解　説

3. ハードウェアから見た命令セットアーキテクチャ

3.2 ミニコンピュータ・ワークステーションの命令セットアーキテクチャ†

森　良　哉††

1. はじめに

ミニコンピュータには汎用中・大型機に近いものから，マイクロプロセッサに近いものまである．上位の機種では高速化が進み，場合によっては汎用機以上のコストパフォーマンスが実現できる．すなわち，ミニコンピュータは汎用機並みの速さをもち，かつ汎用機の難点であったシステム柔軟性，拡張性に優り，保守，運用面，異機種接続，ネットワーク制御，特殊装置の接続における容易さ，コストの低さを特徴としている．また，個人用の小型のミニコンピュータが機能・性能を上げ，新たな製品分野となったのがワークステーションである．

本稿は，RISC (Reduced Instruction Set Computer)/CISC (Complex Instruction Set Computer) の説明が本題ではないがミニコンピュータ・ワークステーション分野では RISC/CISC の議論がさかんであり，この話題が中心となり，命令セットアーキテクチャを狭義の命令セットに限って述べる．

RISC の命令セットはワークステーションで多く採用されている．RISC の研究は当初，ミニコンピュータの分野で始まったのであるが，それはミニコンピュータの比較的下位の機種において，ハードウェアのコストに制限のついた状況で性能向上をはかるためであった．

一方，汎用中・大型機に近い上位のミニコンピュータでは，ハードウェア実装により1命令を1マシンサイクルに近い速さで実行できるようになっていたり，多くのソフトウェア資産をもっているため，CISC を引き続き採用している．

ミニコンピュータの上位機種では，用途により CISC の命令が必要な場合がある．ワークステーションでは用途が絞られ，実行環境も UNIX, C 言語であり，RISC が適している．

RISC/CISC のいずれであっても，現在では，実行性能を問題とするようになっている．

2. ミニコンピュータの命令セット

2.1 命令の種類と機能

ミニコンピュータの命令セットは多種多様である．ミニコンピュータとして製造・販売されているものはシリーズ数でも百種以上あるが，汎用中・大型機に近いものから，マイクロプロセッサに近いものまであり，各機種の命令セットも大きく異なっている．また，下位の機種では汎用マイクロプロセッサであるモトローラ社の MC 68020 やインテル社の 80386 などを CPU としているものも多い．

図-1 に命令数からみたミニコンピュータの命令セット分布（オプション命令は含まない）を示す．

RISC 命令セットとしているものは命令数 80 から 120 の機種に分布している．多くのミニコンピュータの命令セットは 120 から 220 命令であり，これらは汎用コンピュータの基本命令セットを継承している．命令セット数が 250 を越えるものは独自の命令を増やし，CISC 化が進んだといえ，もっとも命令数の多いものは命令数 550 以上に及んでいる[1]．

基本的な命令の分類は，演算命令，転送命令，制御命令，入出力命令である．演算命令には固定小数点演算，論理演算，浮動小数点演算などがある．転送命令にはレジスタ・レジスタ間，レジスタ・メモリ間，メモリ・メモリ間の転送がある．制御命令は無条件分岐，条件分岐などがある．

ミニコンピュータでは上位の機種では汎用機なみに高級言語命令，10進演算命令，関数演算命令，文字列処理命令，データ変換命令を備えた機種も少なくない．また，OS のファームウェア化によって性能向上，保護の強化，コスト低減をはかるために，数多く

† Instruction Set Processor Architectures of Minicomputers and Workstations by Ryoya MORI (Toshiba Corporation, Fuchu Works, Process Computer Development Department).

†† (株)東芝府中工場産業用電算機開発部

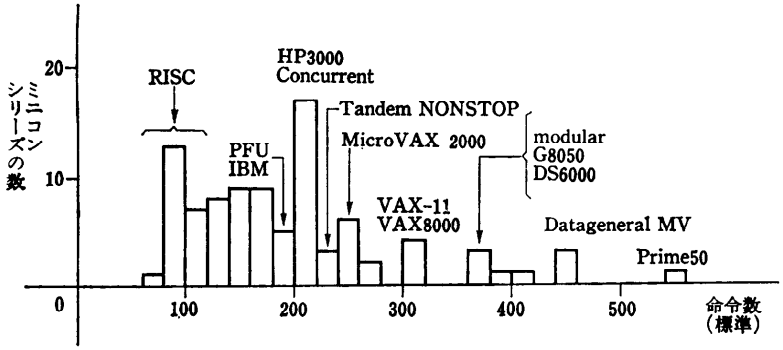
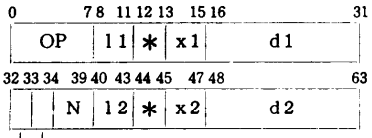


図-1 ミニコンピュータの命令数分布

10 進 演 算 命 令



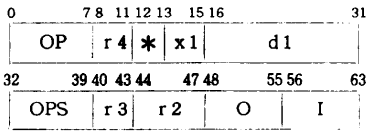
パック形式 10 進数の加算を行う.

