マイクロコンピュータマニュアル No.456

SHARP

Z-80 DMA テクニカル マニュアル

Z-80 DMA/Z-80A DMA

SHARP

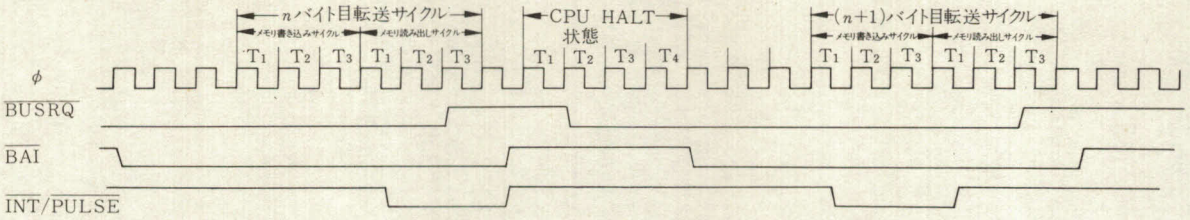
Z-80DMA 使用上の注意事項

1．プログラム作成時

ブロック転送後の割り込み、またはバイト一致での割り込みが発生した後、ブロック長（WR0）または一致バイト（WR3）を変更する場合、変更に先だって必ず「割り込みディセーブル・コマンド(WR6、AF_H)」を書き込まねばならない。

2．パルス発生機能使用時

パルス発生機能をバイト・モードで使用時、パルスは2回に分かれて発生します（$\overline{BAI}$が非アクティブになる期間があるため)。また、オフセット値とブロック長の下位8ビットが等しいとき、一度DMA動作を完了した後オフセット値を変えずに再びDMA動作を行わせると、1バイト目の読み出しサイクル中にパルスが発生する。



（注） 本タイミング図では、CPUはHALT状態にある。

バイトモード時のパルス発生タイミング

Z-80 DMA テクニカル マニュアル

SHARP

目　　次

第1章　序　　論

1.1　DMAの有用性

Z-80 DMAデバイスについて詳述するまえに、ダイレクト・メモリ・アクセス（DMA）およびダイレクト・メモリ・アクセス・コントローラ（DMAC）全般について述べる。これによって、本マニュアルを通じて使われている8ビット・シングル・バス・マイクロコンピュータに関連した、機能および用語背景を明らかにする。

本マニュアルではDMAとDMACを特に区別せず、単にDMAとのみ記述することにする。

DMAは、CPUとは独立にデータの高速ブロック転送の制御を行うデバイスである。これは、一般的にはメモリと入出力デバイス間の転送、あるいはその逆の転送を意味するが、DMAの中には従来のCPUによって行われていたような他のタイプの転送ができるものもある。たとえば、Z-80 DMAはメモリーメモリ、入出力ー入出力間転送のみならず、転送と同時に、あるいは転送に関係なく1つのバイト中の特定のビット・パターンをサーチすることができる。

CPU転送：　DMAを含まないシステムでは、データ転送をCPUを介して行うため、ソフトウェアでこれを監視しなければならない。これは、転送ブロック中のデータの各バイトの入力、出力、そしてトラッキングのための命令シーケンスの実行を含むのが通常である。

図1.1は、1つのデータ・ブロックを一度に1バイトずつ転送するために必要なメモリからのフェッチ、そして従来のCPUによって実行すべき最少限の命令シーケンスを示したものである。実際にはほとんどのCPUの場合ここに示されたものよりさらに多くの命令を必要とする。

この方式の場合、CPUによる転送は比較的低速で長時間にわたってCPUが拘束されることになる。さらに、レスポンス時間（最初のバイトの立ち上がり時間）も一般に遅くなる。これは、入出力デバイスはそのレディ信号を割り込みとして使用するのが通常で、この割り込みサービス・ルーチンが最初のバイトの転送にかなり

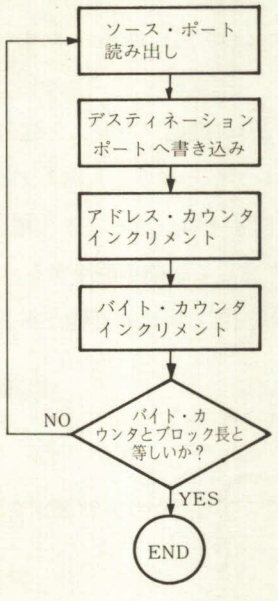


図1.1　代表的なCPU入力シーケンスの例

のタイムラグを引き起こすからである。

Z-80とZ-8000 CPUに共通した特徴は、ブロック転送とストリング・サーチ命令を有し、単一命令で最高64Kバイトまでのデータを処理できることである。単一ブロック転送命令は、バイトの全シーケンスについて図1.1に示したすべての機能を繰り返し実行し、その達成可能な転送率は他のCPUに比較するとはるかに高い。4 MHzのZ-80A CPUでは毎秒200Kバイトを転送でき、4 MHzのZ-8000 CPUでは毎秒800Kバイトを転送することが可能である。これは一部のDMAよりも高速である。したがって、Z-80およびZ-8000ファミリにおいて、CPUのブロック転送上の問題は一般的には転送のスピードではなく、立ち上がりにおけるレスポンス時間である。ブロック転送命令の実行には、通常次のうちのいずれかの方法が使われる。

- 入出力デバイスがCPUに割り込みをかける。そして、CPUの割り込みサービス・ルーチンの中でブロック転送命令が実行される。これには、割り込み処理能力が最も早い機種に含まれる4 MHzのZ-80AおよびZ-8000ファミリでさえ少なくとも5～10μsのレスポンス時間を要する。
- 入出力デバイスがレディとなるまえに、CPUがデバイスのサービス・ルーチンの実行を開始する。そして、フラグ・ビットがCPUによってつねにポーリングされる。ビットによってデバイスのレディ状態が表示されると、CPUはブロック転送命令にジャンプする。この方法では、場合によってはレスポンス時間が5μs以下になるが、CPUを完全に占有することになる。
- 入出力デバイスが事実上レディになるまえに、CPUは割り込みサービス・ルーチンの中でブロック転送命令の実行を開始する。ただし、$\overline{RD}$および$\overline{CE}$がアクティブになった直後、入出力デバイスは$\overline{WAIT}$によってCPUをアイドル状態にする。入出力デバイスがレディになるとともに$\overline{WAIT}$は解除され、転送は完了する。この場合、レスポンス時間は最良（4 MHzのZ-80AまたはZ-8000 CPUで250 ns）となるが、バスを完全に拘束する。

要約すると、ほとんどのCPUにおいて転送時間もレスポンス時間も遅すぎるといえる。Z-80およびZ-8000 CPUの場合、転送時間は相当早くすることができるが、割り込みによる転送状況によってはレスポンス時間が長すぎるケースが出てくる。

DMA転送： DMAは、CPUを介することなく、CPUの場合に必要な命令のフェッチもなく、データのソースおよびデスティネーション間の直接転送を行うものである。すなわち図1.1に示したすべてのステップをハードウェアが実行するわけである。

たとえば、メモリー入出力間転送の場合、転送に先立ってメモリの開始アドレスと転送されるブロック長がCPUによってDMAに書き込まれる。CPUによってDMAがレディとなり、入出力デバイスのRDYがアクティブになるとDMAは直ちに制御を開始し、データの転送を開始する。ほとんどの場合、CPUはこの間アイドル状態となる。転送の完了とともにDMAはCPUに信号を発して制御を解除する。

以上のことから、次のうちいずれか1つ、またはそれ以上の状況あるいは必要性があるときDMAが使用される。

- CPUに対する入出力が多すぎて他のタスクが正常に実行できない。
- CPUの能力以上の転送速度が要求される。
- CPUが楽に処理できるよりも高速の転送レスポンス時間（立ち上がり）が要求される。

小型で低性能のシステムは、一般にDMAなしで使用される。中性能システムでも、CPUが充分な速度で転送を処理し、かつ他の仕事もできる場合はDMAなしの設計が可能である。

転送およびレスポンスに速度が要求される場合、この性能向上のための有力な候補となるのがDMAである。DMAはCPUよりも早く転送処理をするばかりでなく（Z-80およびZ-8000 CPUは例外となることがある)、元来そのレスポンスはより早いものであり、CPUのレスポンスについて先述したと同様の技術を使うことによって、これをさらに早くすることが可能となる。

次に、DMAが最適なケースをいくつか示す。

- ディスクおよびディスケットのコントローラ。
- CRT入出力のようなスキャン操作。
- データ・アクイジション。
- メモリーメモリ間転送。
- メモリ・サーチ。
- 一時記憶（入出力ー入出力間転送)。
- IEEE 488のような並列バス・システム。
- オプティカル・ファイバとの接続。
- ネットワーキング、マルチプロセシング、マルチプログラミングにおけるブロック転送。

速さという長所に対して、DMA動作中はCPUは通常アイドル状態となってシステム・バスを完全あるいは部分的にも制御できないという短所がある。このことは、システムのトータル処理能力に影響するばかりでなく、メモリ・リフレッシュ、その他の割り込みにも影響を及ぼす。

このような問題については、第7章 性能上の限界で詳しく述べることとする。Z-80ファミリ間の速度の比較は、次章 Z-80 DMAの機能説明 で述べる。

1.2 DMAの特性

ほとんどのDMAはプログラムしなければならない。なぜならば、最低の条件として、データ転送に先立ちブロック長（バイト・カウント）と開始メモリ・アドレスをCPUからDMAに書き込む必要があるからである。転送の進行に従って開始アドレスはインクリメントあるいはデクリメントされ、バイト・カウンタは0から特定のブロック長までインクリメントされる。

しかし、これ以後はその特性および能力の面でDMAごとに大きく異なってくる。次に記すのはDMAについての概況であり、特に必要な場合のみしかZ-80 DMAについては言及しないこととする。

ポートおよびチャネル： すべてのデータ転送にはソース・ポートとデスティネーション・ポートがある。たとえば、メモリー入出力間転送では、メモリがソース・ポートであり、入出力がデスティネーション・ポートである。この2つのポート間のデータ交換を制御し取り扱う手段がチャネルと呼ばれるものである。1つのチャネルにはアドレスおよびバイト・カウントのためのハードウェア、バス制御および全転送プロセスのコーディネーションが含まれる。

チャネル内の各ポートは、DMAのアドレス発生機構あるいはハードウェアによって指定される特定のロケーションを有する。他の8ビットDMAと異なり、Z-80 DMAは各バイトの転送中にメモリおよび入出力

ポートのアドレスを発生する。他のDMAでは、入出力ポートはハードウェアによって行っている。

DMAの中にはマルチプルチャネルを有するものもある。これは、一般にマルチプル転送の取り扱いが可能であること、そして1つのDMAから複数の入出力デバイスにデータを転送できることを意味する。しかし、DMAは一度に1つの読み出しまたは書き込みサイクルしか実行できないから、マルチプルチャネルといえども、一定の速さのもとではシングルチャネルよりもスループットが高いことにはならない。Z-80はシングルチャネル・デバイスであり、最大転送率が最も速いこと、そしてアドレス・バス上に真の入出力ポート・アドレスを乗せるという特長を有している。さらに、Z-80が2つのアドレスを発生できるという能力は、シングルチャネル内でメモリーメモリ間転送ができることを意味している。他の機種ではこれが全くできないか、もしくは2チャネルを必要とする。

バイト・サーチの機能もまたZ-80 DMAのシングルチャネルのもう1つの特長であり、他の8ビットDMAでは不可能なものである。転送中にバイトをDMAの内部レジスタにロードし、データが一度ロードされるごとにデータをマスクし、御制バイトと比較できるのはZ-80 DMAのみである。有効な比較がこれによって行われ、Z-80 DMAに種々の動作を実行させる。

転送方法： 入出力処理の場合、従来のCPUの命令によるものと直接メモリ・アクセスによるものの相違について先にふれた。図1.2は、従来のCPUの命令だけでなく、DMA転送の2種類の方法とともにZ-80およびZ-8000 CPUのブロック転送命令をも比較することによって、このテーマを拡大したものである。この図には、1バイトのデータの転送に要する読み出しおよび書き込みサイクルを示す。

図1.2aは従来のCPUの入出力命令による動きを示したものである。読み出しおよび書き込みサイクルの数は大体の数であり、多くのCPUの場合これより多くのサイクルが必要である。少なくともCPUの命令によって図1.1に示されたすべてのステップが実行されるとともに、次のバイトの転送がレディかどうかをテストするといったハウスキーピング的作業も実行される。

図1.2bはZ-80およびZ-8000 CPUのブロック転送命令を示したものである。これも概略であって、初期化後は特にZ-80 CPUの場合、1読み出しサイクルおよび1書き込みサイクルより多くの命令を必要とする。しかし、1つのブロック転送命令で最大64Kバイトまでのデータを転送することが可能である。

図1.2cは、シーケンシャルまたはフロースルーDMA転送を示したものであり、ここではバイトはソース・ポートからDMAに読み込まれ、つづいてデスティネーション・ポートに書き込まれる。これはZ-80 DMAの場合、Z-80 CPUに外部論理回路を付加せずに行える方法である（そのタイミング特性およびCPUへのインターフェイスはこの場合極めて均一である）。シーケンシャル転送の場合の速度は、シリアルの通信方式の大部分、そして他の入出力またはメモリ・デバイスの多くの能力と同等あるいはそれを超えるものとなる。

図1.2dは、同時またはフライバイDMA転送を示したものであり、バイトは同じマシン・サイクル内で読み出しと書き込みが行われる。読み出しおよび書き込みの制御線は同時にアクティブとなり、ソースおよびデスティネーションは信号によって決定される。この信号はメモリ読み出し入出力書き込み、または入出力読み出しメモリ書き込みを指定する信号である。これは転送方法の中で最も早いものであるが、外部論理回路が要求されるために、メモリおよび入出力へのタイミングのインターフェイスが多分に複雑なものとなる。

いくつかのDMAで使われるもう1つの方法は、トランスペアレントまたはサイクル・スチール転送と呼ばれるものである。これは図1.2cおよびdと同様の働きをするものであるが、バスを制御するかわりに、通

常であればダイナミック・メモリがリフレッシュされるCPUサイクルと、DMAデータ転送をインターリーブさせる。この方法もまた外部論理回路を必要とし、DMAの発生中はメモリ・リフレッシュを禁止する。さらに、この方法ではDMAのスループットが減じるケースもある。

DMA転送はすべてCPUによるダイナミック・メモリ・リフレッシュに対して割り込みをし、そのほとんどの場合、CPUをアイドル状態にする。そのため、より速いDMA転送速度を目的とする際は、こういった点を考慮することが重要である。

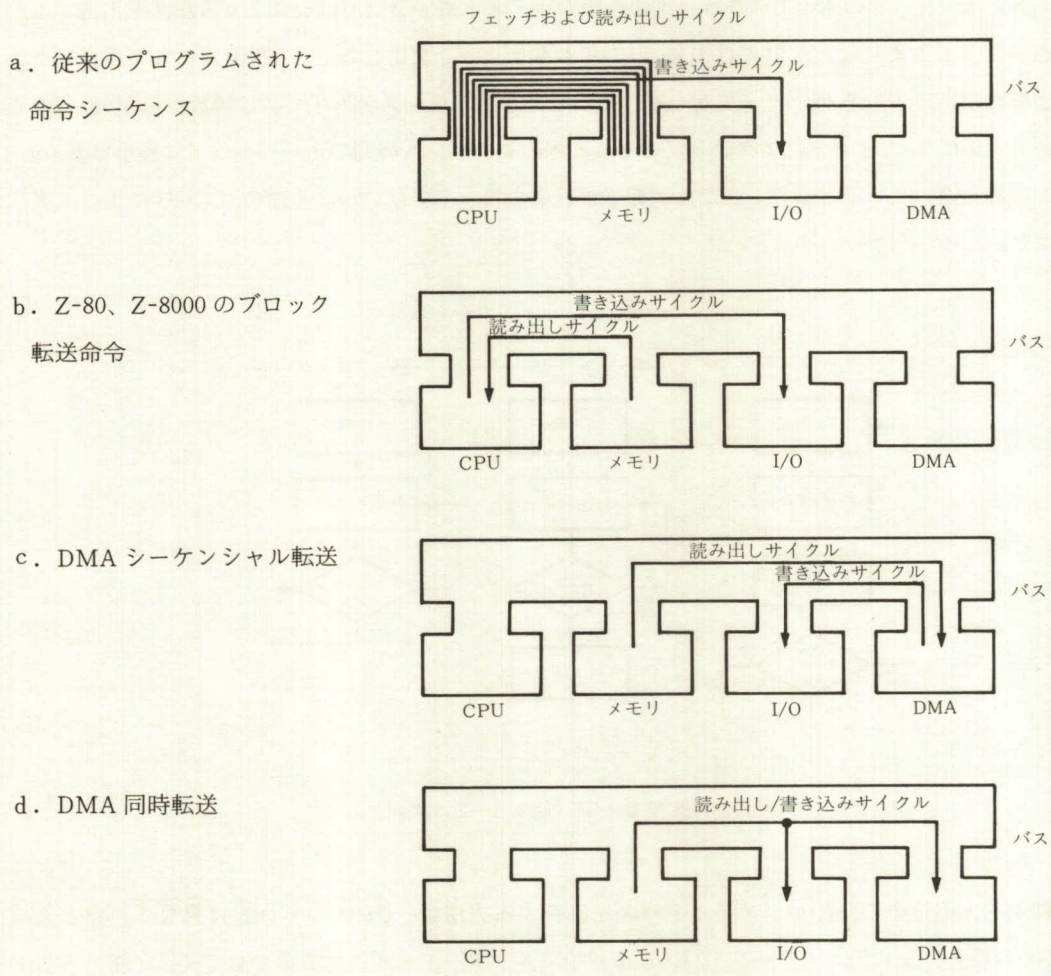


図1.2　入出力転送方法の概念的比較

動作モード：　図1.2cおよびdに示した各方法に対して3種類の動作モードがある。本マニュアルではこれをバイト、バースト、および連続モードと呼ぶことにするが、これはときにはシングル、デマンド、およびブロック・モードと呼ばれるものである。図1.3は各モードについてその代表的なシーケンスを、入出力デバイスのRDYがDMAに対してアクティブとなった後、DMAの動作がエンド・オブ・ブロックあるいはその他の終了状態に到達するまでを示したものである(図2.4−2.6にこれをさらに拡大して示す)。

バイト・モードにおいて、入出力デバイスのRDYがアクティブのとき、DMAは一度に1バイトずつ転送する。システム・バスの制御は各バイト動作ごとに解放されCPUに復帰する。次のバイトを転送するまえに、DMAがCPUに対してシステム・バス制御の新しい要求を出すまで、CPUは他の命令を実行する。バイト・モードは、ともにCPUとDMAの機能をインターリーブできる点でトランスペアレント転送方法に類似している。しかし、バイト・モードは各バイト転送ごとのバスの要求と解除の機能を含んでいる。

バースト・モードは最も一般的なモードで、システム・バスの制御を得て、入出力デバイスのRDYが非アクティブになるまでDMAはバイト転送を続ける。この間CPUは通常アイドル状態となっている。RDYが非アクティブになると、DMAはシステム・バスの制御を解放しCPUに復帰する。

連続モードでは、データのすべてのブロックの転送が完了するまでDMAはシステム・バスを占有する。ブロック転送が完了するまえに入出力デバイスのRDYが非アクティブになると、DMAはこれが再びアクティブになるまでただ待っているが、バースト・モードの場合と異なりシステム・バスは解放されない。RDYが瞬間非アクティブになり、すぐまたアクティブになるときのレスポンス時間のオーバヘッドは最小であるので、連続モードは最も速いモードとなる。しかし、転送中はこのモードではいかなる命令もCPUによって実行されることはない。

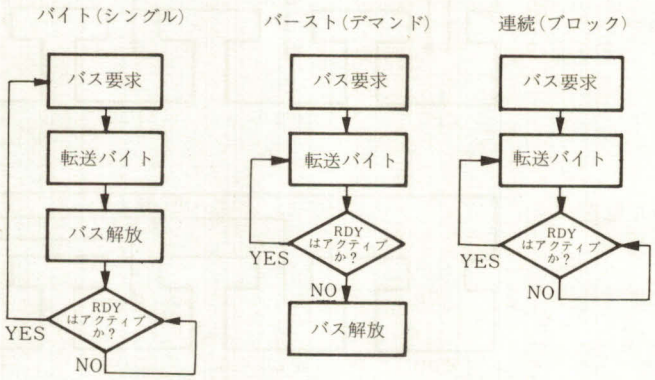


図1.3　動作モード（図2.4−2.6参照）

バスの制御：　大部分のDMAがシステム・バスを制御する方法は、CPUのそれとは異なる。たとえば、DMAの多くの場合、システム・データ・バスに直線的にインターフェイスするのではなく、メモリ・アドレスの一部分をデータ・バス上にマルチプレクスし、さらにこれは外部論理回路によってラッチされることが必要である。また、CPUが発するすべてのバスの制御信号をほとんどのDMAが発するのでもない。そのためDMAの動作中はバスの制御はある程度欠如しているといえる。

