Ⅲ 入出力機能の活用技法

学習目標
入出力機能の活用技法に関して、マイコン応用システムにおける入出力機能をデータの流れに従って理解させる。また、割込みの必要性と働きの概要を理解させるとともに、ダイレクトメモリアクセスを行うハードウェア構成とその動作についても理解させる。

1 入出力機能と入出力イベントとの関係

(1) 入出力機能とは

コンピュータのシステムがプログラムの指示に従って行った処理の結果を、何らかの方法でコンピュータシステムから外部に示す必要がある。また、処理に必要なデータを外部から取り込む必要もある。このような働きを実現するための機能を入出力機能という。また、このような機能を持ったものとして、パーソナルコンピュータのディスプレイ、キーボード、プリンタなどがある。

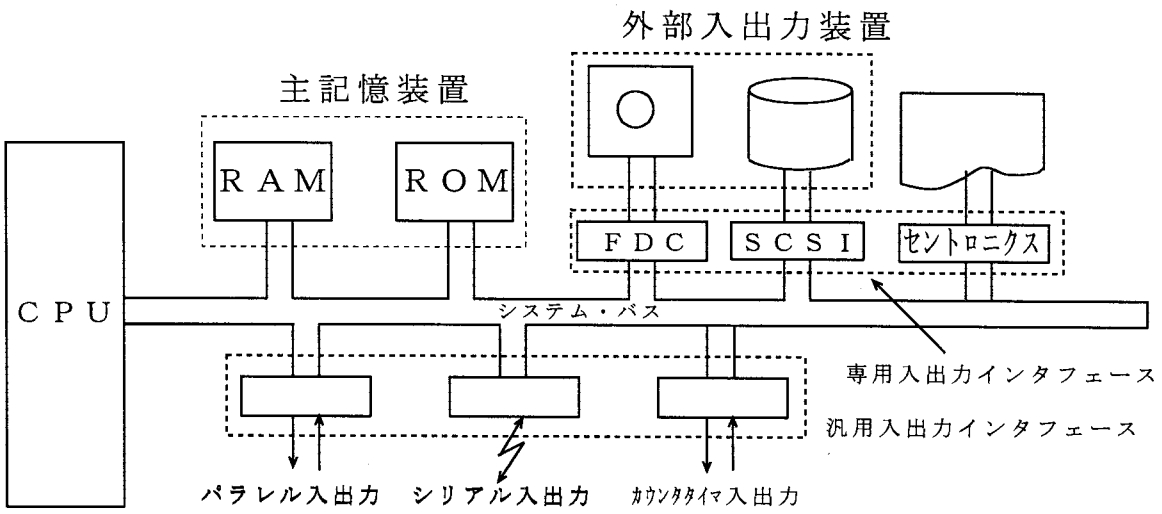
コンピュータシステムとディスプレイやキーボードなどの外部装置の間には、処理速度の違いやデータのやり取りの方法などに違いがある。これらの間でデータの受渡しを行うためには、その違いを吸収するための調整装置として、入出力インタフェース（Ｉ／Ｏインタフェース）が必要である（図Ⅲ－１参照）。

主な入出力装置については、標準的なデータのやり取り方法が定まっている。代表的なものにプリンタとの間のセントロニクスインタフェース、高速磁気ディスクとの間のＳＣＳＩインタフェース、モデムとの間のＲＳ２３２Ｃインタフェースなどがある。

