II. 基礎編

2. マイクロコンピュータのハードウェア (II)

日本電信電話公社 池野 信一
武蔵野電気通信研究所

6. マイクロコンピュータのアーキテクチャ

今回は、いよいよ本体の話に入るわけであるが、最初にその大まかな構成と動作について簡単に述べることにしよう。アーキテクチャというのは、使う立場から見たハードウェアの論理的構成のことをいうのである。

図 28 は大まかな基本構成を示したもので、全体の制御と演算を受け持つ CPU (中央処理装置)、データを記憶するメモリ、そして I/O (入出力) 装置との間でデータを出し入れするための I/O ポートなどが主な構成要素となっている。

その基本的動作は、

装置間のデータのやりとり
CPU 内でのデータの処理

という2つからなるということができる。

図では2つのメモリとひとつの I/O ポートしか書いていないが、実際にはもっと多数のメモリモジュールや I/O が接続されるのが普通である。このように多くの装置の間でのデータのやりとりには、バス方式が威力を発揮する。

バスというのは、共通に引かれた信号線で、各装置からの信号がこれに“乗り込む”ようにしたものであって、装置間に別々の線を張るより、はるかに経済的で使い易

図 28 マイクロコンピュータの構成

“Microcomputer Techniques for TV and Radio Engineers; 3. Microcomputer Hardware (II)” by Nobuichi Ikeno (Musashino Electrical Communication Laboratory, NTT, Tokyo)

い。ただ、電気信号のことであるから、同時に2つ以上が乗り入れるわけにはゆかず、ひとつの装置から信号が出ているときは、他の装置はその通信のじゃまにならないようにフローティング状態にならなければならない。

どの装置をアクティブにするかを指定する信号も、装置番号を表わすアドレスという形でバスで送ることができるし、信号を送り出す役か受けとる役かの区別をするなどの制御信号も別のバスで送ることができる。

こうして、普通、コンピュータの中には、データバス、アドレスバス、コントロールバスの3種類のバスが設けられている。

このように説明すると、前回のべたメモリの内部構造もまたバス方式を使っていることに気がつくであろう。番地をつけられた個々の語がすべて共通のバスにつながれ、アドレスで選択され、R/W 信号で読み書きの区別がつけられてデータの出し入れをするわけである。

逆に見れば、I/O などとも読出しあるいは書込み専用のメモリと見なすこともできるわけで、実際に CPU から見るとき区別をしないコンピュータも多い。

装置間のデータのやりとりといったが、そのほとんどは CPU の制御のもとで、CPU とメモリ、あるいは CPU と I/O との間で行われる。この場合には信号のぶつかり合うこともなく簡単である。

ところで、高速の I/O 装置とのデータのやりとりをするとき、いちいち CPU にデータを読込んでからメモリに移すというのではスピードが間に合わない場合がある。こんなときは、CPU の制御をはなれて、I/O とメモリの間で直接データを受け渡すことができるようにしたものもある。

これを DMA (ダイレクトメモリアクセス) というが、そのためには CPU を一時止めるか、CPU がバスを使わない短い時間を利用してやるとしなければならぬ。

以上、バスの話ばかりしてきたが、マイクロコンピュータでは、CPU、メモリなど、ほとんどLSI化されているので、目に見える配線はそれらをつなぐバスだけといってよいほどである。バスを理解すれば、マイクロコンピュータのハードウェアのかんりの部分を理解したといつてよいであろう。

次にCPU内での処理であるが、これはむしろソフトウェアの話のときに詳しく述べた方がよいと思われるので、ここでは概略の説明だけしておく。

CPUは、それ自身では2数の加算とか、メモリのある番地からデータを読出すとか、単純な1つずつの動作しかできない。これに、ある手順に従ってこのような基本動作を順に実行させ、まとまった仕事をさせるには、次々に命令を与えてやる必要がある。

電卓ではそのつど人間がボタンを押して計算を進めるわけであるが、コンピュータの場合は、この命令の系列を符号化してあらかじめメモリのなかに記憶させておくことにより、CPUがこれを自分で1つずつ読出して解釈実行するのである。そしてパラメータを変えながら同じ手続きを必要な回数繰返すとか、いろいろの制御ができるようになっており、まとまった複雑な仕事を高速で処理することができるようになっている。