8ビットのDMAの中でも、Z-80 DMAはZ-80 CPUと全く同じ読み出しおよび書き込みサイクルのバス制御信号を発し、またデータおよびアドレス・バスに対してCPUと全く同じ論理的および電気的インターフェイスを有するという特長をもっている。これは、他のシステム・コンポーネントにはZ-80 DMAとCPUの区別がつけられず、これらのデバイスによる制御は完全に同一であることを意味する。シーケンシャルDMA転送方法（読み出しサイクルに書き込みサイクルが続く）において、これはまた他のDMAのとき

と異なり、外部のインターフェイス論理回路を使わずにZ-80 DMAの端子を直接Z-80 CPUの端子に接続できることを意味する。この特性のために、設計をかなり単純化でき、部品の数も少なくできる。

プログラム性： DMAがいかに開始してデータを転送し終了するかは、転送のまえにCPUによってDMAに書き込まれた制御情報によって決定する。DMAの転送後、次の転送条件を決定するためにCPUはステータス・レジスタを読み出す。

プログラム性の程度は、種々の転送タスクを処理するDMAのフレキシブル性に直接関係している。ほとんどのDMAは、そのプログラム性では相当制限されている。これに対して、Z-80 DMAは140ビット以上の制御情報を有しており、広範囲におよぶタスクおよびシステムにデバイスを対応させるためにこれを用いている（動作と動作の間に変更することも可）。

たとえば、Z-80 DMAは、エンド・オブ・ブロック、バイトの一致、あるいはRDYの状況に対して目標が達成されたとき、CPUにストップあるいは割り込みをかける。転送を継続あるいは繰り返すなどをプログラムすることができる。あるいは、バイト・モードで転送中にそのバッファされたアドレス・カウンタを再ロードして、新しいロケーションからすぐ次の転送を開始することができる。また、全体の読み出しまたは書き込みサイクルのタイミングは、Z-80ファミリの標準タイミングより早くも遅くもすることができ、他のCPUのメモリまたは入出力デバイスの要求に合致するように、各ポートに対して個別に修正できる。

これらの項目については後の章でさらに詳しく説明する。ここでは、Z-80 DMAに関するより詳しい説明の準備としての一般的な概略にとどめた。

第2章　Z-80 DMAの機能説明

2.1 概　説

Z-80 DMAは、広範囲におよぶ8ビットCPUの中で、データ転送とサーチを行うものである。DMAがCPUからイネーブルされると、システム・アドレス・バス、データ・バス、および制御バスを完全に制御する。このことからZ-80 DMAは特別の目的を有したプロセッサという点でユニークなものである。このDMAは、システム・バスに対して完全なインターフェーシング機能を有する。たとえば、Z-80 CPUシステムの中で、データ転送実行のために、普通はZ-80 CPUが発するのと全く同一の3ステート状態を含むシグナル・レベルとタイミングをZ-80 DMAは発生することができる。しかも、ほとんどの場合外部のTTLゲートを使わないで行えるが、他のDMAでは外部にTTLゲートを必要とする。

データおよびデータ・フロー動作のための広大なプログラム性によって、Z-80 DMAは特殊目的用転送プロセッサと呼べるものである。これはCPUの負担を軽減するばかりでなく、システム設計者の負担をも軽減する。

他の面でもこのZ-80 DMAはユニークである。第1にポート・アドレスは1つでなく2つ発生できる。そして両アドレスをも可変または固定のいずれにもできるので、メモリーメモリ、または入出力ー入出力間転送を1チャネルで実行できる。他のDMAの場合は、1チャネル以上を必要とするか、このような転送はすべて実行不可能かのいずれかである。

このZ-80 DMAのチャネル能力は、いろいろな速さのデバイスもサービスすることができるので、他のいかなる有効なモノリシックDMAのチャネルの能力よりもすぐれている。サイクル拡張のための$\overline{\text{WAIT}}$を有していることに加えて、基本的な読み出しおよび書き込みサイクルを、種々のタイミング条件に応じてプログラムすることも可能である。多くのチャネルが必要なときは、Z-80 DMAを複数個用いることができる。その場合の割り込み構造は、多くの場合に高速な対応ができるようになっている。割り込み信号とベクトルはいくつかの条件下で発生可能である。さらに、Z-80 DMAはデータを自分自身の中をパスすることができる唯一のDMAであり、データをビットごとにマスク可能なバイトでマスクし、一致バイトに対して比較することも可能である。以下にZ-80 DMAの特長の概要を示す。各特長については本章あるいは他の章でさらに詳述する。

以後、本マニュアルではZ-80 DMA、あるいは単にDMAについて述べる。これには2.5MHz Z-80 DMAあるいは4MHz Z-80A DMAデバイスが含まれる。ともに全く同じ特長をもつが、異なる点はスピードのみである。

- ○ 極めて万能なシングルチャネル。
- ○ デュアル・ポート・アドレス発生（両ポートともインクリメント、デクリメント、あるいは固定アドレス可能）。
- ○ バッファされたアドレスおよびブロック長レジスタ。
- ○ ブロック長は最大64Kバイト。
- ○ クロック・レートは2.5または4MHz（Z-80またはZ-80A DMA）。
- ○ データ・レートは1.25または2Mバイト/秒（Z-80またはZ-80A DMA）。
- ○ 転送、サーチ、あるいは転送/サーチ動作。

- ビット単位でマスク可能なバイト・サーチ機能。
- シーケンシャル（フロースルー）または同時（フライバイ）転送。
- Z-80 およびその他多くの CPU との互換性。
- バイト、バースト、連続モード。
- 自動リスタート。
- 可変サイクル・タイミング。
- $\overline{\text{WAIT}}$ によるサイクル拡張可能。
- 内部で修飾可能な割り込みベクトル。
- レディ、エンド・オブ・ブロック、バイト一致でのプログラム可能な割り込み。
- バス要求および割り込みのためのハードウェア優先度デージー・チェーン。
- 外部デバイスに対して周期的にパルス発生可能。
- 21の可変制御レジスタ。
- 7つの読み出し可能なステータス・レジスタ。
- プログラム可能な強制レディ条件。
- RDY のアクティブ状態をプログラム可能。
- プログラム可能な DMA イネーブル。
- 完全なシステム・バスのマスタリング。
- Z-80 システム内でのシーケンシャル転送に外部論理回路不要。

2.2 プログラミング

Z-80 DMA は、CPU に有効な21の書き込み可能な8ビット制御レジスタと、7つの読み出し可能な8ビットステータス・レジスタを有する。DMA がバスを制御していないときはいつでも制御バイトを DMA に書き込んだり、あるいは DMA からステータス・バイトを読み出すことができる。

DMA に書き込める制御バイトには、イネーブル、ディセーブル、リセット、開始アドレス、転送またはサーチ継続、バイトおよびアドレス・カウンタ・クリア、ステータス・ビット・クリアなど、すぐその動作に影響を与えるコマンドが含まれる。さらに、モードを設定する多くの制御バイトも書き込むことが可能で、これには動作クラス、動作モード、ポート構成、開始アドレス、ブロック長、アドレス計数規則、一致バイトおよび一致マスク・バイト、割り込み条件、割り込みベクトル、エンド・オブ・ブロック規則、RDY および $\overline{\text{WAIT}}$ の規則、その他が含まれる。

読み出し可能なステータス・レジスタには、レディ、エンド・オブ・ブロック、バイト一致、割り込み条件を示す汎用ステータス・バイト、さらに現在のバイト・カウントおよびポート・アドレスのためのレジスタが含まれる。このような機能についてはプログラミングに関する章で詳述するが、その多くは一般的な範囲では以下にも記述する。

2.3 動作クラス

Z-80 DMA には 3 種類の基本的な動作クラスがあり、下記に示すようにこのうちの 2 つはそれぞれさらにサブクラスに区分される。

○ DMA の 2 つのポート間のデータ転送

　シーケンシャル転送（フロースルー）

　同時転送（フライバイ）

○ シングル DMA ポートにおけるバイト中の特定のビット・パターンに対するサーチ

○ 2 つの DMA ポート間の転送とサーチの組み合わせ

　シーケンシャル転送／サーチ

　同時転送／サーチ

図2.1はこれらのクラスを示したものである。転送の 2 つのサブクラスを a，b に、サーチのみのクラスを c に、そして転送中サーチの 2 つのサブクラスを d，e に示す。いずれのときも、一定のバイトを転送あるいはサーチ中は、DMA はシステム・アドレス、データ、および制御バスを完全に制御する。DMA のポートとは、データのソースおよびデスティネーションであり、ここではポートとはメモリあるいは入出力デバイスの意味で使われる。

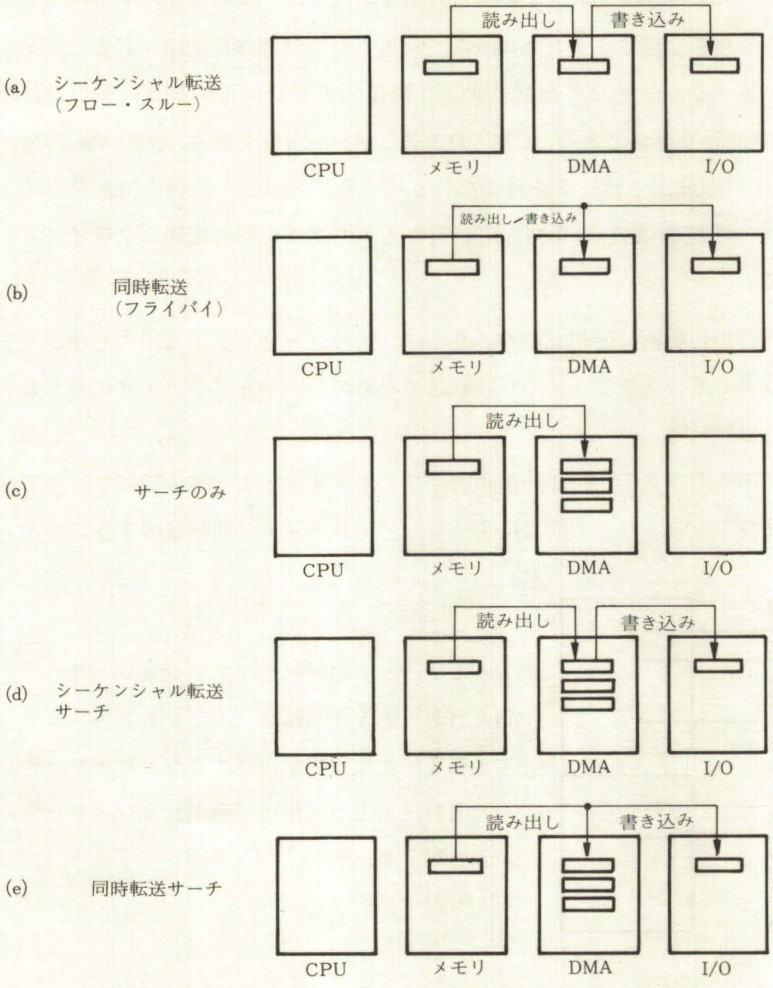


図2.1　動作クラス

シーケンシャル転送はフロースルー転送とも呼ばれ、各バイトの転送には読み出しサイクル、そして続いて書き込みサイクルを必要とする。DMA はデータ・バスを介して内部レジスタにバイトを読み出し、データ・バス上のバイトを次の書き込みサイクルに続ける。Z-80 CPU および他の一部の CPU システムでは、シーケンシャル DMA 転送はDMA および CPU 間に外部の論理回路を必要とせずに実行できる。

同時転送は、フライバイ転送とも呼ばれるものであり、各バイトは1つのマシン・サイクルの中でソースから DMA に読み出され、同時にソースから直接デスティネーションに書き込まれる。したがって、この転送はシーケンシャル転送のときの2倍の割合で実行されるが、制御バスに正しい信号を同時に発生させるために、少なくとも1つの外部論理回路を必要とする（アプリケーションの項を参照)。

サーチのみのクラスでは、データ・バスを介してソース・ポートから DMA に直接読み込まれ、ここで一致バイトと比較される。オプションとして、一致バイトは他のバイトによってマスクし、データおよび一致バイト中の一定のビットのみを比較することができる。Z-80 システムおよび他の一部の CPU システムの中において、サーチのみのクラスは DMA と CPU 間に外部論理回路を必要としない。

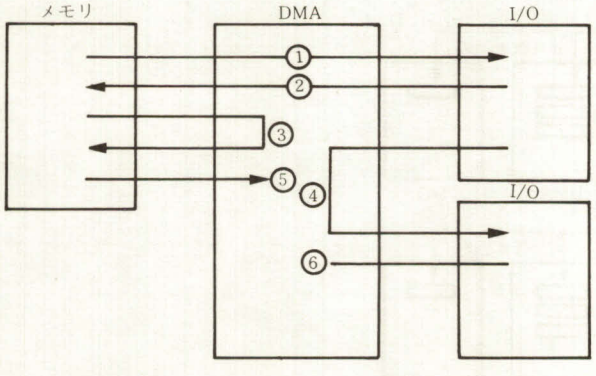
シーケンシャル転送／サーチにおいて、データはシーケンシャル転送クラスと同じ方法で転送され、同時にサーチのみのクラスと同様にサーチされる。このクラスも Z-80 および他のいくつかのシステムでは外部論理回路を必要としない。

同時転送/サーチにおいては、データは同時転送クラスを同じ方法で転送され、同時にサーチのみのクラスと同様にサーチされる。この場合、速度を2倍にするために少なくとも1つの外部論理回路を必要とする。

図2.2は、ソースおよびデスティネーションとして選択できる2種類のアドレス可能なポートについて、Z-80 DMA の動作の各クラスの機能を示すものである。DMAのアプリケーションで最も一般的なものは、メモリー入出力あるいは入出力ーメモリ間転送でサーチを含まないものであり、ほとんどの DMA デバイスもこの動作に限定されている。同時サーチ機能は Z-80 DMA に特有なものである。同時転送（フライバイ）はメモリー入出力間転送に限定される。

メモリーメモリ間転送はメモリ内容の再配置に便利である。さらに、メモリ・マップド入出力もサポートする。レディ条件を DMA にプログラムして、メモリーメモリ間転送のためのレディをアクティブにしておくことができる（プログラミングの項を参照)。

入出力ー入出力間転送は、入って来るデータの一時記憶が必要なリアルタイム・データの取得などのアプリケーションに使うことができる。オプション・サーチ能力によって、メモリーメモリ間転送のようにバイトの



①メモリー入出力間転送（オプション・サーチ）
②入出力ーメモリ間転送（オプション・サーチ）
③メモリーメモリ間転送（オプション・サーチ）
④入出力ー入出力間転送（オプション・サーチ）
⑤メモリ・サーチ
⑥入出力サーチ

図2.2　Z-80 DMA の基本的機能

一致がある場合いろいろな動作に分岐させることが可能である。転送を伴わないメモリ・サーチおよび入出力サーチはZ-80 DMAに特有なもので、エンド・オブ・ブロック、キャラクタ・チェック、またはブロック中の他の特別なバイトを検出するのに便利である。

2.4 動作モード

動作の各クラスについて、Z-80 DMAは次の3つの転送/サーチ・モードのいずれかで動作するようにプログラムできる。

バイト・モード： データ処理動作は1回に1バイトである。各バイト動作ごとにシステム・バスは解放され、CPUに復帰する。次のバイト動作のためには再びバス要求を必要とする。このモードはシングルあるいはバイト・アット・ア時間モードとも呼ばれることがある。

バースト・モード： データ処理動作は、そのDMAに対するポートのRDYが非アクティブになるまで続く。そしてDMAは現在のバイト動作の完了後ストップする（システム・バスを解放する）。このモードはデマンド・モードとも呼ばれる。

連続モード： データ処理動作は、システム・バスが解放される以前にプログラムされたデータ・ブロックの最後のバイトの転送が完了するまで、あるいは一致のストップの条件となるまで続く。そのまえにポートのRDYが非アクティブとなるときは、RDYが再びアクティブになるまで単に停止する。このモードはブロック・モードとも呼ばれる。

どのモードにおいても、DMAがデータを読み込むと、他の信号の状態（ポートのRDYを含む）に関係なく、そのバイト動作を定められたシーケンスに従って完全に実行する。図2.3は1つのバイトのシーケンシャル転送/サーチに見られるシーケンスを示したものであり、動作モードには関係ない。はじめにソース・ポート・アドレスが可変アドレス・ポートとしてプログラムされるときは、これがインクリメントあるいはデクリメン

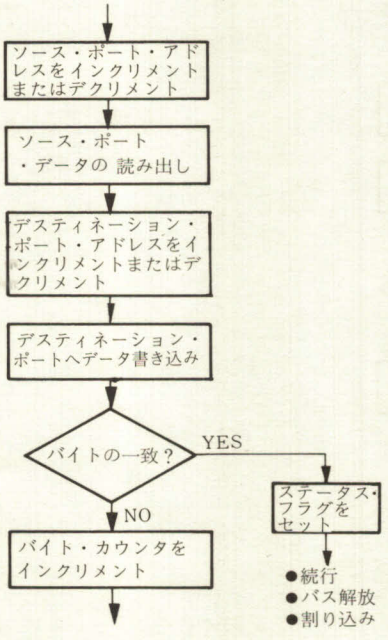


図2.3　1バイト転送／サーチ

トされる。次に、バイトがそのポートからDMAに読み込まれる。デスティネーション・ポートが可変としてプログラムされているときは、次にデスティネーション・ポートがインクリメントあるいはデクリメントされる。バイトは次にデスティネーション・ポートに書き込まれる。サーチ機能が含まれるときは、データは一致バイトと比較される。データが一致しないときは、DMAは単にバイト・カウンタをインクリメントし続行する。データが一致したときは、ステータス・ビットがセットされてDMAはそのはじめのプログラミングに応じて、バイト・カウンタをインクリメントして続行するか、あるいは停止するか（バスを解放する）、あるいはCPUに割り込む。図2.4－2.6は、図2.3に示した1バイトの動作の前、途中、および後の各モードの動作を示すものである。

バイト・モード（図2.4）の動作は、CPUからのイネーブル・コマンドと入出力デバイスのRDYをテストすることからはじまる。RDYがアクティブのときは、DMAは$\overline{\text{BUSAK}}$を通じてシステム・バス（アドレス、データおよび制御バス）を要求し、CPUが$\overline{\text{BUSAK}}$を出した直後にDMAはバス制御を解放する。次に、図2.3に示したように1バイトの転送/サーチが実行される。この後、バイト・カウンタがプログラムされたブロック長に達したかどうかを見ることによってエンド・オブ・ブロックをテストする。ブロックの終わりに達していないときは、DMAはバスをCPUに解放する。終わりに達しているときは、ステータス・ビットをセットして、はじめのプログラミングに応じた動作が行われる。DMAが各バイトの終わりごとにバスを解放することによって、CPUは次のバイトの転送のために再びバスをDMAに解放するまで、少なくとも1以上のマシン・サイクルを実行する。このことは、他のモードのときに比較するとDMAの動作は低速であるが、バイト・モードにおいては割り込みアクノリッジ、ポーリング、メモリ・リフレッシュのようなCPUの機能は、DMA転送とインターリーブすることができることを意味する。

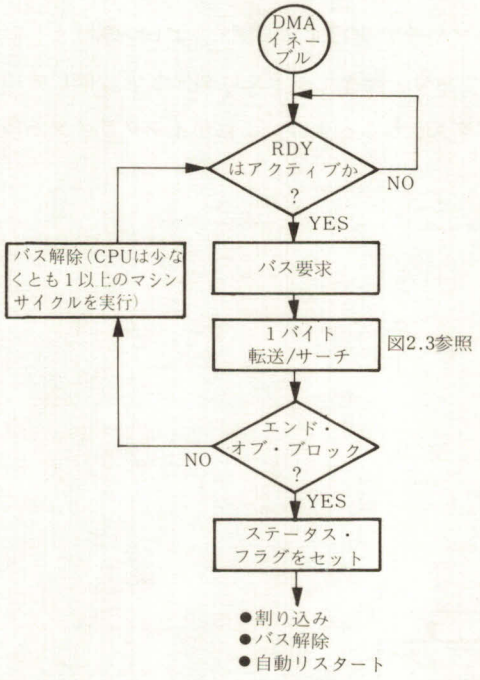


図2.4　バイト・モード

バースト・モード（図2.5）においてバス要求の方法は先と同じであるが、一度DMAがバスの制御を得ると図2.3に示したように入出力ポートからのRDYが非アクティブになるか、エンド・オブ・ブロックに達するか、あるいはバイトの一致に達するまでDMAは転送を続行する。エンド・オブ・ブロックに達するまえにRDYが非アクティブになると、DMAはバスをCPUに解放し、RDYが再びアクティブになるまでRDYをテストし続ける。その後再びバスを要求し転送を続行する。このことから、DMAは実際にバスを使用できるまでバス要求をせず、一度バスを獲得すると最高速度で転送するので、汎用アプリケーション用として最も有用性が高いものである。しかし、転送が長い場合は、CPUの他の機能はDMA転送の全期間中にわたって停止するという支障をこのモードは有する。

