

低レベル並列処理機能を備えたダイナミック・マイクロ プログラム制御計算機QA-2のアーキテクチャ

AN ARCHITECTURE OF A DYNAMICALLY MICROPROGRAMMABLE

COMPUTER WITH LOW-LEVEL PARALLELISM : QA-2

柴山 繁, 富田 眞治, 萩原 宏
Kiyoshi SHIBAYAMA Shinji TOMITA Hiroshi HAGIWARA

山崎 勝弘, 北村 俊明, 栗山 和則, 中島 浩
Katsuhiko YAMAZAKI Toshiaki KITAMURA Kazunori KURIYAMA Hiroshi NAKASHIMA

新 実 治 男, 山 下 博 之, 中 田 登 志 之, 藤 井 誠
Haruo NIIMI Hiroyuki YAMASHITA Toshiyuki NAKATA Makoto FUJII

(京都大学・工学部 KYOTO University, Faculty of Engineering)

1. ま え が き

我々は、低レベル並列処理方式とマイクロプログラム制御方式とを採用することにより、高速性と柔軟性を兼ね備えた可変構造形実験用マイクロプログラム制御計算機QA-1 (Computer with Quadruple ALU's) の開発を完了して以来、マイクロプログラム制御方式の柔軟性を生かして種々の応用分野へQA-1を適用し、そのシステム評価を行なってきた。^[1]

QA-1のアーキテクチャ上の特徴は、i) 4個の算術論理演算装置(ALU)とこれら複数ALUから同時アクセス可能な共有レジスタを装備して、レジスタレベルの並列処理を行ない処理の高速化を達成する；ii) 水平形マイクロ命令(1語長=160ビット)の独立なフィールドで4個のALUを同時に制御し、可変構造形計算機としてのシステムの柔軟性を高める；iii) 仮想制御記憶方式により、大規模ユーザマイクロプログラミングを実現する；などである。

QA-1の設計目標は、i) 図形処理、画像処理、信号処理などの大量均質データの高速処理を必要とする応用分野において、並列処理方式を利用することにより、中小形の専用コンピュータと同程度、または通常のミニコンピュータの50~100倍程度のリアルタイム処理能力を達成すること；ii) 汎用エミュレーションや高級言語処理などの各種逐次処理応用に適用できる柔軟性を有し、大規模ユーザマイクロプログラミングを可能とする強力な可変構造形の実験用コンピ

ュータとして利用できること；iii) ii) の特徴を充分に生かし、研究室において、並列処理と逐次処理が同一コンピュータ上で効率よく統合されたインテリジェント・サテライトを構成できること；の3点であった。

ii) の応用分野におけるQA-1の代表的な適用例は、リアルタイムカラーアニメーションシステムであり、QA-1は200本以上のベクトルから構成される2次元カラー図形によるアニメーションを1秒間に12コマの割合で生成するリアルタイム処理能力を有している。^[2] これは、通常のミニコンピュータの処理能力の50~100倍程度の機能と判断して良い。ii) の応用分野におけるQA-1の適用例は、各種の真/仮想計算機のエミュレーションである。エミュレーションにおけるソフトウェア/ファームウェア/ハードウェアのトレードオフなど、QA-1の可変構造形計算機としてのシステム評価、検討のためには多様なエミュレーションクラスへのQA-1の応用実験が必要であるため、商用ミニコンピュータ(HITAC-10, PDP-11/40)の機械語命令、PASCAL向きスタックマシンの命令語、および直接実行形高級言語計算機の命令語(BASIC)の各々を対象言語として種々の実験を行なってきた。^{[3][4][5]} これら逐次処理指向の応用においても、HITAC-10, PDP-11/40クラスのミニコンピュータの数倍から数十倍の能力が検証され、QA-1の低レベル並列処理方式がある程度有効であることが確認されている。

多数のユーザが、広範な応用にQA-1を適用

して来た結果、QA-1設計時には把握し得なかつた種々のアーキテクチャ上の欠点を同時に指摘されてゐる。また、過去6年の集積回路技術の目ざましい発展による、アーキテクチャ改良の可能性やハードウェア実装上の問題点も明示されてきた。QA-1を用いた実際的な応用システムの構成過程で得られた貴重な経験を集約し、高機能、高速度のLSI論理素子、高速大容量メモリ素子を採用することによって、QA-1のシステム設計時にはハードウェア化し得なかつた高機能の複素制御方式を実装した大規模実験用可変構造計算機QA-2の設計が可能となった。

2. QA-2の設計方針と特徴

QA-2の適用分野は、QA-1の応用範囲であった各種リアルタイム処理(図形処理、信号処理、画像処理)、および商用ミニコンピュータや仮想の高級言語計算機、専用計算機のエミュレーションなどをすべて包括するものである。従つて、QA-1の基本設計思想であり、上記の応用分野に有効であった「低レベル並列処理方式とダイナミックマイクロプログラム制御方式の有機的結合」方式は、QA-2でも基本アーキテクチャとして継承され、QA-1のハードウェア構成方式の根本的な改良によって、処理速度、システムの柔軟性、およびマイクロプログラムの生産性を飛躍的に高め得る様にQA-2を設計した。