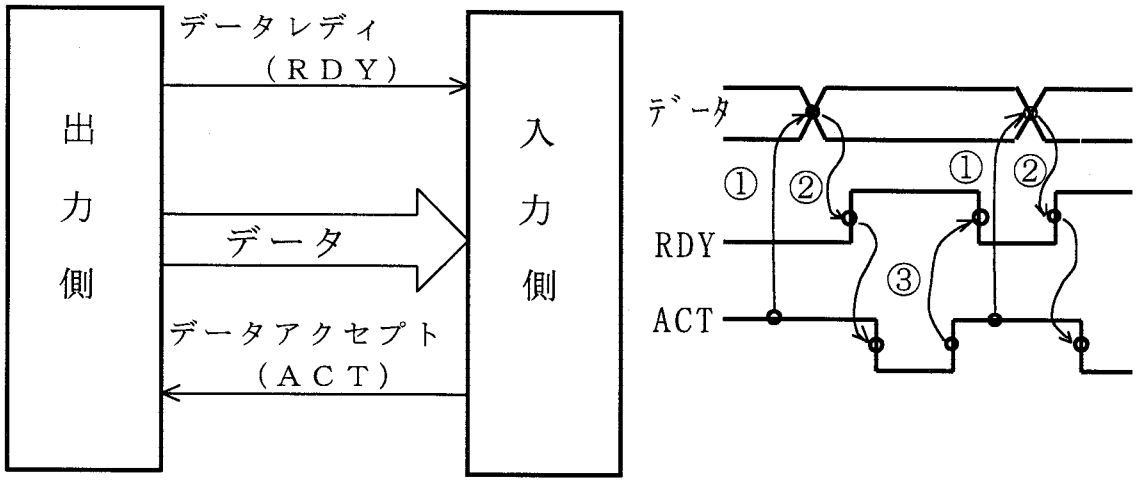
この入出力インタフェースには、特定の入出力装置とのやり取り専用に作られたものと、多用途に用いられる汎用のものがある。

(2) ハンドシェークによるデータの扱い

並列データ伝送を行うとき、互いにデータの送受信を確認し合いながら行うことをハンドシェークという。図Ⅲ－２のようにデータ送信を知らせるデータレディフラグ（ＲＤＹ）と、データ受信を知らせるデータアクティブフラグ（ＡＣＴ）とを準備する。手順は図Ⅲ－２のようになる。データを送る側は、ＡＣＴ信号が”Ｈ”になるのをみて、データを出力する（①）。その後、ＲＤＹ信号を”Ｈ”にしてデータ送信を知らせると、データ受信中ＡＣＴ信号が”Ｌ”になる（②）。データ受信が終了すると、ＡＣＴ信号が”Ｈ”になるので、ＲＤＹを”Ｌ”にして次のデータ送信に入る（③）。このような方法は、出力側と入力側の転送速度が大きく違っていても、確実にデータ転送ができる。



図Ⅲ－１　マイコン応用システムにおける入出力機能



図Ⅲ－２　ハンドシェーク

(3) 時間的制約のあるイベントの扱い

一般に、ハンドシェークによる入出力機器は、マイコンが他の処理を実行していて入出力機器に対する命令が多少遅れても待っていてくれる。しかし、ある時間内に処理しなければならないものや、ある時間内に消滅してしまうような入出力信号もある。このような時間的制約のある入出力機器の信号の発生（イベントという。）に対しては、割込みという方法がある。

2 割込み機能

(1) 割込み機能の必要性

人間の日常生活でも、何かしているときに急に別の何かをしなければならず、そちらの処理をしてから元に戻ることがある。例えば、本を読んでいるときに電話がかかると、本を読むのを中断して、電話を受けその用件を済ませてから再び本を読み始める。これも我々の日常業務の中の割込みの一種である。しかし、これだけのことでも結構多くのことを人間は処理している。

① 本を読むという仕事をしながら、電話が鳴ったということを知ることができる。
② 電話に出る前に、今まで読んでいた本を再び読み始められるように何らかの印を残す。
③ 電話に出て相手の要望に応じた適切な処理を行う。
④ そして、再び何事もなかったように同じページを開き読み始める。

これと同じことをコンピュータシステムにおいて行えば、割込み処理が実現する。

コンピュータシステムにおける割込み（Ｉｎｔｅｒｒｕｐｔ）とは、”何らかの要求によって現在の処理を中断し、特別な処理を行うこと”である。この何らかの要求とは、高速な対応が必要なシステムの資源から発生するもので、例えば、通信回線に接続されたシリアルポートからのデータ入力であるとか、システム作動中の電源異常であるとか、通常の処理より時間的に緊急を要す処理要求である。

従来は、ＣＰＵに通常はある処理をさせながら、条件がそろったときに別の処理をさせることで、高価なＣＰＵを遊ばせずに使うことが割込み処理の目的であったが、最近ではむしろユーザインタフェースの向上や、非同期に入力されるデータの取りこぼしを防ぐために使われる例が多い。

例えば、電源の異常やシステムの異常などの緊急な事態に対処する場合や、低速な入出力機器とのデータ転送を待ち時間なしに効率よく行う場合、またいつ発生するかわからない事柄を、別の処理をしながら待つときなどに使う。