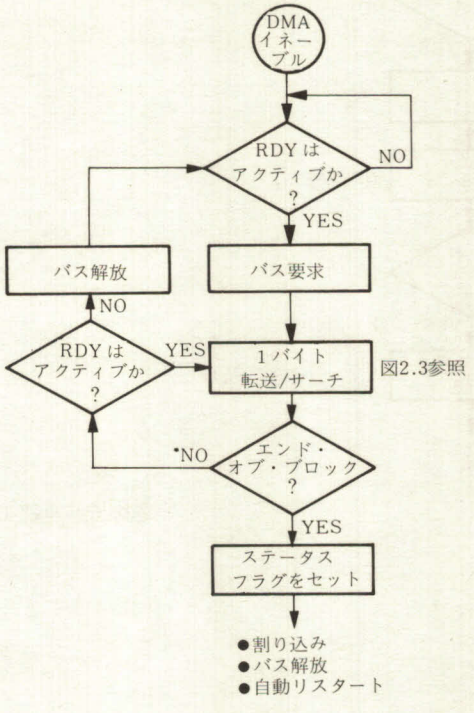


図2.5　バースト・モード

連続モード（図2.6）において、他のモードと同様にDMAはバスを要求し、バースト・モードと同様にバイト転送を繰り返す。しかし、バースト・モードの場合と違って、エンド・オブ・ブロックまたはバイトの一致によって停止するまえにRDYが非アクティブになる場合、バスはDMAにより保持されたままになる。RDYが再びアクティブになるまでDMAはバスを保持し、単にアイドル状態となり、そして転送シーケンスを完了する。このモードでは、ブロック転送の完了後バスを解除し、再びこれを要求するという必要が省かれるので、3つのモードの中で最も速いモードである。しかし、このモードではシステム・バスはつねにDMAによって強制的に占有される。このモードは、非常に速い転送が必要なとき、そしてCPUの能力への要求がある程度緩和されるときにのみ使われる。たとえば、ディスクからメモリへロードするようなシステムのときである。

DMA のデータ読み出しが高速バッファ方法のため、次のバイトが読み出されるまでは1バイト動作は完了しない。これは、バイト・モードにおいても転送あるいはサーチの全ブロック長は2バイト以上で、DMA にプログラムされたブロック長は、希望するブロック長より1バイト少なくなければならないことを意味する。この現象は、内部構造の章の中のアドレスおよびバイト計数の項で詳述する。

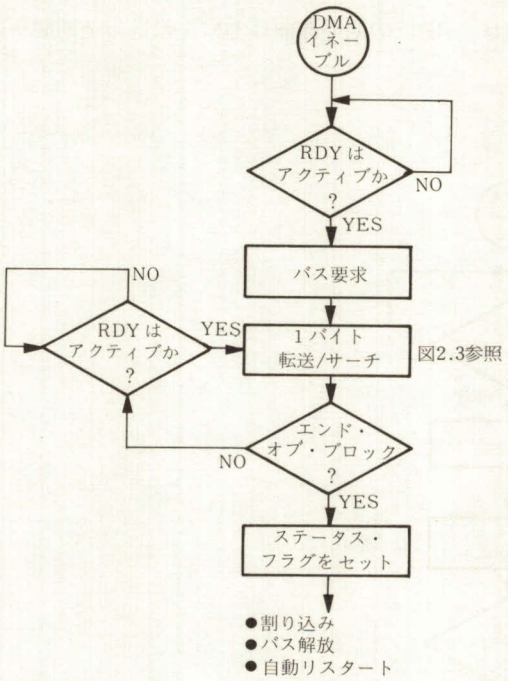


図2.6　連続モード

2.5　転送速度

8ビット DMA デバイスの中でZ-80 DMA は最高の最大転送率を有する。この転送率は同時転送動作クラスのとき達成されるもので、より一般的なシーケンシャル転送クラスと違って少なくとも1つの外部論理回路を必要とする。しかし他のすべての8ビット DMA も何らかの外部論理回路を必要とするから、正当な速度比較は可能である。

表2.1は、DMA 動作のいろいろなクラスにおいて達成できる最高転送率を示したものである。比較のために、CPU の最高転送率も示してある。DMA 転送についてはバーストまたは連続モードの割り込みを含まないときを想定してあり、読み出しおよび書き込みサイクルは2サイクルを想定してある（次の可変サイクルの項を参照）。

バイト・モードの転送速度は、DMA のバイト転送の間 CPU がバスを保持する時間によって左右される。このため、この速度は CPU の見地から最もよく用いられる。

	Z-80 (2.5 MHz)	Z-80A (4.0 MHz)
DMA 同時転送 DMA サーチのみ DMA 同時転送／サーチ	1.25Mバイト／S	2Mバイト／S
DMA シーケンシャル転送 DMA シーケンシャル転送／サーチ	0.625Mバイト／S	1Mバイト／S
CPU ブロック転送命令	0.125Mバイト／S	0.200Mバイト／S

表2.1　転送およびサーチ最高速度（バーストおよび連続モード）

表2.2は、バイト・モードのDMA転送により生じるZ-80スループット減少率（転送キロボー当たり）を示したもので、この減少率をCPUが6つの（実際上の下限）命令から成る割り込みサービス・ルーチンを使って、バイト転送を実行するときに生じるであろうスループット減少率と比較してある。アプリケーションの章のDMAおよびZ-80 SIOの項に、このデータに関してさらに詳しく述べる。このデータでは、読み出しまたは書き込みにつき最小2クロック・サイクルよりも長いサイクル・タイミングによるシーケンシャルDMA転送を想定してある。したがって、2クロック・サイクルの同時転送のCPUスループットに対する減少率はさらに低いものになる。

表2.2は、バイト・モードのDMA転送によるCPUのスループット減少率が、通常の割り込みモードによってCPUが自己のバイト・モードで入出力を処理するときに比べると、約1/5になることを示している。表2.1では、バースト・モードおよび連続モードによるDMA転送率が、Z-80 CPUのときに比べると10倍までは速くできることを示している。

	Z-80 (2.5 MHz)	Z-80A (4.0 MHz)
DMA シーケンシャル転送 DMA シーケンシャル転送／サーチ	0.065%	0.041%
CPU 割り込みによる転送	0.340%	0.213%

表2.2　バイト・モード転送のキロボー当たりのZ-80 CPUスループット減少率

2.6 アドレス発生

各転送動作において、2つの16ビット・アドレスがDMAによって発生する。1つはソース・ポート、そしてもう1つはデスティネーション・ポートである。DMAがソースを読み出しているかあるいはデスティネーションに書き込んでいるかによって、この2つのアドレスはアドレス・バスにマルチプレクスされる。

2つのポートを仮にポートAとポートBとする。ポートA、Bともにソースまたはデスティネーション、メモリまたは入出力とし、固定または可変アドレスとすることができる。

可変アドレスは、プログラムされた開始アドレスから自動的にインクリメントまたはデクリメントできる。固定アドレスは入出力デバイスに便利で、DMAの固定アドレス発生能力によって入出力デバイスの転送/サーチ・イネーブルの結線を省略できる（チップ・イネーブルの結線は必要であるが、これはすべての周辺回路についても同じである）。

2つの読み出し可能なアドレス・カウンタが各ポートの現在のアドレスを保持する。これらのカウンタは各ポートの開始アドレス・レジスタとは異なる。すなわち、カウンタはレジスタによってバッファされている。そのため、DMAがバスを解放している限り（たとえば、バイト・モードのときのバイト転送後）新しい開始アドレスをDMAに書き込むことができる。したがって、現在のブロックの転送完了まえに、新しいデータ・ブロックの新しい開始アドレスをDMAにロードすることが可能である。新しい開始アドレスをロードしても関連したポートのアドレス・カウンタの内容は影響をうけない。

次に、DMAのアドレス発生能力の用途を部分的に要約する。

- ○ 基準アドレスから開始してカウント・アップまたはカウント・ダウンする。
- ○ 1つのアドレス・シーケンスの完了後、自動的にはじめにステップ・バックする。
- ○ 次のシーケンスのために、新しい開始アドレスをロードしたり、あるいはまえの開始アドレスを再ロードする。

2.7 バイトの一致（サーチ）

バイトの一致（サーチ）は、単一の機能として、あるいは転送と同時に行うことができる。バイトの一致があるときは、読み出し可能なステータス・レジスタ内のステータス・ビットがセットされ、DMAが次のうちいずれが1つを行うようプログラムできる。

- ○ バイトの一致の直後停止する（バスを解放する）。
- ○ バイトの一致の直後停止してCPUに割り込む。
- ○ エンド・オブ・ブロックでDMAが停止するときCPUに割り込む。

DMAに書き込まれた一致バイトは、一致バイト内の特定のビットのみをサーチするデータの対応ビットと比較できるように、別のバイトでマスクすることができる。

2.8 割り込み

DMAは、次の3つの条件のときCPUに割り込みできるようプログラムが可能である。

- ○ レディに対する割り込み。
- ○ バイトの一致に対する割り込み。
- ○ エンド・オブ・ブロックに対する割り込み。

はじめの条件（入出力ポートのRDYがアクティブになる）の場合、DMAがバスを要求するまえに割り込みを発生させる。他の2つの条件では、DMAの停止（バスを解放する）後、DMAがCPUに割り込む。バイトの一致またはエンド・オブ・ブロックでのDMAの停止は、それぞれ別にプログラムする。

これらの条件（RDYがアクティブになる、バイトの一致、またはエンド・オブ・ブロック）のいずれでも読み出し可能なステータス・ビットがセットされる。さらに、これらの条件のうちのいずれの割り込みがプログラムされても割り込み保留ステータス・ビットがセットされ、上述のいずれのタイプについてもオプションとしてDMAの割り込みベクトルを修飾することができる。

DMAは、リアルタイム・アプリケーションにおいて、高速に割り込みサービスができるZ-80の巧妙な割り込み方式を採用している。CPUがそのモード2の割り込みを使用するZ-80 CPUシステムにおいて、DMAは内部で修飾可能な8ビット割り込みベクトルをCPUに送り、CPUはこれにさらに8ビットを加えて割り込みルーチン・テーブルのメモリ・アドレスを形成する。このテーブルは割り込みルーチン自身の開始アドレスを含む。このプロセスにおいて、CPUの制御は割り込みルーチンに移行し、割り込みアクノリッジ後に実行する次の命令が割り込みルーチン自身のはじめの命令となる。

2.9 自動リスタート

DMAはブロック転送を自動的に繰り返すことができる。この機能によってバイト・カウンタはクリアされ、開始アドレス・レジスタの内容をアドレス・カウンタに再ロードする。

この自動リスタート機能によって、CRTのリフレッシュや他の多くの繰り返し動作のときのCPUのソフトウェア負担を軽減できる。さらに、CPUはバイト・モードで転送中（または、バースト・モードで転送中RDYが非アクティブでバス解放のとき）、異なった開始アドレスをバッファ・レジスタに書き込むことができ、自動リスタートは新しいアドレスから開始することが可能となる。

2.10 パルス発生

256バイトのインターバルで信号を出力するDMAのパルス出力を使って転送されたバイト数を、外部デバイスは知ることができる。動作の開始において、インターバルのシーケンスは1から255バイトの範囲でオフセットさせることができる。

パルス信号は$\overline{\text{INT}}$から出力するが、この信号は$\overline{\text{BUSRQ}}$と$\overline{\text{BAI}}$および$\overline{\text{BAO}}$がともにアクティブのときのみ出力するため、CPUがこれを割り込み要求と誤まることはない。このような条件下では、Z-80 CPUは$\overline{\text{INT}}$をモニタしないからである。

2.11 可変サイクル

Z-80 DMAは、プログラムによって動作サイクル長を変更できるという特長がある。これはいろいろなCPUや他のシステム・コンポネントの動作速度に関する特別な要求にDMAを追随させ、データ転送率を最大にするときに有用である。また、外部論理回路を省略し、CPUのソフトウェア負担をも軽減する。

この可変サイクル機能には2つの面があり、第1に、ソースおよびデスティネーション・ポートに関連した読み出しおよび書き込みの全サイクル（期間）を、2、3、または4クロック・サイクル（$\overline{\text{WAIT}}$を使うときは4以上）に独立的にプログラムできるので、DMAの制御信号の立ち上がる時点を変化させることが可能

である。第2に、データ転送に関連した各ポートの4つの信号（$\overline{IORQ}$、$\overline{MREQ}$、$\overline{RD}$、および$\overline{WR}$）のそれぞれ立ち上がりエッジを½クロック・サイクル早く終了させることができる。これによってさらにフレキシブルなものとなり、データが変化を開始するまえに、通常より早く$\overline{RD}$または$\overline{WR}$を非アクティブにすることが可能となる。図2.7は一般的な機能を示したものであり、これについてはタイミングの章で説明する。

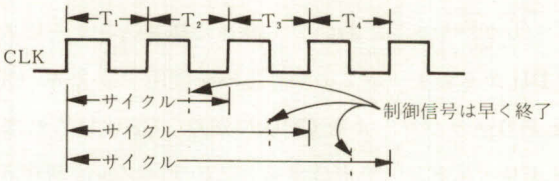


図2.7　可変サイクル長

2.12　目標と動作

表2.3は、プログラムの仕方によってDMAに一定の動作をとらせるための目標の概略を示したものである。目標とはDMAの内部レジスタ、入出力デバイスからの信号、あるいはデータ・バス上のDMAに対する命令の条件のことである。

状　況	左の状況のとき起こり得る動作	状　況	左の状況のとき起こり得る動作
エンド・オブ・ブロック	1．バス解放 2．CPU割り込み 3．自動リスタート	RDY非アクティブ	1．バス解放 2．中断(連続モードのみ)
バイトの一致	1．バス解放 2．CPU割り込み 3．コンティニュー	RDYアクティブ	1．バス要求 2．CPU割り込み
パルス制御バイトがバイト・カウンタ下部に一致	1．パルス発生	RETI命令（CPUからの割り込み復帰命令）	1．バス要求

表2.3　状況と動作

第3章　端子信号

図3.1および図3.2に従ってZ-80 DMAの外部端子の機能を説明する。

A_0－A_{15}　**システム・アドレス・バス（3ステート出力）**

A_0－A_{15}は、DMAによって作られたアドレスが出力されるバスで、ソースおよびデスティネーション・ポートに送られる。ポートはメイン・メモリまたは周辺入出力デバイスである。

$\overline{BAI}$　**バス・アクノリッジ入力（入力、アクティブ“Low”）**

$\overline{BAI}$は、システム・バスがDMA制御のために解放されたことを示す。複数個のDMA構成では、最高位の優先順位のDMAデバイスの$\overline{BAI}$を、CPUの$\overline{BUSAK}$に結線する。低い優先順位のDMAの$\overline{BAI}$は、それより優先順位の高いDMAの$\overline{BAO}$に結線する。

$\overline{BAO}$　**バス・アクノリッジ出力（出力、アクティブ“Low”）**

$\overline{BAO}$は、複数個のDMA構成において、CPUがバスの制御を切り放したことを示す。$\overline{BAI}$と$\overline{BAO}$をデージー・チェーンに接続し、複数個のDMA間でバス制御の優先順位を決定する。DMAにおけるデージー・チェーンは、IEIとIEOを接続した割り込みデージー・チェーンと異なり、1つのDMAが既にバス制御を得ているとき、より高い優先順位のDMAが制御を強制排除することはできない。バスを有しているDMAは、その優先順位に関係なく動作を完了できる。

$\overline{BUSRQ}$　**バス要求（入出力、オープン・ドレイン、アクティブ“Low”）**

$\overline{BUSRQ}$は、システム・アドレス・バス、データ・バス、制御バスを制御するための要求をCPUに送る。複数個のDMAが$\overline{BAI}$と$\overline{BAO}$によってデージー・チェーン接続されている場合、このDMA以外のDMAがいつバス要求を出したかを検出し、そのDMAの動作が終了するまでこのDMAがバス要求を出すことを保留させる。$\overline{BUSRQ}$は双方向であるため、DMA間にバッファを置くことはできない。しかし、CPUに対しては単方向なので、CPUの間にはバッファを置くことができる。通常$\overline{BUSRQ}$は、1.8kΩのプルアップ抵抗を接続する。

$\overline{CE}/\overline{WAIT}$　**チップ・イネーブル/ウエイト（入力、アクティブ“Low”）**

$\overline{CE}/\overline{WAIT}$は、通常$\overline{CE}$として動作するが、$\overline{WAIT}$として働くようにプログラムすることもできる。CPUからの$\overline{CE}$として使用する場合、$\overline{IORQ}$がアクティブで、システム・アドレス・バス上の入出力ポート・アドレス（16ビットまで）がそのDMAのアドレスで、制御バイトをCPUからDMAに書き込めるとき、この$\overline{CE}$はアクティブになる。DMAが

CPU からバス要求に対する $\overline{\text{BUSAK}}$ を受け取った後、この端子をメモリあるいは入出力デバイスからの $\overline{\text{WAIT}}$ として使用する場合、この端子は DMA の動作サイクルに待ち状態を挿入することができ、DMA の動作速度を遅くしてメモリまたは入出力の動作速度に合わせることができる。$\overline{\text{CE}}$ / $\overline{\text{WAIT}}$ を CPU の $\overline{\text{BUSAK}}$ にマルチプレクスさせる方法については、アプリケーションの章で説明する。

CLK　**システム・クロック（入力）**

CLK は、標準 Z-80単相クロックで、2.5 MH_Z（Z-80 DMA）または 4MH_Z（Z-80A DMA）である。これより遅いシステム・クロックでは、プルアップ抵抗を含む TTL ゲートを使用し、そのタイミングおよび電圧レベルに合わせる。高速システムにはアクティブ・プルアップを含むクロック・ドライバを用い、V_{IH}仕様および立ち上がり時間条件に合わせる。いずれの場合でも10kΩ（最大）のプルアップ抵抗を使用し、DMA リセット時の適正な電力を得ることが必要である。

D_0－D_7　**システム・データ・バス（3テート入出力）**

D_0－D_7は、双方向のデータ・バスである。この端子によって、CPU からの制御バイト、DMA からのステータス・バイト、メモリまたは周辺入出力デバイスからのデータを転送する。DMA によるデータ転送やサーチは、DMA がこの端子および A_0－A_{15}を制御してるときにのみ実行可能である。これらのバスを CPU が制御しているときは、DMA の制御またはステータス・バイトの書き込み読み出しができる。

IEI　**割り込みイネーブル入力（入力、アクティブ“High”）**

IEI は、IEO とともに使用し、割り込みデバイスが 2 個以上のときの優先順位を決めるためのデージー・チェーンを形成する。この端子が“High”の場合、より高位の優先順位の他のデバイスからの割り込みがないことを示し、この DMA がイネーブルされていれば割り込むことが可能である。

IEO　**割り込みイネーブル出力（出力、アクティブ“High”）**

IEO は、IEI とともに使用し、割り込みデバイスが 2 個以上のときの優先順位を決めるためのデージー・チェーンを形成する。この端子はIEI が“High”で DMA が割り込みを要求していないときにのみ“High”となる。高位の優先順位のデバイスが、その CPU の割り込みサービス・ルーチンのサービスを受けている間、この信号を“Low”にして優先順位の低いデバイスの割り込みを抑える。割り込みデージー・チェーン内のデバイスは、優先順位の低いデバイスに対するサービスが完了するまえに、より高い優先順位のデバイスが割り込み可能である。

$\overline{\text{INT}}/\overline{\text{PULSE}}$　**割り込み要求（出力、オープン・ドレイン，アクティブ“Low”）**