大規模ユーザマイクロプログラミングにおいて、ユーザは論理的なプログラム構造(データ構造、ALU演算や制御構造によって記述するアルゴリズム)をホスト計算機によって実行される物理的なマイクロプログラム構造(物理的なレジスタや主記憶空間におけるデータ表現、マイクロ命令を制御するハードウェア)に変換する。多様な応用に対して、この論理的な構造と物理的な構造とのギャップをできるだけ限り小さくできるハードウェア構成方式が、ホスト計算機の処理能力、システムの柔軟性、およびマイクロプログラムの生産性を向上させる主要な要因となる。特に、マイクロプログラミングが難しくなるのは、ホスト計算機のハードウェア設計時にマイクロプログラムの生産性についての配慮が

なされてゐないことに起因してゐる場合が多い。従つて、QA-1の開発によって得られた経験から、マイクロプログラム制御による可変構造形ホスト計算機の性能評価の基準として、処理能力とシステムの柔軟性と併せて、マイクロプログラムの生産性をハードウェアレベルで考慮することが重要であると確信してゐる。

これらの基準を考慮に入れて新たに設計したQA-2のアーキテクチャの特長は、次の様にまとめられる。

(1)同一機能をもつ4個のALUによる低レベル並列/連鎖演算方式。

QA-2では、4個の可変長ALUが1つの水平形マイクロ命令(1語長=256ビット)の相異なるフィールドで独立に制御され、均一で大容量のレジスタファイル(6kビット)を共有して動作できる。4個のALUは相互に独立な4つのオペランド群に対して同時に並列演算を実行できるだけでなく、1個の演算結果が他の演算の入力オペランドとして使用される場合には、ALU連鎖演算機能により1つのマイクロ命令内でALUの逐次演算も実行できる。(図1参照)ALU連鎖演算方式を付加したことにより、QA-2では、QA-1が充分効率良く適応できなかった逐次処理に対して、大幅な処理速度の改善が達成できる。

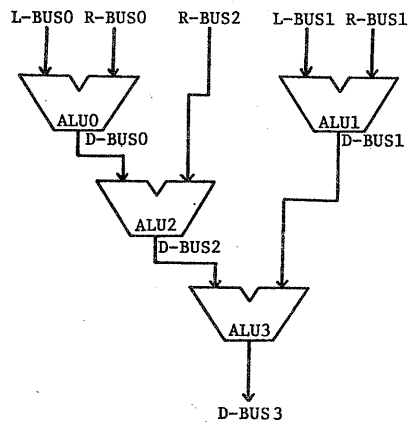


図1: ALUの連鎖演算機能。

(2)マイクロプログラムの生産性の向上を指向した複素制御構造。

大規模ユーザマイクロプログラミングを実現するためには、ユーザが見たハードウェアの並列演算・制御構造が一様であること、さらに

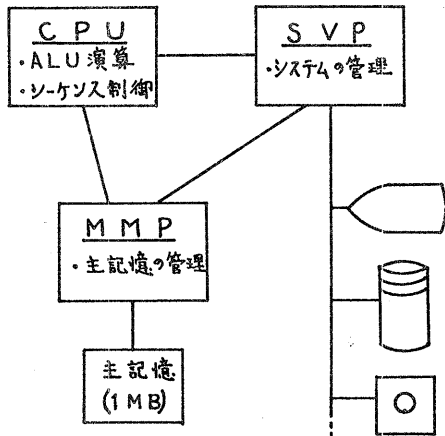


図2: QA-2の機能分解

ユーザが記述する論理レベルのプログラム構造と物理的なハードウェア構造との対応関係を容易に与えることが要求される。例えば、高度な制御構造は、高機能のマイクロプログラムを作成する場合に必須のものであり、IF文、CASE文などの強力な条件判定分岐機能は、汎用ハードウェアとして実装されていなくてはならない。QA-2では、ハードウェアレベルの演

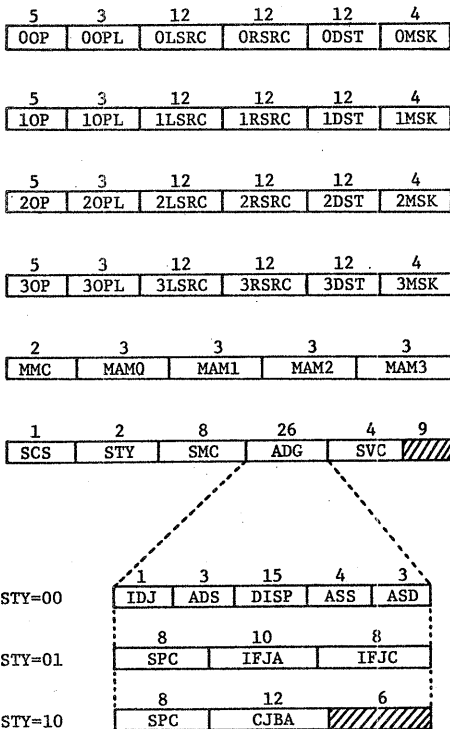


図3: QA-2のマイクロ命令形式

算・制御構造およびデータ構造と、汎用性が失われぬ範囲でユーザ記述レベルの論理的プログラム構造に近づけることを試みるため、ファームウェア工学の観点に立脚した高機能の演算・制御構造を採用している。

(3)処理機能の分散化および高機能化。

図2に示した様に、QA-2はALU演算部や順序制御部など超高速に動作する機能部分(CPU)と、主記憶管理プロセッサ(MMP)やシステム管理プロセッサ(SVP)とを独立に構成し、これらを非同期に制御する方式を採用