ＣＰＵによって何かを制御しようとするとき、必ず外部の信号によって処理先の変更などを行う必要が生じる。この外部の信号によって処理の流れを変える方法には二通りある。一つは、プログラムによって常時信号を監視する方法であり、ポーリングと呼ばれている。もう一つは割込み処理による方法である。これらの処理方法の違いをプリンタを例に図Ⅲ－５にフローチャートで示す。割込み処理機能を使うことにより、プリンタ用バッファに空きがある限り、プリンタのＢＵＳＹ待ちをせずにＣＰＵは処理を進められる。

(2) 割込みの処理動作

コンピュータは、通常外部の時間の経過と無関係に動作し、与えられた命令を逐次実行する。しかし、コンピュータが外部の入出力機器とデータのやり取りをしたり、機械など

を制御したりする場合には、外部の時間に合わせて動作しなければならない。

例えば、コンピュータが外部からデータを受け取る場合、ＣＰＵはＩ／Ｏ用の周辺ＬＳＩにデータが到着するまで待ち、到着したら即座にそれを読み出す。このとき周辺ＬＳＩからＣＰＵに対して、データの到着を知らせる方法があればいちいちじっと待っていなくてもすむ。このために用いられるのが割込みである。

ＣＰＵは、周辺ＬＳＩから割込み要求信号が入力されると特定のサブルーチン（割込みサービスルーチン）を起動する。周辺ＬＳＩからデータを読み出す割込みサービスルーチンを作っておけば、いつデータが到着しても受け取ることができる。

マイコンにおける割込みとは、あるプログラムを実行中にそのプログラムの中にはジャンプ命令もコール命令もないのに、ある条件が成立したとき突然別のアドレスに実行を移すことである。そして、その条件に対応する処理をしてから、再び途中まで実行していたプログラムに戻る。

(3) 割込み要求の種類

一般に、割込みにはソフトウェア割込みとハードウェア割込みの２種類がある。ソフトウェア割込みというのは、ＣＰＵの命令によって割込みが発生するもので、ＯＳやモニタプログラムのようにハードウェアに密着した処理を行うソフトウェアの中で使用される。ＭＳ－ＤＯＳのＩＮＴ２１Ｈを使ったシステムコールなどがその例である。

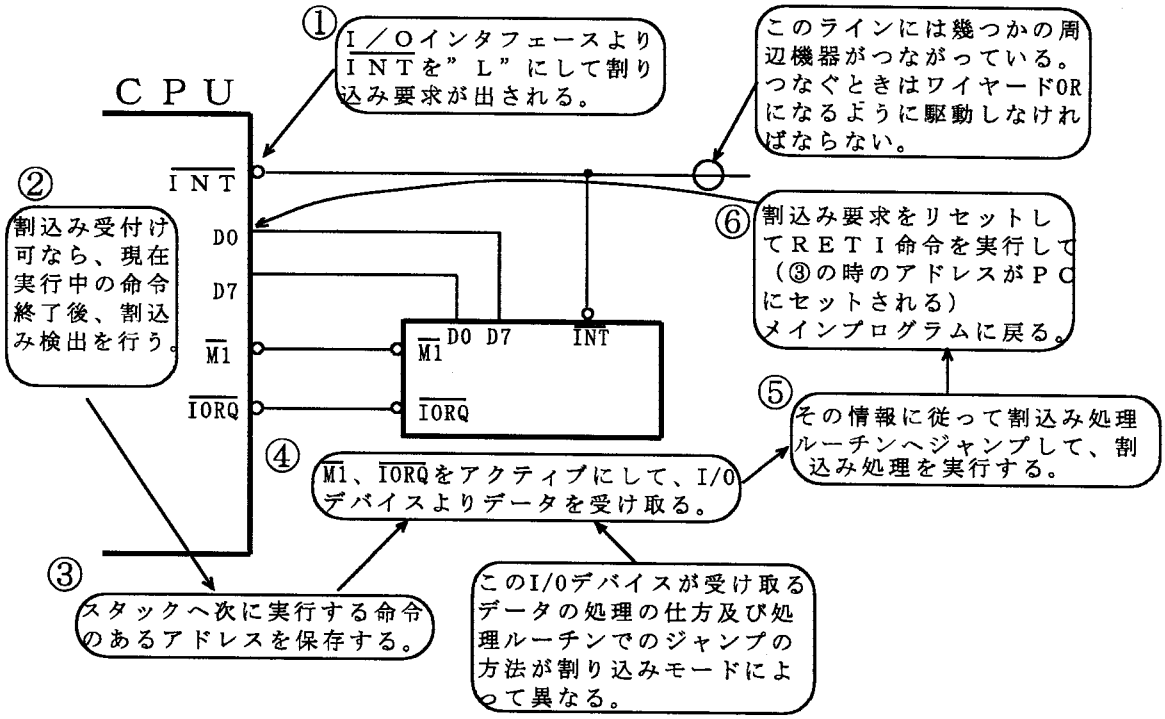
これに対してハードウェア割込みとは、ＣＰＵの割込み入力端子にハードウェアによって信号を入力し、割込み処理に移るものである。一般に、ソフトウェアによってマスク（割込みを受け付けないようにすること。）が可能なものと、マスクできないものがある。また、それぞれの割込みがタイミング的に同時に発生することが予想されるため、各割込み要求には優先順位が付けられているのが普通である。