$\overline{\text{INT}}/\overline{\text{PULSE}}$ は、DMA がバスを制御していない場合“Low”となって CPU に対して割り込みを要求する。これに対する CPU のアクノリッジは、$\overline{\text{M1}}$ サイクル中に $\overline{\text{IORQ}}$ 出力を“Low”にすることによって行う。DMA の $\overline{\text{INT}}$ はプルアップ抵抗を付けて CPU の $\overline{\text{INT}}$ と結線し、システム内の他のすべての $\overline{\text{INT}}$ とも結線して使用する。$\overline{\text{PULSE}}$ は、外部デバイスに対して周期的にパルスを発生させるために用いられる。ただし、これは DMA がバスを制御しているときに限る。(例：CPU の $\overline{\text{BUSRQ}}$ および $\overline{\text{BUSAK}}$ がともに“Low”で、CPU が割り込みに応答することができないとき)。

$\overline{\text{IORQ}}$　**入出力要求（3ステート入出力、アクティブ“Low”）**

$\overline{\text{IORQ}}$は、入力として用いる場合、アドレス・バスの下半分に CPU からの制御、あるいは CPU へのステータス・バイトを転送するための正しい入出力ポート・アドレスが出ていることを示す。DMA は、$\overline{\text{CE}}$、$\overline{\text{WR}}$、または $\overline{\text{RD}}$ が同時にアクティブになるとアドレス指定される。出力として用いる場合、DMA がシステム・バスの制御を得た後、アドレス・バスに DMA のデータ転送に関連した他の入出力デバイスへの正しい16ビットのポート・アドレスが出ていることを示す。$\overline{\text{IORQ}}$ および $\overline{\text{M1}}$ がともに DMA に対して同時にアクティブ入力のとき、CPU による割り込みアクノリッジを示す。

$\overline{\text{M1}}$　**マシン・サイクル1（入力、アクティブ“Low”）**

$\overline{\text{M1}}$ は、現在の CPU のマシン・サイクルが命令フェッチであることを示す。これは DMA の割り込み構成の中で2つの目的を有する。第1は、割り込みサービス・ルーチンの終了時に、CPU がデータ・バスよりフェッチする割り込み復帰命令（RETI）を DMA が検知するときに使われる。第2は、$\overline{\text{M1}}$ および $\overline{\text{IORQ}}$ がともに DMA に対してアクティブ入力のとき、割り込みアクノリッジとしての意味をもつ特別な2バイトの命令フェッチの場合、$\overline{\text{M1}}$ はその2バイトの OP コードがフェッチされるごとにアクティブとなる。

$\overline{\text{MREQ}}$　**メモリ要求（3ステート出力、アクティブ“Low”）**

$\overline{\text{MREQ}}$ は、A_0－A_{15}上にメモリ読み出しまたは書き込みに対する正しいアドレスが乗っていることを示す。DMA がシステム・バスの制御を得た後、メモリからまたはメモリへの DMA 転送要求を示す。

$\overline{\text{RD}}$　**読み出し（3ステート入出力、アクティブ“Low”）**

$\overline{\text{RD}}$ は、入力として用いる場合、この DMA が選択されていれば CPU が DMA の読み出しレジスタからステータス・バイトを読み出すことを示す。出力として用いる場合、DMA がシステム・バスの制御を得た後、メモリまたは入出力ポート・アドレスからの読み出しが DMA によって制御されていることを示す。

RDY　　**レディ（入力、アクティブ“Low”または“High”にプログラム可能）**

RDYは、DMAによってモニタされ、DMAに関連した周辺デバイスが読み出しや書き込みの準備ができたか否かを判定する。DMAがイネーブルされたとき、RDYは間接的にDMAのアクティビティを制御する。RDYによってDMAのアクティビティが制御される方法は、動作モード（バイト、バースト、連続）によって異なる。RDYをアクティブにするには、制御レディ条件をプログラムすることによってシミュレートできる。これはメモリーメモリ間転送に便利である。RDYはCLKに同期させることが望ましい。たとえば、CLKの立ち上がりエッジにおいてRDYはアクティブでなければならない。これは特に連続モード動作のときに重要である。

$\overline{WR}$　　**書き込み（3ステート入出力、アクティブ“Low”）**

$\overline{WR}$は、入力として用いる場合、CPUがDMAの書き込みレジスタに対して制御バイトを書き込むことを示す。出力として用いる場合、DMAがシステム・バスを得た後、メモリや入出力ポート・アドレスへの書き込みがDMAに制御されていることを示す。

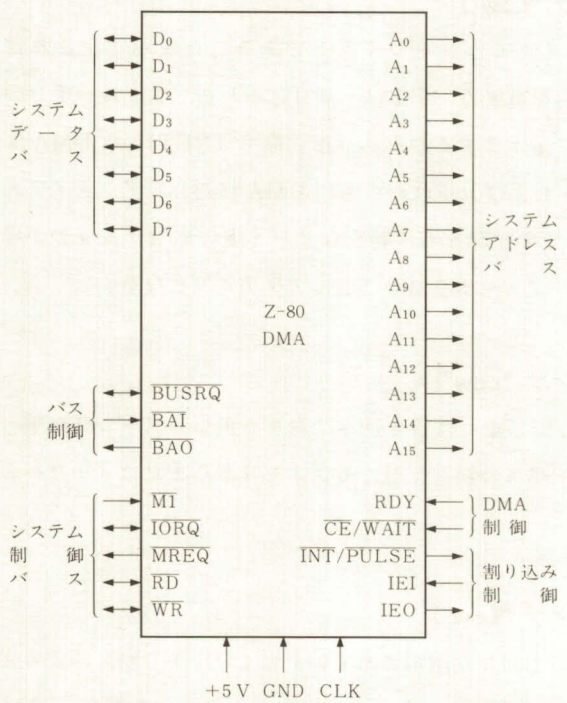


図3.1　端子説明

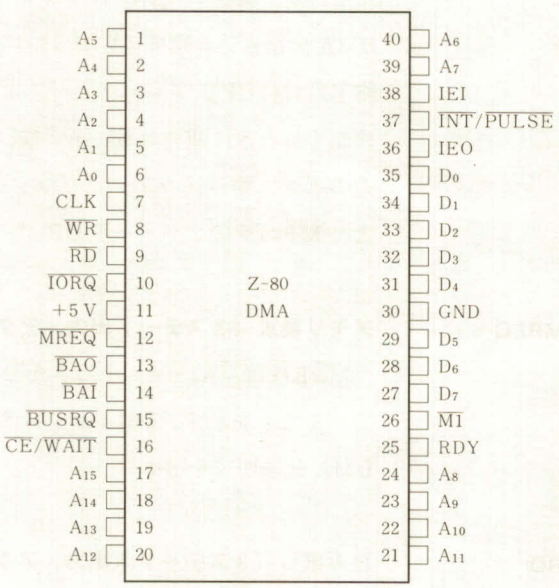


図3.2　端子配置図

第4章　内部構成

4.1 概　説

Z-80 DMA の内部構成は、8ビット・データ・バス、16ビット・アドレス・バス、およびシステム制御バスとのインターフェイスのためのドライバおよびレシーバ回路を含む。Z-80 CPU を用いたシステムにおいて、動作がシーケンシャル転送およびサーチに限定されている場合、DMA の $\overline{CE}/\overline{WAIT}$ 以外はバッファを使用しないで CPU の対応する端子に直接接続できる。これについてはアプリケーションの章で説明する。

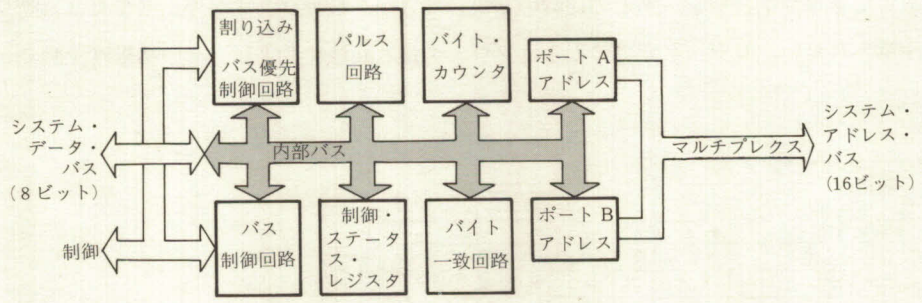


図4.1　Z-80 DMA のブロック図

図4.1は、DMA の内部データ・バスとシステム・データ・バスとのインターフェイスの方法と、すべての内部論理回路とレジスタとの接続を示したものである。DMA の単一の転送チャネルのポートAおよびポートBから発生したアドレスは、システム・アドレス・バス上にマルチプレクスされる。

DMA の専用論理回路は、外部バスとのインターフェイス、内部バス制御、バイトの一致、バイト計数、周期的パルス発生、CPU 割り込み、バス要求、およびアドレス発生などの種々の機能を有する。

4.2 制御およびステータス・レジスタ

DMA には、21個の書き込み可能な制御レジスタと7個の読み出し可能なステータス・レジスタがあり、CPU はこれらを設定し、モニタすることができる。すべてのレジスタは8ビット（データ・バスのビット数）で、2バイト情報は連続したレジスタに保持される。

21個のデータの書き込み可能な制御レジスタは、WR0から WR6まで7個の基本レジスタ・グループに分類され、その大部分は複数のレジスタ構成である。

各グループの基本レジスタは、制御ビットとそのグループ内の他のレジスタのアドレス設定のポインタ・ビットを含む。1つのグループ内のレジスタへの書き込みは、まずポインタ・ビットを設定したデータを基本レジスタに書き込み、次にそのグループ内でポインタによって示された1または2個以上の他のレジスタに書き込むことによって行われる。この技術についてはプログラミングの章で詳述する。図4.2に書き込みレジスタを示すが、多くのモードおよび機能は各グループの基本レジスタのバイトの単一のビットで設定されるので、書き込みレジスタの名称は DMA のプログラム性を完全に表現したものではない。

図4.2は書き込みレジスタを示したものである。これらのレジスタへのデータ・バスからの入力を図示して

ある（この図と読み出しレジスタを示す図4.3を比較すること）。

WR0 および WR4 グループ内の 2 個の16ビット開始アドレス・レジスタ、そして WR0 グループ内の16ビット・ブロック長レジスタには、16ビット・カウンタが付属している（カウンタはその関連レジスタと異なり、書き込むことはできない）。2個のアドレス・カウンタは、アドレス・バスに出力すべきアドレスを発生する。これは、バイト・カウンタと同様に、データ・バスを通じて CPU によって読み出すことが可能である。3つの書き込み可能なレジスタは、読み出し可能なカウンタのバッファとしても動作する。レジスタの内容はブロック転送中、カウンタの内容に影響することなく変更できる。このことは、たとえばバイト・モード転送において、バイト転送後 CPU がバスを制御している間に制御バイトを DMA に書き込むときに便利である。これによって、異なる場所にある次のブロック（自動リスタート・ブロックのこともあり得る）をすばやく開始させることができる。ブロック長レジスタとの比較の結果、ブロック長カウンタが停止（または自動リスタート）することに注意されたい。レジスタ変更のときはブロック長も同様に変化し、その結果は予測できない。

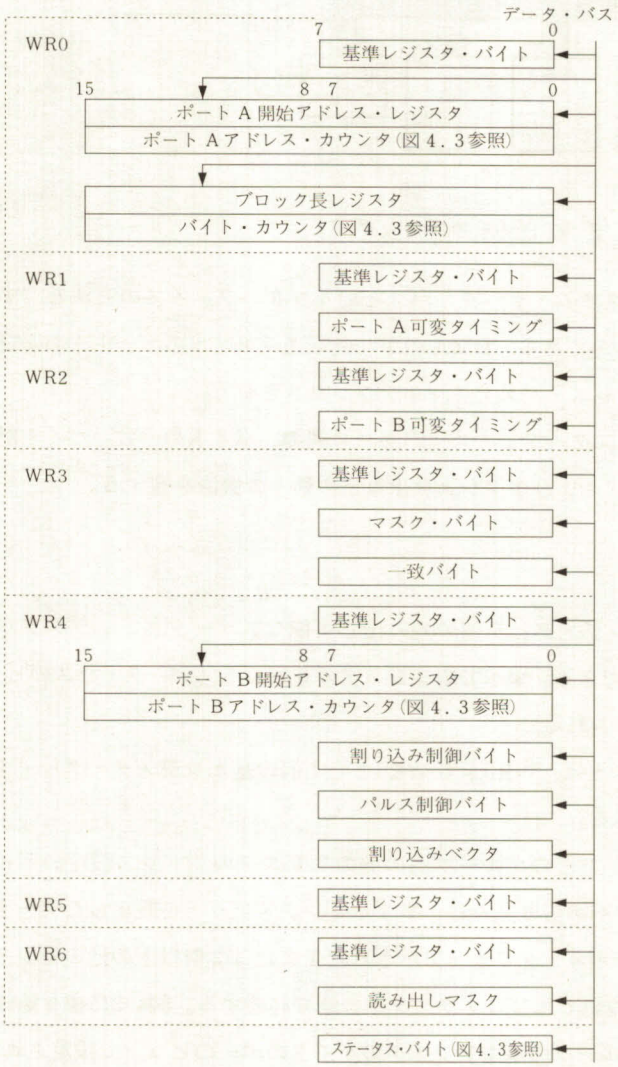


図4.2　書き込みレジスタ

図4.2のパルス制御バイト（WR4 グループ内）もまたWR0 内のバイト・カウンタと関係を有する。0 から255までのオフセット値をロードすることができ、この値はバイト・カウンタの下位バイトとつねに比較される。$\overline{\text{INT}}$ は一致が成立するごとにパルスを発生するが、これはイニシャル・オフセットから転送またはサーチの256バイトごとである。$\overline{\text{INT}}$ に発生するパルス信号はDMA がシステム・バスを制御しているときにのみ発生するので（例：$\overline{\text{BUSRQ}}$ および $\overline{\text{BUSAK}}$ が同時にアクティブのとき）、CPU はこれを見ることなく外部ゲート、カウンタ、または他のデバイスに向けられる。

図4.3は、データ・バスを通じて読み出し可能な7個のステータス・レジスタを示したものである。書き込みレジスタと異なり、ステータス・レジスタには2次レベルのレジスタまたはグループは含まれない。これらのレジスタは、WR6 グループに書き込まれた読み出しマスクに従ってシーケンシャルにアクセスされる。ただしステータス・バイトは他の読み出しレジスタとは別に読み出し可能である。

各ポートのアドレスは、プログラムされたその開始アドレスに固定されるか、あるいはアドレス・カウンタによってプログラムされた開始アドレスからインクリメントまたはデクリメントされる。DMA にプログラム

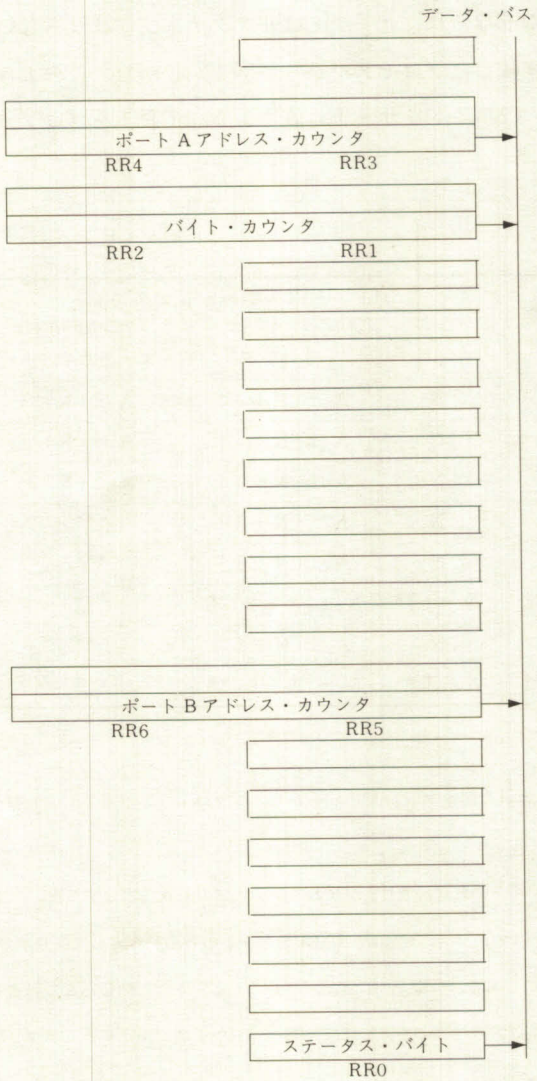


図4.3　読み出しレジスタ

されたブロック長はバイト・カウンタと比較される。これは0からスタートして各バイト動作の完了とともにインクリメントされる（図2.3）。

DMAはデータの読み出しに高速バッファリングまたはパイプライン方式を使う。データの転送およびエンド・オブ・ブロックで停止後、このパイプライン方式ではブロック長レジスタにプログラムされているものより1回だけ多い転送が実行される。この唯一の例外は2サイクル可変タイミングを使う同時転送の場合で、このときはRDYがそのままアクティブであればさらに2バイトが転送される。

表4.1は、エンド・オブ・ブロックでの停止または割り込み（まず停止したのちの割り込みを意味する）を含んだ転送の種々のクラス、およびモードにおけるカウンタの内容を示すものである。

バイトの一致で停止するようにプログラムされたサーチおよび転送/サーチ動作は、表4.2に示すように多少異なった動きをする。一致は次のバイトが読み込まれてはじめて発見される。転送/サーチ動作のすべてのクラスにおいて、転送されるのは一致したバイトである。しかし、同時転送/サーチ動作ではさらにもう1つのバイトが転送されるのが普通である。この唯一の例外は、バーストおよび連続モードにおいてバイトの一致の位置を探している間にRDYが非アクティブになるときである。バーストまたは連続モードでの同時転送/サーチについて見れば、これは一般に問題とならない。なぜならば、サーチとは非アクティブにはならない強制レディ条件またはRDYを用いてメモリ内で行われる連続したプロセスだからである。しかし、このようなケースが実際に起こるような場合は、一致が成立するとき2バイトをサーチし直すようにCPUをプログラムすることができる。

クラス	モード	プログラムされたブロック長	停止時転送バイト	バイト・カウンタ	ソース・ポート・アドレス・カウンタ※	デスティネーション・ポートアドレス・カウンタ※
シーケンシャル転送	バイト	N	N＋1	N	A_S±(N＋1)	A_S±(N)
	バースト	N	N＋1	N	A_S±(N＋1)	A_S±(N)
	連続	N	N＋1	N	A_S±(N＋1)	A_S±(N)
同時転送	バイト	N	N＋1	N	A_S±(N＋1)	※※※
	バースト	N	N＋1	N	A_S±(N＋1)	※※※
			N＋2※※	N＋1※※	A_S±(N＋2)※※	※※※
	連続	N	N＋1	N	A_S±(N＋1)	※※※
			N＋2※※	N＋1※※	A_S±(N＋2)※※	※※※

※　アドレスは、転送目的の最初のアドレスがあるプログラムされた開始アドレス（A_S）からインクリメント（＋1）またはデクリメント（－1）することができる。

※※　2サイクル（可変タイミング）同時転送において、N＋1バイトの転送後RDYがなおアクティブのときに限り生じる。

※※※　同時転送では両ポートをともに可変にすることはできない。このクラスの動作は、DMAのサーチのみの動作としてプログラムされ、可変アドレスは、ソース・ポートとしてプログラムされたものである。実際上、DMAがソース・ポートだと信じ込んでいるものが本当にソース・ポートであったり、あるいはデスティネーションであったりする。これは外部ハードウェアが決定する（アプリケーションの章を参照）。

表4.1　エンド・オブ・ブロックによりDMA停止後のカウンタ内容（転送動作）

クラス	モード	一致成立バイト	停止時転送バイト（転送のとき）	バイト・カウンタ	ソース・ポート・アドレス・カウンタ※	デスティネーション・ポートアドレス・カウンタ※
シーケンシャル転送／サーチ	バイト	M	M	M－1	A_S±(M)	A_S±(M－1)
	バースト	M	M	M－1	A_S±(M)	A_S±(M－1)
	連続	M	M	M－1	A_S±(M)	A_S±(M－1)
サーチのみ、または同時転送／サーチ	バイト	M	M	M－1	A_S±(M)	※※※
	バースト	M	M＋1	M	A_S±(M＋1)	※※※
			M※※	M－1※※	A_S±(M)※※	※※※
	連続	M	M＋1	M	A_S±(M＋1)※※	※※※
			M※※	M－1※※	A_S±(M)※※	※※※

※　アドレスは転送またはサーチの最初のアドレスであるプログラムされた開始アドレス（A_S）から、インクリメント（＋1）またはデクリメント（－1）することができる。

※※　動作の最後のサイクルが始まる直前にRDYが非アクティブとなるときに限って生じる。

※※※　サーチのみのクラスにはデスティネーション・ポートはない。同時転送／サーチの場合、両ポートを可変にはできない。この動作クラスはDMAのサーチのみの動作としてプログラムされ、可変アドレスはソース・ポートしてプログラムされたものに帰する。実際上、DMAがソース・ポートと信じ込んでいるものが本当のソースであったり、あるいはデスティネーションであることがある。これは外部ハードウェアが決定する。

