解　説

マイクロコンピュータ応用の基本的な考え方*

石　田　晴　久**

1. はじめに

マイクロコンピュータ関係では，1個ないし数個のLSIチップからなる中央処理装置(CPU)をマイクロプロセッサと呼び，それに各種メモリ，入出力インターフェース回路，各種周辺機器などをつけてコンピュータの形に構成したものをマイクロコンピュータと呼ぶことが多い．もっとも単にマイクロプロセッサというときは，一般のマイクロプログラム方式の処理装置をさすこともある．さらに前述の微小CPUの意味のマイクロプロセッサにもマイクロプログラム方式のものがあって，ことばは一層まぎらわしくなる．しかしここであえて，マイクロコンピュータと微小CPUという意味でのマイクロプロセッサ（以下ではこの意味に限定する）とを区別したいのは，マイクロプロセッサがいわゆるコンピュータの形で使われるものよりも，いろいろな機器に組み込まれて一見コンピュータとは見えないような形で使われることが多いからである．したがって本論のタイトルも，マイクロコンピュータ応用というよりは，マイクロプロセッサ応用という方が適当かもしれない．

さてマイクロプロセッサを応用するにあたってまず重要なことは，ミニコンなど他の上級機種と比較したときのマイクロコンピュータないしはマイクロプロセッサの特質を考えることである．ここではこれを次のようにとらえることにする．

（1）独立のコンピュータとしてよりも，従来の布線論理回路の代りにシステム・コンポーネントとして組み込む形で使うのに向いている．

（2）完成したプログラムはPROM (Programmable Read Only Memory) に入れて固定化するのが普通である．これによりPROM中のプログラムやデータを少し変えて新たにPROMを用意することにより，多品種少量生産向きの機器が作れる．しかしPROMの予期せぬ変更は一般に困難だから，プログラムの正当性は十分に保証する必要があり，後での仕様変更の必要もないようにしなければならない．

（3）回路の中ではCPU部分は微小であり，コスト上の比重も低い．したがって複雑な割込み処理や多重プログラミングを試みるよりは，プロセス（タスク）ごとに1台というように多数のCPUを使うことを考える方が有利である．CPU（とくに4ビット型）はむしろタダだと考えてよい場合がある．

（4）ハードウェアとしては，メモリ，インターフェース回路，ソケット，プリント板，匡体，電源などCPU以外の部分がコスト的にもサイズ的にも主要部となる．とくにメモリのコストは全体の中で高くなりがちだから，メモリの少くてすむ応用に向く．

（5）プログラムの開発およびテストには，極端に簡単なものを除いては，一般に何らかのコンピュータが必要である．この目的には当のマイクロコンピュータの他，他の上位機種の上で走るクロスソフトウェアを使うこともできる．またPROMへのプログラムの書込みには，PROM書込み器が心要である．

（6）回路工作の道具があれば，素人によるコンピュータの自作も可能である．最近はアマチュア向けのキットも発売されている．

2. マイクロプロセッサの選択

マイクロプロセッサの応用における最も重要な選択は，どの機種を採用するかであろう．この選択はマイクロプロセッサの機種が増えるにつれ[1]，最近はだんだん困難になってきた．またとくにこうしたLSI製品では価格は数量やメーカとの力関係に大幅に左右されるため，単純にコスト・パフォーマンス比を論ずる

* Basic considerations in the application of microcomputers by Haruhisa ISHIDA (Comuter Centre, University of Tokyo)
** 東京大学大型計算機センター

こともできない．次にあげるのはCPUチップの選択にあたって問題になる事項の例である．

（1） 語長——4ビットか，8ビットか，12ビットか，16ビットか，それ以外か

これはもちろん応用の目的，とくに要求される処理のスピードに左右される．価格は語長の短いものほど安いが，同じ処理を行うのにプログラム・ステップが長くなりメモリ所要量が増える可能性がある．10進数を1桁ずつ処理するには4ビットCPU，文字など8ビット（1バイト）ずつの処理に8ビットCPUがよいことは自明である．高速が必要な場合，あるいは12ビットや16ビットでも足りないときは，ユーザ・マイクロプログラム方式（インテル3000シリーズなど）のバイポーラ複数チップCPUも候補にあがるが，これにはマイクロプログラムを書く能力とそれをテストする機器が要求される．