ＣＰＵには、この割込み処理を具体的な実現のために、プログラムの実行中であっても、外部からの処理の要求を検知するための入力信号端子を持っている。この端子が前に述べた電話のベルが鳴っていることを検出する機能に相当する。電源遮断のように、いかなる処理にも優先する緊急な処理のために利用されるＮＭＩ（Non Maskable Interrupt）と呼ばれる割込みと、プログラムによって割込み処理の受付を禁止することのできるＩＮＴ（Maskable Interrupt）と呼ばれる割込みの二つの外部割込み端子を持つのが一般的になっている。

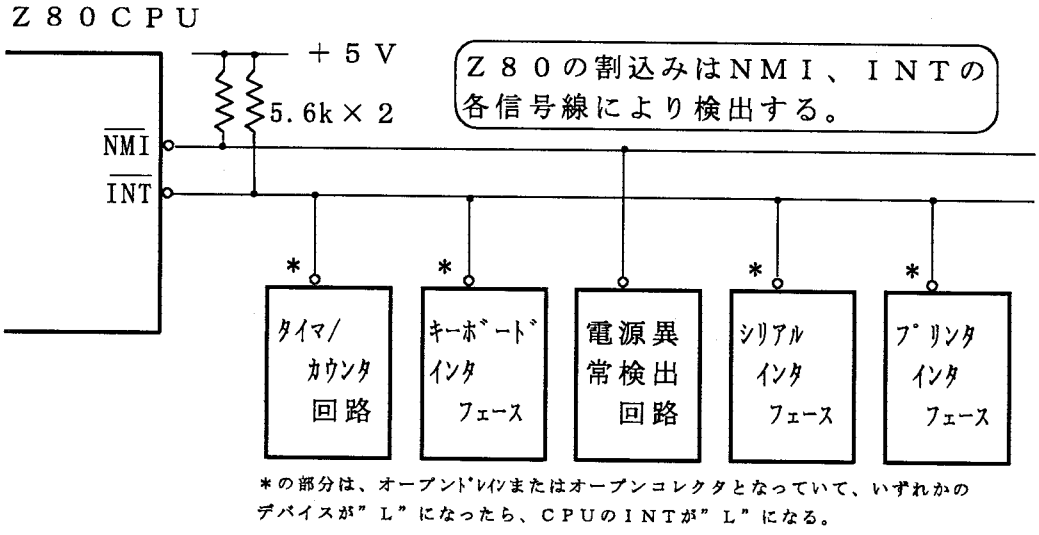
ＮＭＩは、プログラムでマスク（割込みの受付を禁止すること。）ができないため、このような名前になっている。ＮＭＩ割込みは、ＩＮＴ割込みより優先順位が高く、ＩＮＴ割込み処理中でも受け付けられるので、一般には電源の異常に対応する処理などのような最も優先順位の高い処理として使用される。ただし、ＢＵＳＲＥＱ端子がアクティブになっている場合には、ＮＭＩは受け付けられない。

ＩＮＴはプログラムで割込み処理の受付を禁止したり、許可したりすることが可能である。ＩＮＴ割込みは当然のことながらこれより優先順位の高いＮＭＩ割込みの処理中や、ＢＵＳＲＥＱ端子がアクティブになっている場合には受け付けられない。

マイコンシステムでは、一般的に図Ⅳ－４に示すような流れで割込み処理が行われる。また、複数の入出力装置から割込み要求が発生する場合、割込み要求のための信号線の接続方法を図Ⅴ－１０に示す。



