御することが可能であるから、それを応用に合わせて変更することによって、応用指向計算機を容易に構成できるといわれている。更に、最近では高速 RAM が比較的低価格で得られるようになったことから、マイクロプログラムを変更することによって、動的に変化する応用にも適応可能であるとして研究が進められている。

3.1 応用適応形マイクロプログラム方式計算機

さて、一般のマイクロプログラム計算機を用いても、相当程度の応用適応性を得ることはできるが、これらの計算機はあらかじめ定められた機械命令セット等のシステムアーキテクチャを実現するのに都合のよい構

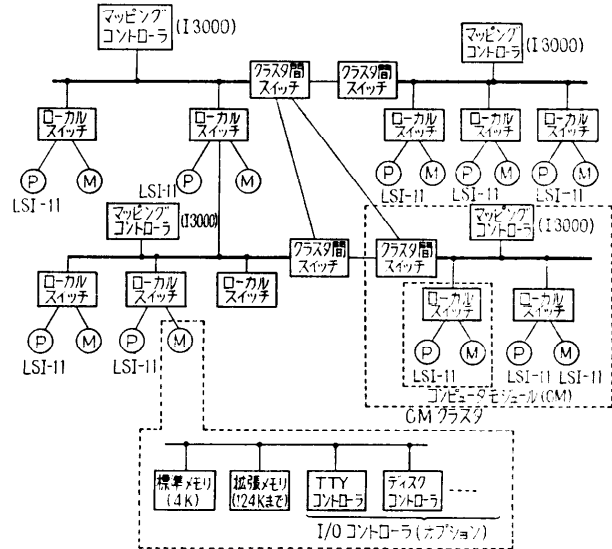


図-3 CM* の構成

造に設計されているから、一般の応用に適応させるには不都合な面も多い。

そこで、任意の応用環境に適応しやすいような一般性のあるマイクロプログラム方式プロセッサ構造の研究が 1960 年代の終り頃から始まった¹⁾。この種のプロセッサは「ユニバーサルホストマシン」、「エミュレーションプロセッサ」等と呼ばれているが、まず、初めにこれに着目したのは L. L. Rakoczi であって、Inner Computer の概念を提唱し¹⁶⁾、SCC 社より、一連の計算機を発売した。なかでも、H. W. Lawson 等が設計した MLP-900¹⁷⁾はマイクロプログラミングによる応用適応性を徹底的に追求したシステムで、現在 USC* にあり、ARPA 網にも接続されている。

MLP-900 は大型すぎて高価であり、わずか 1 台しか製作されなかったが、1971 年に Nanodata 社は R. F. Rosin 等と小型のエミュレーションプロセッサ QM-1 を開発した¹⁸⁾。QM-1 には内部のリソースを結ぶバスとレジスタ間接続を F 記憶と呼ぶ 5 ビットのレジスタで間接指定する方式や、適応性とプログラムの困難さを必要に応じて選択できる 2 レベルマイクロプログラミング**方式等の新しい技術が取り入れられている。

QM-1 は原理的にはマイクロプログラムによる適応性としてはかなり効率を上げられると考えられるが、今のところ具体的成果は発表されていない。

次いで、1972 年には各応用(実際には高級言語)に

* University of Southern California

** 図-4 のように水平プログラム方式のナノプログラムでハードウェア細部を直接制御し、それを 18 ビットの垂直マクロ命令で呼び出す方式。

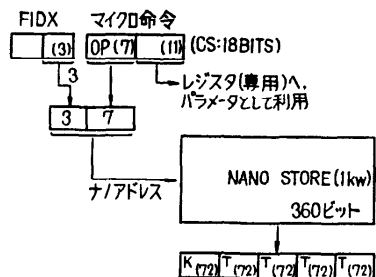


図-4 QM-1の2レベルマイクロプログラミング

適合した中間言語を設定し、それぞれの中間言語をマイクロプログラムで解釈する方式をとった B1700 システムが発表された。この B1700 のプロセッサではこの中間言語の解釈が効率的に行えるように一般性のある構成が取られ、大きなマイクロプログラムを利用できるようにするための主記憶によるバッファ方式(図-5)、任意のビットから1~24ビットの範囲のデータを一度に取ることのできるビットアドレッシング機構等の工夫が取り入れられている。B1700 はマイクロプログラムを利用した応用指向計算機としては商業的にも最も成功し、すぐれたものであるが、今のところそのマイクロプログラム機能は一般には公開されていない。