—DIGIT
—ABSOLUTE

$(EA1 \sim EA1+l1) \leftarrow (EA1 \sim EA1+l1)+(EA2 \sim EA2+l2) * 10^{N}$

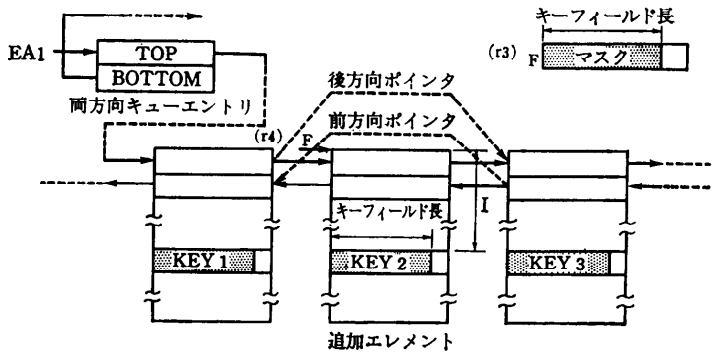
EA1, EA2: 第1，第2オペランドの指定する実行アドレス
l1, l2 : 第1，第2オペランドの指定する領域におかれるパック形式10進数の長さ
ABSOLUTE/DIGIT: パック形式10進数の制御用ビット

キ ュ ー 命 令
AODQ



キーフィールドとマスクの論理積で両方向キューを検索し，キーフィールドとマスクの論理積が昇順になる場所に追加エレメントを挿入する.

$(r2)_F$: キーフィールド長
$(r3)_F$: マスク格納アドレス
$(r4)_F$: 追加エレメントのアドレス
$(r4+1)_F$: 中間情報（命令実行中に割り込みを許す場合に，中間情報を保存する.）
I : エレメントの先頭からのキーフィールドのバイアス値
EA1 : 第1オペランドの指定する実行アドレス



(KEY1∧マスク) < (KEY2∧マスク) < (KEY3∧マスク)

図-2 CISC 命令の例

のシステム制御命令やデータ構造制御命令，たとえば，キュー命令，リスト命令，スタック命令が作られている.

CISC 命令の例を図-2 に示す.

2.2 命令セットの形式

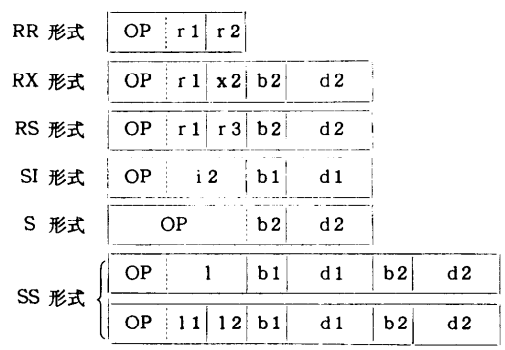
命令セットの形式には，固定長形式と可変長形式がある．固定長形式の命令セットでは，すべての命令は同一の長さをもつ．可変長形式の命令セットでは，命令の長さは命令のタイプに応じて必要な長さに変化する．固定長であると命令のフェッチやデコードが容易であり，ハードウェアの速度が上がり，また命令レベルでパイプラインの処理の効率化がはかれる．一方，可変長形式では命令列を記憶する領域が少なくてすみ，また命令セットの拡張が容易である.

RISC 命令セットの多くが固定長形式の命令セットである．例としてはヒューレットパッカード社のプレシジョン・アーキテクチャなどがある．HP プレシジョン・アーキテクチャでは，命令セット中の全命令長は 32 ビットである．さらに命令内のフォーマットも固定であり，命令コード，オペランドの命令内の位置は一定である[2),3)].