（2） 内部メモリ容量

普通の応用では，プログラムはPROMに内蔵させるとしても，可変なデータやアドレスを格納するにはそれだけのRAM (Random Access Memory——読み書き可能メモリ）が必要である．ところが格納すべき可変データの非常に少い応用では，外部にわざわざRAMをつけることをせずに，CPUの内部メモリ，すなわちレジスタや内部スタック（アドレスやデータを棚積みする場所）だけですましたくなる．こうした応用にはレジスタや内部スタック容量の多いCPUチップがよい．将来のCPUチップには内部メモリ容量がさらに増えることを期待したい．

（3） セカンド・ソース

半導体メーカ関係では，セカンド・ソースということばは，あるメーカが最初に発売したチップと同じ型のチップを他のメーカが正式契約を結んだ上で，あるいは結ばぬまま発売したときに，後者をさしていうことばである．このセカンド・ソースの有無がよく問題とされ，一般にセカンド・ソースはある方がよいとされるのは，半導体製品市場は競争の激烈な世界であり，トップ・メーカといえども倒産の危険は常にあり，ユーザ側からみれば同じ製品を買うのに複数個の相手があった方がよく，また海外メーカの場合は国交断絶なども心配されるからである．参考までに**表-1**にセカンド・ソースの例をあげておく．

もちろんセカンド・ソースが実際に問題になるのは大量の応用の場合であるが，その場合には，たとえ外部的な仕様は全く同一でも，本当にチップが相互に差し換えられるほど電気的特性が一致しているかどうかは十分検討してみる必要がある．

表-1 マイクロプロセッサのセカンド・ソースの例

インテル 8080	日本電気，三菱電機 テキサス・インストルメンツ AMD（アメリカン・マイクロデバイス）
〔他にソフトウェア・コンパティブルなものとして，コムスター社の M-8（マイクロプログラム方式）がある．〕	
インテル 4004	ナショナル・セミコン 日立製作所
フェアチャイルド F-8	モステク
モトローラ 6800	AMI（アメリカン・マイクロシステムズ） 日立製作所，富士通

（4） マイクロコンピュータのキット

実際に回路を組む上での選択としては，LSIチップのレベルから始めて自分（自社）で配線するか，あるいは，CPUボード，メモリ・ボード，インターフェース・ボードのようにすでに（プリント）配線ずみの回路ボードのレベルから始めるか，の決定がある．後者の場合は，特定の機種についてそうしたものが入手しやすいかどうかが重要である．

今日では回路ボードは内外の大小多数の各種メーカから発売されているが，中でも面白いのはアメリカのMITS社より発売されている素人向けキットALTAIR 8800（インテル8080使用）である．カリフォルニアなどではこれのアマチュア・クラブが結成され，ときどき研究発表大会が開かれている．これはコンピュータ技術者人口を拡げる上では無視できない傾向である．日本でもマイクロコンピュータの日曜工作センターの開設が提案されているが，なかなか実現に至らない．

（5） ソフトウェア開発のサポート体制

マイクロプロセッサでもコンピュータの常として，プログラムの開発がコスト的にも応用上の中心事となるから，CPUチップの選択にあたってはソフトウェア開発のサポート体制が整っているかどうかは考慮に入れる必要がある．これは必ずしも当の半導体メーカが提供するものとは限らないが，各種マニュアル，ユーザへの教育，ソフトウェア開発の道具，プログラム・ライブラリなど考えるべきことは多い．

ハードウェアのテスト機器もあることが望ましい．PRO-LOG社のシステム・アナライザM-821（インテル8008用）はその1例である．これはマイクロコンピュータのポータブル・コンソールに相当するもので，多数の表示ランプやスイッチがある他に，回路ボード上で8008型CPUチップを両側からはさめるよ

うになった 24 ピンのディップ・クリップを通して，マイクロプロセッサ本体と接続できるようになっている．

3. メモリ・チップの選択

マイクロプロセッサ用の主メモリとしては，コア・メモリも使用できるが，普通は LSI の RAM, PROM それに ROM が使われる．このうち ROM は記憶内容（プログラムやデータ）を2進パターンとして紙テープなどに用意して，半導体メーカに渡しマスクをおこして製作してもらう必要があり，マスク代と日時がかかることとから大量生産向きである．また中に入れたプログラムやデータに虫がいた場合，ROM は廃棄処分にせざるをえなくなる．

