Vol. 5 No. 1 '64 Jan. 1964

マイクロプログラミングと固定記憶装置*

萩原 宏**

1. まえがき

マイクロプログラム制御の計算機は，control memory に記憶されている制御論理にしたがって計算機の内部の制御が行なわれるものであり，計算機の構成が簡単になり，制御論理の変更，修正が容易になる特長がある．control memory としては読み出し時間が短いこと，経済的なことなどのために固定記憶装置が使われることが多い．

欧洲においては EDSAC-II 以来この方式の計算機がかなり製作されているようであるが，わが国においては，この方式の計算機が少ないため，なじみもうすいと思われるので，まず，マイクロプログラミングの概念，その制御装置，control memory などについて簡単に述べ，筆者らが試作したマイクロプログラム制御の計算機 KT パイロットで使用しているマイクロプログラムの例をあげて，ご参考に供したい．

2. マイクロプラグラミングとは

計算機のプログラムは問題を処理しようとする過程を加算，乗算，飛越しなどの計算機の命令の組み合わせによって構成されたものであり，計算機はこの命令の系列にしたがって，プログラマーの意図した動作を行なうのである．計算機の内部では，これらの命令は，桁移動，カウント，語の転送，その他の，さらに基本的な計算機内部の操作，すなわち micro-operation に分解される．そうして，計算機内部の制御装置で発生される制御信号によってゲートの開閉その他の操作を行なって，上記の micro-operation を実行し，その組み合わせによって計算機の命令を遂行している．

実際の計算機では同時に二つ以上の micro-operation を行なうことも多いので，同時に発生される制御信号によって行なわれるいくつかの micro-operation の組に対して，micro-order というものを考えることにする．計算機のプログラムが命令の組み合わせで構成されているのと対照して，計算機の命令（micro-order に対して macro-order と呼ぶことがある）は micro-order の組み合わせで構成されると考えることができ，これを micro-program と呼んでいる．すなわち，計算機の各 macro-order はそれぞれいくつかの micro-order の組み合わせである micro-program として表現できる．

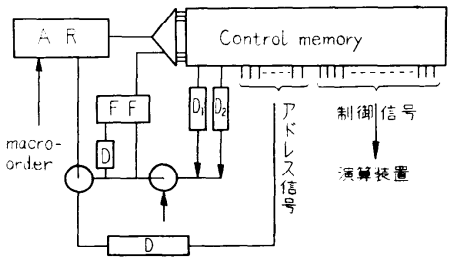
そこで，この micro-program を適当な記憶装置（これを control memory と呼ぶ）に記憶させておき，その micro-order を逐次読み出して来て，順次これを実行させることによって，macro-order を遂行することができる．こういう計算機の制御方式をマイクロプログラム制御と呼んでいる．

マイクロプログラム制御方式は，このように計算機内部の制御論理を control memory に集中化しているために，制御装置は簡単になり，制御論理の変更や修正を容易に行なうことができ，計算機の設計，製作，調整，保守などが非常に容易になる．また，control memory としては一般記憶装置を使うことも可能であり，こうすることにより，プログラムによって計算機の制御論理を変更することもできるようになり，融通性の大きい計算機とすることもできる．

3. マイクロプログラム制御装置

マイクロプログラム制御方式の計算機の代表的なものは EDSAC-II であるが[1]，ここでは主として筆者らが試作した計算機 KT パイロット[2),3)]をもとにして，その制御装置の動作について説明しよう．

第1図にその制御装置のブロック図を示す．まず，命令読出しの micro-program により一般記憶装置か



第1図 マイクロプログラム制御装置

* Microprogramming and Fixed Memory, by Hiroshi Hagiwara
(Faculty of Engineering, Kyoto University)