RISC 命令セットでも IBM 社の RT PC アーキテクチャは2バイト（16 ビット）長命令と4バイト（32 ビット）長命令が混在している．研究段階でのミニコンピュータ 801 プロセッサでは命令セットは4バイト長命令のみの固定形式であったが RT プロセッサではコスト性能比の面からキャシュをはずした

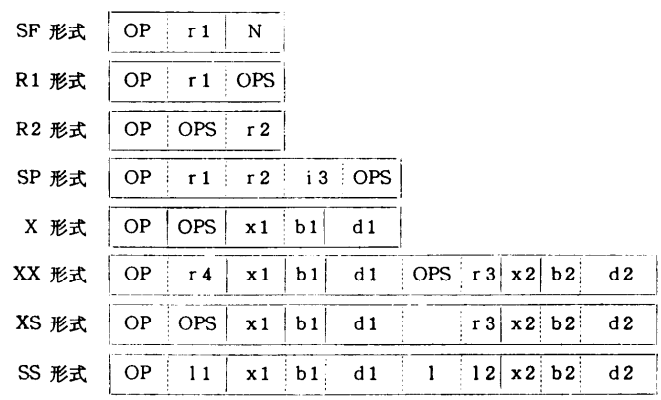
ためにメモリバンド幅を小さくする必要があったからである．使用量の多い命令は2バイト命令にするように考慮されており，平均命令長は約2.5バイトとなっている[4]．

ミニコンピュータには，汎用コンピュータに類似した命令セットのものが多く，特に IBM 社の System/360, 370 の命令セットアーキテクチャの命令形式が基本となっている．この命令セットは命令数 183，命令語長は2バイト，4バイト，6バイトの3種類であり，命令の形式は図-3 に示す6種類であった[5]．



OP　　：命令コード
r1, r2, r3：第1，第2，第3オペランドレジスタ
x2　　：第2オペランドインデックスレジスタ
b1, b2　：第1，第2オペランドベースレジスタ
d1, d2　：第1，第2オペランド変位
i2　　：第2即値オペランド
l, l1, l2：長さ，第1，第2オペランド長
r, x, b は汎用レジスタである．

図-3　命令の形式



x1：第1オペランドインデックスレジスタ
N　：命令コード修飾値
i3：第3オペランド即値
r4：第4オペランドレジスタ
OPS：補助命令コード
x, b を専用レジスタとする場合もある．

図-4　追加された命令の形式

この形式のオペランドは，第1，第2，第3の3オペランドがあり，レジスタ・オペランド，主記憶オペランド，即値オペランドの3種類がある．

その後，各ミニコンピュータが独自に命令を高機能化し，図-4 に示すようなさまざまな形式がつくられた．命令語長が8バイトのもの，第4オペランドがあるもの，命令コードに命令補助コードのあるものなどが追加されている．

可変長形式の1例が，DEC 社の VAX ファミリの命令セットである．この命令セットでは，命令コードは1バイトまたは2バイトで，その後ろに，命令コードに応じて0個から6個のオペランド指定子が続く．各オペランド指定子の長さは1バイトから 10 バイトまで可能であるが，オペランド指定子の形式は固定で，アドレッシング・モードと追加情報からなる[6]．

2.3　命令の CISC 化

初期の（1950 年前後）コンピュータは命令数が少なく，各命令は単機能であった．ミニコンピュータの初期のもの（1970 年前後）も乗除算機能がないなど，演算機能は単純であった．1970 年代後半から，ミニコンピュータの機能，性能の向上がはかられ，高機能命令が追加された．

例）　乗除算命令の追加，アドレッシング・モードの増加

命令の複雑化を推進したのは，半導体の集積度の向上とマイクロプログラミング技術の発達である．

1970 年代では主記憶装置の容量当たりのコストが高く，アクセス速度も演算装置の速度に比べて遅かったため，プログラムのステップ数を少なくし，一つの命令で複数の動作をさせることが必要とされた．

コンパイラ技術が成長中の段階には，高級言語の高機能なステートメントを単純な命令列に変換するのに限界があり，また，ステートメントをそのまま実行するほうが高速なので高級言語計算機の考え方が普及した．そして，セマンティック・ギャップをうめるために高級言語に近い命令語がマイクロプログラミングにより実装された．

また，高機能，高性能な機種を開発するに当たり，旧機種用ソフトウェアとの互換性を保つためにも既存の命令を継承しつつ命令セットが設計された．

命令数の増加の主な要因は次の項目である.