これに対して PROM には，ヒューズを焼切る型のものなど，ユーザ・レベルで書込みはできるが消去は全くできない非可逆型のものと，電荷を絶縁物の中に閉じこめる形で永久記憶を行わせ，外から紫外線をかければそれが消去できる可逆型のものとがある．価格はもちろん石英ガラスの消去用窓がついた後者の方が高い．これらのいずれを使うかは結局コストと消去の必要度および消去の手間とのかね合わせであり，消去可能型の方が常によいというわけではない．

さて PROM にプログラムやデータを書込むには PROM 書込み器が必要である．PROM 書込み器には内外のメーカからいろいろな型のものが出ているが，PRO-LOG 社の 90 型を例にとると次のような機能がある．

（1） PROM の異なる型に応じて回路の1部を差しかえる．これで各種の PROM が扱えるようになる．

（2） PROM チップをソケットにセットし，16 個のキーまたはテレタイプの紙テープ・リーダによりメモリ内容を入れて，1語ずつ書込む．書込みパルスは 40V 程度いるが，メーカ仕様より 40% 程度オーバーチャージする．内部ではパルスを1個加えるごとに書いた内容の読み出しをして，読み出し電圧が規定値に達するまで繰返しパルスを自動的に加える．

（3） 16 キーでアドレスをセットし，PROM のメモリ内容を表示窓に読み出すことができる．

（4） もうひとつ別のソケットに書込みずみのマスター PROM をセットすることができ，このマスターの内容を PROM にそのまま，あるいは 16 ケ所以内の変更を加えた上で別の PROM に書込むことができる．また当の PROM とマスター PROM の内容との比較も自動的に行える．

（5） 全体で約8キロで消去用の紫外線照射器とともにスーツケースに入り，ポータブルである．

このようにアダプティブな書込み，リスト，コピー，編集，比較などの機能をもつ PROM 書込み器は PROM を利用する上の重要な道具である．ベル研究所では計算センターでのサービスとして，処理結果を PROM に書込むサービスを提供することを考えているという．

一方 PROM または ROM 内のプログラムをまず RAM に入れておいて，マイクロコンピュータ上でテストする目的でサイエンティフィック・マイクロシステムズ社では ROM シミュレータを開発している．これは 128 バイトないし 8 キロバイトの RAM を内蔵し，マイクロコンピュータ内の各種(16 ピン/24 ピン) PROM（または ROM）のソケットに差込めるピン型のプローブをもつ機器で，全体としてマイクロコンピュータ側からみると任意の PROM または ROM にみえるように構成されている．これだとメモリは RAM だからテスト中にその内容を自由に変えることができるわけである．

次に主に可変データを入れるための RAM（あるいは主メモリにしてもよいが）には，再生（リフレッシュ）の不要な低容量のスタティック RAM と高密度だがリフレッシュの必要なダイナミック RAM（いずれも普通は MOS）がある．後者はメモリ内容を保持させるために常時リフレッシュ・パルスを供給しなければならないため周辺回路が複雑になるので，ある程度容量の大きな RAM を必要とする応用にしか向かない．以上のべたメモリ・チップの種類の一覧を表-2 に示す．

表-2 の下の部分に示したように，LSI メモリ・チ

表-2　メモリチップの種類

容量（ビット）		256	1,024	2,048	4,096	8,192	16,384
スタティック RAM		○	○	—	○	—	—
ダイナミック RAM		—	○	○	○	—	○
消去型 PROM		—	○	○	○	○	—
非消去型 PROM		○	○	○	○	○	—
ROM		—	○	○	—	○	○
語構成	1ビット／語	256語	1,024語	—	4,096語	—	—
	4ビット／語	—	256語	512語	1,024語	—	4,096語
	8ビット／語	32語	—	256語	512語	1,024語	2,048語