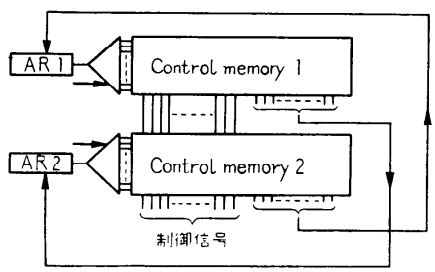
** 京都大学工学部

ら macro-order が取り出され，アドレスの修飾が行なわれたあと，macro-order の操作部は control memory address register (AR) にセットされる．この内容，すなわち macro-order の操作部によって control memory の一語が選択され，図の FF 出力が出ると control memory の中の micro-order にしたがって各種の信号が出力に取り出される．この出力の一部（制御信号）は演算装置に送られ，そのゲートの開閉その他の micro-operation を行ない，一部（アドレス信号）は control memory address 変更信号として，control memory address register の内容を変更する準備をする．

また，control memory の出力の他の一部は，この micro-order を実行するに要する時間に相当する遅延を与える遅延回路（D_1 あるいは D_2）を通って FF をリセットし，同時に control memory address 変更信号を control memory address register に加えてその内容を次の micro-order のアドレスにセットする．AR が新しい状態になると同時に FF はセットされ，AR の新しい内容によって control memory の一語が選択され，その micro-order にしたがって次の制御信号が取り出される．

演算装置の内部条件（たとえば，ある bit が "1" であるか "0" であるか）によって micro-program の sequence を変える必要が生ずるが，このために AR の 1 bit をこの条件によってセットあるいはリセットできるようにしてあり，これによって micro-program の分岐を作ることができる．

また，入出力装置あるいは一般記憶装置のように，動作速度のおそいものの制御のためには，その動作開始信号を送ってあとは独立に動作させて，次の micro-order に移るが，同じ装置を再度使用する際にもし前の動作が終っていなければ，その終了を待たなければならないので，遅延回路（D_1 あるいは D_2）のあとにゲートを設けて，ここで micro-program の進行を止めるようにしている．



第 2 図 control memory を二つ使う方式

上記のような方法をとれば，1 micro-order ごとに control memory の読み出し時間が加わるため動作が遅くなるので，高速に動作を行なわせるためには，**第 2 図**に示すように control memory を二つ用意し，交互に読み出す方式も考えられている[1]．

4. micro-order の情報

前節に述べたことから明かなように，micro-order は次の 3 種の情報を持っていなければならない．

（1） 制御情報 micro-order が制御しようとする演算装置その他のゲート等を指定する情報であって，制御されるべき点に対して 1 bit ずつ割当てる．しかし，こうすると冗長度が大きいので，同時に行ない得る micro-operation の種類を限定して，適当に coding して記憶させておき，読み出したのちこれを decode して各点に対する信号に変換する方法が考えられる．この方法をとれば，一語の bit 数を減少させることはできるが，decoder が必要になり，また decoder 中での時間遅れが加わり，micro-order の種類が限定されるなどの欠点がある．

（2） アドレス情報 次の micro-order の control memory 内のアドレスを与える情報であって，一般に，次の micro-order のアドレスは，現在行なわれている micro-order と演算装置内の条件などによる分岐のための情報によって定まるので，これらを AR にセットするのに必要な信号を発生させなければならない．分岐のための情報を AR の特定の 1 bit（たとえば，最下位の bit）に入れることにすれば，所要の内部条件によってこれを "0" あるいは "1" にするための信号，および AR の他の bit をセットあるいはリセットする信号を送り出せばよい．

（3） 時間情報 micro-order の実行時間は行なわれる micro-operation の種類によって変化するので，その micro-order を実行するのに必要な時間を予め調べておき，これに応じて適当な時間だけ遅れた信号を作り出す必要がある．このため micro-operation の種類に応じて数種の遅延回路の中からその一つを選ぶための情報を与えなければならない．もし同期式の計算機であって，すべての micro-operation が 1 clock time で終了することがわかっていれば，clock pulse を用いればよいので，特に時間情報を control memory の中に入れておく必要はない．

筆者らの試作した計算機 KT パイロットにおいては，構造が簡単であるため，これらの情報は 80 bit であるが，普通の1アドレスの計算機では 150～200 bit であるといわれている．