以上、歴史的意味を持つ応用適応形マイクロプログラム計算機について簡単に述べた。いずれもよく知られているので、詳しくは文献等を参照してほしい。

この他類似のシステムとして、Meta-4 (固定マイクロプログラム)、CONS²⁰⁾ (MIT, Lisp マシン用だが、一般的に設計されている)、ACE プロセッサモジュール¹²⁾ (以下 ACE-PR と書く。応用適応形のマイクロプロセッサ PULCE²¹⁾に制御回路を付加したもの、マイクロキャッシュによって大きなマイクロプログラムを用い得る、FCPU²²⁾ (スウェーデンの Saab 社の商用機、メモリ制御ユニット、演算ユニット、制御ユニットの3つの非同期ユニットからなる) 等の他、大学等での実験もいくつかある^{23), 24)}。

このように、マイクロプログラムによって応用指向

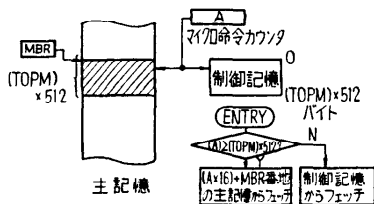


図-5 B1700 マイクロ命令アドレッシング

を実現できるようにした計算機はかなりの数にのぼる。しかしながら、これらを用いて実際に各種の応用に指向したシステムを構成し、成果を上げたという報告はまだ比較的少ない。したがって、それによる効果の程度もはっきりしない面があるが、通常はマイクロプログラムを応用ごとに設定することによって、5~10 倍程度の効果があるといわれている。

3.2 中間言語命令の動的適応化

これまで、応用適応機能を持つ計算機について述べてきたが、応用に適した中間言語を設定し、それをエミュレーションする方式では、この中間言語命令セットをいかに応用にあわせて設定するかが一つの問題となる。これは従来、人間の直観的判断によって行われていたが、応用プログラムの性格が明確に把握できない場合には必ずしもうまく設定ができるとは限らない。そこで、B1700 のように応用そのものではなく、それを記述する言語の特性を中間言語に反映させる、いわば間接方式をとるのが普通であった。

これに対して、最初の中間言語命令セットとしてはミニコンピュータの機械語のようにはん用だが単純なものを用い、そのプログラムを実行させながら、どのような命令群が続けて実行されるかを調べ、それらの内頻度の高いものをまとめた複合命令を新しい中間言語命令としてつけ加えていく方法が考案された^{25), 26)}。例えば、坂村等が HP 2100 で行った結果²⁵⁾では表-4に示すように、かなりの効果が上げられる。

さて、いうまでもないことであるが、この方法は使いつてのプログラムに適用することはできない。しかし、繰り返し利用するプログラムではプログラム一つごとに計算機を応用に適応させることができるし、あらかじめプログラムの特性が分かっているという利点がある。これまでの報告はシミュレータを用いたり、特殊な測定ルーチンを組み込む等、一般的使用には手数がかかりすぎるものであるが、将来はハードウェアモニタ等と組み合わせ、オンラインで動的応用適応化を行うことも可能であるかも知れない。

表-4 応用指向中間言語命令の合成効果

プログラム	全マイクロ命令数	時間 (マシンサイクル数)	合成されたマイクロ命令数	クロールン数
ローダ	7824→1558	18488→8769		4
アセンブラ (パス1)	59625→37855	148947→79608		6
ソーティング (クイックソート)	137092→33985	322501→131249		12
逆行列 (Fortran)	442396→111946	970210→505241		3