ップでは，ビット容量は同じでも語の構成，すなわち語長(2^m)や語数(2^n)の異なるものがある．一般に容量が 2^{m+n} ビットのとき，ピン数は〔信号の時分割制御（マルチプレクシング）をしないとすれば〕$n \times 2^m$ の程度だから，語長が短い方がピンの数は少くてLSIは作りやすい．しかし最近は1語8ビットのメモリ・チップが増えている．

4. プログラマブルなインターフェース用LSIチップの利用

最近のマイクロプロセッサでは，CPUチップの他にインターフェース用のLSIチップも各半導体メーカから多数発売されるようになってきた．それらの著しい特徴はプログラマブル化の傾向，すなわちプログラムでセットできる可変パラメータを含む形で汎用化がはかられていることである．これはインテリジェンスやメモリの分散化の傾向とみることもできる．**表-3**にはプログラマブル・インターフェース・チップの例を示す．

次にプログラマブルの程度をみる目的でインテルの8255入出力インターフェース・チップを例にとり説明する．

図-1 はこの入出力インターフェースの位置を示し，**図-2** はその内部構成を示す[2]．図-2で下側の WR，RD の信号はそれぞれ書出し（CPUよりの出力），読み込み（CPUへの入力）を示し，アドレスの A0, A1 ビットは 00 のときポート A (A7～A0)，01 のときポート B (B7～B0)，10 のときポート C (C7～C0) を選択する信号である．機能がこれだけならこのチップはただの LSI であるが，これがプログラマブルであるゆえんは，内部にグループAおよびBの制御レジスタ回路があり，そのはたらきが (A0=A1=1 のとき) CPU からのデータ・バス上のコマンド（8ビット）で指定できるところにある．このコマンドでは 0, 1, 2 の3種のモードとポートCの何番目のビットをセット／リセットするかが指定できる．

表-3 プログラマブル・インターフェース・チップの例

コミュニケーション・インターフェース (8251)
インタバル・タイマ (8253)
マルチポート入出力インターフェース (8255)
4チャネル DMA コントローラ (8257)
8レベル割込みコントローラ (8259, 以上インテル)
周辺インターフェースアダプタ（モトローラ）
英数字キーボード・ディスプレイ・コントローラ
フロッピー・ディスク・コントローラ
DMA コントローラ（以上各社）

3種のモードのうち，モード0ではポートA，ポートB，ポートC上半分(CA)，ポートCの下半分(CB)の任意の組合せ（16通り）が指定できる．したがって入出力は，4ビット（2種），8ビット（3種），12ビット（4種），16ビット（3種），20ビット（2種），24ビットで可能となる．たとえば12ビット入出力は，AとCA，AとCB，BとCA，BとCBのいずれの組合せでもよい．

またモード1ではポートAまたはBをデータの入力または出力に，ポートCの半分を制御信号およびステータス情報の授受に使用するよう指定できる．さらにモード2ではポートAのみを両方向のデータ入出力に使い，ポートCの5ビットを制御信号／ステータス情報に使用することが可能となっている．