表4.2　バイトの一致によりDMA停止後のカウンタ内容（サーチまたは転送／サーチ動作）

4.3　バス制御

DMAがシステム・バスを制御することによってデータを転送あるいはサーチする方法は、Z-80 CPUが読み出しまたは書き込みサイクルを行うためにシステム・バスを制御するのと全く同じである。特に、DMAは次の線を制御する。

- ○　アドレス・バス（16ビット）
- ○　データ・バス（8ビット）
- ○　$\overline{\text{IORQ}}$
- ○　$\overline{\text{MREQ}}$
- ○　$\overline{\text{RD}}$
- ○　$\overline{\text{WR}}$

さらに、$\overline{\text{CE}}$/$\overline{\text{WAIT}}$を介してDMAが$\overline{\text{WAIT}}$を監視するようにプログラムできる。

DMAがバスを要求し、これをCPUから受け取ったとき、システム上の他のデバイスはこの変化を感知することはできない。この間、CPUはメモリから命令をフェッチすることができないので完全にアイドル状態となる。

バス要求： DMA が CPU に対してバス要求するためには 2 つのイネーブル条件が必要である。

(1) CPU からのイネーブル・コマンド

(2) アクティブ・レディ条件

アクティブ・レディ条件は、入出力デバイスからの RDY がアクティブ、あるいは CPU からの強制レディ・コマンドによって生じる。

DMA は $\overline{\text{BUSRQ}}$ を“Low”にすることでバスを要求する。CPU はこのバス要求に必ず、そしてすばやく反応する。その時間は 1 マシン・サイクル（3～10クロック・サイクル）と 1 クロック・サイクルを加えたもので、これは $\overline{\text{BUSAK}}$ を DMA の $\overline{\text{BAI}}$ への入力として“Low”にすることによって行う。DMA がバスを保持している間、DMA の $\overline{\text{BUSRQ}}$ 出力も CPU の $\overline{\text{BUSAK}}$ もともに“Low”のままである。

DMA の $\overline{\text{BUSRQ}}$ が“High”になるとバスは CPU に解放される。そして CPU の $\overline{\text{BUSAK}}$ は次のクロック・サイクルで“High”になる。DMA がその $\overline{\text{BUSRQ}}$ を“High”にするのは、次を含む種々な条件のときである。

- シングル・バイト転送の完了（バイト・モード）
- RDY が非アクティブとなる（バイトおよびバースト・モード）
- 一致成立時の停止とプログラムされているときのバイトの一致（バーストまたは連続モード）
- エンド・オブ・ブロック時の停止とプログラムされているときのブロックの終わり（全モード）

これらの条件はそれぞれ異なった特性をもっており、タイミングの章で説明する。CPU は DMA から割り込みサービス中、DMA はバス要求は出せない。これは割り込みサービス（IUS）ラッチによって防止されるが、これについては後述する。

バス要求デージー・チェーン： 複数個の DMA を使用する場合、バス要求には優先順位によってデージー・チェーンに結線することができる。この方法を図4.4に示す。

各 DMA の $\overline{\text{BUSRQ}}$ は双方向である。出力として使うときはバス要求を行う。入力として使うときは、デージー・チェーン内の他の DMA のバス要求（$\overline{\text{BUSRQ}}$が“Low”になる）を感知し、その DMA の動作完了までこの DMA のバス要求を待たせる。バスを有している DMA はつねにその動作を完了できる。優先順位の高い DMA もこの期間中はバスを強制排除できない。

デージー・チェーン内の DMA 間の優先順位は、CPU からの距離によって決定される。CPU に電気的に最も近い DMA（$\overline{\text{BUSAK}}$/$\overline{\text{BAI}}$ で規制）の優先順位が最も高位である。優先順位が問題となるのは、複数の DMA が同じクロック・サイクル期間にバスを要求するときのみである。このようにして高位の優先順位の

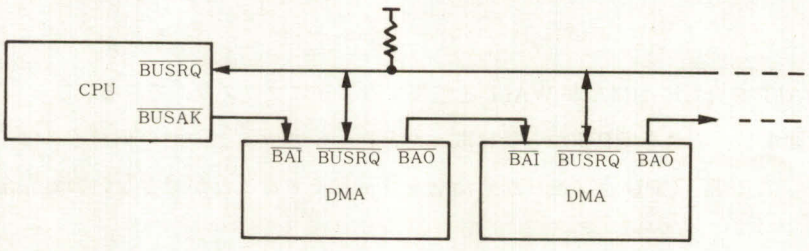


図4.4　バス要求のためのデージー・チェーン

DMA は、より低位の DMA が $\overline{BAI}/\overline{BAO}$ チェーンを通じて $\overline{BUSAK}$ を受け取るのを防ぐ。低位の DMA は、高位の DMA の動作が完了してバスを解放するまでその $\overline{BUSRQ}$ を“Low”に保ち、低位の DMA は即座にバス要求ができるよう準備しているのである。

4.4 割り込み

条件と方法： Z-80 CPU は下記の優先順位に従って外部の事象に応じる。

- バス要求（$\overline{BUSRQ}$）
- ノンマスカブル割り込み（$\overline{NMI}$）
- マスカブル割り込み（$\overline{INT}$）

バス要求（$\overline{BUSRQ}$）に加えて、DMA は通常マスカブル割り込み（$\overline{INT}$）を使用し、これは CPU のモード 2 で使用される。ノンマスカブル割り込み（$\overline{NMI}$）が使われるのは極端に優先順位の高い事象、たとえば電源異常信号などに使われる。これに関する詳しい説明はシャープ Z-80 テクニカルマニュアル（No. 451）第 8 章に記してある。

DMA は、下記の条件において CPU への割り込みをプログラムできる。

- DMA の RDY がアクティブになった後で、DMA がバス要求を出す前（RDY に対する割り込み）。
- エンド・オブ・ブロックで、バイト・カウンタの内容がブロック長レジスタの内容と一致した場合。
- バイトの一致で、マスク・バイト・レジスタによるマスキングの後、一致バイト・レジスタの内容が転送またはサーチ期間のデータ・バイトに対応する場合。

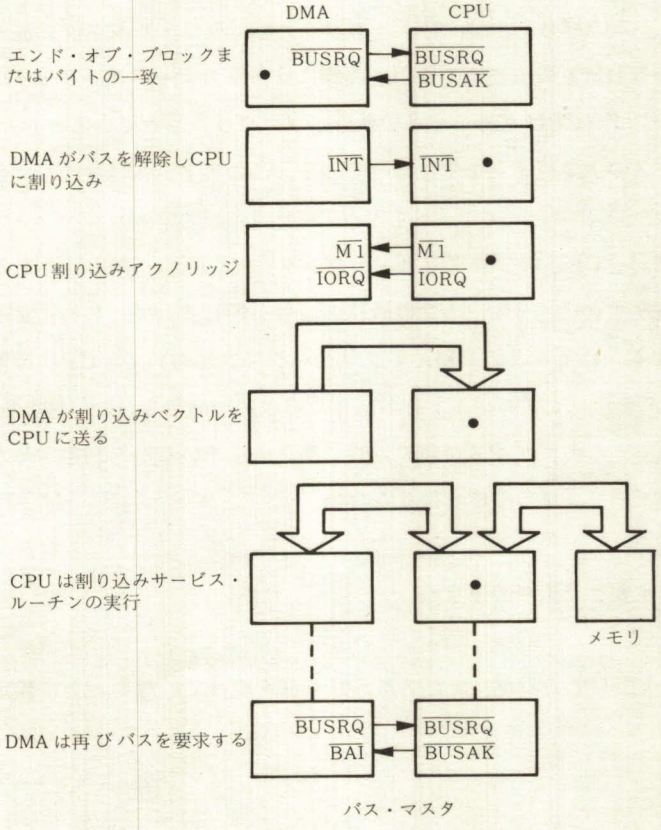


図4.5 Z-80 割り込みシーケンス

CPUに割り込みを発生する場合、DMAはバスを解放しなければならない。これは、DMAがバスを有しているときの$\overline{INT}$上の信号は、外部デバイスへの周期的パルス発生に使われるものであり、Z-80 CPUには感知されないからである。そのため、エンド・オブ・ブロックまたはバイトの一致による停止後、DMAはCPUに割り込むまえにまずバスを解放する。これを図4.5に示す。

DMAが、エンド・オブ・ブロックで割り込み、同様にエンド・オブ・ブロックで自動リスタートするようにプログラムされている場合、割り込みは各エンド・オブ・ブロックで発生する（そして連続動作のためにアクノリッジされる)。しかし、自動リスタートにプログラムした場合、エンド・オブ・ブロックのステータス・ビットは設定されない。そのため、割り込みベクトルはその割り込みの原因を反映することはできない（すなわちステータス・アフェクト・ベクトルは効果がない)。

Z-80 CPUは、1マシン・サイクルの間$\overline{M1}$と$\overline{IORQ}$を“Low”にして割り込みをアクノリッジする（タイミングの章を参照)。これによってDMAは8ビットの割り込みベクトルをデータ・バスに乗せ、どのDMAからの割り込みかがわかる。

さらに、オプションとしてこの割り込みの原因も判明する。CPUはこのベクトルを割り込みサービス・ルーチンへのアクセスに使用し、割り込みサービス・ルーチンが実行される。割り込みサービス・ルーチンはDMAを再びイネーブルしてバス要求を出させ、再び割り込みを行わせる。

割り込みアクノリッジあるいは何らかの形で判定する手段をもっていないCPUの場合、DMAに制御バイトを書き込み（通常は割り込みサービス・ルーチン)、同様な機能をもたせることができる。

割り込みベクトル： Z-80 CPUの割り込みアクノリッジ・サイクルによって、DMAは8ビットの割り込みベクトルをデータ・バスに送る（図4.6a)。このベクトルはCPUによってテンポラリ・レジスタに読み込まれる。これによって通常割り込みをしているDMAおよびその原因（実際にはあるステータス・ビットの現在の状態）が判明する。Z-80 CPUのIレジスタ（CPUがモード2の状態にプログラムされているとき）は、割り込みベクトルとしての16ビット・アドレスの上位バイトを受けもっており、このアドレスはメモリのジャンプ・テーブルのアドレスを指している。

メモリ内のジャンプ・テーブル・アドレスは、CPUのプログラム・カウンタ（図4.6b）へ読み出されるアドレスを含んでいる。このアドレスは割り込みサービス・ルーチンの最初の命令を指していて、これによって割り込みサービス・ルーチンの実行を開始する。ほとんどのDMAアプリケーションにおいて、CPUの割り込みサービス・ルーチンは、CPUのレジスタを介して制御バイトをDMAに書き込む命令を含む（図4.6c)。

割り込みベクトルを含まないCPUにおいて、どのデバイスが何の目的で割り込んでいるかを判定するためには、それぞれの周辺または外部レジスタをポーリングしなければならない。

割り込みラッチ： 割り込み構成には2つの主要ラッチが関連する。

割り込み保留（IP） DMAが割り込み要求を出しているが、まだアクノリッジを受けていないときに設定される。$\overline{INT}$を“Low”にする（図4.7)。

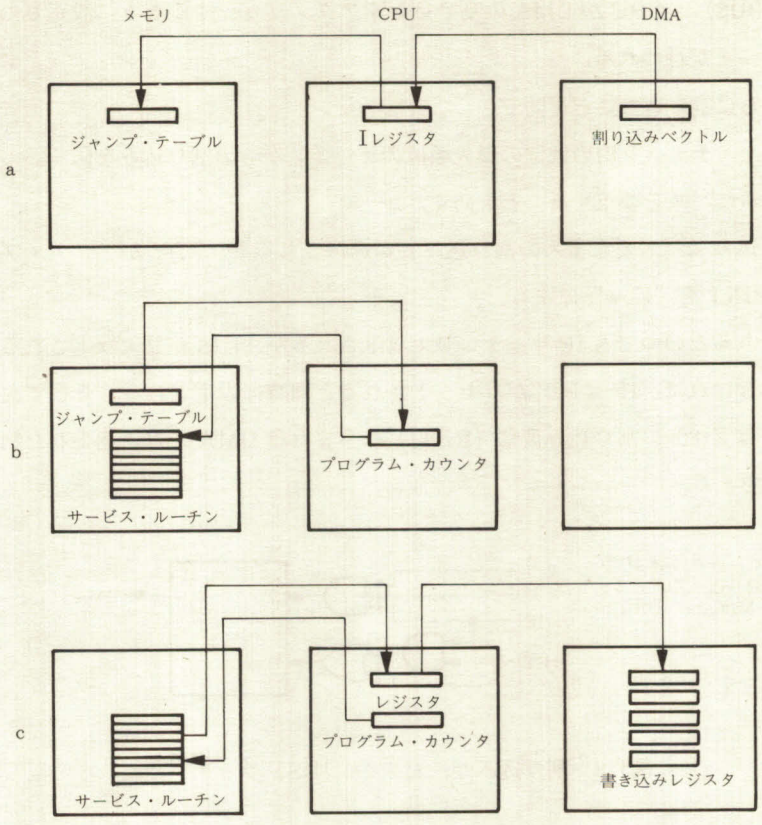
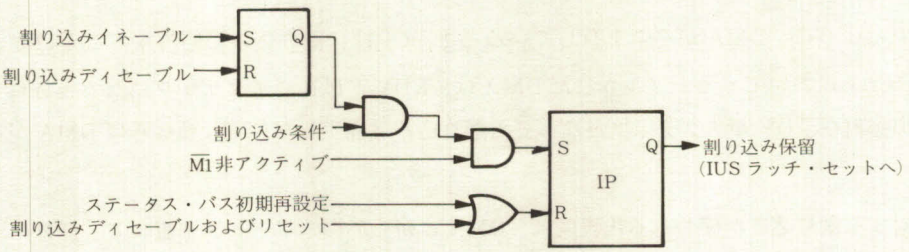


図4.6　Z-80 割り込みサービス・ルーチン



割り込み条件には、プログラムによりエンド・オブ・ブロック、
バイトの一致、または RDY アクティブを含む。

図4.7　割り込み保留（IP）ラッチ

割り込みサービス中（IUS）　CPU がDMA の割り込みをアクノリッジするときに設定される（図4.8）。これによって次の3つのことが行われる。

- ○　このDMA がさらに割り込むことを防ぐ。
- ○　割り込みデージー・チェーン内の低位の優先順位のデバイスからの割り込みを防ぐ。
- ○　このDMA がさらにバスを要求することを防ぐ。

レディに対する割り込み（バス要求まえの割り込み）が選択されると、RDY がアクティブになるとともにIP ラッチが設定され、$\overline{\text{INT}}$ を“Low”にする。

IUS ラッチが設定されるといつでもIP ラッチはリセットされるが、IUS がリセットされるまえに割り込みの発生条件がなくならなければ、IP はIUS がリセットされると同時に設定される。そして次の割り込みを発生させる。IUS ラッチはZ-80 の割り込み復帰（RETI）命令またはDMA に書き込まれた制御バイトによってリセットすることができる。

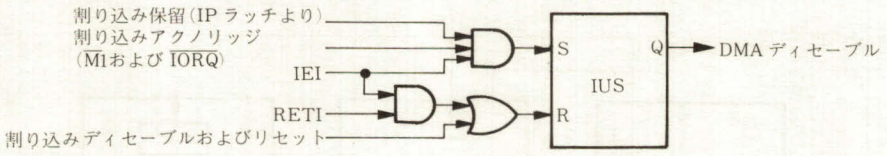
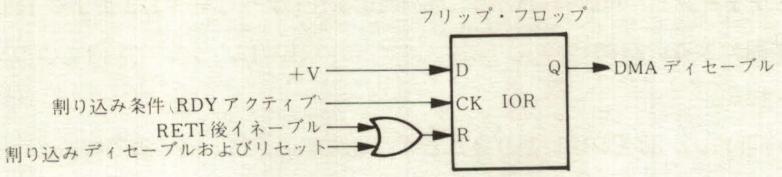


図4.8　割り込みサービス中（IUS）ラッチ

レディに対する割り込み：　通常入出力デバイスのRDY が非アクティブのとき、DMA がCPU によりイネーブルされる。そしてRDY がアクティブに変化することによって $\overline{\text{BUSRQ}}$ が“Low”になる。（図4.9）。クロックの立ち上がりエッジへのセットアップ時間が合っていると、2クロック・サイクル内で行われる。しかし、レディに対する割り込み（バス要求まえの割り込みとも呼ばれる）が選択されると、これは起こらない。この割り込みが使われたとき、RDY がアクティブになるとDMA はCPU に割り込む。CPU の割り込みサービス・ルーチンは、制御バイトをDMA に書き込み、これによってサービス・ルーチンの終了でDMA はバス要求を出す。

先にもふれたように、DMA がバスを有しているとき、CPU は割り込みに応答することはできない。このように連続モードにおいてイネーブルされたDMA は、RDY が最初にアクティブになるときにCPU に割り込むが、その後再びRDY がアクティブとなっても割り込みは発生しない（なぜならばDMA がバスを解放しないから）。

レディに対する割り込みが使われる代表例は、DMA に新しい開始アドレスを置いて、任意に決定されるメモリの一部へデータを転送するときである。



注：このラッチは、レディに対する割り込みオプションが選択されるときにのみ設定される。

図4.9　レディに対する割り込み（IOR）ラッチ

割り込みサービス・ルーチン：　DMA のモード設定（通常は電源投入および初期状態設定によって行う）の柔軟なプログラム性に加えて、種々の割り込みサービス・ルーチン用に設計された多数のコマンド（制御バイト）がある。次章プログラミングの中でこのコマンドについて詳述するが、理解を助けるためにここでも簡単に概説しておく。

割り込みサービス・ルーチンに制御バイトを使う代表的な機能には、下記が含まれる。

- DMA リセット
  - バス要求のために DMA をイネーブル
  - バス要求のために DMA をディセーブル
- リセットおよび DMA の割り込みをディセーブル
  - DMA の割り込みをイネーブル
  - DMA の割り込みをディセーブル
- 新しい開始アドレスおよびブロック長をロードする
  - 以前のアドレス計数を続行、ブロック長カウンタ・クリア
- 強制レディ条件
- ステータス・バイトの読み出し
  - ステータス・レジスタ読み出しシーケンスの開始
  - ステータス・クリア

Z-80 CPU における割り込みサービス・ルーチンは、つねに割り込み復帰命令（RETI）で終わる。

割り込み復帰：　割り込みサービス・ルーチンの終わりに、Z-80 CPU は割り込み復帰命令（RETI）を実行する。これは CPU を割り込みサービス・ルーチンから復帰させるものである。

DMA は、また RETI 命令を同時にデコードし、データ・バス上のデータを命令として認識するものである（DMA の $\overline{M1}$ 入力が“Low”のとき）。これによって DMA に少なくとも 1 つのこと、多くは 2 つのことが生じる。

- DMA 内の割り込みサービス中（IUS）ラッチをリセットし、その IEO が“High”となって低位の優先順位のデバイスの割り込みを可能とする。
- 再びバス要求できるように DMA をイネーブルする（これはレディに対する割り込みオプションで、DMA イネーブル制御バイトが使われているときにのみ生じる）。

Z-80 以外のシステムでは、制御バイトを使ってこれらの動作をシミュレートする。

割り込みデージー・チェーン： 複数のDMAは、図4.10に示すようにそのIEIおよびIEOを使って、デージー・チェーンに結線できる。Z-80 ファミリにおいて、このIEI/IEOチェーン内でのDMAの位置によって優先順位が設定される。

複数の周辺機器が同時にZ-80 CPUに割り込むとき、優先順位が最も高い周辺機器（デージー・チェーン端部の＋5 Vに一番近い）がサービスを受ける。その割り込みベクトルを受けることによって、それがどの周辺機器であるかをCPUは知ることができる。IEI/IEOチェーンがデータ・バス上に割り込みベクトルを乗せることを許可するのは、優先順位の最も高い割り込み周辺機器のみだからである。割り込みベクトルをもっていないZ-80 CPU以外のシステムにおいては、すべての周辺機器のステータスを読み続けることによって、どの周辺機器が割り込みを得たかを判別できる。

優先順位を得るには、デバイスのIEIが“High”でなければならない。そのデバイスがサービスを受けるには、そのIEOを“Low”にして他の下位のデバイスの割り込みを防がなければならない。チェーン内の次のデバイスはこの“Low”条件をそのIEOを“Low”にすることによって他の下位のデバイスに伝え、さらにそれを次々と伝えて行く。