4. レジデュアル制御と可変論理機構

応用適応形のマイクロプログラム計算機において、一つのエミュレーションの間には不変で、かつ繰り返して利用される情報をレジスタ等にセットしておき、一解釈せずにハードウェアに利用させる方法をレジデュアル制御といっている。ここではこれをもう少し拡大解釈して、応用の特性を表わす情報をレジスタ内あるいは半固定的にセットアップし、ハードウェアがシステムの制御にこれを利用することによって、計算機を各応用へ適応させる技法を総称することにする。以下、この範疇に含まれる二、三の技法とシステム例を説明する。

4.1 デスクリプタ

デスクリプタとして最も普通のものはデータセグメントデスクリプタであって、Burroughs社の計算機等で古くから用いられている。データセグメントごとにおかれ、その位置と長さを初めとし、データのタイプ、保護情報等を記憶しているから、ハードウェアは各メモリアクセスごとにこれを参照し、アドレスの計算や適当なチェックを行うことができる。

もちろんこの程度ではデスクリプタが応用環境を定義しているとはいえないであろう。しかし、デスクリプタにはセグメント単位に変る特性ならなんでも入れてよいわけであるから、できるだけ多くの応用特性規定情報を格納し、計算機の動作をより応用指向にできるはずである。例えば ACE-PR のセグメントデスクリプタには上記の他、データユニット長、キャッシュ制御情報、外部アクセス路情報等も記憶されている¹²⁾。

さて、上記はデータのセグメントに対するデスクリプタであったが、デスクリプタが記述するのはデータでなくてもよい。例えば、先に述べたように、QM-1ではバスとレジスタの接続をF記憶というレジスタで間接制御しているが、これも結合路を記述するデスクリプタと見ることができる。この他、中間言語命令解読のためのジャンプテーブルも応用を記述するデスクリプタの一種といえるし、ALU の動作を記述するデスクリプタ等も考えられる。

表-5 論理語長アクセス方式の例

プロセッサ	最小アドレス単位	ユニットサイズ
B1700	1	1~24
MINIC-S	8	8 * ($n=1, 2, \dots, 16$)
ACE-PR	1	2 * ($n=0, 1, \dots, 5$)
FCPU	1	1~64

このようにデスクリプタ方式は一つの応用を実行中に変化しない状態を記述し、計算機を応用指向化して処理効率を高めるのに効果的である。もちろん、そのためのハードウェア量の増加は免れ得ないが、最近ではハードウェアコストが低下しているので、その点はあまり問題とはなるまい。むしろ、どのようにして、応用環境を記述し、それを効率的に解釈するどのようなハードウェア機構を用意すべきかという点では、データセグメントの場合を除き、未成熟な点が多いと考えられる。

4.2 データ長のアラインメント

一般の計算機で取り扱われるデータの長さは通常固定長または、基本語長の整数倍を用意するにすぎず、応用に対する適応性が小さい。そこで、最近では主記憶の物理語長と応用プログラムが要求する論理語長の間でマッチングを取り、データ長を調整する回路を備えた計算機が現れ始めている(表-5)^{12), 19), 22), 27)}。

4.3 半固定可変論理機構

デスクリプタ等の間接方式ではいくらうまくいっても完全にハードウェアを応用に合わせて設計するのに比べ、どうしても一段劣る効果しかあげられない。そこで、最も処理能力に影響を与える部分だけを応用に合わせてハードワイアドできるようにシステムを構成しておく技法が考えられた。これを半固定可変論理機構と呼ぶことにする。

さて、この種の機構の初めは MLP-900 に用いられたランゲージボードである¹⁷⁾。ランゲージボードは主に中間言語命令の解読部分を、対象に合わせてハードワイアドできるようにしたもので、最大4種類の応用に合わせたボードを持つことができる。同様のものは FCPU でも可変論理セット (VLS) という名称で4組まで持つことができる²²⁾。