- 旧機種との互換性のための引継.
- RAS 命令（信頼性，可用性，保守性）.
- OS 強化命令，高機能命令.
- 高級言語対応命令.
- 仮想記憶対応，マルチプロセッサ対応.
- リアルタイム処理での割り込みに対する不可分な処理対応.

CISC 傾向が進むにつれて，実際には使用頻度の非常に低い命令までハードウェア化されたためいくつかの弊害が生じた．それらを以下に示す.

・命令が複雑化したため，デコードに要する時間が増加した.

・アドレッシング・モードの増加によりデータバスにマルチプレクサやバスドライバが加えられ，ゲート数が増加し配線が長くなり，遅延時間が増加した.

・ハードウェア規模の増大による大容量電源と発熱量増加に対する強力な冷却能力が必要となった.

・ハードウェア開発に要する時間が増加した.

2.4 RISC の特長[7]

CISC で発生した弊害は，汎用機では対策が容易であったが，コスト面でハードウェア量の制約が厳しいミニコンピュータ，特に中位以下の機種には大きな問題となった.

そこで，CISC 化の行き過ぎに対する見直しが始まり，基本的な演算と操作を中心とする小さな命令セットを基本概念とする RISC 技術が 1980 年代前半から育ってきた．それは，CISC 化を支えてきたマイクロプログラム技術の成熟期に一致している.

RISC の文字どおりの意味は「小さい命令セット」であるが厳密に定義されてはいない．しかし，一般に RISC と呼ばれる場合の特長として次の 6 項目があげられている.

（1） 少ない命令数と少ないアドレッシング・モード.

（2） 固定命令形式.

（3） 1 命令を 1 マシンサイクルで実行.

（4） メモリアクセスを LOAD/STORE 命令などに限定（ロード・ストア方式）.

（5） マイクロプログラム制御ではなく，ワイヤド・ロジックで命令実行.

（6） 最適化コンパイラ.

2.5 CISC の RISC 的アプローチ

RISC のアイディアは，過度に CISC 化したミニコンピュータのアーキテクチャに対し反省をうながした．CISC のマシンの再検討が行われ，はじめは，RISC の特長とされていたものが CISC にも利用されている．また，パイプラインを意識した最適化コンパイラ，大きなレジスタファイルのアイディアなどは，CISC での研究に RISC での研究が刺激を与えた．現在，RISC 製品にしかみられない特長は次の二つになっている.

・命令が固定長.

・メモリ参照が LOAD/STORE 命令のみ.

たとえば，VAX 8700 では命令セットレベルでは従来のアーキテクチャを踏襲しているが，マイクロ命令レベルでは RISC ふうのアプローチをとっている．主なポイントは次のような項目である[8].

（1） マイクロ命令での単一命令形式の採用.

（2） マイクロ命令でのサイクル・タイムの縮小.

（3） マイクロ命令でのディレイド・ブランチ.

（4） 高頻度命令を選択的に高速化（ハードウェア化，手続き呼び出しの高速化）.

また，大きなレジスタ・ファイルの効果はこれを採用した CISC でも顕著である．ディレイド・ブランチ機構は CISC でもマイクロプログラムのレベルでは従来から採用されていた.

上位のミニコンピュータでは，その用途の一つであるリアルタイム処理において，CISC 命令が有効である．たとえば，カウンタ操作，すなわち，読み出し，加／減算，書き込みの一連の処理が不可分であったりキュー操作中の割り込みを抑止したりする場合があるからである．これを RISC 命令セットで実現しようとすると，スーパバイザコールのオーバヘッドがあったり，割り込みの禁止のようなシステムに重大な影響を与える命令を非特権モードで利用できるようにしなければならない.

一方，RISC 製品として市場にあるものは，純粋な RISC の定義にこだわることなく，さまざまな工夫が適用され RISC ふう製品となっている.

AT & T ベル研究所の CRISP アーキテクチャでは，C 言語を効率よく実行するために，小さな命令セットではあるが，可変長命令形式を採用し，プログラムからみえる汎用レジスタがないメモリ・メモリ間アーキテクチャを採用している．最適化コンパイラがなくても容易にコード生成ができるように考慮されており，ミニコンピュータあるいはワークステーションとしての製品はまだ出ていないが，UNIX，C 言語

の環境が主流のこの分野では影響力がありそうである[9)-11)].

