高集積マイクロコンピュータに適したマイクロプログラム制御方式†

前島英雄†† 桂 晃洋††
安田 元††† 木原利昌†††

本論文では，高集積マイクロコンピュータに適した新しいマイクロプログラム制御方式について述べる．マイクロコンピュータの性能・柔軟性・信頼性を向上するためのマイクロプログラム制御方式として2つの主要な手法を提案する．

まず，高い性能及び柔軟性を得るため，マクロ命令解読機能をマイクロプログラム・メモリ（μ-ROM）と一体化する構成をとる．このような構成に伴い，マクロ命令の解読は命令デコーダといった付加機構を介することなく，直接，μ-ROM のアドレス・デコーダ上で実現する．すなわち，マクロ命令からマイクロ命令実行までの応答時間を縮めることで高速性を得る．また，柔軟性に関してはマクロ命令にページ情報を持たせることでμ-ROMを論理的なページに分割し，ページ間でのアドレス・パターンに独立性を与える方法で実現できることを示す．次に，高信頼性を得るため，マイクロ命令の実行シーケンスの合理性をチェックする．すなわち，マイクロ命令中に実行可能な位相を指す位相指示ビットを与え，これを基準クロックの位相と比較することで実現する．

以上の手法により，マクロ命令の解読時間が省略されるとともに，複数の命令セットがμ-ROM の内容だけを変更することで柔軟に対処できる．さらに，マイクロ命令の実行レベルでマイクロプログラムの暴走を効果的に検出し得る．

1. まえがき

近年，半導体技術とりわけ MOS (Metal Oxide Semiconductor) 技術の進歩には著しいものがあり，その寸法は年20%ほどの速度で微細化されつつある．これに伴い，マイクロコンピュータはより高集積になってきており，現在では VLSI (Very Large Scale Integration) と呼ばれるデバイスも出現している．これらの LSI には多くの機能がわずか 5～6 ミリ角のチップ上に集積されており，主メモリ用の ROM (Read Only Memory)，RAM (Random Access Memory) や周辺 I/O (Input/Output) を搭載した4～8ビット・シングルチップ・マイクロコンピュータ[1]，ミニコンピュータの性能・機能を備えた 16 ビット・マイクロプロセッサ[2)~5)]といった形で出現している．これらの LSI はほとんどが微細な MOS プロセスを用いており，数万素子を1チップに集積している．

以上のような状況の下では，膨大な論理がゆえに設計作業はなかなか収束せず，高集積化に伴って開発期間がますます延びてしまう．設計作業の単純化あるいは機械化が VLSI 設計技術として重要なファクタになりつつある．この段階ではもはやデバイスの問題ではなく，VLSI に向いた論理回路の問題に帰着する．マイクロコンピュータのような論理 LSI においてもメモリ LSI と同様に規則性を持ち，単純かつ整然とした構造が要求される．結局，半導体プロセスの微細化に対応した設計技術が不可欠なものとなっている．

これまで，多くのマイクロコンピュータではいわゆるランダム論理方式がとられていた．これは言い換えれば1つのマイクロコンピュータに対する専用制御方式であるため，論理回路としては最適化されており，高速かつチップ面積小という大きな利点を生み出していた．反面，論理の柔軟性に欠けている．高集積化に伴い，論理の複雑さ（設計の難しさ）がゆえに論理不良が発生しやすく，その発見・修正も困難になってくる．さらに，命令機能の増強やファミリ展開に対しても同様に膨大な作業量を必要とする．これに対し，ランダム論理をマイクロコード化し，これをROMに集約するマイクロプログラム制御方式は論理を規則化するとともにプロセッサに柔軟性を与える特徴がある．

コンピュータの分野では，古くからハードウェアのソフト化（ファームウェア）を強く意識し，マイクロプログラム方式を積極的に導入している．これはシス

† A Microprogram Control Method Applied to Highly Integrated Microcomputers by HIDEO MAEJIMA, KOYO KATSURA (The 8th Dept., Hitachi Research Laboratory, Hitachi, Ltd.), HAJIME YASUDA and TOSHIAKI KIHARA (Microcomputer Device Engineering Dept., Musashi Works, Hitachi, Ltd.).