このような可変論理機構はその部分だけハードウェアで作成するのであるから、効果の上では申し分ない。しかし、あまり自由度を与えても、その設計が複雑になりすぎ、使いにくくなるから、どのような部分まで応用指向に設計できるようにするかは、かなり難しい問題である。応用指向のハードウェアが容易に設計できるようにするためには、次に述べる PLA 等を効果的に使い得るようにする工夫が必要である。

4.4 P L A

PLA (Programmable Logic Array) は最近マイクロプロセッサと共に脚光を浴びている LSI 素子である。これを用いれば任意の論理回路を簡単に構成する

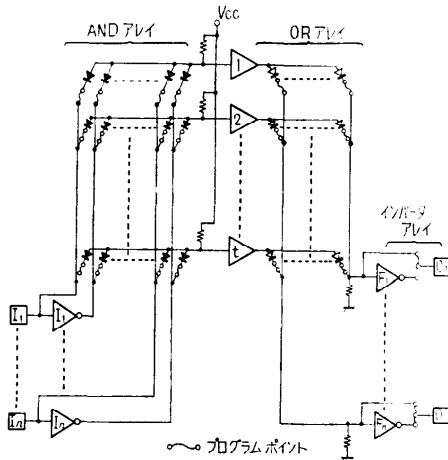


図-6 PLA の基本構造

ことができるから、比較的小形の応用指向プロセッサの設計や上記の半固定論理機構用に便利であると考えられる。ここではごく簡単に原理だけ説明しておく。

PLA は図-6 に示したように AND アレイ, OR アレイ, インバータアレイで構成される。まず, AND アレイで入力変換の論理積, すなわち主項を作り, 次に OR アレイでその論理和を求める。このようにして, 10 変数前後の 10 個前後の論理関数を 1 個の PLA を用いて構成することができる。

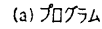
普通の PLA はこのようにアレイ構造を利用して、任意の組み合わせ論理回路を LSI で構成できるようにしたものであるが、記憶用のフリップフロップを内部または外部に用意すれば順序回路を作ることも難しくなく、

PLA を用いた応用指向プロセッサとしては IBM 7441 型端末制御装置の例が報告されているが²⁸⁾、7 個の PLA で通常の IC 1,731 個を置き換え、小型化、設計製作時間の短縮に成功したといわれる。PLA は応用指向計算機をハードウェアから作る場合の技術としてその将来性が期待される。

4.5 データフロー計算機

最後に、本章の標題とは若干離れるが、応用指向の
はん用計算機としてユニークな計算機を紹介しよう。
すなわち、MIT の Dennis 等が研究中のデータフロー
計算機である²⁹⁾。

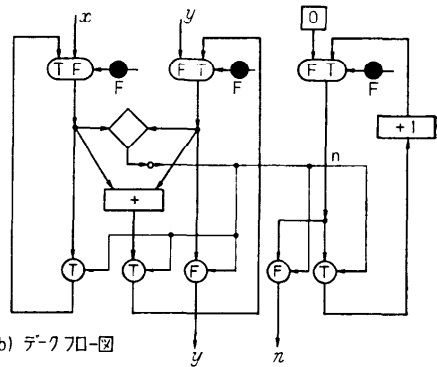
データフロー計算機はデータフローの形で表わしたプログラムをそのまま実行する。図-7 の例のように、データフローとは一種の流れ図であって、計算におけるデータの流れを中心に書いたものである。このデー



```

in put  y, x
n := 0
while  y < x do
  y := y + x
  n := n + 1
end
output y, n