3. ワークステーションの命令セット

RISC アーキテクチャの採用により話題が多いのがワークステーションの分野である．ただし，オフィスワークステーションの分野では CPU はインテル社の 8086 系，80286，80386 マイクロプロセッサ，モトローラ社の MC 68000 系マイクロプロセッサなどが用いられている．エンジニアリングワークステーションの分野ではモトローラ社の MC 68000 系マイクロプロセッサを CPU とするものが多いが，独自の RISC アーキテクチャをもつ製品が増えてきた[12)]．

従来は，ワークステーション専門のメーカでは標準的なマイクロプロセッサ MC 68000 などを用いることにより，アプリケーションソフトウェアの移植性に強みをもっていた．一方，ミニコンピュータメーカは，上位のミニコンピュータとのソフト互換性をポイントとしていた．

RISC アーキテクチャを採用したワークステーションも純粋に 2.5 で述べた全項目を満たすものは少ない．現状では，RISC 命令セットに必ず含まれるべき命令というものは決まっていない．命令セットに含んではならない命令というものもない．さまざまな定義がなされている．たとえば，カリフォルニア大学の RISC Ⅱの命令数が 39 であったのに対し，各 RISC ワークステーションの命令数は 80〜120 と多くなっている[13)]．

IMB の RT プロセッサでは，「単純な 1 サイクル命令の概念を採用」という意味で RISC と呼んでおり，必ずしも命令数が少ないという意味ではない．さらにハードウェアがそのままでサポートできる範囲内であればコンパイラにとって有効な機能，コンパイラが頻繁に使う機能を命令セットに含む方針である．したがって，CISC ふうな “Shift Left Paired Immediate Plus 16” といった命令が命令セットに含まれている[4)]．

ワークステーションは RISC を採用しやすい環境にあると言える．すなわち，UNIX をベースとして，ソフトウェアはすべて高級言語（C言語）で作る傾向が最初から強く，過去のソフトウェア資産のしがらみも少ない．実装上，大きさの問題，コストの問題からハードウェア量に制限がある．多ユーザではなく，タスク数が少なくてよい．タスク切り替えが少なく，OS のオーバヘッドが目立たない．エンジニアリングなど用途を絞れ，RAS より性能にポイントを置く．などが理由としてあげられる．

今後 RISC の採用により，ワークステーションはさらに小型化が進み，また RISC チップにコプロセッサをつけることにより，特化が進むであろう．

RISC アーキテクチャのワークステーションをいくつか紹介する．

（1） Apollo 1000 PRISM

最大 4 プロセッサ・ユニットのマルチプロセッサ構成が可能．1 プロセッサ・ユニット内でも複数の命令を並列に実行できる．命令数は 103，命令長は 32 ビットで，すべての演算を 1 サイクル内で実行．プロセッサ・ユニットは整数演算プロセッサ，浮動小数点加算プロセッサ，浮動小数点乗算プロセッサからなる．

（2） HP プレシジョン アーキテクチャ

ワークステーションだけでなく，ミニコンピュータにも RISC を採用している．命令数 138，命令長 32 ビットで固定形式，ハードワイヤドで 1 サイクル／命令，メモリアクセスは LOAD/STORE のみである．HP 社の従来のアーキテクチャをエミュレートする場合も考慮されている．

（3） IBM RT PC アーキテクチャ

RISC アーキテクチャの ROMP を CPU とする．命令数は 118，2 バイト命令と 4 バイト命令が混在するが，多くは 2 バイト命令であり平均の命令長は約 2.5 バイトである．マイクロプログラム方式のプロセッサであるが大部分の命令は 1 サイクルで実行される．

（4） SUN SPARC アーキテクチャ

ワークステーション専用チップの命令セット，命令数は 89 であり，命令長は 32 ビットで命令形式は 3 種類に大別できる．メモリアクセスは LOAD/STORE のみ．演算命令の基本は 2 ソースレジスタ，1 ディスティネーションの 3 レジスタオペランド指定である．1 ソースレジスタは 13 ビット符号即値データとすることも可能である．ほとんどの命令が 1 サイクルで実行される．

4. 技術の実際

ミニコンピュータの命令セットに関する技術の話題は，パイプライン，マイクロプログラム技術とその生産性，WCS（書き込み可能制御記憶）とキャシュメモリ，最適化コンパイラなどがあるが，ここではパイプラインと最適化コンパイラについて述べる．

4.1 パイプライン

RISC/CISC のいずれであっても，1命令を1マシンサイクルで実行することを目標とする．そのために一つの命令を数段のステージに分割し，パイプライン処理を行う．パイプラインの1段を1マシンサイクルで実行することにより，見かけ上の1命令／1マシンサイクルの実行効率を出す．