†† 日立製作所日立研究所第8部
††† 日立製作所武蔵工場マイクロコンピュータ・デバイス設計部

テマティックな設計による設計のしやすさと仕様変更・追加を容易にする柔軟性を選んだからである．最新のマイクロコンピュータ[2),4)]でも同じ思想に立って，マイクロプログラム方式が採用されつつあり，VLSI 化にとって有力な解であることが示されている．この理由は，MOS プロセスの微細化に伴い，マイクロプログラム方式の欠点といわれている低速，チップ面積大の問題が解決しつつあるからである．

本論文では，高集積のマイクロコンピュータにおける論理方式として，マイクロプログラム制御方式をとり挙げ，その高速性，柔軟性，信頼性につき論じる．高速性，柔軟性に関しては，マクロ命令の解読機能を内蔵するマイクロプログラム・メモリ (ROM) 構造を，信頼性に関しては，マイクロプログラム・シーケンス監視によるハードウェア・エラーの検出等の新しい方式を提案する．

2. マイクロコンピュータ論理方式の必須条件

2.1 高 速 性

プロセッサの性能は，デバイスの性能と論理方式（アーキテクチャ）に依存する．前章では，VLSI に向いた論理方式としてマイクロプログラム制御方式を提案した．本節では，マイクロプログラム制御機構を高速化する必要性を示す．始めに，従来のマイクロプログラム制御構造を **図 1** に示し，その問題点を明らかにする．この構成では，まず，マクロ命令が命令レジスタ (IR) に読み出され，命令デコーダ (ID) によって解読される．解読結果はマイクロプログラム・メモリ (μ-ROM) の1アドレス（エントリ・アドレス）として得られる．これを ROM アドレス・レジスタ (RAR) に格納してマイクロ命令の1つをマイクロ命令レジスタ (MIR) に読み出す．このマイクロ・シーケンスのタイム・チャートを **図 2** に示したが，これから明らかなようにマクロ命令の解読期間 (ID の信号遅延時間) に「待ち時間」が生じてしまう．これはマクロ命令実行に要するマイクロ・サイクル数を常に1サイクル増加する．したがって，マイクロ・サイクル数の少ない命令にとって不都合なものとなり，プロセッサの高速性に重大な影響を与える．マイクロ・サイクル数の実効的な短縮はパイプライン制御等の諸技術によりある程度まで可能である．しかし，LSI では自由度の高い回路を駆使できる大きな利点があり，これによって解決することが論理設計の単純化の見地からも理想である．

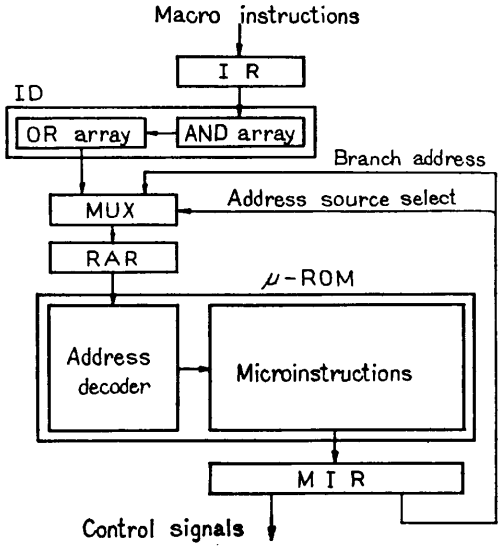


図 1 従来のマイクロプログラム制御構造

Fig. 1 Organization of a traditional microprogram control unit.

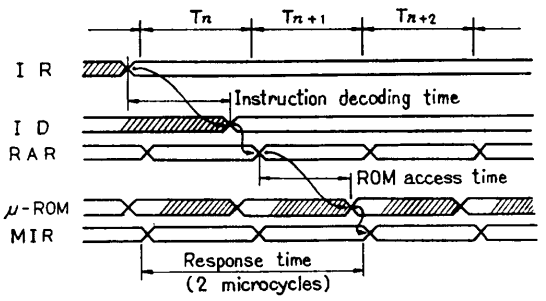


図 2 従来構成におけるタイム・チャート

Fig. 2 Time chart of the traditional Organization.

本論文では，こうした発想からマクロ命令解読機能内蔵形の μ-ROM 構造を提案するものである．

2.2 柔 軟 性

開発した1つのマイクロコンピュータにおいて，その論理構成の柔軟性が高ければファミリ展開によって適用範囲が広がり，ハードウェアのリピータビリティを高める．ところが，通信・端末制御等各分野ごとに最適なハードウェアを同時に内蔵することは汎用のマイクロコンピュータにとって難しい．すなわち，応用に適する命令セットがそれぞれの分野ごとに必要である．図1に示したマイクロプログラム制御構造によって得られる利点は，第1に，μ-ROM 中のマイクロ命令の追加・変更によって論理修正を容易にする柔軟性

にある．第2に，マクロ命令デコーダ (ID) でマクロ命令解読の自由度を維持するため，より柔軟性を高めることにある．したがって，マクロ命令解読機能内蔵形の μ-ROM には上記2点の機能を組み込ませる必要がある．

2.3 信 頼 性

マイクロプログラム制御構造における信頼性の配慮は，一般的に，マイクロ命令を格納する μ-ROM のパリティ・チェックにとどまる．これは，まさに実行しようとするマイクロ命令のパターン・チェックであり，マイクロ・シーケンスには全く係わり合いがない．ところが，最も致命的なのはマイクロ・シーケンスの乱れ，すなわち，暴走である．