図Ⅲ－３　割込み処理の手順



図Ⅲ－４　割込みを要求する入出力装置の接続方法

3 割込み機能を活用した入出力

(1) 割込み処理プログラムの基本

割込み処理は、それまで実行中のプログラムの流れを突然変え、用が済むと再び何事もなかったように戻らなければならない。このため、ＣＰＵのレジスタやメモリを不用意に破壊しないように考慮してプログラムを組まなければならない。

通常のプログラムを”メインルーチン”と呼ぶのに対し、割込み要求に応じてその処理を行うプログラムを”割込み処理ルーチン”と呼ぶ。

基本的には、割込みが受け付けられた際に、システムとしては割込み処理ルーチンが終了した後にメインルーチンに戻るためのアドレス（割込みを受け付けた直後のアドレス）を保存しておかなければならない。そのため、通常はスタックエリアを使う。割込み処理中のスタックには、この他にメインルーチンで使用していた汎用レジスタやフラグレジスタの内容も保存される。

このように割込み処理ルーチンの先頭でアドレスやすべてのレジスタをスタックに退避し、処理が終わったら退避していたレジスタを戻して実行中のプログラムに戻る。

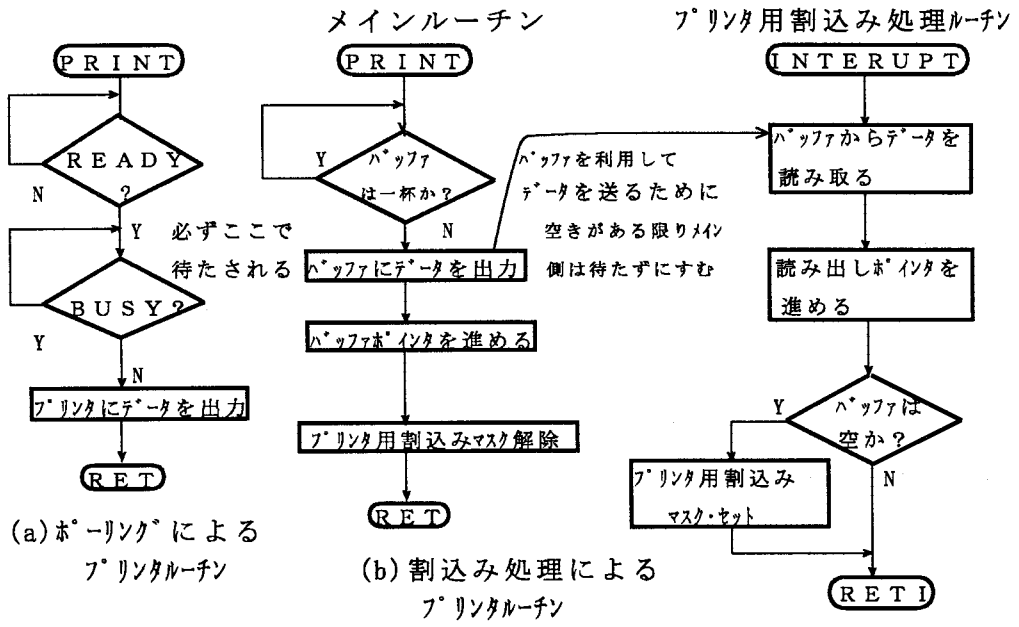
例えば、図Ⅲ－６に示すように、プログラムを実行中に割込みがかかったとする。割込み処理ルーチンに行く前にあるレジスタの値をメモリスタックに保存せずに、割込み処理ルーチンでそのレジスタに別な値が代入された場合、これでは割込みがかかる前にそのレジスタに入っていた値が、割込みから戻ってきたときには全く別の値に変わっていて、その後のプログラムの実行がおかしくなる。

マイコンを接続する周辺ＬＳＩの中には、あるコマンドを書き込むと一定時間以内に次のデータを書き込まないと正常な動作を保証できないというものなどがある。このようなデバイスのコントロールプログラムを組むとき、プログラム的にはコマンドを書き込んで、次のデータを一定時間以内に書き込んだとしても、その間に割込みが発生した場合、実際の実行時間はその一定時間を超えてしまうかもしれない。