このように、仕事の手順を命令の系列の形に書き表わしたものがプログラムであり、これをあらかじめメモリのなかに入れておくやり方が“ストアプログラム方式”である。

プログラムをとり変えれば同じコンピュータでいろいろ違った仕事をさせることができるが、マイクロコンピュータでは、固定した用途に使われることも多い。この場合、プログラムをROMに書き込んでおくのが安全で便利である。

ハードウェアに対応して、プログラムのことをソフトウェアということがある。コンピュータのすばらしい機能は、ハードウェアとソフトウェアの協同により発揮されるわけであるが、プログラムは簡単に書き換えることができ、柔軟性があるということからソフトというらしい。とはいえ大規模なソフトウェアは変更が容易でなく、ハードより硬いとさえいわれることがある。

さて、命令は普通図29のように、命令の種類を表すオペレーション部と、命令の動作に関連するパラメータを指定するオペランド部からなる。これらはそれぞれ命令コード、アドレスとも呼ばれる。命令の長さは必ずしも一定でなく、オペランド部のないものもある。

図29 命令の形

図30 命令の実行

CPUのなかには、このような命令の系列を実行するためいくつかのレジスタと制御回路が入っている。図30にはそのレジスタの主なものが書いてあるが、アキュムレータはデータの演算用のレジスタで、プログラムカウンタは次に実行すべき命令のアドレスが入っているレジスタである。カウンタという理由は、アドレスを送り出したあと、自動的にインクリメントされる(値に1が加えられる)からである。

さて、プログラムカウンタから命令のアドレスが送り出されると、メモリからその番地の内容が読出されて命令レジスタに入り、解読される。その結果、命令にオペランド部が続いていることがわかれば、次のアドレスをプログラムカウンタから送り出してオペランドを読込む。オペランドはたいていデータのアドレスを示すもので、アドレスレジスタに入り、改めてアドレスバスに送り出される。メモリからはそれで指定される番地のデータが読出されてアキュムレータに入れられるという具合になる。こうしてたとえば、

0300番地のデータを読出してアキュムレータにおけるという命令が実行されたわけで、続いてCPUは次の命令の読出しに移るのである。

命令のなかには、プログラムカウンタの内容を書きかえるものもあって、別の新たな番地から実行を始めることができる。これを飛びこし命令という。

データレジスタは2つ以上あることが多い。これにも番号をつけてオペランド部で指定してもよいが、それに必要なビット数は少ないので、命令コードの中にうめこむことが多い。

ほかにインデックスレジスタ、スタックポインタなどという便利なレジスタをそなえたものもある。前者は、命令のアドレス部にその内容を加えてからアドレスバスに送り出すという、いわゆるアドレスのインデックス修飾のために使うことができ、後者はメモリの一部をスタックメモリとして働かせるためのものである。

スタックというのは、穴ぐら式メモリとでもいうべき

もので、データを入口から次々につめておむことができるが、とり出すときは一番上のもの、つまり最後につめこんだものから順にとり出さなければならないというものである。

実際には普通のメモリの一部をこれにあて、スタックの一番上のデータのアドレスをスタックポインタに入れておいて、データをつめておむときは、スタックポインタで指される番地にデータを入れ、スタックポインタの内容をデクリメントする(1つへらす)。とり出すときはその逆をやればよい。スタックは後で述べるサブルーチンとか割り込みの処理にきわめて重要な役割りを果たす。

最後に I/O 部であるが、バスにつながれた I/O ポートは、ラッチやゲートで作られ、命令に従ってデータをバスから受けとり、信号形式の変換などをするインタフェースを通して各種の入出力機器、あるいは制御すべき装置に送ったり、逆にこれらの装置からのデータを受けてバスに送り出す役割りを果たすところである。また単にデータばかりでなく、装置の状態の変化に応じて、CPU に割り込みをかけるなど、制御信号の受け渡しも行う。

インタフェースから先は応用によりきわめてバラエティに富んだところであり、工夫の楽しめるところでもある。

7. マイクロプロセッサ

CPU はかなり複雑な回路で構成されるが、マイクロコンピュータでは、これが1個ないし数個のLSIチップにまとめられている。これがマイクロプロセッサで、1個の場合、ワンチップCPUということもある。

マイクロプロセッサも、各社から次々に新型が発表されて、きわめて種類が多くなったが、いくつかの特徴で分類することができる。まず、その処理するデータ長から見ると、

4ビット、8ビット、12ビット、16ビット

ビットスライス

などに分けられる。4ビットのものは、キャッシュレジスタなど、簡単な制御用に便利で、数としては最も多く出ているようである。しかし、最も手頃で使いやすく、多方面に使われているのは8ビットのもので、種類も多い。ただ、汎用コンピュータとしてはビット数の多い方が便利で、いずれ16ビットが主流になってくると思われる。