表1: マイクロ命令フィールドの機能

フィールド名	機能
0-3OP	各ALU演算の種類を指定する。
0-3OPL	各ALU演算のオペランドデータ長を指定する。
0-3LSRC	各ALU演算の左入力オペランドのソースレジスタを指定する。
0-3RSRC	各ALU演算の右入力オペランドのソースレジスタを指定する。
0-3DST	各ALU演算の結果を格納するデスティネーションレジスタを指定する。
0-3MSK	各ALUの左右入力オペランドおよび演算結果に対するマスク操作(を行な)マスクレジスタ組を指定する。
MMC	MMPに対するコマンドであり、主記憶のリード/ライト、ブロック転送、サーチを要求する。
MAM0-3	MMPに対してリード/ライト操作方式の詳細を指定する。
SCS	マイクロ命令の実行と、次に実行するマイクロ命令のフェーズをオーバーラップさせるかを指定する。
STY	順序制御における分岐形式を指定する。
SMC	マイクロ命令実行によるステータスの収蔵方法を指定する。
SVC	SVPとの通信を行なう。
IDJ	ADSおよびDISPフィールドで作成されたアドレス情報を直接アドレスとするか間接アドレスとするかを選択する。
ADS	次に実行するマイクロ命令アドレスを生成するためのアドレス修飾データのソースを指定する。
DISP	次に実行するマイクロ命令アドレスを生成するためのベースとなるインデキスイットフィールドである。
ASS	順序制御用に回避すべき情報のソースレジスタを指定する。
ASD	順序制御用に回避すべき情報のデスティネーションレジスタを指定する。
SPC	256ビットのステータスレジスタ(SSR)から、順序制御に使用する8ビットのステータス情報を選択する。
IFJA	IF条件による2方向分岐の場合の分岐形式を指定する。
IFJC	SPCフィールドによって生成された8ビットのステータス情報からIF条件を生成する。
CJBA	CASE条件による多方向分岐の場合の分岐形式を生成する。

した。各プロセッサは互いにバスで密結合され、CPUのフリーズ状態を極力減らして、CPUの並列処理能力を最大限に発揮できる構成となっている。特に、SVPによって、多重プログラミングの制御下で制御記憶は動的に入れ換えられ、多数のユーザの仮想計算機がQA-2上で並列に動作可能である。

3. QA-2のハードウェア構成

QA-2における、最も重要な改良点は、処理速度、システムの柔軟性、およびマイクロプログラムの生産性の向上を図るために、一様な並列演算のためのハードウェア構成を採用し、多様な応用に対して論理/物理構造のギャップまでできる限り小さくすることができるとアーキテクチャを採用した点である。

以下本章では、QA-1とQA-2の機能を対比して詳述し、QA-2アーキテクチャの改良点を明らかにする。QA-2の各機能を制御するマイクロ命令は1語256ビット水準形であり、その形式を図3、各フィールド機能を表1にまとめて掲げ、ハードウェア構成の詳細に併用する。

3.1 演算部の構成(図4)

(1) ALUの構成形式

[QA-1] 1マイクロ命令でALUは16ビット固定長のものが最大4個並列に制御できる。多倍長データの演算(例えば64ビット)を行なう場合には、ALUを(4個)連結して制御しなければならぬので、1マイクロ命令における論理的な

並列演算度が(1に)減少し、論理レベルでユーザが考えた並列演算度との間にギャップが生じる。従って、処理速度とコストに、マイクロプログラムの生産性が低下する。

[QA-2] 1マイクロ命令で最大4個のALUが並列に制御できる。1個のALU演算のデータは8, 16, 24, 32, 40, 48, 56, 64ビットのいずれでも可能な可変長形式である。1マイクロ命令で最大4組の演算の実行が保障されているので、ユ