1命令の機能分割は基本的には次のようになる．

（1） 命令フェッチ．

（2） デコード．

（3） オペランド読み出し．

（4） 操作．

（5） 結果の書き出し．

パイプラインでは，これらのステージの長さが等しいほうがよく，もっとも時間の長いステージがマシンサイクルを決めるため，短いステージを合わせて1段とする．

RISC では，この詰合せを行いやすい．これは，命令のデコード時間が短い，オペランドがレジスタで読み出し時間が短い，機能が小さく操作時間が短い，結果の書き出し先がレジスタで時間が短いなどの理由による．したがって，パイプラインの段数は3段前後である．

図-5 に RISC でのパイプラインの例を示す[14]．

パイプラインの段数が多いと次の問題が発生する．

・オペランド・ハザードの頻度が増加する．

・パイプライン制御および分岐時のパイプラインキャンセルのオーバヘッドが増加する．

・多くの命令は3から4段で実行を終えるので，無駄な段数実行が増加する．

一方，CISC のマシンではパイプラインの1段にかかる時間を短くするために，段数を増やしている．**図-6** に DS 6060 の例を示す．この例では8段のパイプラインであるが，I, D, R のステージでは命令不在やメモリリクエストの待ち，C, M, E, W のステージではオペランド不在や操作の待ちが発生する場合がある．そのため，Qステージで4レベルの待行列で分離を行い，パイプラインの流れを効率良くしている．

VAX 8800 では CISC である命令を，RISC ふうのマイクロ命令にマッピングし，RISC 型のユニフォームパイプライン処理ユニットを通している．パイプラインのステージは次の5段である．

（1） デコード．

（2） マイクロ命令デコードとシーケンシング．

（3） マイクロ命令オペランド転送，レジスタ読み込み．

（4） マイクロ命令実行．

（5） 結果のレジスタへの書出し，メモリアクセス．

4.2 コンパイラ技術の実際

RISC/CISC のいずれの命令セットであってもパイプラインを意識したコンパイラ，最適化コンパイラの実現がシステムとして大きなポイントとなる．

最適化コンパイラは，C言語に関して一般に論じられるが，これからのミニコンピュータやワークステーションの用途の広がりを考えると他の言語でも重要となるであろう．

ここでは，1例として PROLOG の場合を示す．PROLOG の高速処理系は，処理をハードウェアでサポートしようとするのが主流になっており，WAM (Warren's Abstruct Machine) コードを直接実行する専用マシンが作られている．これは PROLOG がパターンマッチングやバックトラッキングなどの手続き型言語とは異なる実行方式をもつからである．しかし，システム全体としての性能，実行効率を上げるために主導権をソフトウェアで支配しようとする流れがミニコンピュータやワークステーションにはある．