ビットスライスというのは、1つでは2ビットか4ビットの処理しかできないが、これをならべて長いビット数のものができるようになっているもので、高速用のものが多い。

表3 マイクロプロセッサ

品名	データ幅	メーカー	プロセス	ピン数	開発年	セカンドソース等
4004	4	Intel	PMOS	16	71	
μCOM-4	4	日電	NMOS	23	72	
4040	4	Intel	PMOS	24	74	
TMS-1000	4	TI	PMOS	28	75	
8008	8	Intel	PMOS	16	72	
8080	8	Intel	NMOS	40	73	TI, AMD, 日電, 三菱
M6800	8	Motorola	NMOS	40	74	AMI, 富士通, 日立
PPS-8	8	Rockwell	PMOS	42	74	
F-8	8	Fairchild	NMOS	40	75	MOSTEK
6502	8	Mos Technology	NMOS	40	75	
COSMAC	8	RCA	CMOS	40	75	
SC/MP	8	NS	PMOS	40	75	
Z-80	8	Zilog	NMOS	40	76	(8080の拡張版) シャープ
TLCS 12A	12	東芝	PMOS	36	74	
IM6100	12	Intersil	CMOS	40	75	PDP8 とコンパチブル
PFL-16A	16	パナファコム	NMOS	40	75	
CP 1600	16	GI	NMOS	40	74	
TMS9900	16	TI	NMOS	64	76	
3000	2ビット	Intel	STTL	28/40	74	高速
Am 2901	4スライス	AMD	STTL	28/40	75	
M 1C800	4イス	Motorola	ECL	48	76	

次に製造プロセスで分類すると、

PMOS, NMOS, CMOS, バイポーラ

などとなる。NMOS が最も普通で、CMOS は小電力用、バイポーラは高速用である。

必要な電源について見ると、5V、12V、-5V など3電源のものや、2電源のもの、5V単一のものなど、いろいろであるが、特殊なものを除き次第に5V単一に統一される傾向にあるようである。

表3に主なものの一部をあげた。このほかにも沢山あり、まだ新型も次々に生れるようである。現在のところ最もよく使われているのは8080系で、やはり先発の強みであろうか。これにつぐのはM6800のようである。

すでに述べたように、コンピュータを動かすにはソフトウェアが必要であるが、プロセッサが違つと一般に同じソフトウェアをかけることができない。ソフトウェアの開発にはかなりの労力を必要とするので、これになるべく広く流用できることが望ましい。そこで同じアーキテクチャをもつマイクロプロセッサが別の社で作られることも多い。これがセカンドソースである。同じプロセ

ッサが世の中に多数あるということは、使う側にとっては心強いといえよう。

さて、マイクロプロセッサの最近の傾向のひとつは大規模集積化で、すでに CPU とメモリ、I/O ポート、あるいはメモリと I/O ポートなどを1つのチップにまとめたものが始まっており、こうなると、ワンチップコンピュータということになるわけで、ほんとにコンピュータが1つの部品になってしまったという感じである。

さて、ここで、8080 A (8080 の改良版) を例にとって、その具体的構成をしらべてみることにしよう。8080 A は図 31 のように 40 ピンのパッケージに入っている。40 ピンというのは、現在のところ、最も標準的なサイズである。

アドレス線は A0 から A15 までの 16 本、つまり、2 バイト長で、 $2^{16}=65536$ (通称 64K) バイトまでメモリ番地が指定できる。16 進表現を使えば、0000 番地から FFFF 番地までというわけである。

データ線は D0 から D7 までの 8 本で、命令やデータはすべてここを通して出入りされる。つまりこれは双方向信号線である。8080A の場合、さらにステータス信号というものもここから送り出される。マイクロプロセッサでは、パッケージのピン数に制限があるから、このように同じピンを通して、別の信号を時分割的に出入りすることはよく行われる。

8080A の前身である 8008 では 18 ピンのパッケージであるため、アドレスの 16 ビットも 2 回に分けてデータ線に送り出されるようになっている。このような方式では、バスにラッチをつなぎ適当なタイミングで信号をとり出して分離することが必要である。