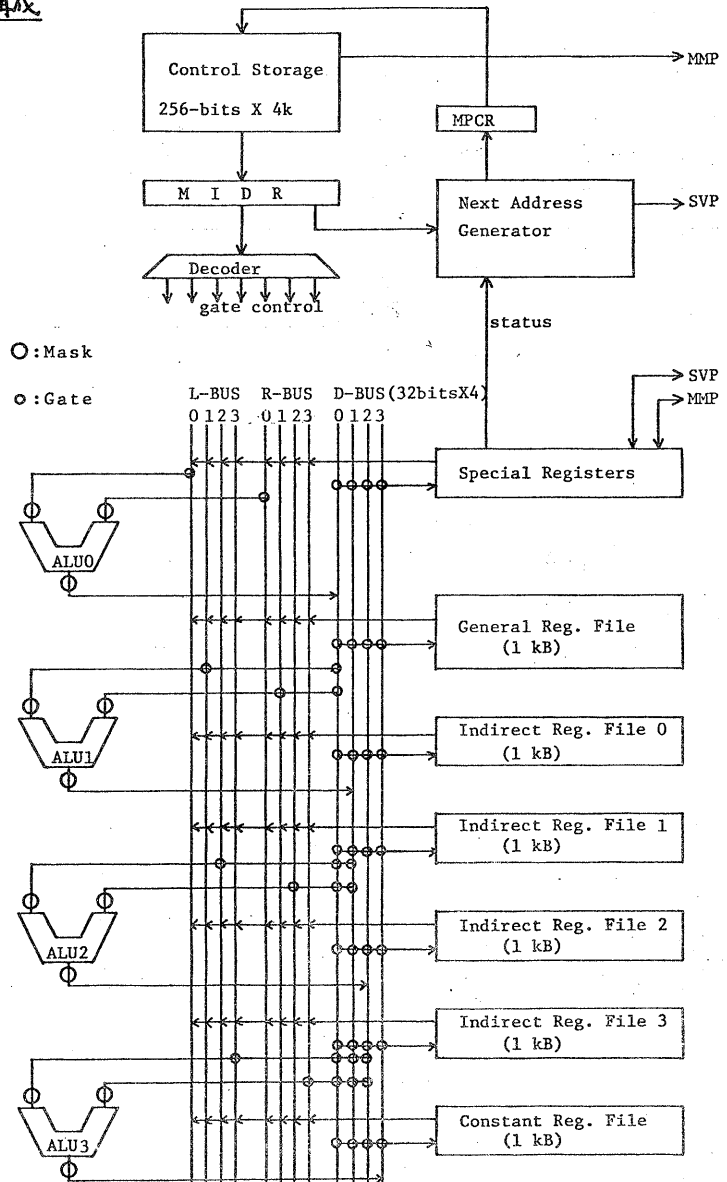


図4: QA-2演算部の構成

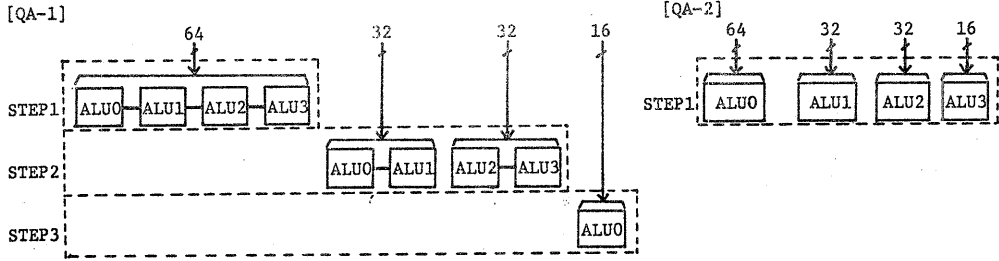


図5: ALU並列演算機能の比較

ユーザが考えたプログラムの論理構造と物理的なALU構成との対応が容易であり、マイクロプログラムの生産性を向上する。(図5参照)

また、ALUごとに可変長データを処理できるので、ALUの入出力データに対するマスク操作を行なうマスクレジスタとして、各データ長に対して各々16組の大容量ファイルが装備されている。(図6参照) さらに、浮動小数点形式データに対する演算機能として、正規化回路などをハードウェアでサポートしているほか、ALU演算モジュールとして、乗算器、並列シフト回路、バイト/ビット操作回路を装備している。

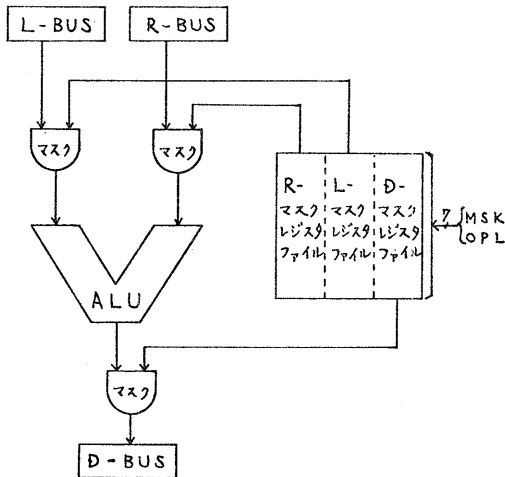


図6: QA-2のマスク操作機能

(2) レジスタの構成方式

[QA-1] 4個のALUから同時アクセス可能な共有レジスタの個数は15個と少なく、また共有レジスタ内のオペランドは各ALUの右入力にしかなり得ないことや、直接アクセスあるいは間接アクセスのみでレジスタがあることなど、レジスタ空間が「様でない」ので、ユーザは多く

の回数に対して効率が良いレジスタ割り付けを行なうために苦慮しなけれならぬ。また、共有レジスタとローカルレジスタ間でデータ転送に要するオーバヘッドが大きく、レジスタレベルでの並列処理の効率低下の一因となっている。

[QA-2] 全レジスタ空間は、いずれのALUから見ても一様である。(汎用: 1kバイト, 定数格納用: 1kバイト, 間接アクセス専用: 4kバイト) レジスタファイルは、図7の様に256x4ビットのRAMチップを用いた8ポート方式で構成することにより、4個のALUによる同一RAMチップへのアクセス競合を解決している。

これらにスタック機能を与えるために、間接アクセス方式の強化を図った。間接アドレスレジスタは16個あり、各々アクセスデータ長に合わせて自動的にインクリメント/デクリメント可能なカウンタ機能を装備している。

直接/間接アクセス可能な大容量の共有レジスタ空間が構成されるので、複雑なレジスタ割り付けが不要となり、またレジスタ間転送のオーバヘッドも生じないので、レジスタレベルの並列処理方式の利点を最大限に生かせると同時にマイクロプログラムの生産性を飛躍的に増大させることができる。

(3) ALUの連鎖演算方式 (図8参照)