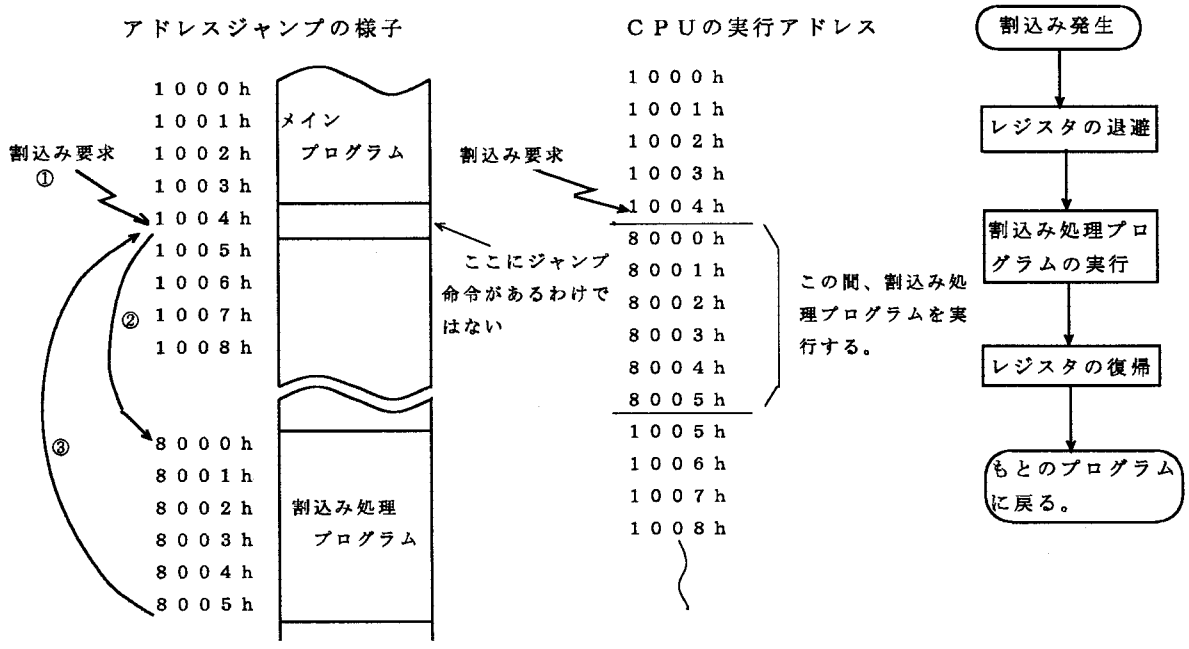
また、割込みを発生する周辺ＬＳＩの初期化プログラムも、割込み受付禁止状態で実行しないと、意味のない割込みが発生することがある。

このような場合はその処理に入る前に割込み受付を禁止し、できるだけ短い時間で処理を済ませて割込み受付を許可する。

この”できるだけ割込み受付禁止状態を短くする”というのがポイントで、不必要に長い時間割込み受付を禁止したままの状態では、割込みが発生しても処理できなくなる。



図Ⅲ－５　ポーリング処理と割込み処理の流れの違い



図Ⅲ－６　割込み処理プログラムの実行

(2) 割込み処理の具体的な方法

割込み処理の具体的な手順は次のようになる（Ｚ８０ＣＰＵを使用する場合)。

① 使用する周辺素子、割込み処理の内容によって、使用する割込み処理のモード（モードＯ～２）の検討を行う。

② 使用するモードに応じて、割込みコントローラ及び周辺素子の選択を行う。

③ 使用するモードが決まったら、次の事項について検討する。

- モードＯでは、０～３８ＨまでのそれぞれのＲＳＴ命令のジャンプ先
- モード１では、３８Ｈ一つが割込み処理ルーチンの入り口となる。
- モード２では、ＣＰＵのＩレジスタと周辺装置が割込み要求時に出力する割込みベクタから合成されるアドレスが、割込み処理ルーチンの入り口を示すアドレスの入っているテーブルを示す。

④ 具体的な割込み処理ルーチンのコーディングを行う。このとき、割込み処理ルーチンの中で使用するレジスタは、スタックへ保存する。

(3) 入出力を割込み処理とすることのメリット

入出力を割込み処理とした場合、以下に示すメリットがある。

① 入力要求を取りこぼさない。

ソフトウェア的に割込みをマスクできないＮＭＩを使うと、ＣＰＵがどんな状況にあっても、その最優先割込み処理へプログラムの実行が移る。

② いつ起きるかわからない入出力要求を待ち続けずに済む。

入出力デバイスからの入出力要求に割込み処理を使用すると、ＣＰＵは常に入出力を待ち続けることなく、入出力要求があった場合のみ、割込み処理プログラムに従って必要な処理が実行される。