ϕ_1 , ϕ_2 はクロック信号の入り口である。システムの動作はこのクロックに歩調を合わせて進められ、処理速度がこれで決定されるから、水晶振動子を用いた正確な発振器で作られるのが普通である。クロックを速くすれば速度は上がるが、許される範囲が定められていて、8080 A の場合、クロック周期は $0.48 \sim 2.0 \mu\text{s}$ となっている。

図 32 はその波形で、各部の時間間隔について制限があり、電圧レベルも 9V 以上必要で、普通の TTL レベルではない。カウンタとゲートを組み合わせ、出力にオープンコレクタを高い電圧にプルアップしたものを使えばできるが、クロックジェネレータという IC を使うのが便利である。最近はこの回路もプロセッサに内蔵する傾向にある。

その他、沢山信号線があるが、一応ざっと説明しておこう。

SYNC (出) プロセッサの動作はいくつかのマシンスサイクルに分かれるが、各サイクルの始まりを示す信

図 31 8080 A

図 32 クロック

号がこれである。

RESET (入) これを 3 クロック以上 high (H) にすると、レジスタ類が 0 にクリアされ、0 番地からプログラムがスタートする。M 6800 のように、スタート番地が任意に指定できるものもある。

HOLD (入) これを H にすると、プロセッサはストップして HOLD 状態になり、HLDA 信号を出す。HOLD 状態では、アドレスバス、データバスがフローティング状態になってバスから切りはなされるから、I/O とメモリ間で DMA ができる。

READY (入) これが low (L) になると WAIT 状態になり、WAIT 信号を出す。アドレス信号は出たままになるから、遅いメモリの場合、CPU を待たせるのに使う。

DBIN (出) これが H ならデータ線が受け入れ状態であることを示す。メモリの読出しコントロールに使う。

$\overline{\text{WR}}$ (出) これが L なら、データ線にデータが送り出されることを示す。メモリ書込みのコントロールに使われる。上のバーは負論理出力であること、つまり H, L が逆になっていることを示す。

INT (入) 外部から割り込み信号を与えるところである。割り込みというのは、現在実行中のプログラムを一時中断させ、緊急な仕事を先にやってからもとのプログラムの実行を再開させることで、いろいろな使い方があ

図 33 8080A のブロック図

るが、ある仕事をやりながら、外部装置の動作完了を待つなど、機器の制御にきわめて便利なものである。詳細はソフトの項で述べることにする。

INTE (出) 割り込み受けつけ可能であることを示す。

このほか、先ほど述べたように、データバスに送り出される次のような信号がある。

INTA (D0) 割り込みが受けつけられたことを示す。

$\overline{WO}$ (D1) これがLなら、そのマシンサイクルは書き込みサイクルであることを示す。 $\overline{WR}$ とほぼ同じである。

STACK (D2) アドレスバスにスタックポインタ内のアドレスが送り出されることを示す。この信号でメモリを切りかえて使えば、スタック専用のメモリとすることができて、64Kバイトまでのアドレスの範囲をスタックと関係なく使うことができるようになるが、普通はアドレスに余裕があるからこのようにする必要はない。

KLTA (D3) HALT 状態であることを示す。

OUT (D4) 出力命令のサイクルであって、アドレスバスには出力機器番号が出ることを示す。

M1 (D5) 命令読出しの最初のサイクルであることを示す。

INP (D6) アドレスバスに入力機器番号が送り出されることを示す。

MEMR (D7) 読み込みサイクルであることを示す。

さて、CPU の中味についてはあまりくわしく知る必要はないが、一応 図33 にその概要を示しておいた。レジスタが沢山あり、アキュムレータ (Aレジスタ) のほかに B, C, D, E, H, L などがあるが、いずれも8ビットで、(BC), (DE), (HL) のように2つずつ組にして16ビットとしても使える。HL のペアはデータのアドレスのレジスタともなり、これで指定される番地と各レジ

図 34 命令実行のタイミング

タの間でデータのやりとりができる。

図には書かなかったが、ALU に付属して演算結果の状態を表わすフラグのレジスタがある。

さて、命令の実行はいくつかのマシンサイクルに分かれ、各サイクルはまたいくつかのステートからなる。図 34 は「STA B2 B3」という命令のタイミングを表わしたものである。これは3バイトの長さをもつ命令で、第2バイト B2 の内容 (これを <B2> と書く) は下位アドレス、<B3> が上位アドレスを表わし、「アキュムレータの内容を <B3> <B2> 番地に書き込め」という意味をもつ。