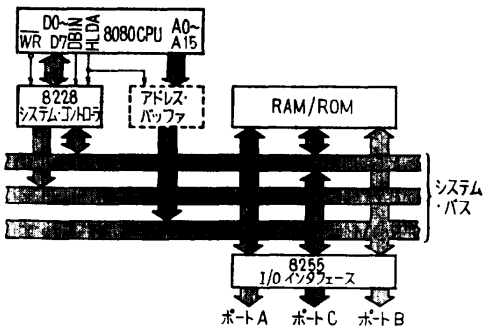

図-1 プログラマブル入出力インタフェイスの位置

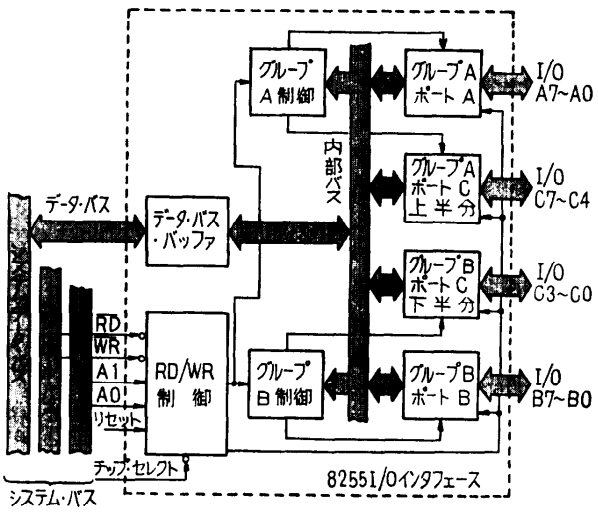

図-2 8255 入出力インタフェースチップの内部構成

これからもうかがえるように，こうしたインターフェース・チップは非常に多様な使い方ができる上，インターフェースの機能のかなりの部分がソフトウェアに移るから，ハードウェアの設計製作はかなり容易になる．ただ現在のところこうした LSI チップはまだ高価であり，インターフェースは従来の個別回路または標準の MSI で組む方が安く，従来のハードウェア設計者にはその方がやりがいがあるといわれる．しかし今後は論理回路をマイクロプロセッサに置きかえるのと同じ理由(複雑さへの対処，低コスト，高信頼性)で，インターフェースにも積極的に利用すべきだと思われる．またチップ自体のコストを下げるには大量生産が必要であるが，そのためにはチップが多品種少量生産になるのは好ましくない．この意味では少品種のインテリジェントなインターフェース・チップができれば標準化されて，多目的に大量に使われることが望ましい．しかしどのようなプログラマビリティが望ましいかをきめるにはユーザ側でもこの種のチップにもっと経験を積む必要があるかもしれない．

5. システム開発の道具立て

すでにのべたようにマイクロプロセッサを中心とするシステムは，ハードウェアとしてはだんだん作りやすくなってきた．そこでユーザ側でのシステム開発上の最大の課題は他のコンピューター般と同様にソフトウェアをどう開発するかという問題になる．この問題は本質的にはミニコン以上の上位機種のソフトウェア開発と同じであろうが，ただプログラム言語でアセンブラ語が中心となっていることや，純手続き部分(PROM 領域)と可変データ部分 (RAM 領域) を分けて書かねばならない点，また上位機種を使ってソフトウェア開発が行われうる点などやや事情が異なる面がある．

ここではマイクロコンピュータのソフトウェアを開発するのに必要な道具あるいはシステムおよび開発の方法を次の4形態に分けて考えてみたい．

PROM 書込み器による方法
基本マイクロコンピュータによる方法
拡張マイクロコンピューたによる方法
上位機種のクロスソフトウェアによる方法

(1) ハンド・アセンブルと PROM 書込み器

これは主として4ビット CPU 用に比較的単純なプログラムを開発する場合にとられる方法で，コンピュータを使用せず，先に紹介したような PROM 書込み器をもっぱら使う方法である．手順は次のようになる．

a．アセンブラ語でプログラムを書く．
b．机上デバッグを十分行う．
c．次にプログラムを手で16進数（または8進数）にアセンブルする．
d．PROM 書込み器のキーボードを使ってそれを消去可能型 PROM に書込む．
e．PROM を実際にマイクロコンピュータ回路に差込んで実地テストを行う．万一虫がいたら，PROM の内容を紫外線で消去し，上記ステップ d に戻ってやり直す．

これは恐らく設備投資がもっとも少くてすみ，いかにもマイクロコンピュータにふさわしいプログラム作成法である．しかしもちろんプログラムがある程度以上長くなるとこれでは不便になろう．これはコンピュータを使わない点が面白い．

(2) ファームウェア付基本マイクロコンピュータ

マイクロプロセッサ応用システムで必要とされるプログラムをテストするための最もオーソドックスな方法は，やはり当のマイクロプロセッサを内蔵するマイクロコンピュータを利用することである．最小限必要なシステム・プログラムは，テレタイプを制御するモニタ，テキスト・エディタ，ローダ，アセンブラ，デバッグ用プログラムなどであろう．