本論文では，マイクロ・シーケンスの合理性チェックとして位相比較方式を提案し，簡単なハードウェアの付加によってマイクロコンピュータのハードウェアの中枢であるマイクロプログラム制御部の信頼性を高める手段を示す．

3. マクロ命令デコード方式

図1で示したマイクロプログラム制御構造における命令デコーダ (ID) は，マクロ命令語を対応するマイクロプログラムのエントリ・アドレスに変換する機能を持つものである．マクロ命令語の読み出しからマイクロ命令読み出しまでの一連の流れに注目すると，2つの情報変換がなされていることが分かる．すなわち，第1の変換はマクロ命令語からマイクロプログラムのエントリ・アドレスの生成，第2の変換はエントリ・アドレスからマイクロ命令語の読み出しである．これをマクロ命令からマイクロ命令読み出しの直接変換に置き換えれば，図1における命令デコーダ (ID) が省略できる．このようなマクロ命令からマイクロプログラムへの直接マッピングを実施するプロセッサはすでに存在しているが[6]，本論文で提案するマッピング方式は次の2点を特徴とするものである．

(1) 柔軟な制御構造

マクロ命令は一般にオペコード，レジスタ，アドレッシングなど複数フィールドから成る．提案するマイクロプログラム方式では，これらすべてを μ-ROM 上で処理する事によりマクロ命令形式に制約されない柔軟な制御構造を得ようとするものである．

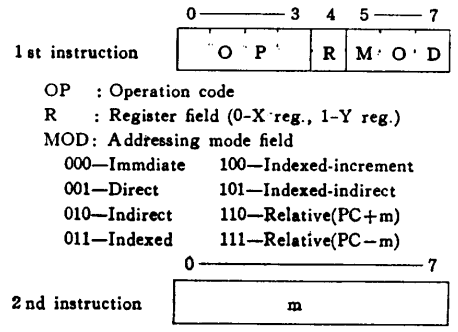
(2) LSI の特徴を活かした μ-ROM 構造

LSI では限られたチップ面積の中に所定の機能を集積しなければならない．反面，目的に合せて回路設計が行えるという大きな利点がある．本論文では PLA (Programmable Logic Array) と ROM を一体化し，アドレスとこれに対応するマイクロ命令パターンを同時設計する事で効率のよい μ-ROM 構造の実現を図る．

以上の特徴を持つ μ-ROM の構成及びその原理を1つのマクロ命令形式を例にとって述べる．

3.1 マクロ命令形式

図3 に8ビット・マイクロプロセッサのマクロ命令形式を一例として示す．第1語目は各種演算モードを指定する OP，演算レジスタを指定する R，各種アドレッシング・モードを指定する MOD の3つのフィールド，第2語目はアドレスあるいはイミーディエイト・データとなる m フィールドから成る．本マクロ命令形式を例題に，これを解読しマイクロ動作の制



OP : Operation code
R : Register field (0-X reg., 1-Y reg.)
MOD: Addressing mode field
000—Immdiate 100—Indexed-increment
001—Direct 101—Indexed-indirect
010—Indirect 110—Relative(PC+m)
011—Indexed 111—Relative(PC−m)
m : Immediate data/Address

3 図 標準命令フォーマットの一例

Fig. 3 Example of standard instruction format.

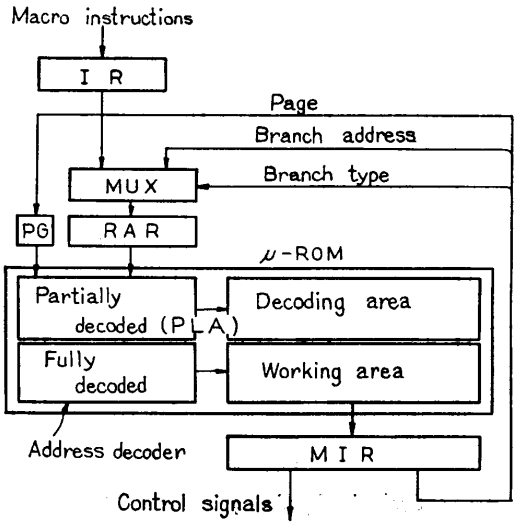


図 4 提案するマイクロプログラム制御構造

Fig. 4 Organization of the proposed microprogram control unit.

御を行うマイクロプログラム制御構造を示す.

3.2 新マイクロプログラム制御機構の構造

前記したように，本論文で提案する μ-ROM ではマクロ命令の解読機能を含むため，**図4**に示す2つの部分に分けられる．第1の部分はマクロ命令解読を行うデコード領域，第2の部分は通常のマイクロ動作を行うワーク領域である．後者については従来の構造（図1）と全く同じであるから本論文では省略し，前者についてのみ述べる.