各マシンサイクルの頭では SYNC 信号が出て、データバスにはステータス信号 (SI) が送り出される。第1のマシンサイクルは命令フェッチサイクルで、PC から命令アドレスが送り出されて、命令コード (INSTR) が読み込まれる。点線で書いたのは CPU への入力を表わしている。第2、第3のサイクルで、オペランド <B2> <B3> が読みまれ、第4サイクルでは <B3> <B2> がアドレスバスに送り出され、データバスにはアキュムレータ (A) の内容が送り出され、書き込み制御の $\overline{WR}$ がLになってメモリに書き込まれるのである。

図では READY 入力が H の状態にある場合を示したが、図の下に示したように、T2 ステートのとき、READY が L であると Tw という待ち状態に入り、WAIT 信号が出る。READY がHになれば T3 へ進み、動作が継続される。

おそいメモリを使うときはその速度に合わせて適当な期間 READY を下げてやればよいわけであるが、WAIT 出力を READY につないでやると、各サイクルご

図 35 CPU まわり

とに T_w が1つ入り、簡単にメモリアクセスの時間をのばすことができる。

ただし、速いメモリと遅いメモリを共用するときは、全部遅くなってしまってややもったないが、回路が簡単であるので使われることがある。

8. マイクロコンピュータの具体的構成

マイクロプロセッサについて勉強したから、これを使ってコンピュータを構成してみることにしよう。

図 35 は CPU まわりの回路の一例である。アドレスバスとデータバスにはバスドライバをつけてあるが、小規模のシステムではつけなくてもよい。8080Aの出力は1.9mA位で、普通のTTLなら1個強、MOSメモリなら10個位はドライブできるであろう。各ドライバは、HLDA信号が出たとき、フローティング状態になるように制御される。データバスの8216は図36のような双方向性バスで、DBINにより方向が切りかえられる。8216は、左側の4,2のピンなど2つずつまとめてつなぎ、3ピンの側をバスにつなぎ。DBINなど、インバータ2個で中継してあるのは、増幅器としてドライブ能力を増すためである。

クロック回路には8224というクロックジェネレータを使っている。これには、RESET信号などのタイミングを整える機能もあり、また、SYNC信号を使って、ステータス信号をラッチするためのストローブパルス(STSTB)を作り出す回路も入っていて便利である。

8080Aの入力ピンへの信号は、TTLレベルより少し高くなっているため、TTLでドライブするときはプル

図 36 双方向性バスドライバ 8216

図 37 システムコントローラ 8228

アップ抵抗が必要であるが、8224のようなファミリICを使えばその必要はない。

右下の74LS75はラッチで、データバスから出るステータス信号を取り出し、各種の制御信号を作るためのものである。この部分も、システムコントローラ8228を使うと、図37のように簡単になる。8228の中には、双方向性バスドライバも入っているのである。BUSENはデータバスをフローティングにしてDMAを行うための入力、HLDAを反転したものをつないでおけばよいであろう。

このように、各マイクロプロセッサには、それと組み合わせると便利なファミリICが出ているから、それらを利用すると回路は簡単になる。

次はメモリである。数Kバイト程度のRAMには前回述べた2102を使うのが便利であろう。もっとも最近

図 38 小規模メモリの構成例

は 4 K ビットのスタティック RAM が出回り始めたから、これを使えばさらに小形にできる。

小容量のメモリの場合は 256×4 ビットあるいは 128×8 ビットという形のメモリ素子を使うのがよいであろう。前者の例は 2101, 2111, CMOS の 5101 などがあり後者の例としては 6810 などがある。

図 38 は 2111 の例で、2 個で 256 バイト、4 個で 512 バイトになる。チップセレクト $\overline{CS}$ が 2 本出ているから、アドレスのデコードに便利である。図では A 8 で 256 バイト 2 組の切りかえを行い、A 15 で ROM と RAM の切り換えを行っている。OD ピンはメモリを読むときだけ出力が出るようにコントロールするためのもので、R/W は書き込みのコントロール用である。

μPD 454 は ROM で、 $\overline{CS}$ が 1 つだけであるので、外部のゲートで制御信号を作っている。

さらに ROM や RAM の容量をふやした場合には、1 組の素子だけを選ぶチップセレクト信号をアドレスから作るのに、アドレスデコーダが必要になる。これには前回述べた 7442 とか、74139, 74155 などを使えばよい。