この場合，マイクロコンピュータとしてどの規模のものを用意するかが問題となるが，設備投資を低く抑えたい場合は，若干の RAM と入出力機器としてテレタイプのついた最小構成のマイクロコンピュータとならざるをえない．しかしこれではシステム・プログラムをテレタイプの紙テープ・リーダでいちいちロードするのに長い時間がかかってきわめて能率が悪くなる．そこで少くとも高速の紙テープ・リーダが欲しくなるが，マイクロコンピュータの場合は，PROM が利用できることを考えると，システム・プログラムをすべて PROM に入れて自分のファームウェアとして固定化するのも一方法である．ただ PROM はまだ高価だから，システム・プログラムはできるだけ単純にして短かくする必要がある．たとえばアセンブラのマクロ定義機能などは入れないようにせざるをえないであろう．

こうしたシステム・プログラムを PROM に入れておけば，それはいちいちロードしなくても電源を入れればすぐ動き，またユーザ・プログラムが仮に暴走しても PROM 内のシステム・プログラムは絶対にこわ

れない，すなわちメモリ保護が自然にかかるといった利点がある．この目的に必要な PROM の個数は，チップ当り 2,048 ビット(256 バイト)の PROM の場合，ある例では次のように合計 20 個になっている[3]．

モニタとデバッガ	(1,024 バイト)	4 個
ローダ	(257 バイト)	1 個
単純エディタ	(1,792 バイト)	7 個
単純アセンブラ	(2,048 バイト)	8 個

こうしたシステム・プログラムが常駐になってしまえば，ユーザ・プログラムの方はテレタイプの紙テープ・リーダや紙テープ・パンチの毎秒 10 字の速度でもまずまずの仕事ができる．

(3) デラックスなマイクロコンピュータ

ソフトウェア開発の能率を上げる目的で，マイクロコンピュータにかなりの設備投資ができる場合には，ミニコンと同様にあるいはそれ以上にデラックスな構成にすることが可能である．ただこの場合，メモリや周辺機器が相対的に高いため CPU が微小で安いというマイクロコンピュータの利点はもはや生きないが，開発すべきプログラムのテストのために，同じマイクロプロセッサを用いるということが主眼となる．

デラックスなマイクロコンピュータでは主メモリは 64キロバイト程度まで実装でき，周辺機器としては次のようなものが使われる．

ディスケット（フロッピー・ディスク）
カセット磁気テープ
高速紙テープ・リーダ／パンチ
ラインプリンタ
CRT 文字ディスプレイ／キーボード

また外のマイクロコンピュータ回路へ接続するためのプローブやPROM書き込器をつけることもでき，システム・プログラムもディスク・オペレーティング・システム込みとなってくる．前述のようなROM シミュレータや外のマイクロコンピュータ回路に対するシステム・アナライザの機能をもたせる試みも行われる．

(4) 上位機種のクロスソフトウェア

コンピュータとしてみた場合，マイクロコンピュータは基本構成のものはもとより，上記デラックスな構成のものでも，ミニコン以上の上位機種に比べれば能力は足りない．そこでミニコン以上の上位機種の能力を使ってソフトウェア開発を行うこともよく行われる．この場合は，システム・プログラムとして，クロスアセンブラ，シミュレータ，クロスコンパイラなどのいわゆるクロスソフトウェアが使われる．上位機種としてはアメリカでは TSS もよく使われるが，わが国では TSS の利用は，電通 GE の TSS のようにアメリカで開発されたクロスソフトウェアがわが国の端末からもそのまま使えるという場合が多いようである．上位機種によるソフトウェア開発には，上位機種の能力や設備がそのまま使える利点があるが，クロスソフトウェアがかなり大きくなり，使用コストが必ずしも低くないこと，プログラムのランがシミュレータを通した間接的なものとなるため，タイミングや割込み処理の微妙なプログラムは完全なテストができないなどの問題もある．最近は電電公社の TSS でもクロスソフトウェアが利用できるようになっている．

以上いずれの道具立てを採用するかは，開発すべきソフトウェアの規模，利用可能な設備や人員，投資額など数多くの観点からきめなければならない．強いていえば(2)のファームウェア付マイクロコンピュータはマイクロコンピュータらしい使い方といえよう．

6. 高水準言語の利用と PROM モジュール

現在のところマイクロコンピュータのプログラミングにはもっぱらアセンブラ語が使われている．これはとくにプログラムを入れる PROM のメモリ容量を低く抑えたいという経済的な要求から，コンパクトなオブジェクト・プログラムが必要とされるからである．しかし上位機種の歴史が示すように，これではソフトウェアの生産性の向上やコスト・ダウンは望めないから，今後は高水準言語もだんだん使われてゆくであろう．