マクロ命令を解読する際には，図3に示した形式からも明らかなように OP，R，MOD の各フィールドが互いに独立しているため，命令語のパターンから1つのフィールドを切り出してデコードしなければならない．この実現にはROMのアドレス・デコードを部分的に行えるようにすることで対処できる．ハードウェアで考えると ROM ではなく，PLA に相当する．すなわち，マクロ命令の内容に応じてマイクロ命令が決定されるデコード・サイクルにおいては，PLA で構成したデコード領域に配置したマイクロ命令の1つを読み出す．**図5** に MOD フィールドのデコード例を示す．このデコードには，OP，Rフィールド（ビット0～4）は不要であり，その部分はアドレス・デコーダにおいて "don't care" としてハード的にマスクをかけている．また，それぞれのアドレスに対応するマイクロ命令は1つずつとし，それ以後に続くマイクロ命令はワーク領域に配置して通常のマイクロプログラム制御と同様に扱う.

以上のように，μ-ROM としてデコード領域に PLA を用い，ワーク領域に ROM を用い，これらを1つの ROM として構成することでマクロ命令解読機能をμ-ROM中に埋没する構造が得られる．また，

210	7654321 0	MICRO INSTRUCTION
000	XXXXX000	NO OPERATION
000	XXXXX001	MAR <— MDR
000	XXXXX010	MAR <— MDR
000	XXXXX011	MAR <— IXR + MDR
000	XXXXX10X	MAR <— IXR + MDR
000	XXXXX110	MAR <— PC + MDR
000	XXXXX111	MAR <— PC − MDR

Page　Don't care　MOD field

図5 MOD フィールドのデコード例
Fig. 5 Example of MOD field decoding.

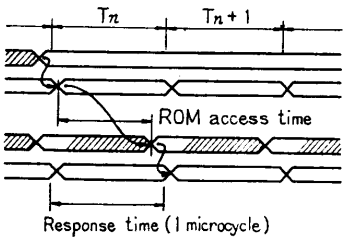


図6 新構成におけるタイムチャート
Fig. 6 Time chart of the new organization.

図6に本μ-ROMの動作タイミングを示すが，命令デコードに要する時間がなくなり，マクロ命令が命令レジスタ (IR) に取り込まれ，1マイクロ・サイクル後にはこの内容に従ってマイクロ命令が実行できる.

以上のように，マクロ命令デコーダを μ-ROM に集約し，かつ高速に動作する構造はマイクロプロセッサのように回路を自在に組み合せることのできる LSI に応用することによってその真価を発揮し得る高集積論理デバイスに向いたマイクロプログラム方式である.

4. マイクロプログラム・フロー制御方式

本章では，マクロ命令解読機能内蔵形 μ-ROM において，マイクロプログラムのシーケンスがいかに成されるかを示す．マクロ命令形式は図3，μ-ROM の構造は図4のものを用い，まずマイクロ命令仕様を示した後に実際のフローを説明する.

4.1 マイクロ命令仕様

マイクロ命令形式を **図7** に示す．本節では，この中で本論文の特徴とするマイクロ・シーケンス制御に関係するフィールドについてのみ述べる.

（1） B (Branch Type) (ビット 20)

μ-ROM のアドレス源を選択するフィールドで，マクロ命令あるいはマイクロ命令のアドレス・フィールドのいずれか一方を選択する.

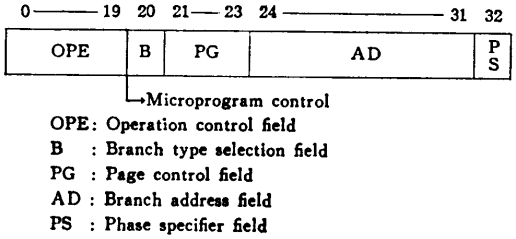


図7 マイクロ命令フォーマット
Fig. 7 Microinstruction format.

(2) PG (Page) (ビット 21～23)

μ-ROM における各種マイクロ命令群の存在するページを示すフィールドである．マイクロ命令が次のマイクロ命令へ分岐して行く際に，どのようなアドレス・デコードを行うかを指示する．図3のマクロ命令形式では，OP，R，MOD の3つの命令解読ページと1つのワーク・ページを持つ．この場合，μ-ROM は4つのページに分離すれば良い．

(3) AD (Address) (ビット 24～31)

μ-ROM のページ内アドレスを示すフィールドであり，1ページは最大256語である．

以上の情報を元に，次節にはページ制御を示す．

4.2 ページ制御

図3に示したマクロ命令を図4のマイクロプログラム制御構造及び **図 8** に示すレジスタ構成を持つデータ処理装置で実行する場合を想定して，マイクロ命令のページ制御原理とそのフローを示す．

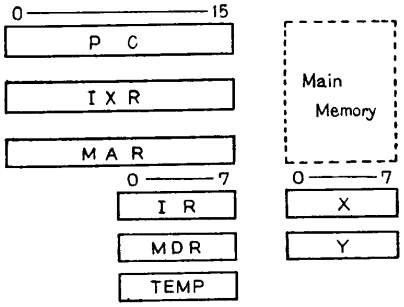


図 8 基本レジスタ構成
Fig. 8 Organization of basic registers.