[QA-1] ALU連鎖演算機能をもたないので、逐次演算が連続する処理において、マイクロ命令の演算制御フィールドに遊ばが生じ、本質的に逐次処理の多い応用に対しては、低レベル並列処理方式は充分有効には機能し得なかった。すなわち、処理効率が良いマイクロプログラムを作成するために、ユーザは、1マイクロ命令で同時実行可能なALU演算の組み合わせを、論理的に逐次処理の多いプログラムの構造から発見するという煩わしい最適化操作を余儀なくさ

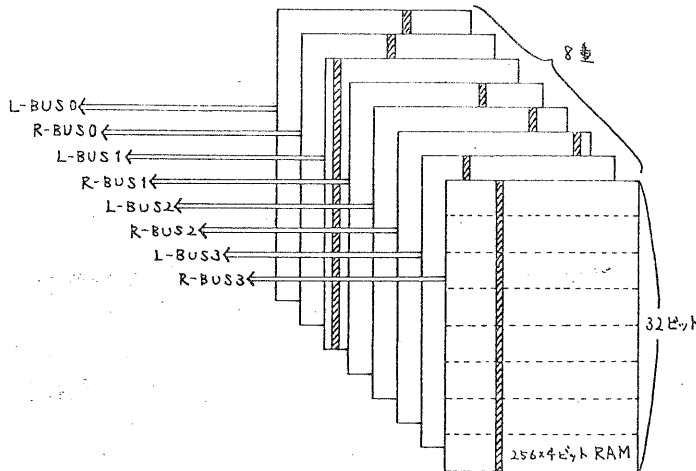


図7: QA-2レジスタファイルの構成

れ、マイクロプログラムの生産性や完成したマイクロプログラムの読み易さなどを著しく低下させる結果となっている。

[QA-2] 1マイクロステップで、あるALU演算の結果をそのまま別のALU演算の入力データとして用いるALU連鎖演算が可能であり、ユーザは最大4個までのALU演算を並列、逐次演算の区別なく1マイクロステップで行なうことができる。従って、1マイクロ命令で制御するALU機能レベルが向上し、オーバヘッドの大きい一時退避レジスタの存在も無くなるので、水平形マイクロ命令による低レベル並列処理機能も有効に動作する。ユーザは、マイクロプログラムの論理構造をそのままQA-2のALU演算機能の物理構造に対応させることが容易となる。

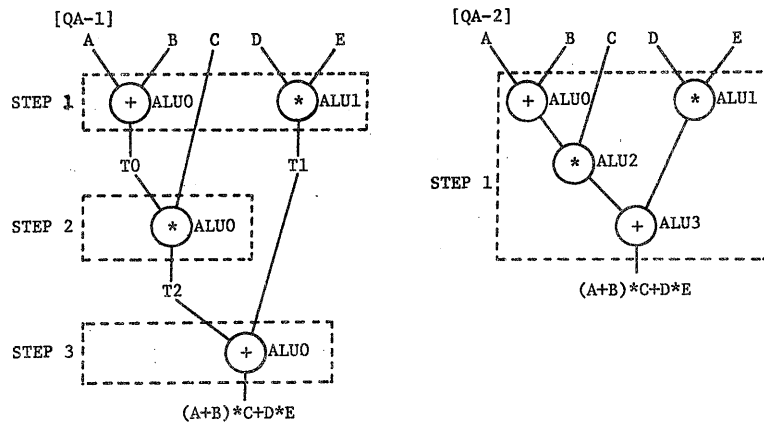


図8: ALU連鎖演算の使用例

(4) ALU機能の一様性。

[QA-1] ALU片のみに特殊レジスタが多数付加され、逆形式命令(80ビット長)では、このALU片のみを制御している。さらに、固定長ALUの連鎖順位は固定されているので、ユーザは使用するALU番号を充分考慮に入れてマイクロプログラミングを行なう必要がある。

[QA-2] 4個のALUは機能的にすべて等価であり、各ALUに専有のローカルレジスタは無く、レジスタ空間はすべてのALUに共有されている。ユーザは、論理的

に4個の演算のみを意識すれば良く、その演算がどのALUハードウェアで実行されるかを考慮する必要がないので、ALU割り付けが処理速度を大きく左右することは無い。

(5) 定数オペランド演算機能

[QA-1] マイクロ命令のフィールドを流用して定数オペランドの発生や演算を行なうので定数データ長が限られる。また、16ビットの定数データを発生させるためだけに1マイクロ命令の1ALU演算制御フィールドが必要となり、実行時のオーバヘッドが大きい。

[QA-2] 実行時に必要となる定数データは、実行前にSVPによって大容量共有レジスタファイルの一部に割り付けられるので、定数発生のためのマイクロ命令は不要である。定数は、マイクロ命令のフィールドに格納されるので、そのデータ長が制限されることはなく、8~64ビットの任意のデータ長が選択できる。

3.2 主記憶部の構成

[QA-1] 主記憶アクセスは各メモリバンク(4バンク)に固定されたMAR, MDRを通じて行なわれる。各MARへのアドレスデータのセット弁送は3種類用意されているが、各ALUの出力を各MARに同時に書

き込むことができないので、アドレスデコーダセットが逐次処理となり、主記憶の並列アクセス機能が有効に動作しない場合がある。主記憶のアクセス単位は16ビット(固定長)であり、記号データ、画像データの処理を行なう場合には、バイト操作を多用しなければならぬ。また、効率の良い主記憶管理を行なうためには、ユーザが各バンクに対する物理的なデータ配置を充分考慮する必要がある。