表-1 に汎用ミニコンピュータ

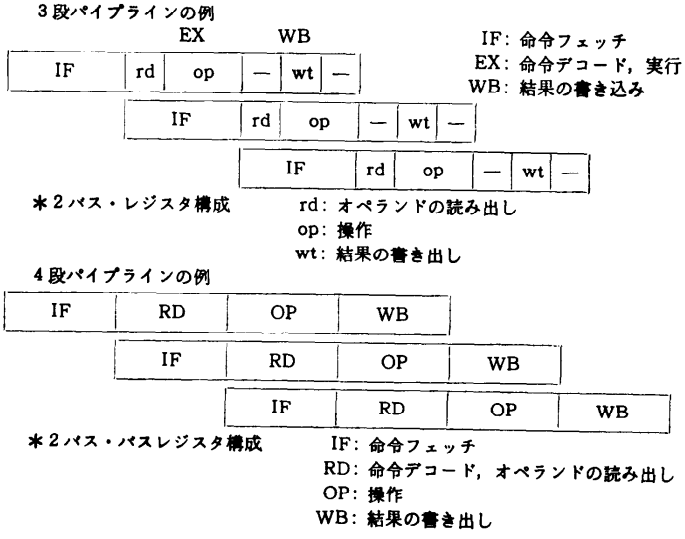
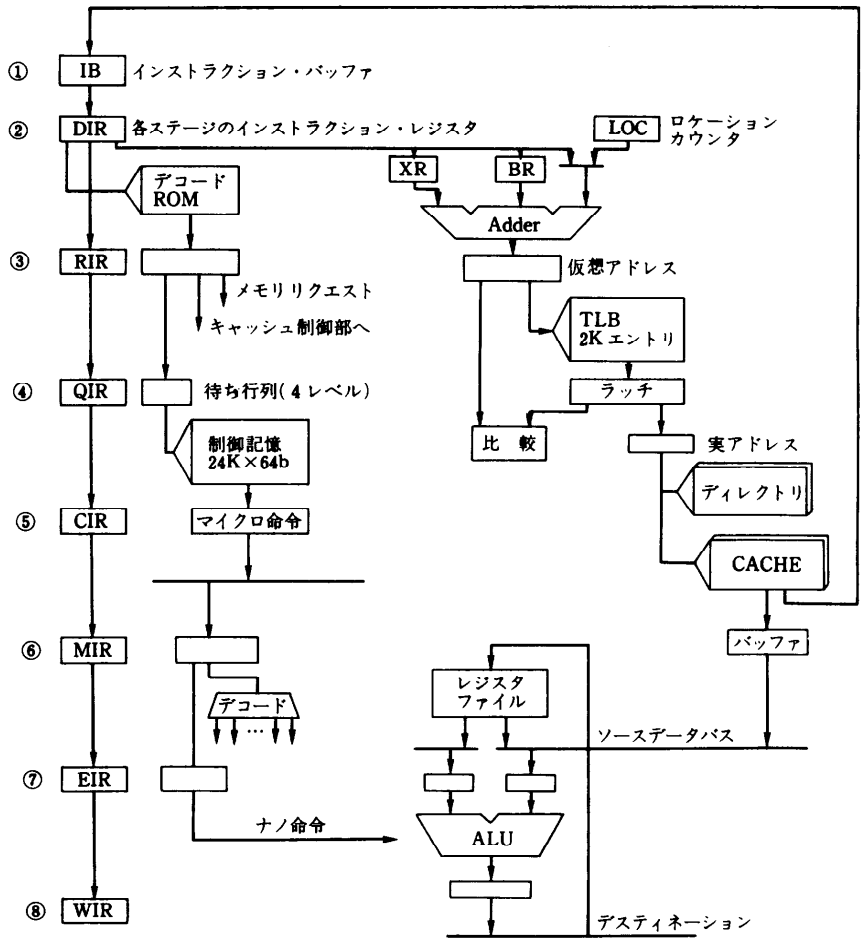


図-5 RISC のパイプライン



① Iステージ：命令フェッチ
② Dステージ：命令のデコード，仮想アドレス計算
③ Rステージ：メモリリクエスト出力，アドレス変換
④ Qステージ：解読済み命令のキューイング，制御記憶のアクセス
⑤ Cステージ：オペランドキャッシュの読み出し
⑥ Mステージ：マイクロ命令の実行，ソースデータの出力
⑦ Eステージ：演算実行
⑧ Wステージ：演算結果のデスティネーション

図-6 G8000 のパイプライン

G 8000 (CISC), ワークステーション AS 4260 (SPARC =RISC), ワークステーション AS 3260 (MC 68020=CISC) の最適化 C-PROLOG コンパイラの実行効率比較を示す[15),16)].

表-1 最適化 C-PROLOG コンパイラの実行効率比較
単位 KLIPS キャッシュミスヒット有

機種 / プログラム	G8000 (CISC)	AS 3260 (汎用マイクロプロセッサ)	AS 4260 (RISC チップ)
append	280	189	787
nreverse	224	186	694
qsort	110	98	315
8 queen	129	126	303
マシンサイクル	80 ns	40 ns	60 ns

最適化コンパイラは，G 8000 ではパイプラインを乱さないことにポイントが置かれるが，SPARC の命令セット (RISC) では，メモリアクセスを減らすことにも注力される．

CISC の汎用ミニコンピュータ G 8000 においても最適化したオブジェクトコードで使用する命令は，RISC の命令セットに含まれている命令と同様となる．これらの測定対象で，RISC になく CISC 命令セット内で有効であるのは，BFCR/BTCR (Branch on False/True Condition by Register) 命令のみである．

5. おわりに

上位のミニコンピュータでは，TSS 環境を取り上げても数 10 のユーザをサポートしており，タスク・スイッチングの高速性やデータ・ハンドリング能力が大きな要素を占め，CISC のほうが有利である．また，データベース管理などの事務処理やプロセス・コントロール分野では，頻繁にスワッピングされる大容量のディスク装置やリアル・タイムで入出力されるプロセス I/O 装置を管理するため，強力なバスをもち，専用の命令を備える必要がある．