(1) 命令フェッチ・サイクル

プログラム・カウンタ (PC) の内容に従い，主メモリからマクロ命令語の2語目までを読み出す．命令フェッチはすべての命令で共通であるため，命令語の内容に無関係である．したがって，次に示す2マイクロ命令は μ-ROM のワーク・ページ (ページ0) に配置される．

- MAR←PC, PC←PC+1, M(R), IR←(M), PAGE←0 (第1命令).
- MAR←PC, PC←PC+1, M(R), MDR←(m), PAGE←1 (第2命令).

第2マイクロ命令でページを1に換えるのは，次のオペランド・フェッチ・サイクルに移るためである．

(2) オペランド・フェッチ・サイクル

図3に示したごとく，第1語目の命令中の MOD フィールドでは8種類のオペランド・フェッチ形式が指定され，その実行は μ-ROM のページ1に格納されたマイクロ命令によって成される．ページ1のアドレス・デコーダはビット5～7の3ビットだけをデコードする (図5参照)．間接アドレッシングの場合でのオペランド・フェッチ処理を次に示す．

- MAR←MDR, MDR←(M), PAGE←0 (間接アドレス).
- MAR←MDR, MDR←(M), PAGE←2 (オペランド).

第1マイクロ命令のページ0指定は，次のマイクロ命令が AD フィールドによって決まるためである．また，第2マイクロ命令でページ2を指定するのは，

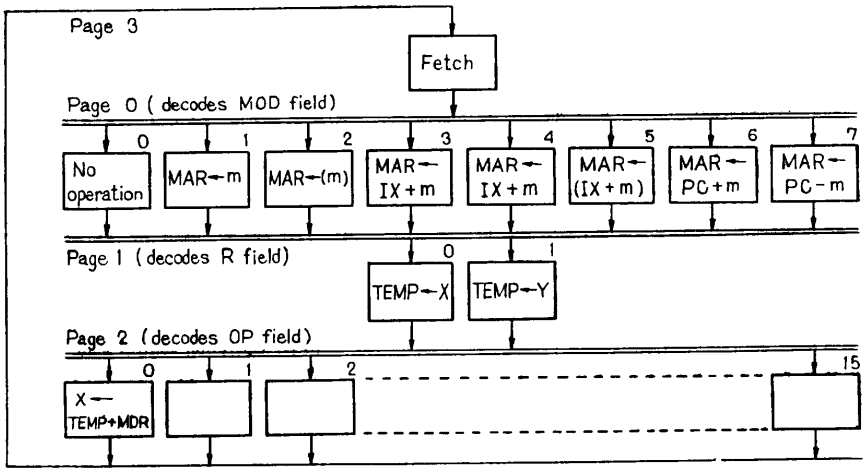


図 9 マイクロ・シーケンスの状態遷移
Fig. 9 State transition of micro sequence.

次のレジスタ読み出しサイクルに移るためである．

（3） レジスタ読み出しサイクル

第1語目の命令中の R フィールドでは，X，Y いずれかのレジスタを指定する．したがって，μ-ROM のページ2ではビット4の1ビットだけをデコードする．そのマイクロ命令は次のとおりであり，次に続く OP フィールドのデコードのためにページ3を指定する．

- TEMP←X, PAGE←3 （レジスタ X の読み出し）．

（4） 命令実行サイクル

レジスタの読み出しが終るとオペランドはすべてそろう．最後に，OP フィールドで指定する演算を行い，マクロ命令の処理は完了する．以後，再び命令フェッチ・サイクル（ページ0）へ分岐し，マクロ命令処理が繰り返される．加算の場合のマイクロ命令を示す．

- X←TEMP+MDR, PAGE←0 （加算実行）．

以上のマイクロ動作を含むマイクロ・シーケンスのフローを **図 9** に示す．図3のマクロ命令形式は理想的にフィールドが区分されているため，命令デコードが非常に簡潔であった．しかし，現実の命令形式には複雑な構成のものも少なくない．この場合，ページをさらに細分化し命令語の部分的デコードを増やすことによって対応し得る．

5. マイクロ・サイクル位相比較方式

5.1 異常検出

マイクロプログラム制御回路はマイクロコンピュータの動作を規定する中枢である．したがって，この部分の動作が正常か否かを常に監視しておくことがハードウェアの信頼性を保証するために基本的な策であろう．その意味でマイクロ命令パターンのパリティ・チェックは1つの効果的な手段である．ただ，この場合に検出され得るのは1つ1つの独立したマイクロ命令パターンの不合理性にとどまる．すなわち，μ-ROM パターンのビット方向の誤りを探すものであり，マイクロ命令間のつながり（マイクロ・シーケンス）には無関係である．そこで，本論文ではマイクロ・シーケンスの合理性チェックとしてマイクロ・サイクルの位相比較によりマイクロ命令のフロー方向に関してエラー検出を行う方式を提案する．