割り込みがアクノリッジされると（図4.5）、そのCPUの割り込み構成はディセーブルされる。他のデバイスが再び割り込むためには、割り込みイネーブル命令によって再びイネーブルされなければならない。これは、通常割り込みサービス・ルーチンの中で行われる。サービス・ルーチンのはじめの方でこれが行われる場合、より高位の優先順位をもつ周辺機器は、CPUがまだそのサービス・ルーチンを実行中でもCPUに割り込むことが可能である。こうして、ネスト構造の割り込みが許可され、高位の優先順位の周辺機器はより低位の周辺機器のサービス・ルーチンの実行を中断する。

バス要求を行うデージー・チェーンには、この強制排除またはネスティング機能をもっていない。そのかわり、バスを得ることができる周辺機器は、そのタスクが完了するまでバスを保持できる。

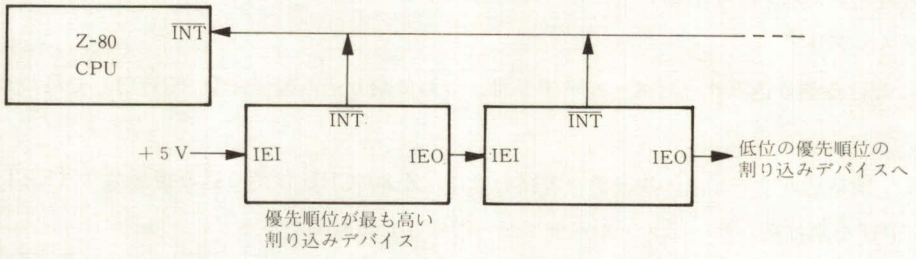


図4.10　割り込みデージー・チェーン

サービス要求のポーリング： CPUが割り込みを直接感知できないときは、図4.11に示すようにCPUは外部ゲートで$\overline{INT}$をポーリングすることができる。

ポーリングは下記のように行われる。

- ○　制御バイトによりDMAの割り込み構成をイネーブルする。
- ○　ステータス・ビットのポーリングにより割り込み要求を知る。

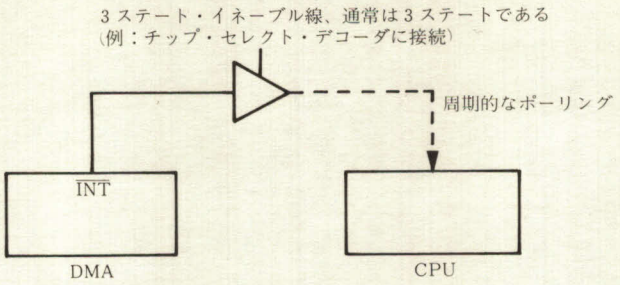


図4.11　サービス要求ビットのポーリング

第5章　プログラミング

5.1 概　　要

DMA を使用するにはまずプログラムすることが必要である。電源投入後において、その制御レジスタには有用なディフォルト値は含まれていない。

電源投入後プログラミングによってリセット命令、基本動作モード、その後コマンドが順次 DMA に書き込まれる。これは一般に、ブロック転送後またはサーチ後、ステータスの読み出し、開始アドレスの変更、および割り込みとバス要求論理回路の再イネーブルなどの目的のためのサービス・ルーチンの中で行われる。

Z-80 DMA にはプログラム可能な基本的な状態が２つある。

イネーブル状態：　システム・バスの制御を獲得して、両ポート間のデータ転送またはシングル・ポートからのデータ・サーチができる。

ディセーブル状態：　ここではバス要求もデータ転送もできない。

表5.1にこれらの状態およびそのサブ状態を詳述する。DMA は、電源投入後リセット命令を書き込むことによって自動的にディセーブル状態になる。CPU から DMA へのプログラム・コマンドは、イネーブル／非

	ディセーブル	イネーブル		
		非アクティブ（停止）	アクティブ	
			中　断	動作中
DMAの状態	DMAはバス要求ができない（CPUへの$\overline{\text{BUSRQ}}$入力を“Low”にできない）	DMAはバス要求ができ、この状態に入る直前にバスを得ることが可能であったが、現在はバスを有していない。	DMAがバスを有しているが、現在は動作していない。	DMAがバスを有しており、3つのモード(バイト、バースト、連続)のいずれかで転送／サーチ中である。
CPUがDMA制御バイトを書き込めるか、あるいはDMAステータス・バイトを読み出せるか？	Yes	Yes.（ただし、はじめにDMAディセーブル・コマンドの書き込みが必要）	No	No
この状態となる外部要因	電源断	●自動リスタートの場合を除くすべてのエンド・オブ・ブロック●すべての場合のバイト一致●バイトとバースト・モードにおけるRDYの非アクティブ●$\overline{\text{BAI}}$の非アクティブ●電源不充分	連続モードでRDY非アクティブ	●バースト・モードでRDYアクティブ、DMAイネーブル●CPUによりRETI命令フェッチでDMAイネーブルかつRDYアクティブ
この状態となるDMAコマンド（WR 6 制御バイト）	●DMAイネーブル・コマンド（および、他のコマンドが先行していないときはステータス・バイト初期状態設定コマンド）以外のコマンド●DMAディセーブル・コマンドは特にこの目的のために設計されている。	RDYが非アクティブで強制レディ・コマンドが使われていないときのDMAイネーブル	連続モードでRDYが非アクティブのときのDMAイネーブル	●RDYがアクティブあるいは強制レディ・コマンドが使用されているときのDMAイネーブル。そしてこのコマンドが割り込みサービス・ルーチンに含まれていないとき。

表5.1　DMA ステータス

アクティブ状態で書き込むことができるが、これによってDMAは自動的にディセーブル状態となり、CPUによってDMAイネーブル・コマンドがDMAの書き込みレジスタ6に書き込まれるまで維持される。

Z-80 ファミリにおいて、通常DMAはシステムの入出力領域内に周辺機器として存在する。この目的のためにアドレス・バスの下位バイトからその $\overline{CE}$ がデコードされ、すべての制御バイトおよびステータス・バイトが同じ入出力ポート・アドレスへの書き込み、あるいはこれからの読み出しとして行われる。このとき、OTIR（Z-80 CPUの命令）などの出力命令が使用される。

DMAを、メモリ・マップド入出力構成の中で使用することは可能であるが、これには何らかの外部論理回路が必要となる（アプリケーションの章で説明する）。メモリからDMAの内部レジスタに制御バイトを転送して、DMA自体をプログラムすることが不可能なためである。

Z-80 システムにおいてDMA割り込みベクトルが使用される場合、Z-80 CPUはモード2のマスカブル割り込みにプログラムしなければならない。

5.2 書き込みレジスタ

電源投入後の初期状態設定には、DMA内のすべての関係するレジスタに制御バイトを書き込むことが必要である。ここでは、制御バイトの書き込みができる書き込みレジスタ（WR0－WR6）のそれぞれについて説明する。WR6に書き込まれた制御バイトはコマンドを呼ばれることが多い。なぜならばCPUの割り込みサービス・ルーチンや、DMAの電源投入時の初期状態設定以外にシステム動作にもよく使われるからである。

内部構成の章に書き込みレジスタの組織概図を示し（図4.2）、そのアクセスの一般的な方法を説明する。すなわち、DMAの制御は、はじめに1つのバイトを基本レジスタに書き込むことによって識別されたレジスタ・グループ（WR0－WR6）に、1または2以上のデータを書き込むことによって行われる。すべてのグループは基本レジスタをもっており、ほとんどのグループにはさらに関連したレジスタをもっている。1つのグループ内の関連したレジスタは、まず基本レジスタにデータを書き込むことによって順番にアクセスされる。基本レジスタには、DMA機能制御のための制御ビットと、その基本レジスタ・グループ内の1または2以上の関連レジスタへのポインタ・ビットがある。

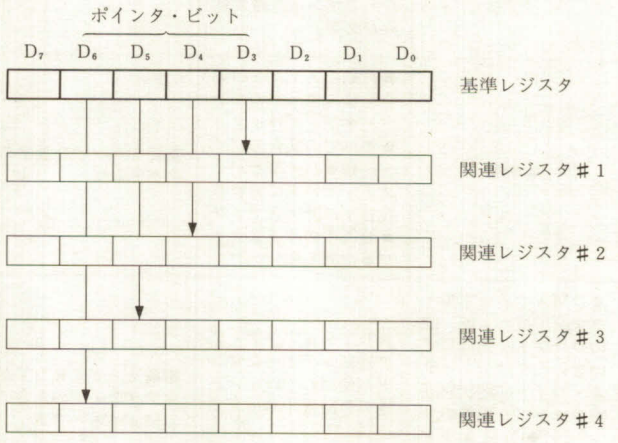


図5.1 書き込みレジスタ・ポインティングの方法

図5.1（WR0）はこれを示している。この図において、同じグループ内の関連レジスタへの書き込みのシーケンスを、関連レジスタの縦方向の位置によって示す。たとえば、DMA に書き込まれるバイトが WR0 を識別するビット（ビット D_0、D_1 および D_7）を含み、同時に関連レジスタ 2 および 4 を示すビット位置にポインタ・ビットを含んでいるとき、基本レジスタの後に DMA に書き込まれる次の 2 バイトは、この順にこれらの 2 つの関連レジスタ内に保持される。

図5.2および5.8に、7 個の基本レジスタとその関連したレジスタのそれぞれについて詳しく示す。図4.2と異なり、この 2 つの図には開始アドレスおよびブロック長レジスタに関連する16ビット・カウンタは含まれていない。

5.3 書き込みレジスタ 0 グループ

WR0 基本レジスタはビット 7 の 0 およびビット 0 と 1 の 0、0 以外の組み合わせで識別する（図5.2）。これは次の条件の設定に使用される。

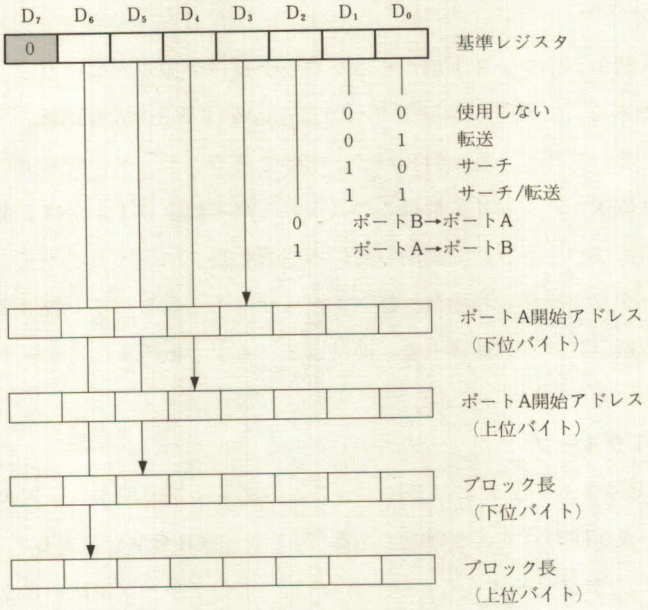


図5.2 書き込みレジスタ 0

動作クラス： ビット 1 と 0 でシーケンシャル転送（0、1）、サーチのみ（1、0）、またはシーケンシャル転送/サーチ（1、1）の動作クラスを設定する。同時転送または転送/サーチはサーチ（1、0）を選択し、外部ハードウェアで完全転送のための適切なバス制御信号を発生させることによって得られる（アプリケーションの章を参照）。

ソースおよびデスティネーション： ビット2はソース・ポート、およびシーケンシャル転送動作の場合はデスティネーション・ポートを宣言する。ビット2が0のときはポートBがソースであり、ビット2が1のときはポートAがソースである。サーチのみの動作ではソース・ポートだけである。同時転送または転送/サーチ動作のとき（クラスがサーチに設定されている）は、外部配線によってデスティネーションを決定する。

（注意） 転送方向を現設定から変更するのは、他の何らかのバイトをDMAに書き込むことによって、DMAをディセーブルした後にすること。これは、転送方向が変更されようとするときは、WR0がDMAに書き込まれる最初のレジスタとなってはならないことを意味する。

ポートA開始アドレス： ポートAがソースまたはデスティネーションとして使用される場合、その開始アドレスをプログラムすることが必要である。これは、基本レジスタのビット3および4を1にして、DMAに書き込んだ次の2つのバイトがポートA開始アドレスとなり、それぞれ下位および上位バイトとして認識される。このアドレスはWR1のビットと3－5のビット・パターンの内容に応じて各様に解決される。これらのビットによりこのアドレスはメモリまたは入出力、固定または可変、（可変の場合は）インクリメントおよびデクリメントの組み合わせとして指定される。ポートAが固定アドレス・デスティネーション・ポートの場合は、固定アドレス・デスティネーション・ポートの次の項を参照。

ブロック長： 電源投入時のディフォルト値からはブロック長は予知できないので、すべての動作にはブロック長の書き込みが必要である。これらのレジスタへの書き込みはWR0基本レジスタのポインタ・ビット5と6の設定によって行われる。ブロック長は64Kバイトまでである。データの読み出しはパイプライン方式のため、実際にサーチまたは転送されるバイト数はここに入れられる数より1または2多い。これについては、内部構成の章のアドレスおよびバイト計数で説明している（表4.1）。

ブロック長を零とプログラミングした場合、転送またはサーチされるバイト数は$2^{16}+1$バイトとなる。そのため、書き込める最も短いブロック長は1で、通常は2バイトの転送またはサーチとなる（表4.2）。

5.4 書き込みレジスタ1グループ

図5.3に示すように、ビット0－2および7によってこのグループの基本レジスタを識別する。このグループが使われるのはポートAが使われるときのみである(例：ポートBをソースとして、これでサーチ、同時転送、または同時転送/サーチにプログラムすることはできない)。このグループは次のような特性を指定する。

デバイスのタイプ（ポートA）： ビット3によってポートAをメモリまたは入出力として識別する。これによって正しい制御線（$\overline{\text{MREQ}}$または$\overline{\text{IORQ}}$）がこのポートを含むサイクルに対してアクティブとなる。

固定または可変アドレシング（ポートA）： ビット4と5によって、転送またはサーチされるデータの各バイトに対してポートAがインクリメントするか、デクリメントするか、あるいは固定かを指定する。動作においてデータの最初のバイトはWR0にポートAとして入れられる開始アドレスを使う。インクリメントまたはデクリメントはその動作の2番目のバイトによって開始する。

可変サイクル（ポートA）： ビット6が0の場合DMAの可変サイクル・タイミングは使われない。そのかわり、タイミングの章に述べるように、読み出しおよび書き込みサイクルのための標準Z-80 タイミングが使用される。ビット6が1のときは、WR1基本レジスタの後にDMAに書き込まれる次のバイトが、ポートAの可変タイミング・バイトとなる。これによってポートの読み出しおよび書き込みサイクルの長さを短くできる。$\overline{\text{IORQ}}$、$\overline{\text{MREQ}}$、$\overline{\text{RD}}$ および $\overline{\text{WR}}$ をアクティブにすることを含み、DMAの全体のサイクル・タイミングを選択するには、ビット1と0によって下記のように指定できる。

4クロック・サイクル

3クロック・サイクル

2クロック・サイクル

さらに可変タイミング・バイトのビット7、6、3および2によって、各線をビット1と0で指定するより1/2サイクル早く終わらせることができる。この効果についてはタイミングの章で詳述する。

シーケンシャル転送または転送/サーチにおいて可変サイクル・タイミングが使用されるときは、$\overline{\text{IORQ}}$ について特別な注意が必要となる。入出力ーメモリまたはメモリー入出力間転送の場合、メモリ・ポートはその $\overline{\text{IORQ}}$ が早く終了するようにプログラムしなければならない（通常は、$\overline{\text{IORQ}}$ はメモリとは関係ないものであるにもかかわらず）。入出力ー入出力間転送の場合、両ポートともその $\overline{\text{IORQ}}$ は早く終了することが必要である。

この状態が生じるのは可変タイミング仕様が採用されているとき、$\overline{\text{IORQ}}$ が他の制御線と異なるクロック・サイクル・エッジで論理レベルを変更するという事実からである。

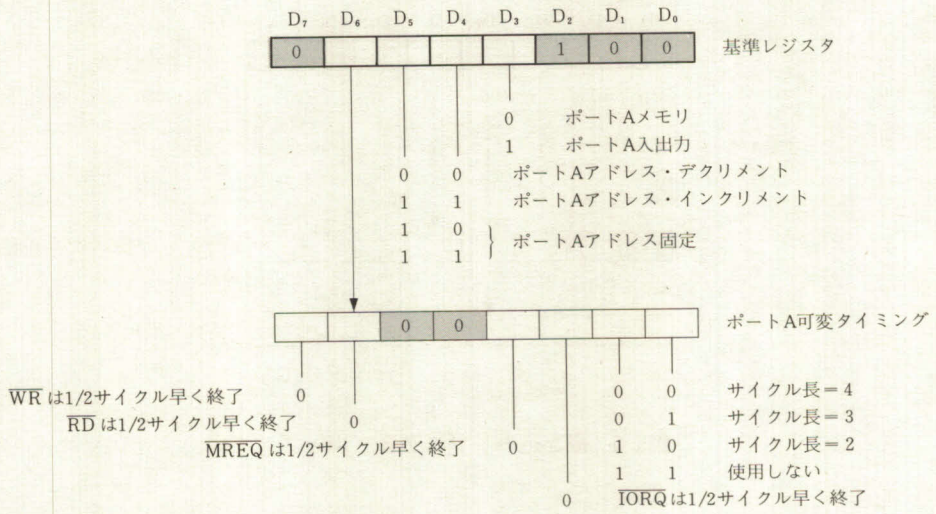


図5.3　書き込みレジスタ1グループ

5.5　書き込みレジスタ2グループ

図5.4に示すように、ビット0－2および7によってこのグループの基本レジスタを識別する。このグループが使われるのはポートBが使われるときのみである（例：ポートAをソースとしてサーチ、同時転送、同時転送/サーチにプログラムすることはできない）。書き込み方法はWR1と全く同じである。

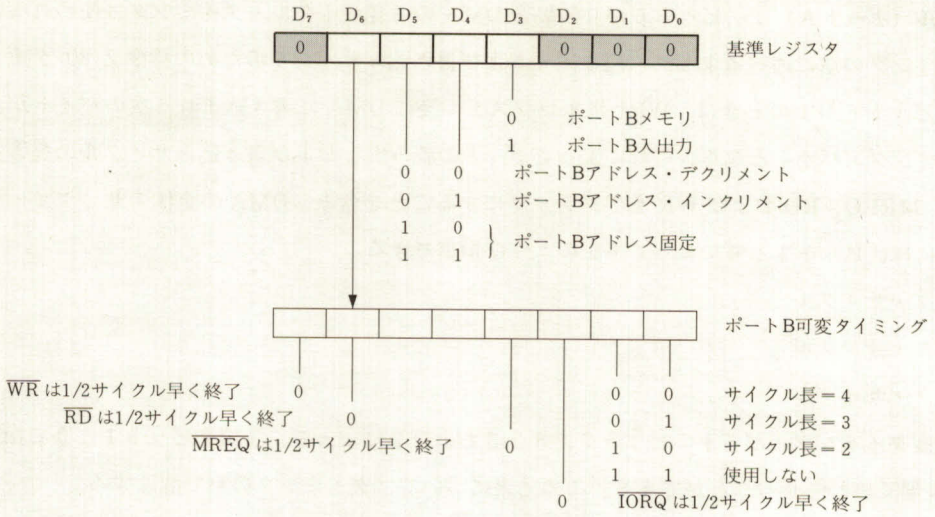


図5.4　書き込みレジスタ２グループ

5.6　書き込みレジスタ３グループ

図5.5に示すように、ビット０、１および７によってこのグループの基本レジスタを識別する。このグループは、主にサーチ動作における一致の成立による停止条件および一致バイト自体を示すのに使われる。これはバス要求および割り込みのイネーブルを１バイト早く行うことができる。このグループの各機能を次に記す。

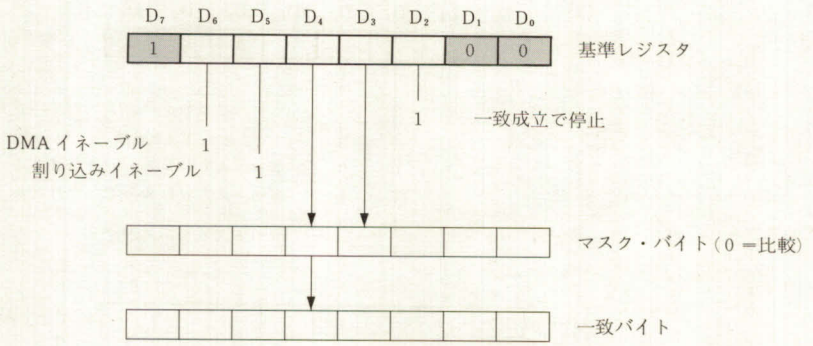


図5.5　書き込みレジスタ３グループ

一致成立による停止：　この基本レジスタのビット２を１に設定すると、データが一致バイト（後述）と一致したとき DMA は停止してバスを解放する。このビットを設定するときは、WR0 でサーチまたは転送/サーチ動作を指定しなければならない。このビットが零（一致成立で停止しない）のとき一致が成立すると、ステータス・フラグがステータス・バイトに設定され、一致に対する割り込みのオプションは残存する（WR4参照）。サーチを含まない同時転送を行うには、サーチでの一致成立に対する停止または割り込みは使用されない。