[QA-2] 主記憶は1Mバイト包装され、サイクルタイム4200 nsecで、CPUと独立にMMPによって制御される。CPUにおけるALU演算と非同期に動作可能であり、CPUから主記憶アクセスの必要が生じた場合は、マイクロ命令の主記憶制御フィールドを利用してMMPに主記憶アクセスを依頼する。MMPとCPUとのインタフェースは、共有レジスタファイル空間に割り付けられたMARとMDRであり、これらの組が4組装備されている。各MAR, MDRは同一機能をもち、任意のALUからアクセス可能である。1マイクロ命令で異なる4個のアドレスへの主記憶アクセス(1組のMAR, MDRにつき最大4ビット)が可能である。MAR, MDRの組々々には、他にRWCRというカウンタが付加されており、可変長データに対するアクセス制御やMARの自動カウント機能に使用される。MMPが、同一バンクへのアクセス競合を高速に処理するので、ユーザが可変長データの割り付けを意識する必要はない。(図9参照)

MMPはCPUと独立に動作可能であり、CPUから依頼があれば、CPU動作とは並列に、主記憶データのサーチやブロック転送を実行することも可能である。サーチ機能は、ユーザの指定した4ビット以下のデータを指定されたアドレス範囲内でサーチして、その有無およびアドレスを応答するものである。ブロック転送機能は、ユーザの指定に従って主記憶内のデータを移動するものである。

MMPには、仮想制御記憶方式による制御記憶の入れ換え用とビデオ信号の接続用に超高速バーストモードDMAバスが2本、外部I/O接続用にサイクルスチールモードDMAバスが2本、CPUとSVPとの通信用にプログラマモードバスが2本接続される。

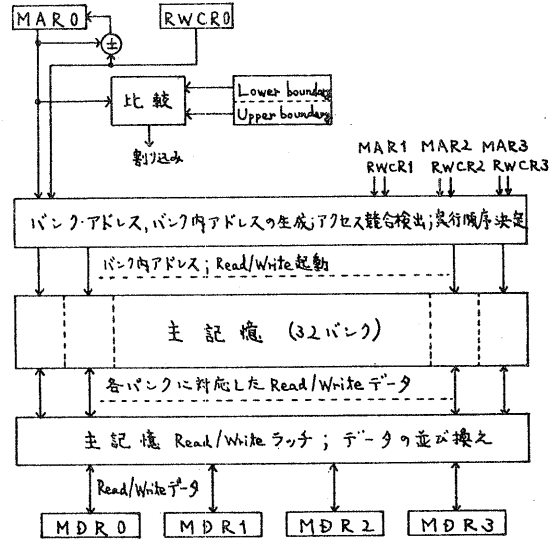


図9: QA-2のMMPの構成

主記憶は、制御記憶やCPU内の順序制御用の各種の長のバックアップ記憶としても利用され、これらの入れ換え動作の管理はSVPが行ない、SVPの制御下でMMPとCPUが通信する。

3.3 制御部の構成

(1) マイクロ命令の形式

[QA-1] 逐次処理の多い応用において、制御記憶を表番することがない様に、ALUの中のみを制御する短形式命令(長形式命令の半分の語長)が設けられている。短形式命令と長形式命令の混在は制御記憶内の構造を複雑にし、ユーザが制御記憶の効率的利用を意識しすぎるあまり、マイクロプログラムの生産性を低下させる一因となっている。

[QA-2] 逐次処理に対してはALU連鎖演算機能を利用するなど、演算部において根本的な対処をしているので、マイクロ命令の各フィールドの使用効率は良くなる。従って、マイクロ命令形式は1種類であり、大量の制御記憶と合わせて、ユーザマイクロプログラミングにおけるマイクロ命令形式が原因となる制約は存在しない。

(2) マイクロプログラムの分岐形式

[QA-1] 条件分岐方式には、マイクロ命令の長短両形式の混在、アドレススタックの容量不足などから種々の制約があり、実行効率の良いマ

マイクロプログラムを作成するためにユーザは、問題の本質とは直接関係しない部分にも考慮を払う必要がある。また、複数の分岐形式を1つのマイクロ命令で組み合わせることによって(図10参照)、高機能の順序制御を行なうためには複数のマイクロ命令を必要とするなどの空間的かつ時間的オーバーヘッドが大きい。

[QA-2] 低レベル並列処理方式によるALU演算機能は高く、それを充分に生かすためには、順序制御方式の柔軟性が不可欠である。また、ダイナミックマイクロプログラム制御方式を有効に機能させるためには、順序制御部の制御論理が多数の応用に適応可能であることが必要である。QA-2は、順序制御の論理を固定したハードウェアのみで構成するのではなく、高速アクセス可能なRAMチップを多数使用して書き換え可能な表を構成し、その中に制御論理を書き込んでおく方式を採用している。ALU演算部からのステータス信号とマイクロ命令の順序制御フィールドによって、これ5の表が駆動され、次に実行すべきマイクロ命令のフェッチを行なう。表は実行前にSVPによって書き換えられ、またユーザがSVPを使用して自由に書き換えられるので、制御論理の書き換えだけでなく、制御論理の大半をユーザ自身の問題向に適応させるダイナミックマイクロプログラミングが可能である。