図 10 はマイクロ・サイクル位相比較回路の一例を示したものである．本回路の動作原理を以下に示す．

クロック発生器 (CPG) はマイクロ命令の実行に用いるクロック ϕ を2分の1に分周する回路である．これによって得られるクロック E をマイクロ・サイクルでの位相比較に用いる基準クロックとする．したがって，マイクロ命令は E あるいは $\hat{E}$ のいずれかの位相で実行されることになる．図7に示したように，マイクロ命令は1ビットの PS (Phase Specifier) フィールドを持ち，その内容は次に示す関係で規定される．

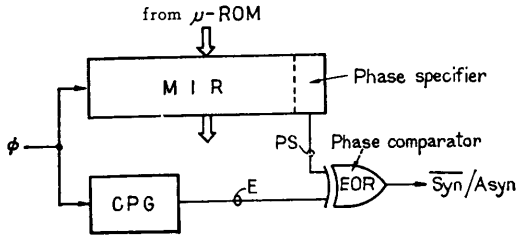


図 10 マイクロサイクル位相比較回路の構成

Fig. 10 Configuration of microcycle phase comparator.

PS=1：E 位相で実行可能なマイクロ命令．

PS=0：$\hat{E}$ 位相で実行可能なマイクロ命令．

1つのマクロ命令処理を達成する一連のマイクロ命令群のすべてのマイクロ命令中の PS フィールドに，実行順に1（E 位相）と0（$\hat{E}$ 位相）を交互に割り付けていく．この際，クロック E との同期関係を保つように配慮しなければならない．すなわち，マイクロ命令の分岐時には，分岐先マイクロ命令の実行可能な位相に合せるため同期用のダミー・マイクロ命令を挿入する必要がある場合も生じる．

μ-ROM から順次読み出されるマイクロ命令はマイクロ命令レジスタ (MIR) にクロック ϕ でセットされる．したがって，MIR の PS フィールドの出力は，正常であればクロックと全く同一のクロックを発生することになる．図10に示すように，2つのクロック E 及び PS を単なる Exclusive OR によって比較すれば，マイクロプログラム制御シーケンスの異常が検出できる．ハードウェア上のインターミテントなエラーによってマイクロ・シーケンスが乱れた場合（暴走），クロック E と PS との同期関係が失われ，逆位相の形で検出される．**図 11** に正常な場合とアドレス・デコードの際のエラー等で発生した異常状態の場合の位相関係を示した．

5.2 異常処理

前節に述べた異常検出によって得られる信号 ($\overline{\text{Syn}}$/Asyn) はマクロ命令処理を中断するために，図4に示

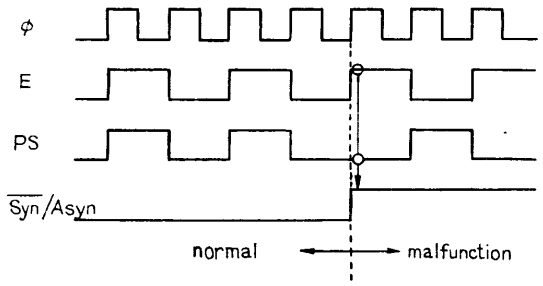


図 11 位相比較回路のタイム・チャート
Fig. 11 Time chart of the phase comparator.

した ROM アドレス・レジスタ (RAR) をクリアする．これによって，マイクロプログラムは強制的に μ-ROM の特定領域へ分岐し，ここでリスタート処理あるいは割込み処理といった異常処理を行うようにする．

以上述べたマイクロ・サイクル位相比較方式は，パリティ・チェックと同様に異常検出率 100% とはなり得ないが，特にハードウェアの暴走を防止する 1 つのチェック・ポイントの役目を果す．

6. 特殊マイクロプログラム制御

第 3，4 章では提案するマイクロプログラム制御方式の基本原理を述べた．本章では，その応用として異なる命令セットの共存法及びその効果について示す．

6.1 異なる命令セットの共存法

一般にマイクロコンピュータはデータ処理を主とする標準的な命令セットを持つものが多い．ところが，その応用を考えると通信，端末，シーケンス制御，直接ディジタル制御といった特殊な分野も少なくない．この場合にはビット処理や平方根・不感帯・飽和等の標準命令とは大分異なった特殊命令が必要となる．マイクロプログラム制御方式のコンピュータではマクロ命令の「空きコード」を特殊命令に当て，ファームウェア技術によってこれに対処していた．ところが，命令コードの空きが少ない場合には，標準命令の一部を削除し，新しい命令を加えることも多い．また，特殊命令の理想的な形式が標準命令のそれと大きく異なる場合，ハードウェアの制約から結局は標準に近い形式にせざるを得ない．そのため効率の良い命令セットが得られないという問題も生じる．そこで，全く形式の異なる複数の命令セットを 1 つのマイクロコンピュータに共存できれば，以上のような問題点は解決する．以下，本論文の提案するマイクロプログラム制御構造によって実現する手段を示す．