③ 一定周期ごとの入出力が行いやすい。

コンピュータシステムが持っている内部時計を使って、一定周期で割込みがかかるようにすると、容易に一定周期の入出力処理が行える。

このように、コンピュータシステムの入出力処理において、割込み処理は応答性及び処理効率の点で重要な機能であるといえる。

4 ダイレクトメモリアクセス機能

(1) ダイレクトメモリアクセスの必要性

マイコンによって何らかの処理を行うとき、Ｉ／Ｏ装置とメモリ間やメモリーメモリ間など様々なところでデータ転送が必要になる。ソフト的にこれらの処理を行う場合、メモリーメモリ転送では、データポインタ及びループカウンタを用意してＬＯＡＤ／ＳＴＯＲＥを繰り返すことになる。また、Ｉ／Ｏの場合はポーリングによるか割込み処理によって行うことになる。

Ｉ／Ｏの場合、ポーリングによる方法は処理としては最も簡単な方法であるが、データ転送中にほかの処理が一切できないことや転送するためのプログラム実行時間の関係で、ある程度以上に転送レートを上げることができない（ほぼ１０～２０μｓ（ｍａｘ）の周

期くらい)。

割込み処理による場合は、データ転送中に他のことを行うことができる点で有利であるが、割込みのために余分な処理が必要であるので、転送レートはポーリングによる場合とほぼ同程度かやや劣る。

この問題を解決するために、ＣＰＵを介さずハードウェアによってデータの転送を行う方法が考えられており、これをＤＭＡ（Ｄｉｒｅｃｔ　Ｍｅｍｏｒｙ　Ａｃｃｅｓｓ）と呼ぶ。この方法を使うことで、データの発生速度がＣＰＵの処理能力を上回る場合にも対応できるほか、ＣＰＵが転送以外の処理を並行して実行することも可能となる。

(2) ダイレクトメモリアクセスの動作

ＤＭＡでは、ＣＰＵは外部バスの入出力を停止して、代わりに専用のＤＭＡコントローラがアドレスバスやコントロールバスの信号を出力する。このように、データ、アドレスの各バスをＣＰＵとＤＭＡ転送用ハードウェアが共用して使うために、その調停が必要になる。

ＤＭＡ転送を行うには、ＤＭＡコントローラがＣＰＵに対してバス要求信号を出す。ＣＰＵは、それを受付けると命令の実行を中断し、バス受付信号を出して外部バスを開放する。図Ⅲ－７に専用ＤＭＡコントローラを使った場合の基本構成を示す。

実際の転送処理を大別すると、図Ⅲ－８のように１回の転送に限ってバスを開放し、転送ごとにＣＰＵにバスを返す方法（シングル転送）と、ＤＭＡ制御回路が優先となりＣＰＵを止めて連続転送（ブロック転送）を行う方法がある。

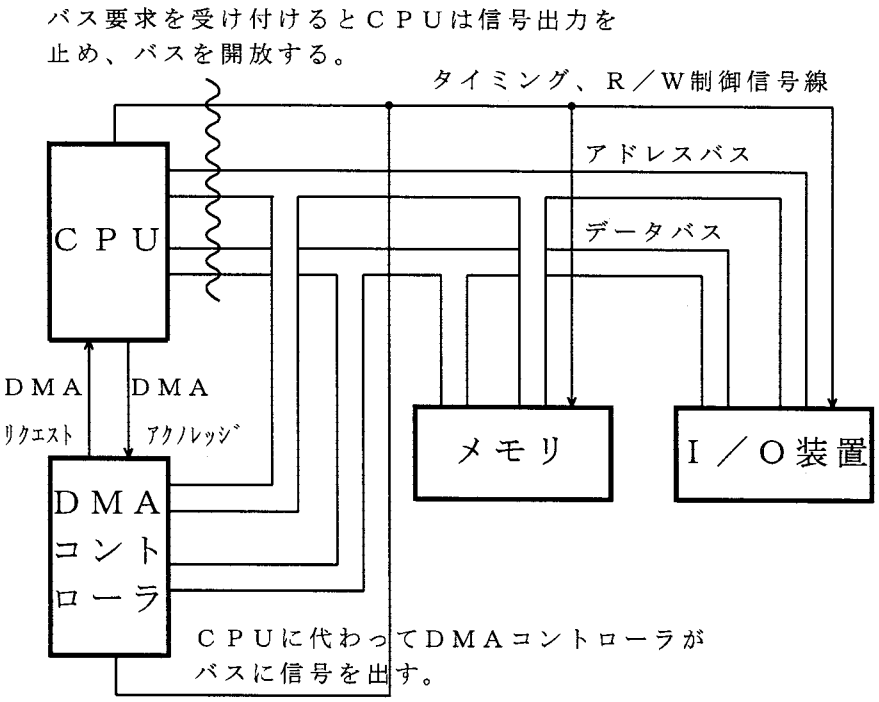
シングル転送では二つの仕事を並列的に実行できるが、ブロック転送よりデータを転送する速度が落ちる。一方、ブロック転送ではＤＭＡが動作している間は、ＣＰＵは全く動作できない。