また、QA-2の順序制御部は上記の表から、分岐先アドレスのみを生成するのではなく、分岐形式も生成するので、条件分岐とサブルーチンコールを同一マイクロ命令で行なうことが可能になった。(図10参照) 構造をマイクロプログラミングの考え方をマイクロ命令レベルで実現しているので、論理レベルと物理レベルのギャップが少なく、実行効率の良いマイクロプログラムを容易に作成し得る。

[QA-1]

```
IF ZOA=1 THEN GOTO LABEL1;
IF SOA=1 THEN GOTO LABEL1;
CALL SUB2;
GOTO LABEL2;
```

LABEL1:CALL SUB1;

LABEL2:

[QA-2]

```
IF ZOA=1 OR SOA=1 THEN CALL SUB1 ELSE CALL SUB2;
```

図10: 分岐形式の使用例

(3) ステータス制御方式。

[QA-1] ステータスレジスタの概念はなく、90個のステータス用フリップフロップを、マイクロ命令のフリップフロップ制御フィールドによって、ハードウェアとして固定された組み合わせから選択して設定する。それ以外の組み合わせを選択する場合には、複数のマイクロ命令を使用しなければならぬので、順序制御部のみ負担がかかってALU演算が実行できない場合も生じる。

[QA-2] 256ビットのステータスレジスタ(SSR)が装備され、96種のステータスを使用してSSRの各ビットごとに同時に、ステータスのセット/リセット/トリガを行なうことができる。SSRの設定はハードウェアの表(SMT)を用いて間接的に指定する方式なので、ユーザはこの表を自由に書き換えれば、ステータスの制御方式をユーザ自身の応用に適した構造にすることが出来る。

(4) マイクロ命令の条件分岐方式。

[QA-1] マイクロ命令のテスト条件指定フィールドにより、190種類のステータス組み合わせを判定条件として指定できる。しかし、これらの組み合わせとそれを使用した論理演算はハードウェアで固定されており、複雑な判定条件を設定するためには複数のマイクロ命令を必要とする場合がある。分岐先の限られた4方向分岐形式は存在するが、ステータスフリップフロップの内部をそのまま多方向分岐の判定条件とすることはできないので、4方向以上の多方向分岐の場合は間接多方向分岐を採用しなければならない。ユーザは、物理レベルの制約をマイクロプログラミング時に意識しなければならない。

[QA-2] 図11に従い、QA-2マイクロ命令の典型的な分岐方式について詳述する。分岐方式は3種類に大別される。この中、i)はマイクロ命令のADGフィールドを直接使用してマイクロプログラムのシケンスの分岐形式を決めるが、ii)およびiii)はいずれもCPUのステータスから各種の表を駆動してADGフィールドと同一形式のアドレス情報を生成し分岐形式を決める。

i) 無条件分岐: ADGフィールド(I DJ, A D S, D I S P, A S S, A S D)をそのまま使用して順序制御を行なう。

ii) IFJ条件による2方向分岐：①SPCフィールドで索引したSPTの情報によって、SSRの256ビットが58ビットのステータス情報を選択する。②このステータス情報とIFJCフィールドによって、TFT(論理関数を格納する表)を索引し真理値を決定する。③その真理値とIFJAフィールドによってIFJTを索引し、i)におけるADGフィールドと同一形式の情報を作成する。④以下の処理はi)と同様である。

iii) CASE条件による多方向分岐：①SPCフィールドで索引したSPT情報によって、256ビットが58ビットのステータス情報を選択する。②このステータス情報とCJBAフィールド(CJTのアドレス)を加えることによって生成されたアドレス情報がCJTを索引し、i)におけるADGフィールドと同一形式の情報を作成する。④以下の処理はi)と同様である。

3.4 システムの管理方式
[QA-1] 商用用の汎用ミニコンピュータHITAC-10が低速プログラムモードバス方式で、HITAC-10IIが高速DMA方式で結合され、HITAC-10

と10IIとによって構成されるミニコン複合体がシステムを管理する。ミニコン複合体とQA-1とのデータ転送バス幅は16ビットであり、HITAC-10の入出力制御機能や機械語命令レベルも低いので、QA-1に密結合した高速処理は困難であり、HITAC-10, 10II上で効率の良いシステム管理プログラムを作成するのは難しい。