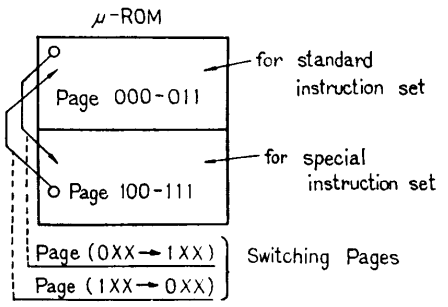


図 12 命令セットの切り換え
Fig. 12 Switch between two instruction sets.

(1) ページの区別

第 4 章で示した標準命令セットのページは，ワーク・ページが 0，命令解読ページが 1～3 の 4 種であった．特殊命令セットでは残りのページ 4～7 を配分して構成する．

(2) ページの切り換え

標準命令セットと特殊命令セットとの間の切り換えは，それぞれの命令セットに CALL 命令を備え，これによって動的にページを移す．**図 12** に示したように，標準命令セットから特殊命令セットへ制御を渡す場合には，PG＝4～7 のいずれか 1 つを含むマイクロ命令を標準命令の 1 つで実行する．反対の場合には，PG＝0～3 のいずれか 1 つを含むマイクロ命令を特殊命令の 1 つで実行すればよい．

(3) 特殊命令のデコードおよびページ制御

標準命令についてはすでに述べたので，ここでは特殊命令に関して示す．特殊命令の形式は **図 13** に示したものを例として考える．本形式は図 3 に示した標準命令形式とフィールド名 (機能) 及び区分が大分異なっている．そのため，μ-ROM の命令解読ページにおけるアドレス・デコーダのパターンは **図 14** に示すようなものとなる．すなわち，この命令セットでは 2 つの OP 1，OP 2，R の 3 フィールドに分割されているため，次の順序でデコードされ，実行されて行く．

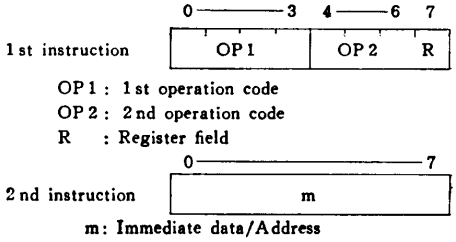


図 13 特殊命令フォーマットの一例
Fig. 13 Example of special instruction format.

2 1 0	7 6 5 4 3 2 1 0	MICRO INSTRUCTION
1 0 0	X X X X X X X 0	TEMP<— X
1 0 0	X X X X X X X 1	TEMP<— Y
1 0 1	0 0 0 0 X X X X	NO OPERATION
1 0 1	0 0 0 1 X X X X	TEMP<—TEMP+MDR
1 0 1	0 0 1 0 X X X X	TEMP<—TEMP-MDR
1 0 1	0 0 1 1 X X X X	TEMP<—TEMP*MDR
1 0 1	0 1 0 0 X X X X	TEMP<—TEMP**MDR
1 1 0	X X X X 0 0 0 X	NO OPERATION
1 1 0	X X X X 0 0 1 X	MAR <— PC,PC+1
1 1 0	X X X X 0 1 0 X	MAR <— IXR,IXR+1

Page　OP1　OP2　R　X: Don't care

図 14 特殊命令のデコードパターン

Fig. 14 Decoded pattern of special instructions.

まず，R フィールド（ビット 7）により演算レジスタを選択する．命令実行時には，最初に OP 1 フィールドの内容で分岐し，マイクロ動作の後に OP 2 フィールドの内容で再び分岐して別のマイクロ動作を行う．

以上のように，μ-ROM 上でページを明確に区分するとともに，デコード方法を命令形式に合せて変えることにより，全く異なった命令形式を持つ複数の命令セットの制御手順を 1 つの μ-ROM に共存させることが可能である．

6.2 効　果

本論文で提案するマイクロプログラム方式は，高速性，柔軟性，信頼性を高めることを意図するものである．これらの効果をまとめる以下のようになる．

（1） 高 速 性

新しいマイクロプログラム方式は，マクロ命令デコーダを μ-ROM と一体化したため，マクロ命令を読み出してから対応するマイクロ命令を読み出すまで 1 マイクロ・サイクルで実現できる．同期メモリ・インタフェースの場合，従来方式の 1/2 となり，特に処理の単純なマイクロ・サイクル数の少ない命令で効果が大きい．