クロスコンパイラ用としては，すでにインテル8008/8080（およびそのセカンド・ソース）用には PL/M と呼ばれる PL/I のミニサブセットに当るような言語が使われている．このクロスコンパイラは互換性を重んじて基本 FORTRAN で書かれ，そのソース・プログラムは FORTRAN で数千行にも及ぶほう大なプログラムであるが，コンパイル自体やオブジェクト・プログラムの効率はともかくとして，PL/M でプログラムを書けば，とくに大きなプログラムの場合ほどソフトウェアの生産性があがることは確かである．いずれにせよクロスコンパイラという形であれば，上位機種を使うのだからマイクロコンピュータ用の高水準言語の利用が可能となるのは当然といえる．

むしろ今後の興味ある問題は，マイクロコンピュータ自体で高水準言語がコスト的にいつ利用可能になるか，すなわちコンパイラやインタプリタがいつ利用可

能になるかである．そもそもマイクロコンピュータの機械語はミニコンよりも低レベルであり，いわゆる垂直形のマイクロプログラミング用のマイクロ命令に近い．したがって上述のとくにインタプリタを PROM に入れて固定化することを考えると，これは高水準言語を直接にマイクロプログラムでインプレメントするのと非常に近い．

既存の言語の中でマイクロコンピュータ用としてよくあげられるのは，FORTRAN, APL, BASIC などのサブセットである．これらの利用はすでに試みられている．この他たとえばナショナル・セミコンの 16 ビット・ワンチップ CPU (PACE) をベースとするデラックス・マイクロコンピュータには PL/M コンパイラが予定されている．またブロック図言語や 4 ビット CPU 用の専用言語を組み込んだ例も発表されている[4]．高水準言語のコンパクトなコンパイラあるいは PROM によるコンパクトなインタプリタの直接インプレメンテーションは今後の重要な研究開発項目となるであろう．

ソフトウェアに関してもうひとつ興味がもたれるのは，サブルーチンやプログラム・ライブラリなどソフトウェア・モジュールの PROM によるハードウェア・モジュール化である．ソフトウェアを PROM の形で交換したり，売買したりすることは現在でも可能である．現に前述のファームウェア化されたシステム・プログラムのヴァージョンアップは PROM の差し換えという形で行われる．

しかし現在の多くのマイクロコンピュータのアーキテクチャは，上の意味のソフトウェアのハードウェア・モジュール化には，アドレス付けが固定されているために全く不向きである．これを本格的に行うには，PROM 上のアドレスが相対アドレスあるいは論理アドレスとして CPU に解釈され，適当なアドレス変換が行われるフローティング・アドレス機構あるいはハードウェア・リンケージ・エディタのようなものが必要である．また PROM のプログラムには変更がきかないから，変更の必要があるときはそれに簡単にパッチ（そこだけ制御を他のメモリ部分に移してそこを実行）する機能あるいは RAM 付 PROM などもあれば面白い．とにかく特定の絶対番地にしばられることなく，あるサブルーチンを含む PROM が取付けられるようになれば，ソフトウェアを含んだ PROM が商品として売買できるようになるであろう．

7. マイクロプロセッサ・システムの信頼性

マイクロプロセッサのような複雑な LSI の信頼性を論ずるには，どの位の割合でこわれるかという物理的な信頼性の他に，内蔵されている回路に論理的な虫がないかどうかという意味の論理的な正当性も問題になる．とくに最近は半導体製造技術が進歩し，回路は望めばいくらでも複雑なものが作れるが，論理的な正当性が保証できないことで複雑さが限定される傾向があり，技術が人間を追い越したような面がある．マイクロプロセッサが本当に設計通りに動いているかどうかをどうチェックするかは今後ますます重大な問題となることであろう．

マイクロプロセッサに関連して，よく心配されるのは，PROM の信頼性，すなわち書込んだ内容が果して消えないで永久にもつものかどうかという問題である．実際 PROM が不良品だと分ったとき，もともとの PROM が不良だったか，書込み方が悪かったかは問題となりうるところである．消えるかどうかについては，PROM 書込みの微調整に左右される可能性もありうる．しかし実用上は，書込みを十分よくやっておけば内容が消える心配はないとされている．