5. Control Memory

マイクロプログラム方式の計算機の control memory としては，上述の情報を記憶しておき，必要に応じて読み出されるものであれば，いかなる型の記憶装置であっても差支えない．したがって一般記憶装置と同様のものを用いることももちろん可能である．しかし，計算機の制御論理を記憶しており，1 step ごとにその内容をとり出してくるので，その内容の読み出しに時間がかかれば，それだけ計算機の動作がおそくなるので，なるべく高速で読み出し得るものが望ましい．

また，micro-program を計算機のプログラムで変更することを考えなければ，書き込みのできない読み出しのみのいわゆる固定記憶装置で差支えない．そのために現在まで発表されているマイクロプログラム制御の計算機では，多くが control memory として固定記憶装置を使用している．

control memory として用いられる固定記憶装置としては，ダイオードマトリックスあるいはフェライトマトリクスが広く用いられているようである．これらは micro-program がきまれば，それに対応してダイオードを固着し，あるいはコアを貫通して巻線を施して固定される．したがって，これらは固定記憶装置というよりもむしろ制御論理を集中的に配線したものといった方がよいかもしれない．これらは micro-program の変更や修正は容易ではないが，極めて高速にその内容を取り出すことができ，また経済的でもあるので広く使われている．

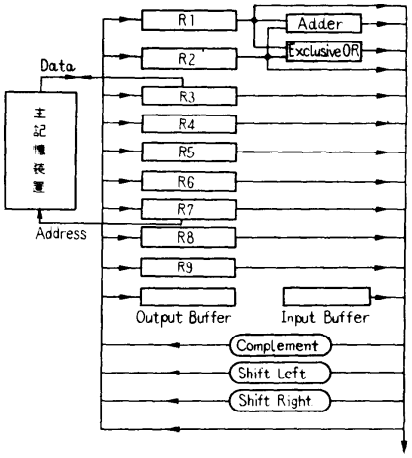
筆者らは KT パイロットにおいて，一部はダイオードマトリクスを用いたが，一部は micro-program を変更できるようにするために，フォトトランジスタを用いた固定記憶装置[4] とパッチボードを採用した．これらはいずれもダイオードマトリクスと同程度の高速動作を行なうことができ，また極めて容易にその内容を変更できるので，便利に使うことができる．

6. マイクロプログラムの例

筆者らが KT パイロットで使用しているマイクロプログラムの例について説明しよう．

6.1 KT パイロットの概要

マイクロプログラムの例をあげて説明するのに必要な KT パイロットの構成の概要について述べる．



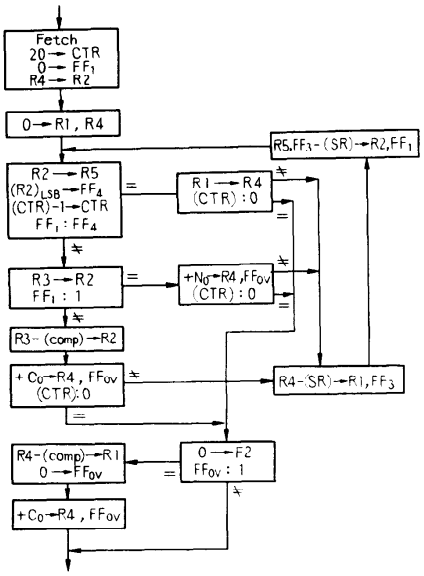
第3図　KT-パイロットの演算装置

KT パイロットは非同期，並列形の計算機で，一語 20 bit（符号を含む）で，その演算装置の主要部は**第3図**に示すような構成である．ここで R1～R6 のレジスタはそれぞれ 20 bit で，R1，R2 は adder および Exclusive OR 回路に直結されている．R3 は主記憶装置との情報の授受を行ない，R7 は主記憶装置に対するアドレスを保持する．R8, R9 は macro-order の処理のためのものである．この他にマイクロプログラムのループの計数などのためのカウンタ（CTR），CTR の零検出器，シフトの際の溢れ bit の保持その他の目的の補助フリップフロップ（FF_1, FF_2, etc）などが用意されている．また，負の数は2の補数表示を採用している．