```



(b) ~~テ-7 フ□-㊄~~

図-7 データフロー図によるプログラム

タフロー図では問題の特性（特に並列構造）が明確に表現されることになるから、一般のプログラムよりも応用の特性を、より直接的に表現しているといえることができる。

いいかえれば、従来のプログラム表現では問題の特性がばやけていて、応用指向のプロセッサを設計するにしても、応用特性の把握は人間の直観的判断や高級言語という間接的手段を通してでしかなされていなかったのに対し、プログラム特性の明確に出る表現法をそのまま実行できるように考え出されたのがデータフロー計算機であるということができる。

さて、Dennis のデータフロー計算機の基本構成は図-8 のようになっている。メモリには3語(普通は操作指定と2つのデータセル)を一つの命令セルとした命令群が入っており、データがそろると、これらの3つ組みはアービトレーション網を通過して、その操作コ

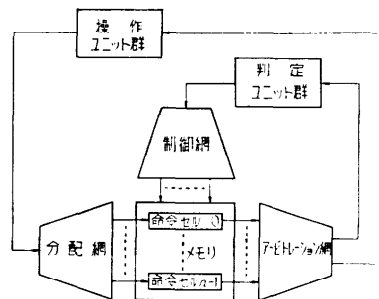


図-8 データフロープロセッサの構成

ードの示す操作ユニットあるいは判定ユニットへ送りこまれ、その結果は次のノードに対応する命令セルへ分配網からもどされる。このようにデータフロー計算機は3つ組みの命令セルの内容がそろいさえすれば直ちに対応する処理を行うことができ、問題の持つ並列性をそのままの形で実行することができる。さらに操作ユニット群は独立しているから、この機能に応用に合わせることも容易である。

5. む す び

以上、応用指向型のシステムやプロセッサを構成する技術を考察し、それをを用いたシステム例について述べた。既に触れたように、これらの技術は適応度、コスト、柔軟性等においてそれぞれ特徴を持っており、これらをうまく組み合わせ用いることが重要である。最近の LSI 技術とは、いずれの技術もかなりマッチングがよいから、今後、更に進歩を期待することができるであろう。

応用指向型のシステムはその応用に関する限り、はん用システムより処理効率がよいことは明らかであるから、大量生産できない応用でも、低コストで構成することが可能になれば大きな発展を期待することができる。しかし、そのためには本稿で述べた構成技術そのものの発展のみならず、i) 機能あるいは応用の特性を如何に正確に把握(測定も含む)し、表現するか、ii) その結果から、上記の技法を如何に組み合わせ、短時間で目的の応用指向計算機を設計できるようにするか、そして iii) 複数の機能指向プロセッサを組み合わせ一つの実用システムを構成する場合には、いかに機能を分配し、制御するか、ということも重要である。例えば i) の中にはマシンハードウェアの記述言語やハードウェアモニタ等による特性評価方式が、ii) の中には i) によって得られた記述からシステムを設計する言語や、シミュレータ等が含まれる。また、iii) については別稿に詳しく述べられる³⁰⁾。

これまでの応用指向計算機構成技術は技法そのものに比べ、上記 i) ~ iii) の面の検討が比較的遅れていたように思われる。こうしたものを統一的にまとめた応用指向計算機構成技術が発展することを期待したい。