5 ダイレクトメモリアクセス機能を活用した入出力

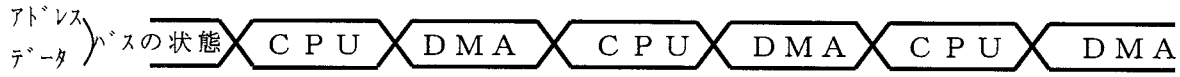
(1) ダイレクトメモリアクセス機能を活用した入出力の応用例

マイコンの誕生当初、メモリへプログラムを書き込む際に、メモリアドレスやバイナリデータをスイッチで設定してマニュアル操作でメモリに書き込んだり、またマニュアル操作で読み出したデータをＬＥＤ表示させるためにダイレクトメモリアクセスを用いていた。

実際の利用例としては、ディスク装置などのデータ転送やＣＲＴ表示装置などが一般的である。ＤＭＡ転送処理は、ミニコンや大型計算機で考えられてきた方法である。しかし、最近ではＣＰＵのファミリーＬＳＩの中に専用のＤＭＡコントローラが用意されており、使いやすくなっていることもあり、マイコンシステムでもよく利用されている。

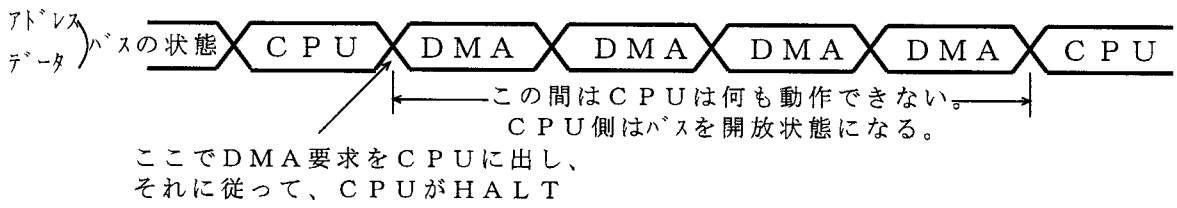


図Ⅲ－７　専用ＤＭＡコントローラを使ったときの基本構成



ＣＰＵとＤＭＡが交互に動作する。

(1) シングル転送による方法



(2) ブロック転送による方法

図Ⅲ－８　ＤＭＡ転送のバスタイミング

演習問題

問１　マイコンのインタフェースに関して、次のａ～ｅの記述に最も関連の深いと思われる項目を解答群の中から選べ。

ａ　プリンタ　　ｂ　フロッピィディスク　　ｃ　モデム　　ｄ　キーボード

ｅ　計測器　　ｆ　ハードディスク

解答群

ア　セントロニクス　　イ　ＧＰ－ＩＢ　　ウ　スイッチマトリクス

エ　ＳＣＳＩ　　オ　ＲＳ－２３２Ｃ　　カ　ＤＭＡコントローラ

問２　割込み処理の手順に関して、次のフローチャートの空欄に入れるべき適当な字句を解答群の中から選べ。

解答群

ア　割込みコントローラの初期化

イ　優先度の低い割込みマスクの解除

ウ　優先度の高い割込みをマスクする。

エ　レジスタを復帰する。

オ　割込みベクタを書き換える。

カ　優先度の高い割込みマスクの解除

キ　優先度の低い割込みをマスクする。

ク　レジスタを退避する。

ケ　レジスタをクリアする。