第1図に示した制御装置の control memory よりの制御信号は各レジスタの入出力ゲート，加算器，補数回路，シフト回路の出力ゲート，その他に送られて，情報の転送その他を行なうようになっている．

6.2 マイクロプログラムの流れ図

第4図に Booth の方法を用いた乗算のマイクロプログラムを示す．ここで，macro-order のアドレスは修飾されたのち，レジスタ R7 に入れられているものとし，レジスタ R4 の内容と R7 の内容で指定される主記憶装置の内容との積をレジスタ R4 と R5 に入れるものとする．なお一つの box に書かれた



第4図　乗算のマイクロプログラム

micro-operation の組が一つの micro-order になっている．

使用した記号の意味は次のとおりである．

Fetch：主記憶装置からの読み出し．

Store：主記憶装置への書き込み．

R4→R2：レジスタ R4 の内容を R2 に転送する．

R3-(comp)→R2：R3 の出力を補数回路を通して R2 に転送する．

R4-(SR)→R1, FF_3：R4 の出力を**第3図**の Shift Right の回路を通して右へ1 bit シフトし R1 へ転送すると同時に，最下位から溢れたビットを FF_3 に入れる．

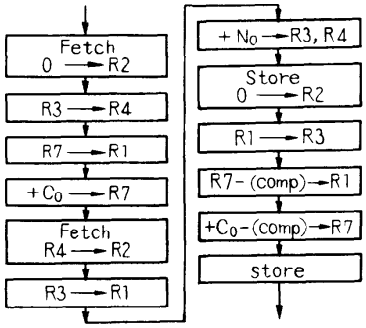
FF_1：FF_4：補助フリップフロップ FF_1 と FF_4 の内容を比較する．

(CTR)：0：カウンタ（CTR）の内容の零検出

$+C_0$：加算器に initial carry を入れて和を取り出すことを示す．

$+N_0$：加算器に initial carry を入れないで和を取り出すことを示す．

また，**第5図**に相加法による擬似乱数発生のマイクロプログラムを示す．これは macro-order のアドレスで指定される番地の内容を x，次の番地の内容を y とするとき，$z=x+y \pmod{2^{20}}$ を R4 におき，x を y で，y を z でおきかえる操作を行なうもので，一つ



第5図　擬似乱数発生のマイクロプログラム

の macro-order によって擬似乱数を発生さすことができる．したがって，この macro-order を利用することによって，乱数を頻繁に使用する Monte-Carlo 法による計算などを能率よく行なうことができる．

このようにして構成されたマイクロプログラムは，それぞれの micro-order に対して control memory の location を割当て，その micro-operation を行なわせるに必要な制御信号および次の micro-order のアドレス情報などを定めて，これを control memory にセットしてやればよい．ただし，macro-order に対応する micro-program では，その第1ステップの location は macro-order の code にしたがって定まることに注意しなければならない．

7. むすび

マイクロプログラム制御方式の計算機は，計算機の構成が簡単になるための利点の他に，マイクロプログラムを容易に変更できるような control memory を用いれば，**6節**で述べたように，問題に適した macro-order を設定して処理することができるので，能率のよい動作をさせることができると考えられる．このために，control memory として高速読み出しができ，しかも容易に内容を変更できる固定記憶装置が開発されることが望まれる．

参考文献

1) M.V. Wilkes, W. Renwick, D.J. Wheeler: The Design of the control unit of an Electronic Digital Computer, Proc. IEE. Vol. 15, Part B, No. 20 (1958-3) pp. 121-128

2) 萩原，天羽，松下，山内：KT パイロットモデル計算機，昭和36年度情報処理学会全国大会

3) H. Hagiwara, K. Amo, S. Matsushita and H. Yamauchi: The KT-Pilot Computer, Proc. of IFIP Congress 62

4) 萩原，天羽他：フォトトランジスタを用いた超高速固定記憶装置，昭和37年度電気通信学会全国大会．

（昭和38年10月17日受付）