（2） 柔 軟 性

新方式と従来方式の柔軟性を定量的に評価することは非常に難しい．その理由として，従来方式ではマクロ命令の各フィールドの長さ及びその配置によって命令デコーダ (ID) の内容が大きく異なるからである．従来方式で新方式と同一の柔軟性を持てる構成を得ようとすれば，マクロ命令が図 3 に示したように 3 フィールドに分割されているという条件を付けた場合，3 つの 8 入力 8 出力デコーダを持たなければならないことを意味する．これはそれぞれのデコーダでマクロ命令のどの部分（フィールド）をデコードするかが不定だからである．したがって，従来方式で完全な柔軟性を持つためには相当大きなデコーダを必要とするし，レイアウト設計も複雑になる．

これに対し，新方式ではマクロ命令の任意フィールドの切り出しを μ-ROM のアドレス・デコーダ上で行い，同時にマイクロ命令の読み出しも行うため，命令形式によって変化するのは μ-ROM でのマイクロ命令語数だけである．さらに，μ-ROM の内容の変更だけであらゆる命令形式に対応でき，レイアウト設計はきわめて簡単なものとなる．

（3） 信 頼 性

ハードウェアの中枢であるマイクロプログラム部におけるマイクロ・シーケンスの合理性チェックにより，プロセッサの暴走を検出することで信頼性を高める．本論文の例では，マイクロ・サイクルの 1/2 周期の基準クロックの位相で比較している．したがって，マイクロ動作中に何らかの誤りが生じた場合，次のマイクロ命令が現在のものと同相で動作するものを読み出す確率は 50% であり，異相で動作するが間違ったマイクロ命令を読み出す確率は残りの 50% である．前者の誤りは検出され，後者は検出されない．パリティ・エラー検出では発見されない誤りを検出するものである．

7. む す び

高集積マイクロコンピュータに適する新しいマイクロプログラム制御方式について報告した．マイクロプログラム方式は，制御論理をマイクロコード (ROM) 化しているため，整然としたハードウェア構成が得られるとともに論理変更に強いものである．反面，従来のマイクロプログラム方式はマクロ命令からマイクロ命令起動までの応答が遅いという欠点を持っている．また，論理の柔軟性の点をみても，任意の命令セットを解読することは命令デコーダが複雑になり，その面積が増加するため十分とはいえない．

本論文では，高速性と柔軟性を同時に満たすマクロ命令解読機能内蔵形のマイクロプログラム制御構造を提案した．高速性に関しては，マクロ命令をマイクロプログラム・メモリ (μ-ROM) のアドレス・デコーダ

で直接解読することによって実現した．すなわち，命令デコーダと同等の役割を持つ PLA をデコード領域として μ-ROM に一体化した．柔軟性に関しては，マイクロ命令にページ情報を持たせることで μ-ROM を論理的に分割し，ページ間での独立性を得る方法で実現した．すなわち，μ-ROM において，マクロ命令の任意フィールドを切り出してデコードする領域とマイクロ命令のアドレス・フィールドをデコードする領域との間のアドレス・パターンに重複を許すようにした．

この結果，任意のマクロ命令形式の命令セットに対し，すべてμ-ROM上で解読が可能となり，複数の命令セットを同一の μ-ROM に収納できることが分かった．標準の命令セットだけでなく，問題向きの特殊命令セットをも合せ持つことができる柔軟性の高いマイクロプログラム制御機構を提供し得るものである．

また，マイクロプログラム制御機構の信頼性向上の一方法として，マイクロ・シーケンスの位相比較による合理性チェック方式を示した．これによって，システムの中枢となるハードウェアの暴走防止を一歩進めることができた．

なお，本方式の基本思想に立った汎用マイクロプロセッサは現在試作中であり，製品化を予定している．

最後に，本稿の研究に当たり，有益なご助言をいただいた日立製作所・武蔵工場中野浩行氏，同日立研究所増田郁朗氏，浜田亘曼氏に謝意を表します．

参考文献

1) Bursky, D.: MOS/digital/analog mix=low-cost, high-performance Single-Chips μCs: Electronic Design, Vol. 27, No. 13, pp. 43-48 (1979).
2) Morse, S. P. et al.: The Intel 8086 Micro processor: A 16-bit Evolution of the 8080; Computer, Vol. 11, No. 6, pp. 18-27 (1978).
3) Shima, M.: Two Versions of 16-bit chip span microprocessor, minicomputer needs; Electronics, Vol. 51, No. 26 (1978).
4) Stritter, E. et al.: A Microprocessor Architecture for a Changing World: The Motorola 68000; Computer, Vol. 12, No. 2 (1979).
5) 前島他：制御用 16 ビットマイクロコンピュータのアーキテクチャ，情報処理学会論文誌，Vol. 21, No. 3, pp. 208-215 (1980).
6) Fiala, E. R.: The Maxc System; Computer, Vol. 11, No. 5 (1978).

(昭和 55 年 10 月 6 日受付)
(昭和 56 年 7 月 13 日採録)