一致バイト： 基本レジスタのビット4を1に設定する場合、サーチされるすべてのデータと比較される一致バイトを指定しなければならない。次の機能に示されるように、どのビットを使用するにはWR0においてサーチ動作を指定する必要がある。

マスク・バイト： 基本レジスタのビット3を1に設定する場合、マスク・バイトを指定する必要がある。マスク・バイト中の1に設定されたビットの位置によって、無視されるべき一致バイト（前項）内の同じ位置のビットと比較が行われる。たとえば、マスク・バイトが00001111の場合、一致バイトの上位の4ビットのみがサーチされるデータと比較される。

割り込みイネーブル： 基本レジスタのビット5を1にすると、DMAの割り込み発生を可能にすることができる。この機能はWR6の割り込みイネーブル・コマンドと同じである。

DMAイネーブル： 基本レジスタのビット6を1にすると、DMAはイネーブルとなり、バスを要求できる。この機能はWR6のDMAイネーブル・コマンドと同じで、DMAがCPUからバスを奪うのを許可するまえにDMAに書き込まれる最後の制御バイトとして使用される。この目的のためにはDMAイネーブル・コマンドの方がよいことが多い。

5.7 書き込みレジスタ4グループ

図5.6に示すように、ビット0、1および7によってこのグループの基本レジスタを識別する。このグループは次のような特性を指定する。

動作モード： この基本レジスタのビット5および6によってバイト、バースト、連続の動作モードを指定する。これらのモードについては、図2.3、2.4、2.5、2.6、表4.1および4.2を参照のこと。

開始アドレス（ポートB）： ポートB開始アドレスは、この基本レジスタのビット2および3を1に設定して次の2バイトのデータで指定する。これはポートBが使われるときにのみ必要となり、WR0でこのポートをソース（あるいはデスティネーション）として宣言していれば、DMAが読み出す（あるいは書き込む）最初のアドレスとなる。ポートが固定アドレスでデスティネーションの場合、固定アドレス・デスティネーション・ポートの項を参照。

割り込み： 基本レジスタのビット4は割り込み制御バイトを示し、そのビット4および3はそれぞれ割り込みベクトルおよびパルス制御バイトを示す。割り込み制御バイトはまた次の3つのうちの2つ以上の割り込み条件を指定する。

- 一致成立に対する割り込み（ビット0）。一致成立に対する停止、あるいはエンド・オブ・ブロックに対する停止が同時にプログラムされている場合。
- エンド・オブ・ブロックに対する割り込み（ビット1）。エンド・オブ・ブロックに対する停止が同時にプログラムされている場合。

○　レディに対する割り込み（ビット６）。RDY がアクティブとなってバスを要求するまえの割り込み。

これらのビットを１に設定すると割り込み条件はイネーブルされるが、割り込み回路そのものはイネーブルされない。そのため WR6 の割り込みイネーブル・コマンドあるいは WR3 のビット５によって割り込み回路がイネーブルされる。割り込み制御バイトの関連ビットが零のときは割り込みは生じない。CPU に割り込むまえに DMA がバスを解放する（停止する）ので、図4.4および4.5がこれらの条件に当たる。

割り込みベクトル：　割り込みベクトルの書き込みは割り込み制御バイトのビット４で行う。さらに、割り込み制御バイトのビット５（ステータス・アフェクト・ベクトル）が１のとき、CPU の割り込みアクノリッジへの応答として、ベクトルがデータ・バスに乗せられるまえに、割り込みベクトルのビット１および２に割り込みの原因に応じて修飾される（例：RDY の状態またはステータスがラッチされる）。

自動リスタートおよびエンド・オブ・ブロックに対する割り込みがすでにプログラムされている場合、ステータス・アフェクト・ベクトルのモードを使用してはならない。この場合、各ブロックの終わりに送られる割り込みベクトルは、エンド・オブ・ブロックのステータスに応じて修飾できないからである。

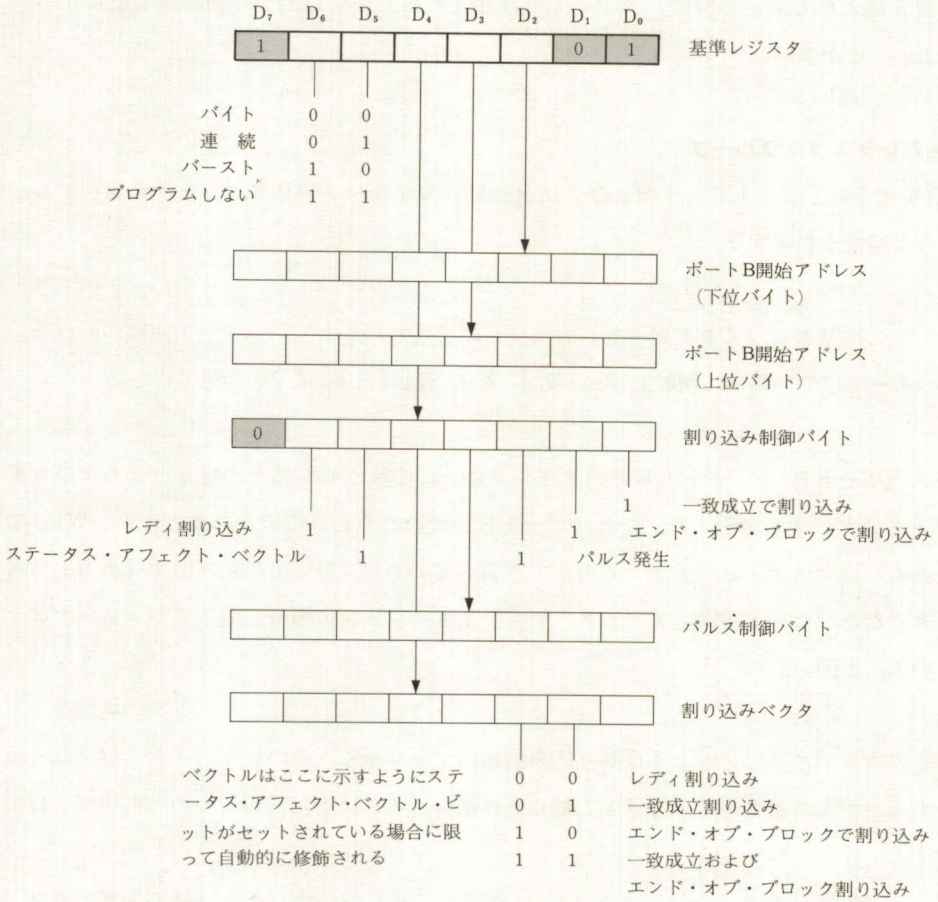


図5.6　書き込みレジスタ４グループ

パルス発生： パルス発生の原因となるのは次のようなものである。

(1) 基本レジスタによって割り込み制御バイトのビット4を1にする。

(2) 割り込み制御バイトのビット2および3を設定する。

(3) パルス制御バイトにオフセット値を書き込む。

パルス制御バイトはバイト・カウンタの下位バイトと比較され、一致が成立すると $\overline{INT}$ にパルスが発生する。その内容は、バイトのはじめのオフセット数の後で転送またはサーチに付き256バイトごとである。

5.8 書き込みレジスタ5グループ

図5.7に示すように、ビット0－2、6および7によってこのレジスタ・グループの基本レジスタを識別する。このバイトは次の特性の指定に使われる。

エンド・オブ・ブロックの動作： ビット5によってWR0にプログラムされたブロック長の終わりで停止（バス解除）またはオートリピートを指定する。ブロックの終わりで割り込みを起こすにはビット5(WR4)を零にする。新しいブロックで続行するには、DMAはエンド・オブ・ブロックのステータスをリセットする必要がある（自動リスタートでは、エンド・オブ・ブロック・ステータス・ビットもともにリセットされる）。

$\overline{CE}$/$\overline{WAIT}$ 使用： ビット4によってDMAの $\overline{CE}$/$\overline{WAIT}$ が次の2つのいずれかに使用されることを指定する。

(1) $\overline{CE}$ のみ。DMAがバスを有していないとき、$\overline{CE}$/$\overline{WAIT}$ はCPUの制御／ステータス・バイトの書き込みまたは読み出しのための $\overline{CE}$ としてのみ機能する(この線をアドレス・バスからデコードする方法についてはアプリケーションの章を参照)。

(2) $\overline{CE}$/$\overline{WAIT}$ マルチプレクス。この線はDMAがバスを有していないときは $\overline{CE}$ のみと同じ機能であるが、DMAがバスを有しているときは、外部論理回路からの外部 $\overline{WAIT}$ 入力を許可し、WR1とWR2の両方または片方にプログラムされているDMAサイクルを拡大することができる（このオプションのハードウェア・インターフェイスについては、アプリケーションの章を参照)。

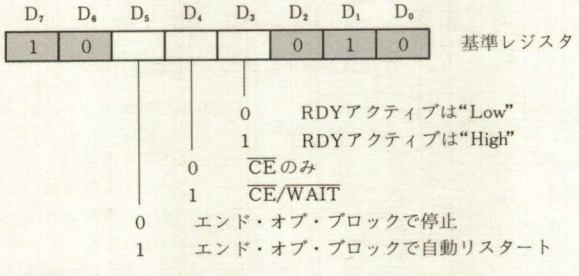


図5.7 書き込みレジスタ5グループ

RDYの状態： ビット3はRDYが“High”のときアクティブ、または“Low”のときアクティブとして、DMAがRDYの機能を識別する。これによって多様なデバイスへの柔軟性に富むインターフェイスが可能となる。

5.9 書き込みレジスタ6グループ

図5.8に示すように、このグループは基本レジスタのビット0、1および7を1に設定する。残りのビットはDMAの初期状態設定後に（例：CPU割り込みサービス・ルーチン内）よく使われる16のコマンドの指定や、読み出しレジスタの読み出しマスクの指示に使われる。

これらのコマンドは、DMAイネーブル・コマンドを除いてすべてDMAをディセーブルするものである。そのためDMAのバス要求開始まえの最後に書き込まれるコマンドは、DMAイネーブル・コマンドでなければならない。

リセット（C3）： このコマンドは電源投入時およびDMAのプログラムの打ち切り時に使用され、次のような機能をもつ。

○ 割り込みおよびバス要求回路ディセーブル。

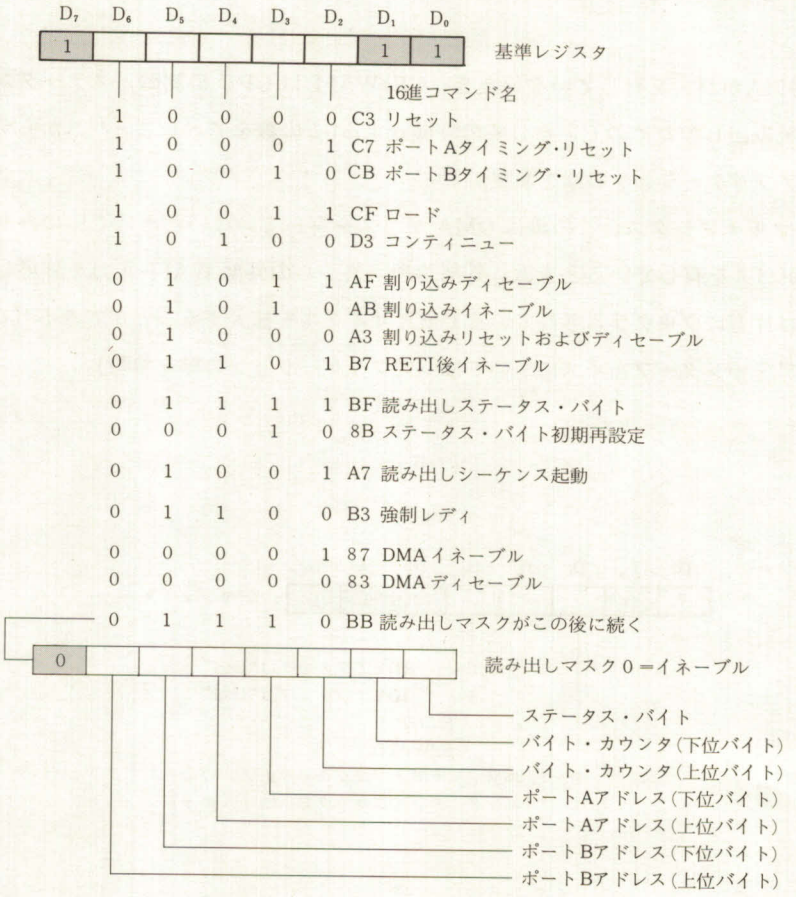


図5.8 書き込みレジスタ6グループ

○　割り込みラッチのリセット。

○　強制レディ条件の解除。

○　自動リスタート機能（WR5参照）のリセット。

○　ウェイト機能（WR5参照）のリセット。

○　ポートAおよびBを標準Z-80サイクル・タイミングに初期再設定（WR1およびWR2参照）。

電源投入時には初期設定プログラムのまえに、1個のリセット・コマンドをDMAに送ることが必要である。動作シーケンスの打ち切りには6個のリセット・コマンドによってリセットを行う(これはWR4には指示することが可能な関連レジスタが5個あることによる)。

リセット・コマンドによってDMAのリセットが完全に行われるものではない。たとえば、このコマンドでは読み出しシーケンスはリセットされず、これは読み出しシーケンス起動コマンドによってリセットされる。

ポートAタイミング・リセット（C7）：　WR1のポートA可変タイミングを標準Z-80タイミングにリセットする(リセット・コマンドにも同じ機能がある)。

ポートBタイミング・リセット（CB）：　ポートAタイミング・リセット（C7）に述べたように、WR2のポートB可変タイミングをリセットする。

ロード（CF）：　このコマンドはアドレス・レジスタ（WR0、WR4）に新しいアドレスを書き込む場合、あるいは同じアドレスで動作を再開するとき（自動リスタートを除く）に使われる。両方の開始アドレス・レジスタの内容をその関連アドレス・カウンタ（図4.2）にロードする。同様にブロック長レジスタに関連したバイト・カウンタをクリアし、内部強制レディ条件を解除する。開始アドレスがはじめの開始アドレスとは異なる場合、ロード・コマンドの書き込みのまえに開始アドレスをWR0，WR4の片方または両方に書き込むことが必要である。

すぐロードできるのはソース・ポート・アドレス・カウンタのみである。デスティネーション・ポート・アドレス・カウンタ（使われているとき）は、デスティネーション・ポート・アドレスのはじめのカウント中にロードされるが、固定されているときはロードされない。この特殊な状態については次の固定アドレス・デスティネーション・ポートの項で述べる。

ロード・コマンドの書き込みのときにDMAが非アクティブの状態（表5.1）となる場合、他のDMA制御バイトをロード・コマンドに先行して書かなければならない。DMAディセーブルなどの他のいずれのコマンドも、このために使うことができる。

ロードによって強制レディ条件が解除されることから、強制レディ・コマンドが使われるときはロードがこれに先行しなければならない。

コンティニュー（D3）：　このコマンドはバイト・カウンタを零にクリアするが、両ポートのアドレス・カウンタは現在の内容を保持する。DMAイネーブル・コマンドの後バイト・カウンタがまた開始したとき、まえと同じところから転送またはサーチを続行する。

コンティニュー・コマンドが使われるのは、いくつかのブロックを転送する場合、各ブロックの転送の終了

を知りたい場合、そしてこれらのブロックをメモリ内の連続した場所に転送するときである。特に、各ブロックの終わりにおける割り込みが必要となる。割り込み後、次のブロックを転送するには、ロード・コマンドよりむしろこのコマンドを使用すること。コンティニュー・コマンドとともに新しいブロック長をWR0に入れることができる。

コンティニュー・コマンドが書き込まれる場合、またはDMAが非アクティブの状態（表5.1）にある場合、他のDMA制御バイトがコンティニュー・コマンドに先行しなければならない。DMAディセーブルのように他のどのコマンドでもこの目的のために使用することができる。

割り込みディセーブル（AF）： このコマンドは、Z-80 CPU以外のシステムにおいて、Z-80 CPUからDMAへの自動割り込みアクノリッジをシミュレートするときに使われる。DMAがZ-80以外のCPUに割り込む場合、そのサービス・ルーチンのはじめの方に、CPUはDMAに対して割り込みディセーブルを書き込む。これによって$\overline{INT}$は非アクティブになるが、ルーチンが続行中はDMAは次の割り込みを送ることはできない。ルーチンの終わり近くになって、CPUはDMAに割り込みイネーブルを書き込み、これによってDMAはイネーブルされ新しい割り込みを発生する。

割り込みをリセットおよびディセーブル・コマンドに比べると、このコマンドの方が範囲は狭い。割り込み保留（IP）および割り込みサービス中（IUS）ラッチをリセットしないからである。

割り込みイネーブル（AB）： 前項の割り込みディセーブルの説明を参照のこと。Z-80 CPUシステムでは、電源投入時にはじめから割り込み論理をイネーブルするために、このコマンドが使用される（DMAはこの回路がディセーブルされて起動する）。しかしこの機能は、RETI命令のフェッチ、およびDMAによるデコードによって行われるから、次の割り込みからはイネーブルする必要はない。唯一の例外は割り込みディセーブル・コマンドが使われる場合で、この場合はDMAの動作を再び開始するには割り込みイネーブル・コマンドも使うことが必要である。

割り込みを発生させるために選択された条件は、DMAがディセーブルされるときでもDMA内にラッチされる。そして割り込みが再びイネーブルされたとき、後の割り込みを発生させることができる。

割り込みイネーブル・コマンドは、DMAの構成が終わってステータス・バイトの初期再設定コマンドが書き込まれた後でなければ書き込んではならない。このコマンドはWR3のビット5に1を書き込むのと同じ効果をもつ。

割り込みリセットおよびディセーブル（A3）： このコマンドが使用されるのは、8080や8085のようなCPUシステムで、Z-80 CPUのように割り込みアクノリッジ機能はあってもRETI命令がない場合である。このコマンドによって次の4つのことが行われる。

- 割り込みサービス中（IUS）ラッチのリセット。
- 割り込み保留（IP）ラッチのリセット。
- 内部強性レディ条件の解除。
- DMAによる以後の割り込みをディセーブルする（割り込みディセーブル・コマンドと同様）。

上述したZ-80以外のシステムではこのコマンドは次のように使用される。DMAによる割り込みが受け付

けられアクノリッジされた後、割り込みベクトルがCPUに送られCPUはこれによりサービス・ルーチンに分岐する。CPUはこのサービス・ルーチンの終わり近くでその割り込み復帰命令を実行するまえに、割り込みリセットおよびディセーブル・コマンド、割り込みイネーブル・コマンド、DMAイネーブル・コマンドの順にDMAに書き込む。

このコマンドの以後に割り込みイネーブル・コマンドが続く場合、このコマンドはZ-80のRETI命令にかわる。これはZ-80システムでは必要ではない。割り込みリセットおよびディセーブル・コマンドは強制レディ条件を解除するから、強制レディ・コマンドが使われる場合は割り込みリセットおよびディセーブルがこれに先行することが必要である。

RETI後イネーブル（B7）： このコマンドはレディに対する割り込みモード（WR4にプログラムされている）で、DMAを動作するときのみ使用される。割り込みから復帰後DMAをイネーブルして再びバスを要求させる。このコマンドはつねにZ-80 CPUシステムにおいてレディに対する割り込み後、さらにバスを要求するために使用する。他のCPU、たとえば8080でも用られることがある。

レディに対する割り込み（IOR）ラッチは、その割り込みの期間中に設定される。このラッチによってRDYがアクティブになってから、RETI後イネーブル・コマンドによってラッチがリセットされるまで、DMAはバスの要求ができない（Z-80および他のいくつかのCPUにおいては、IORラッチとIUSラッチ部分とがオーバラップしてバス要求禁止を行う）。

Z-80 CPUの割り込みサービス・ルーチンでは、DMAコマンドおよびCPU命令の順序は次のとおりである。

```
    ⋮
RETI後イネーブル・コマンド
DMAイネーブル・コマンド
    ⋮
RETI命令
```