最後に、本稿の内容について御意見をいただいた慶大工学部、相磯秀夫教授と所真理雄助手に謝意を表します。

参 考 文 献

- 1) 飯塚肇：適応構造計算機に関する研究，電総研研究報告第 767 号 (1977)
- 2) D. P. Siewiorek & M. R. Barbacci: Some observations on modular design technology and the use of microprogramming, CMU report (1974), also in Infotech state of the art report on microprogramming and systems architecture pp. 509~525 (1975)
- 3) W.A. Clark, et al.: Macromodular computer systems, Proc. SJCC, pp. 335~402 (1967)
- 4) C. G. Bell, et al.: The description and use of Register-Transfer Modules (RTM's), IEEE Trans. C-2, No. 5, pp. 495~500 (1972)
- 5) C.G. Bell and J. Grason: The Register-Transfer Modules design concept, Computer Design, Vol. 10, No. 5, pp. 87~94 (1971)
- 6) S.H. Fuller and D.P. Siewiorek: Some observations on semiconductor technology and the architecture of large digital modules, IEEE Computer, Vol. 6, No. 10, pp. 14~21 (1973)
- 7) C.G. Bell: The architecture and applications of computer modules, Digest of Compcon, pp. 177~180 (1973)
- 8) V.R. Lesser: Dynamic control structures and their use in emulation, CS 309, Stanford Univ. (1972)
- 9) F.E. Heart, et al.: A new minicomputer/multiprocessor for the ARPA network, Proc. NCC, pp. 529~537 (1973)
- 10) S.H. Fuller, et al.: Computer modules: An architecture for large digital modules, Proc. First annual symp. comp. architecture, pp. 231~237 (1973)
- 11) R. Swan: CM* architecture, CMU report draft (1976)
- 12) H. Iizuka, et al.: ACE—A new modular computer architecture, Proc. 2nd US-J comp. conf. pp. 36~41 (1975)
- 13) 寺田他：複数プロセッサシステムのための実験用単位 CPU の構成について，電子計算機研究会資料，EC 73-70 (1973)
- 14) R.G. Arnold & E.W. Page: A hierarchical restructurable multi-microprocessor architecture, Proc. 3rd symp. on comp. architecture, pp. 40~45 (1976)
- 15) J. H. Mcfadyen: Systems network architecture An overview, IBM system J., Vol. 15, No. 1, pp. 4~23 (1976)
- 16) L.L. Rakoczi: The computer-within-a computer. a fourth generation concept, IEEE computer group news, Vol. 13, No. 1, pp. 22~24 (1967)

- 17) H. W. Lawson & B. K. Smith: Functional characteristics of a multilingual processor, IEEE Trans. C-20, No. 7, pp. 732~742 (1971)
- 18) R. F. Rosin, et al.: An environment for research in microprogramming and emulation, CACM, Vol. 15, No. 8, pp. 748~760 (1972)
- 19) W. T. Wilner: Design of the Burroughs B 1700, Proc. FJCC, pp. 489~497 (1972)
- 20) T. Knight: CONS, MIT AI Lab working paper 80 (1974)
- 21) 飯塚肇, 古谷立美: マイクロプロセッサアーキテクチャの一設計, 電子通信学会論文誌, Vol. 59 D, No. 3, pp. 188~195 (1976)
- 22) H. W. Lawson & B. Malm: A flexible asynchronous microprocessor, BIT, Vol. 13, pp. 165~176 (1973)
- 23) M. J. Flynn & C. Neuhauser: EMMY-An emulation system for user microprogramming, Proc. NCC, pp. 85~89 (1975)
- 24) P. Konerup & B. D. Shriver: An overview of the MATHILDA system, SIGMICRO news letter, Vol. 5, No. 4, pp. 25~53 (1975)
- 25) 坂村健也: ダイナミックマイクロプログラミングによる最適命令セットの合成手法に関する考察, 電子計算機研究会資料, EC 75-28 (1975)
- 26) T. G. Rauscher & A. K. Agrawala: Developing application oriented computer architectures on general purpose microprogrammable machines, Proc. NCC, pp. 715~722 (1976)
- 27) F. K. Williamson: A high-level minicomputer, IFIP 74, pp. 44~48 (1974)
- 28) J. C. Logue, et al.: Hardware implementation of a small system in programmable logic arrays, IBM J. R & D, Vol. 19, No. 2, pp. 110~119 (1975)
- 29) J. B. Dennis & D. P. Misunas: A preliminary architecture for a basic data-flow processor, MIT Computation structures group memo 102 (1974)
- 30) 相磯秀夫: 機能分散型計算機システム, 本特集号

(昭和51年12月2日受付)

(昭和52年1月19日再受付)