大容量メモリの場合は、4K とか 8K 程度ずつボードに分け、バスへの出口にバスドライバをつけるようにする。

そのやり方は、CPU とバスとの接続のところと同様で、データバスには双方性のものを使う。アドレスバスはトライステートでなく、普通のインバータでも差支えない。インバータでは信号が反転して、アドレスが変わってしまうと心配されるかも知れないが、書くときと読むときで同じ変換が行われるから、外から見ればまったく差支えないのである。

なお、2102 などではデータの入力、出力が分かれているから、前回の図のように、出力側だけバッファをつける

図 39 出力ポート

るだけでもよいし、両方つけるなら、8216 の 2 つに分かれた方を入力出力に別々につなぐようにする。

容量が大きい場合は、ダイナミック RAM を使った方が安くつくようであるが、リフレッシュ回路により、適当な時間間隔で内容を再生してやる必要がある。Z80 などでは、リフレッシュ用の信号も出るようになっているので、ダイナミック RAM を使うのに便利である。

話を I/O ポートに進めよう。データバスに図 39 のように 7475 などのラッチをつけ、これにアドレスデコーダをつけて、あるアドレスで書き込み信号 ($\overline{MEMW}$) がきたときデータをラッチするようにすれば、CPU から見れば、メモリに書込んだと同じことであるが、ラッチの出力を外にとり出せば、立派な出力ポートとなる。

入力は、ラッチの代りに 74125 のようなトライステートバッファを使い、アドレスデコーダには $\overline{MEMR}$ の方を入れて、あるアドレスで読む信号が来たとき、外からのデータをバスにつなぐようにすれば、メモリ読み込みと同じ形で入力ができる。

このやり方では、メモリと I/O は同じアドレス空間を使うから、たとえば A 14 が 0 ならメモリ、1 なら I/O というように分けて使うのがよいであろう。I/O ポートの数が少なければ、それを区別するためのアドレスデコーダは簡単ですむ。

ところで、8080A の場合は、別に入出力命令というものがあり、これを使うと、CPU から I/O 用の読み書き制御信号が別に出ようになっている。これで I/O ポートを制御すれば、同じアドレスでもメモリと区別できるから、メモリは I/O に関係なく使うことができる。ただ、I/O 命令でアドレスバスには上 8 ビット、下 8 ビットに同じものが出るから、 $2^8=256$ 個までの I/O が指定できることになる。

出力ポートのラッチには、7475 のほか、図 40 のような 8 ビットがひとつに入った 8212 というものがある。これには割込み用の回路もついているし、トライステア

図 40 8212

トゲートもついているから、入力用にも使えるなど、なかなか便利である。

さらに高級な I/O ポートを望むなら、**PPI**(プログラマブルパリアフェラルインタフェースアダプタ)と呼ばれる 8255 がある。これは図 41 のように PA, PB, PC という 8 ビットずつ 3 組のポートがあり、PC はさらに 4 ビットずつに分かれている。そして内部にもう 1 つコントロールレジスタというものがある。その選択にはチップセレクト (CS) のほかにアドレスの A0, A1 をつなぐピンがあり、この 2 ビットで、3 つのポートと 1 つのレジスタの計 4 個のうちの 1 つが指定できるようになっている。

コントロールレジスタの各ビットには図 42 のような役割割りけが割りつけられており、これに適当な 1, 0 のパターンをセットすることにより、いろいろな形で使用することができる。これがプログラマブルという意味である。その詳細を説明する余裕はないが、ひとつの例を示すと、10000011 とセットすれば、モード 0 で PA と PC の上半分が出力、PB と PC の下半分が入力となる。モード 1, 2 では PC は割込みなどの制御信号の受け渡しの役を行い、モード 2 で PA は双方向性のポートとして働く。

その他、USART (Universal Synchronous/Asynchronous Transmitter/Receiver) 8251 というものがあり、CPU から並列のデータを受けとって、直列の信号に変換して送り出したり、逆に直列の信号を受けて、並列信

図 41 PPI 8255

図 42 8255 のコントロールワード

号に直したりする機能をもっている。これに簡単なインタフェースをつければ、テライプや通信線につなぐことができる。

以上、8080Aを使った場合の例を説明したが、他のものでも考えの本質にはあまり違いはない。8080Aの改良版ともいえる 8085 では、クロック回路が内蔵され、ステータス信号も直接ピンに出るようになっている。しかしその代りにアドレスの下位 8 ビットとデータバスが共通のピンを使っている。