そして，従来 CISC アーキテクチャで構成されたコンピュータを製造していたメーカでは，性能問題，ハードウェア開発量の問題だけで RISC の採用を検討することは容易ではない．従来機用に開発されたソフトウェアの蓄積を無視することができないからである．RISC アーキテクチャを指向していても命令語が可変長であったり，マイクロプログラム命令方式を併用しているミニコンピュータはこの点からの制限によるものである．

また，一つのメーカが二つのアーキテクチャをサポートすることは，保守部品や保守人員，あるいはソフトウェア開発が二重になることを意味しており，メーカにとって大きな負担となる．

ワークステーションでは，プログラムは UNIX 環境で動作し，小規模の科学技術計算処理などの比率が高く，同時に実行されるプロセスの数もそれほど多くない．したがって RISC との相性がよいといえる．ワークステーションによってはプロセスごとに仮想記憶機構やレジスタ群を用意して，比較的少ないプロセス数ではタスク・スイッチングを高速化しているものもある．またコンパクト化の要請という点からみると，少数の VLSI で構成することを指向する RISC のほうがかなり有利である．

RISC のハードウェア構造が簡単であるので開発工数が少なくて済むという利点は，CAD技術の進展により CISC との差が減少しても無視できない．半導体技術の最新の成果の恩恵は RISC のほうが受けやすく，しばらくはミニコンピュータの性能向上よりもワークステーションの性能向上の伸びが目立つであろう．

参考文献

1) 日経データプロ編集：日経データプロ EDP ミニコンピュータ性能・機能一覧表，pp. DP-1-090-503-pp. DP-090-550, 日経 BP 社 (1988).
2) Fotland, D. A. et al.: Hardware Design of the First HP Precision Architecture Computer, HEWLETT-PACKARD JOURNAL (Vol. 38, No. 3, pp. 4-17 (1987).
3) Fotland, D. A. 他：RISC アーキテクチャを採用した 4.5 MIPS のミニコン，日経エレクトロニクス，No. 415, pp. 185-204 (1987).
4) Richard O. Simpson: IBM RT PC に見る RISC アーキテクチャ，日経エレクトロニクス，No. 415, pp. 185-204 (1987).
5) 金山 裕:アセンブラプログラミング入門 IBM System/370 のための，pp. 1-13, 近代科学社 (1977).
6) 日本ディジタルイクイップメント(株)教育部編；VAX アーキテクチャ・ハンドブック，pp. 63-114, 共立出版 (1984).
7) Patterson, D. A.: Reduced Instruction Set Computers, Comm. ACM, Vol. 28, No. 1, pp. 8-21 (1985).
8) Clark, D. W.: Pipelining and Performance in the VAX 8800 Processor, Proceedings of ASPLOS II, pp. 173-177 (1987).
9) Berenbaum, A. D. et al.: Introduction to the CRISP Instruction Set Architecture, Proceedings of the Spring 1987 COMPCON, pp. 86-90 (1987).
10) Berenbaum, A. D. et al.: Architectual Innovations in the CRISP Micro Processor, Proceedings of the Spring 1987 COMPCON, pp. 91-95 (1987).
11) Bardyopadhyay, S. et al.: Compiling for the CRISP Microprocessor, Proceedings of the Spring 1987 COMPCON, pp. 96-100 (1987).
12) 日経データプロ編集：日経データプロ EDP エンジニアリングワークステーション性能・機能一覧表，pp. DP 1-080-503-pp. DP 1-080-526, 日経 BP 社 (1988).
13) Patterson, D. A. et al.: A VLSI RISC, IEEE Computer, Vol. 15, No. 9, pp. 8-21 (1982).
14) 金城守茂他：スーパーマルチプロセサ DS 6060 (3)：情報処理学会第 33 回全国大会講演論文集，4C-3, pp. 213-214 (1986).
15) 稲葉則夫：SPARC 上で 787 KLIPS がでる Prolog コンパイラを開発，日経エレクトロニクス，No. 454, pp. 110-111 (1988).
16) 松本憲幸他：汎用ミニコンの最適化 C-Prolog 情報処理学会研究報告(記号処理)，88-SYM-44-1, pp. 1-7 (1988).

(昭和 63 年 10 月 7 日受付)