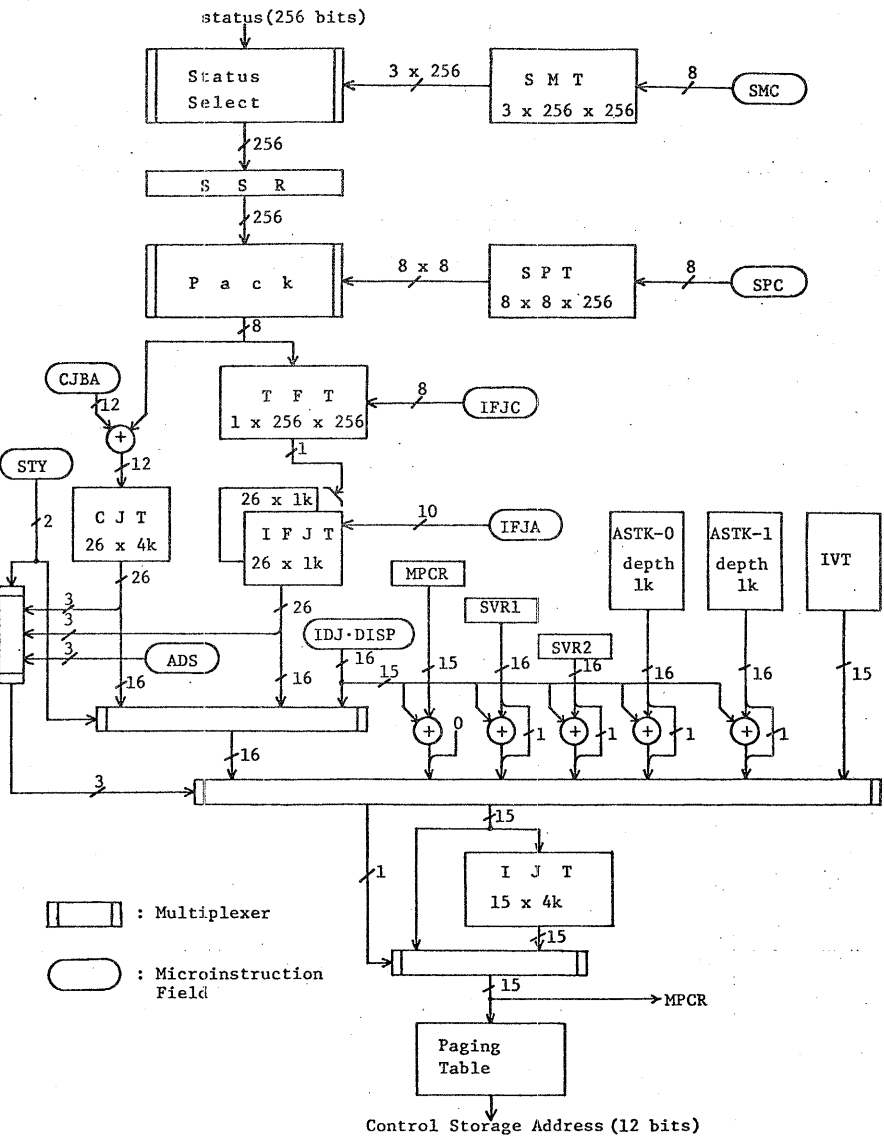


図11: QA-2 順序制御部の構成

[QA-2] QA-2と同時に新しく開発するシステム管理専用プロセッサSVPは、CPUやMMPおよび外部I/Oなど、QA-2システム全体を管理する。SVPは、ECL素子によって超高速処理機能を実現し、かつマイクロプログラム制御としてシステム管理向きの命令を豊富に作成し、CPUやMMPのアーキテクチャの変更や拡張にも柔軟に対処できる様にした。

SVPの機能を個別に掲げると次の様になる。

(1) CPUにおける演算動作の管理。

仮想制御記憶の管理、マイクロプログラムレベルのマルチプログラミングの管理。

(2) マイクロプログラムのデバッグ機能。

CPUの起動や停止の制御、マイクロプログラムのステップ実行、ブレークポイントの設定、CPUファシリティのリード/ライト操作(コントロール機能)

(3) I/Oの管理。

CPUのI/O要求の受け付け、外部I/Oの管理、ファイル管理。

(4) マイクロプログラムの作成支援システム。

プログラムのエディタ、マイクロプログラムのセンプラ、イニシャルロード、マイクロルーチンリネカ。

4. ちよび

QA-1を使用した種々の応用結果を分析して、QA-1のアーキテクチャを評価、検討するとともに、新しい可変構造形計算機QA-2のハードウェア構成方式について概説した。QA-2の設計においては、低レベル並列処理方式とマイクロプログラムの制御方式との有機的結合によって得られる高速性と柔軟性との両立を目指して作成したQA-1の基本思想をさらに掘り下げ、大規模可変構造形計算機として論理/物理構造のギャップを可能な限り小さくし、処理速度、システムの柔軟性、マイクロプログラムの生産性の飛躍的向上を図るため、一様並列演算・制御構造を有する強力なハードウェア構成方式を採用した。QA-2は、商用マイクロプログラムの制御計算機では実現不可能な大規模ユーザ・マイクロプログラミングにより、実/仮想計算機のエミュレーション、各種リアルタイム処理など、広範な応用に対して、より効率的な処理シス

テムを提供できよう。

現在、ハードウェア開発とともに、QA-2用マイクロプログラムの作成支援システムの開発を同時に進めている。ハードウェア本体は、1980年末までに完成予定である。

本研究は、文部省科学研究費補助金によっている。また、本研究に際し熱心な御討論をいただいた萩原研究室の皆様へ感謝いたします。

文献

- [1] 富田, 紫山, 小柳, 萩原: マイクロプログラムの制御による低レベル並列処理コンピュータ, 日経エレクトロニクス, (Apr. 1979).
- [2] 相泉, 川本, 紫山, 富田, 萩原: 実時間色彩動画シミュレーション—マイクロプログラム制御計算機QA-1の応用—, 信学会論文誌: Vol. J63-D, No. 2 (掲載予定).
- [3] 山下, 紫山, 富田, 萩原: ミニコンピュータのエミュレーションとその処理効率の向上について, 信学会技研報 EC79-16, (June 1979).
- [4] 十山, 紫山, 富田, 萩原: QA-1によるPascal用スタック計算機のエミュレーション, 情報会 20th全国大会, 2B-5 (Jul. 1979).
- [5] 北村, 紫山, 富田, 萩原: QA-1のファームウェアによる手続向き言語の処理について, 情報会 20th全国大会, 2B-6 (Jul. 1979).