8080Aについてよく使われる 6800 では、クロックも簡単になり、タイミングも簡単でマシンサイクルが 1 クロック時間になっている。また制御信号類も整理されて少ないし、命令のアドレスモードもすっきりしていて、ある意味で使い易いともいえる。8085 も 6800 も電源は 5 V 単一である。いずれを使うかは人の好みにもよるであろう。

講座 テレビ・ラジオ技術者のためのマイクロコンピュータ

用語集 (3)

〔あ 行〕

アーキテクチャ architecture

使う立場から見たハードウェアの論理的構成。プログラムに必要なハードウェア仕様。広義にはハードウェアの機能構成。

I/O ポート I/O port

コンピュータと外部装置をつなぐためのデータの出入口。

アキュムレータ accumulator

CPU のなかのレジスタの一種で、加減算などの演算回路と連結されている。

アドレスデコーダ address decoder

アドレスはメモリや装置の番号を表わしたものである。このアドレスで指定された番号の出力線だけに信号を出し、装置の指定に使われるのがアドレスデコーダである。

ALU arithmetic logic unit

算術演算や論理演算を行う回路。

〔か 行〕

クロック信号 clock signal

システムの動作の歩調をとるために加えられる周期的パルス信号。

〔さ 行〕

サブルーチン subroutine

ひとまとまりのプログラムで、他のプログラムから呼ばれるという形で使われる。実行が終ると呼んだプログラムに帰る必要があるが、その帰り番地をおぼえておくのに普通スタックが使われる。

スタック stack

つめこみ式あるいはつみ上げ式ともいうべきメモリで、データを入れるときは上に順々につみ上げてゆき、とり出すのも一番上からだけできるもの。いわゆる LIFO (last in first out) 式メモリである。

スタックポインタ stack pointer

普通のメモリをスタックとして使うために、スタックの一番上の位置を示すアドレスを入れておくレジスタ。データの出し入れのとき自動的に増減される。

ステータス信号 status signal

プロセッサなどの状態を表わす信号。たとえばバスへデータを送出中であることを示す信号など。

ストロブパルス strobe pulse

変化する信号のある瞬間の状態をフリップフロップなどにとるために加えられるパルス。

〔た 行〕

DMA direct memory access

外部装置とメモリ間のデータの授受を、CPU を介さず直接

行うこと。バスの使用が CPU とぶつからないよう、DMA のとき CPU を止める方法と、CPU がバスを使わぬ合間をぬって行うサイクルスチール法とがある。

チップセレクト chip select

メモリをいくつかのアドレスの範囲に分けてそれぞれをメモリ IC で作るとき、アドレスがどの範囲にあるかにより、必要な IC チップを選ぶこと。

〔は 行〕

バス bus

複数の装置間に引かれた共通の信号線。各装置間の信号授受が時分割的に行われる。

PPI programmable peripheral interface adapter

汎用の入出力ポートで、内部の制御レジスタに、必要なパターンをセットすることにより、いろいろ動作形式が変更されるようになっているもの。

フラグ flag

演算結果の正負などの状態を表示するもの。

フローティング floating

複数の装置の出力をバスなどの共通の線につなぐ場合、出力のぶつかり合いが生じないように、ひとつの装置のほかは出力インピーダンスの高い、いわば切りはなされた状態にする必要がある。これをフローティング状態という。

プログラムカウンタ program counter

次に実行すべき命令のアドレスが入っているレジスタ。命令を実行すると自動的に次の命令のアドレスに進められる。

〔ま 行〕

マシンサイクル machine cycle

メモリから命令やデータを読み出してレジスタに入れるなど、プロセッサの基本的1動作の時間。1つの命令の実行には1~5程度のマシンサイクルが必要。

命令 instruction

コンピュータにひとつの動作を行わせるための指令情報。動作の種類を表わすオペレーション部と、使われるデータの所在などを表わすオペランド部(アドレス部)からなるのが普通である。

〔ら 行〕

リフレッシュ回路 refresh circuit

ダイナミックメモリは短時間しかメモリ内容を保持できないから、一定時間以内に必ず再生してやる必要がある。この再生を行う回路がリフレッシュ回路である。

レジスタ register

データを一時蓄えておくところ。フリップフロップをならべて構成される。