現在のマイクロプロセッサ・チップやメモリ・チップでは内部的なパリティ・チェックなどの機能は全く入っていない．高信頼性確保の手段はとくにとられていないといってよい．この点も将来は CPU チップにも少くともエラー訂正コード，再実行機能などは入ることが望ましく，メモリ・チップにも循環冗長度チェック (CRC) コードのようなものが使われることが望ましいであろう．

8. 今後の応用と技術開発

マイクロコンピュータの応用が今後ますます盛んになることは確実である．最終的な応用としては，家庭電気器具，ゲーム・リクレーション機器 (FRED＝Flexible Recreation Equipment Device)，スポーツ機器（ゴルフ練習器など）などまで期待されている．ただこうした素人向けの応用の場合には，コンピュータの利用が人間に与える心理的な影響を十分検討し，コンピュータ公害が起きぬように注意深い設計を行い，悪影響が心配されるような場合には製品化を行わないだけの倫理性が要求されよう．最近の応用で面白いのは，コンピュータ・オートメーション社のインテリジェント・ケーブルである．これはコンピュータと

周辺機器をつなぐケーブルの途中に，ピコプロセッサと称するマイクロプロセッサをそう入したもので，ケーブルの途中で直並列変換，エラーチェックなどが行えるようになっている．このように「すべてをインテリジェントに」という哲学に立てばマイクロプロセッサの応用にはほとんど限りがないようにみえる．

最後に今後の技術開発についてすでにのべたことを中心に整理しておく．まず LSI チップについていえば，CPU チップでは，内部スタックや内部メモリの増加，内部チェック機能の強化，フローティング・アドレス機構の組み込みなどを検討したい．メモリ・チップは，やはり内部チェック機能の強化，電気的消去可能 PROM（すでに日電より発表あり），それに RAM 付 PROM，不揮発性 RAM，ファイル型 RAM（たとえばインデックス・シーケンシャル・ファイル用）なども検討に値しよう．インターフェース・チップではプログラマブル化（CPU 内蔵へ）である．いずれにせよ LSI チップでは大量生産によるコスト・ダウンが重要であるから，メーカとしてはチップのできるだけの汎用化，インテリジェント化をはかって統合し，チップの少品種大量生産をはかろうとするであろう．

システム・レベルではマイクロコンピュータを数多く（1,000 台～10,000 台）組合せたネットワークに興味がもたれている．ただこの種のネットワークは今まで論じられている割には実現されていないようであるが将来の可能性としては面白いと思われる．マイクロコンピュータの中には，SMS 社のマイクロコントローラ[5]やテレダイン社のマイクロコンピュータのようにメモリやインターフェースを含む基本的なマイクロコーピュータをハイブリッド技術を使ってひとつのパッケージに収めたものもすでに出ている．将来はネットワークに組合せるのが容易なようなパッケージも需要があれば出現しよう．現在のコンピュータ・システムでも，すべての周辺機器や端末機に個別にマイクロコンピュータをつけてインテリジェント化することを考えれば，全体として 1,000 個程度のマイクロコンピュータを含んだオンライン・システムを想像するのは容易である．

マイクロコンピュータのソフトウェアについては，すでにのべたように，高水準言語の直接利用やファームウェア PROM の相互流通，さらにはユーザ団体によるプログラム・ライブラリーの確立などが今後の課題だと思われる．

参 考 文 献

1) 日経エレクトロニクス，エレクトロニクス（オーム社），電子科学（産報），インターフェース（CQ 出版）などの最近号を参照．
2) 佐々木：格段に I/O インターフェースが進歩した最近のマイクロコンピュータ，日経エレクトロニクス，pp. 63～75，(1975. 7. 28)．
3) 石田：マイクロコンピュータの使い方，産報，(1975)．
4) L. H. Anderson : Development of a portable compiler for industrial microcomputer systems, AFIPS Proc. of NCC, pp. 30～40, (1975).
5) M. Liccardo : Architecture of micro controller system, AFIPS Proc. of NCC, pp. 75～84, (1975).

（昭和 50 年 10 月 27 日受付）