読み出しステータス・バイト（BF）： このコマンドによって次のCPUの読み出しサイクルは、ステータス・バイトのアクセスとなる。これについては読み出しレジスタの項で説明する。

他の読み出しレジスタの読み出しが行われている場合、このコマンドを発するまえに読み出しシーケンス(読み出しマスクによって定義されるように）が完了しなければならない。

ステータス・バイト初期再設定（8B）： このコマンドによってステータス・バイトのビット4および5を初期再設定する。その後のステータス・バイトは表5.2のようになる。

DMAディセーブルまたは他のすべてのコマンドは、エンド・オブ・ブロックまたはバイトの一致成立による停止後、ステータス・バイト初期再設定コマンドのまえに使用しなければならない。DMA内部のハードウェア・レースの状況から、ステータス・ビットの初期再設定によってDMAが停止した条件が取り除かれ、DMAがディセーブルされていない場合はすぐ再びバスを要求することがあり得る。(転送方向を変更する場合、この点からみるとWR6内のステータス・バイト初期再設定コマンドは、WR0バイトに似ている。DMA

ビット	値	意　　味
0	1／0	DMA 動作あり／なし
1	1／0	レディ線アクティブ／非アクティブ
2	X	未定義ビット
3	0／1	割り込み保留／保留でない
4	1	一致不成立
5	1	エンド・オブ・ブロックでない
6	X	未定義ビット
7	X	未定義ビット

表5.2　ステータス・バイト

のディセーブルを確実なものとするには、他の制御バイトをこれらの制御バイトに先行させることが必要である。）

割り込み保留ステータス（ビット３）の初期再設定は、割り込みのアクノリッジ、割り込み処理、割り込みリセットおよびディセーブル・コマンドの書き込みによって行うことができる。DMA 動作ステータス（ビット０）の初期再設定は、ロード・コマンドによって行える。

読み出しマスクが続く（BB）：　このコマンドは読み出しマスク（図5.8）を指す。これは、DMA に書き込まれる次の制御バイトが読み出しマスク・レジスタへ行くことを意味する。読み出しマスクは、読み出しレジスタ RR0 から RR6 の読み出しのための新しいシーケンスの設定に使用され、通常は DMA の電源投入時の初期状態設定の一部となる。

読み出しレジスタは、RR0 からはじまって RR6 で終わる固定シーケンスによって読み出される。しかし、このシーケンスに従って読み出すレジスタは、読み出しマスクをプログラムすることによって限定することができる。読み出しマスクは、読み出されるレジスタの関連ビット・ポジションを１にすることでプログラムできる。たとえば、読み出しマスクが00011001を含んでいるとき、次の読み出しレジスタは次の順序で読み出される。

ステータス・バイト（RR0）

ポートＡアドレス・カウンタの下位バイト（RR3）

ポートＡアドレス・カウンタの上位バイト（RR4）

一度読み出しマスクがプログラムされると、選択されたレジスタのうち最下位の順位のものからはじまるように初期状態設定が必要である。これは、読み出しシーケンス起動コマンドを用いて行う。

読み出しシーケンス起動（A7）：　このコマンドは読み出しシーケンス・ポインタ・コマンドを起動させ、DMA への次の CPU 読み出し命令により、読み出しマスクによって読み出し可能と指定された最初の（下位順位）読み出しレジスタをアクセスする。一度開始すると、読み出しマスクによって指示された読み出しシーケンスは、たとえばさらに読み出しシーケンスあるいは読み出しステータス・バイト・コマンドを出すまえに終了しなければならない。

レジスタは読み出しシーケンス起動コマンドの書き込み後直ちに読み出される必要はない。他のコマンド(読み出しシーケンス起動および読み出しステータス・バイトを除く）を書き込み、はじめの読み出しおよびそれに続く読み出しが実際に実行されるまえに、バス要求、バス解除のサイクルを行うことができる。

強制レディ（B3）： バーストまたは連続モードにおいて、このコマンドは外部のアクティブ・レディ信号に代わって内部のレディ条件をアクティブにする。このコマンドはメモリーメモリ間転送やメモリ・サーチでRDYが必要でないところで使用される。このコマンドを使用する場合、RDYアクティブの“High”や“Low”は考える必要はない。

強制レディ条件は次のコマンドおよび条件によって解除される。

- ○ リセット・コマンド
- ○ ロード・コマンド
- ○ 割り込みリセットおよびディセーブル・コマンド
- ○ エンド・オブ・ブロックによる終了
- ○ バイト一致成立による終了
- ○ DMAによるバス解放

DMAによるバス解放によってレディ条件が解除されるため、このコマンドではDMAが転送できるのはバイト・モードにおいて1バイトのみである。

DMAイネーブル（87）： このコマンドによってDMAはシステム・バスを要求でき、他のすべての条件が満たされている場合（例：RDYがアクティブの場合、または強制レディ条件が存在する場合）は、その動作を進めることができる。DMAをディセーブルしないのはこのコマンドおよびWR3のビット6だけである。DMAに書き込まれる他のすべての制御バイトは、自動的にDMAをディセーブルする。そのためDMAイネーブル・コマンドは、DMAへの他のバイトの書き込み、あるいはDMAからの他のバイトの読み出しの後、つねに最後のコマンドとして使用される。

このコマンドによってDMAのバス要求回路がイネーブルされる。これは割り込み回路へは影響せず、機能やラッチをリセットすることもない。このバス要求イネーブル機能はWR3のビット6と同じである。

割り込みサービス・ルーチンにおいて、DMAイネーブル・コマンドは、CPUがその割り込み復帰命令を実行するまえのDMAへの最後のコマンドでなければならない。

DMAディセーブル（83）： このコマンドはDMAのバス要求を禁止する。DMAのアクションを外部の事象、たとえば電源断などによる停止、エンド・オブ・ブロックによる停止、あるいはバイトの一致による停止などの後、ステータス・バイトを初期再設定するような特別な場合に使用される（ステータス・バイト初期再設定コマンドを参照)。

読み出しレジスタの読み出しは、まずDMAにコマンドを書き込み、その直後あるいは後で行われる。CPUの読み出しは入力命令（Z-80 CPUのINIRなど）を使って、入出力デバイスとしてDMAをアクセスすることによって行われる。

DMAに書き込まれるコマンドは次のいずれかである。

読み出しステータス・バイト： このコマンドによってはじめの読み出しレジスタであるステータス・バイトが、次のDMAに対するCPUの読み出しデータとなる。

読み出しシーケンス起動： このコマンドは読み出しマスクによって定義されるシーケンスに従う繰り返し可能な一連の読み出しへのアクセスを起動する。

これらのコマンドについてはまえにも説明してあり、図5.8には読み出しマスクを図示してある。そこでも述べたように、レジスタの読み出しはこれらの書き込みコマンド、あるいは同じ読み出しシーケンス内のレジスタをアクセスする他のCPUの読み出し命令と、時間的に接している必要はない。

この読み出しレジスタには、さらに2つのコマンドも関連している。

ステータス・バイト初期設定： このコマンドによってステータス・バイトのビット4および5は1として初期再設定される。

読み出しマスクが続く： このコマンドによって、読み出しマスクのプログラムが可能となる。

図4.3および図5.9には7つの読み出しレジスタのグループと書き込みレジスタとの関係が明確に示されている。読み出しレジスタには次のものが含まれる。

ステータス・バイト（RR0）： ステータス・バイトは、他の読み出しレジスタとは独立して読み出しが可能で、そのビットのうち2個は、エンド・オブ・ブロックおよび一致バイトを識別するために初期値再設定することができる。ステータス・バイト中のビットは、次のように定義される。

ビット0　最後のロード・コマンドの後、DMAがバス要求を出しているか否かを示す。1はYES、0はNOを示す。

ビット1　DMAのRDYがWR5のビット3によってアクティブと定義される信号入力を有するか否かを示す。1はRDYアクティブ、0はRDY非アクティブを示す。

ビット2　未定義

ビット3　割り込み保留（IP）ラッチの状態を示す。0は割り込みが保留となっていることを示す（割り込みがアクノリッジされていないときはDMAの$\overline{\text{INT}}$はアクティブである）。1は割り込み保留がないことを示す。

ビット4　0は最後のステータス・バイト・リセットあるいは初期値再設定コマンドの後、一致が成立していることを示す。1は一致が不成立であることを示す。一致がどこかで成立したかは図4.5を参照。

ビット5　0は最後のステータス・バイト・リセット、ロード、コンティニューあるいは初期値再設定の後エンド・オブ・ブロックに達したことを示す。1はエンド・オブ・ブロックに達していないことを示す。DMAが停止するときのカウンタ内容については表4.1を参照。

ビット6　未定義

ビット7　未定義

バイト・カウンタ（RR1, RR2）： この16ビット・カウンタはロード、コンティニュー、またはリセット・コマンドによってのみリセットされる。DMAが転送またはサーチを開始すると、各読み出しサイクルの終わりで1ずつバイト・カウンタはインクリメントし、バイト・カウンタはブロック長レジスタのプログラム内容と比較されて、エンド・オブ・ブロックを判定する。転送で読み出されたバイト数は書き込まれたバイト数とつねに等しい。これは、一度開始した転送は転送／サーチ動作中のバイトの一致で停止してもDMAはこれを完了するからである。

表4.1および4.2はいろいろなクラス、モード、および動作状況の中でデータのパイプライン動作が、転送またはサーチされるバイト数にいかに影響するかを示す。ほとんどの場合、エンド・オブ・ブロックで停止できる転送動作で転送されるバイト数は、プログラムされたブロック長より1つ多い。

パルス発生が使われているときは、WR4のパルス制御バイトの内容は、各バイトの転送後、バイト・カウンタの下位バイトと比較される。

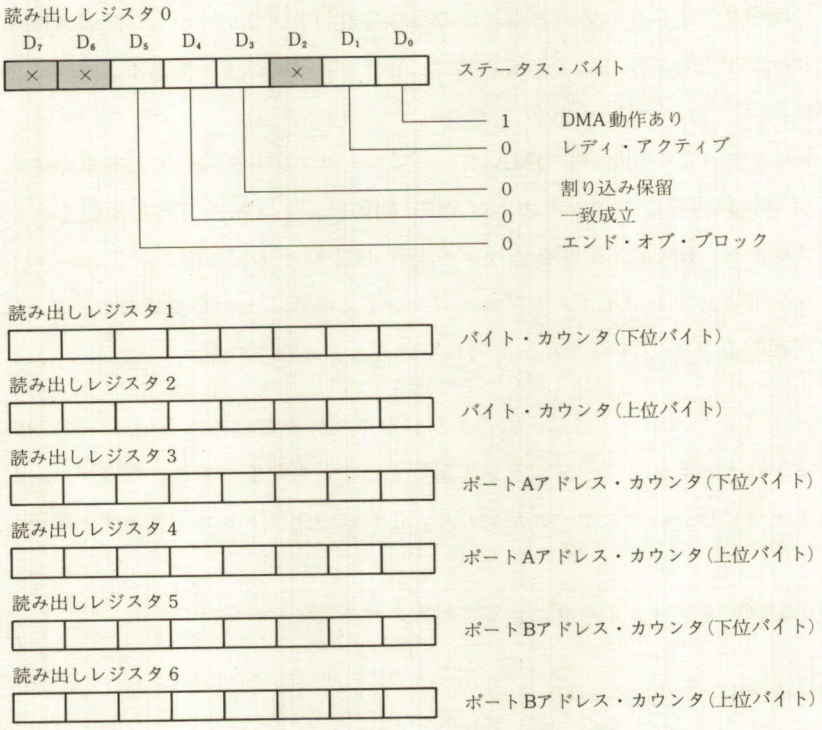


図5.9　読み出しレジスタ0～6

ポートAアドレス・カウンタ（RR3, RR4）： この16ビット・カウンタはWR0のポートA開始アドレス・レジスタからロード・コマンドによって書き込まれる。そしてWR1の指定内容によってインクリメント、デクリメント、または固定のまま残る。表4.1および4.2は、このカウンタがいろいろな転送またはサーチ条件をいかに読み出すかを示す。

ポートBアドレス・カウンタ（RR5，RR6）： このカウンタは上記のポートAアドレス・カウンタと全く類似しており、ポートAまたはポートBが固定アドレス・デスティネーション・ポートの場合、正しく機能させるには固定アドレス・デスティネーション・ポートに記するようにプログラムしなければならない。

5.10 プログラミング・シーケンスの復習

ここではDMAのプログラミングについてその一般的なケース、およびいろいろなアプリケーション、特殊ケースに対するルールを復習する。

DMA初期状態設定： DMAで使用されるすべてのレジスタは、電源投入時においてプログラムする必要がある。どのレジスタも有用なディフォルト値を有していない。これには割り込みのイネーブルおよびステータス・バイトの初期状態再設定の他に、クラスおよびモードの指定、ポートの指定、アドレスおよびブロック長の指定を含む各種機能の初期状態再設定が含まれる。

表5.2は、プログラム打ち切りによる初期状態設定あるいは初期状態再設定の一般的ケースについての制御バイトの書き込みの順序を示唆したものである。このうち特定のアプリケーションには無関係の制御バイトも含まれる。ここにコマンドとして示されているものは、WR 6の制御バイトである。すべてを書き込むと最大35の制御バイトがある。

DMAに書き込まれたすべての制御は、DMAイネーブル・コマンドを除いて、あるいは場合によってはステータス・バイト初期状態再設定コマンドおよびWR0制御バイト（転送方向を変更するとき）を除いて、DMAをディセーブルする。DMAの動作を続行する場合、CPUとDMA間のコミュニケーション後、最後に書き込まれるコマンドは必ずDMAイネーブル・コマンドであることが必要である。さらに、DMAとのコミュニケーションが起こるのは、CPUがバスを有しているときのみである。

ポート指定： ポートAまたはポートBは、同等のプログラム性を有しているから、図2.2に示すようにそのいずれをソースあるいはデスティネーションとして選択することも可能である。指定ポートが同じく固定アドレス・ポートであるときには、特別なケースが生じる。これは固定アドレス・デスティネーション・ポートの項で触れる。

ポート特性は次の制御バイト・グループによって指定される。

ポートA　WR0，WR1，WR6
ポートB　WR0，WR2，WR4，WR6

転送において転送方向（WR0のビット2）を変更するときは、WR0制御バイトの書き込みのまえに、他の制御バイトを先行させDMAがディセーブルであることを確認する必要がある。

アドレス・ローディング： 開始アドレスは、各ポートについてWR0（ポートA）およびWR4（ポートB）を通じて開始アドレス・レジスタに書き込まれる。ロード・コマンドによって開始アドレスはアドレス・カウンタにロードされる。アドレスはカウンタにロードされるまえにレジスタに書き込む。

新しいアドレスは、CPUがバスを有している場合、いつでも、転送と転送の間でも、あるいはDMAが自

初期状態設定／再設定シーケンス	Z-80 CPUの最大バイト数
DMA ディセーブル・コマンド	1
リセット・コマンド(マルチプル)	6
WR0 制御バイト	5
WR1 制御バイト	2
WR2 制御バイト	2
WR3 制御バイト	3
WR4 制御バイト	5
WR5 制御バイト	1
ポートAタイミング･リセット･コマンド	1
ポートBタイミング･リセット･コマンド	1
ロード・コマンド	1
ステータス・バイト初期再設定コマンド	1
読み出しマスクが続くコマンド	1
読み出しマスク制御バイト	1
読み出しシーケンス起動コマンド	1
強制レディ・コマンド	1
割り込みイネーブル・コマンド	1
DMA イネーブル・コマンド	1
	35

表5.3　制御バイト順位

動リスタート・モード（例：バイト転送間のバイト・モード）で動作中でも、アドレス・レジスタに書き込むことができる。しかし、自動リスタート・モード以外の場合、新しいアドレスは使用するまえにロードし直す必要がある。強制レディ条件を使うときは、ロード・コマンドが強制レディ・コマンドよりも先行しなければならない。

固定アドレス・デスティネーション・ポート：　デスティネーション・ポートを固定アドレスとしてプログラムする場合は特殊な状況が生じる。WR6 のロード・コマンドが固定アドレスをロードするのはソースとして選択されたポートであって、デスティネーションとして選択されたポートではない。そのため固定デスティネーション・アドレスをロードするには、一時的にそのポートをソース・ポートとして宣言する必要がある。真のソース・ポートは、その後ソース・ポートとして宣言され（これによって他のポートをデスティネーションとする)、真のソース・アドレスが次にロードされる。

可変アドレス・ソース（ポートA）から固定アドレス・デスティネーション（ポートB）への転送を想定すると、上記の方法は下記のようなステップで行われる。

1．ポートB（固定デスティネーション）アドレスを WR4 に書き込む。
2．ポートBを一時的にソースとして WR0（ビット2＝0）に書き込む。
3．ロード・コマンドによってポートBアドレスをロードする。
4．ポートA（可変ソース）開始アドレスを WR0 に書き込む。

5．ポートAをソースとしてWR0（ビット2＝1）に宣言する。

6．ロード・コマンドにより、ポートAアドレスをロードする。

7．DMAイネーブル・コマンドによりDMAをイネーブルする。

割り込み： 割り込みベクトル（WR4）はこれを使った転送の発生まえに書き込み、初期状態設定または再設定時に、割り込みイネーブル・コマンドによって割り込みをイネーブルにする。Z-80 CPUシステムにおいて、DMA初期状態設定後の割り込みサービス・ルーチンには、ルーチンの終わりに下記のコマンドが含まれるのが普通である。

エンド・オブ・ブロックまたはバイト一致割り込み
　　⋮
DMAイネーブル・コマンド
　　⋮
RETI命令
レディ割り込み（バス要求前）
　　⋮
RETI後イネーブル・コマンド
DMAイネーブル・コマンド
　　⋮
RETI命令

たとえばエンド・オブ・ブロックでの割り込みは、フロッピーディスクの読み出しによって行われることもある。ディスクから128バイト転送する場合、各レコードの終わりにDMAに割り込ませて、CPUにその終了を報告させることができる。そうすればCPUはデスティネーション（メモリ）アドレス・カウンタを読んで、最後のメモリ・ロケーションを見つけることができる（アドレス・カウンタ内容については図4.4参照）。連続したメモリ・ロケーションへの入力の続行のためのサービス・ルーチンは、コンティニュー、ステータス・バイト初期状態再設定、およびCPUの割り込み復帰まえのDMAイネーブル・コマンドを含むのが通常である。データ到着後のDMAの終了のためのサービス・ルーチンには、DMAディセーブル、およびステータス・バイト初期状態再設定コマンドが含まれる。DMAの転送が他のデバイスからの割り込みによって開始される場合、他のデバイスのためのサービス・ルーチンには、DMAのポート・アドレスに書き込まれたDMAイネーブル・コマンドが含まれる。

バイトの一致成立による割り込み（サーチ、または転送／サーチ動作）を、終了バイト、エラー・インジケータまたは他のキャラクタによっても割り込みが発生するようにできる。これによってCPUは一連のデータからこれらのキャラクタを探す必要が省略され、スループットを増大し、CPUのソフトウェアの複雑性を軽減することができる。たとえば通信システムにおいて、DMAはエンド・オブ・テキスト（ETX）またはキャリッジ・リターンをサーチし、完全なメッセージ・フレームが到着したとき、CPUに割り込みをかける。このためのサービス・ルーチンは、エンド・オブ・ブロックの割り込みのものと似ている。

レディ割り込みは多少異なっている。第1に、CPUが割り込みを見れるのはCPUがバスを有していると

きに限られるので（他のタイプの割り込みはバスが解放されるまで処理されない）、割り込みまえにDMAがバスを有することはできない。第2に、DMAをイネーブルするには、レディ割り込み後のサービス・ルーチンにおいて、RETI後イネーブル・コマンドを使用しなければならない。RDYがアクティブとなるときの割り込みの代表的な目的は、転送の行き先を考えるための時間をCPUに与えることである（サービス・ルーチン内で行う）。これはダイナミックなメモリ割り付けを使用したシステムの中で行われ、これによってメモリ割り付け効率が向上できる。たとえば、サービス・ルーチンの中でCPUはメモリ・デスティネーションのための新しい開始アドレスをDMAに書き込み、ロードする。このサービス・ルーチンの終了後はじめてDMAはイネーブルされて、バス要求が可能となる。DMAイネーブル・コマンドに先行するRETI後、イネーブル・コマンドによってレディ割り込みが最初に発生したときにセットされたラッチをリセットする。

Z-80 CPU以外のシステムにおいては、割り込みディセーブル、割り込みイネーブル、および割り込みリセットおよびディセーブル・コマンドが使用できる。これによってZ-80 CPUの割り込みアクノリッジ・サイクルおよび割り込み復帰命令がシミュレートできる。割り込みの実行および割り込みからの復帰には、DMAはこの2つの命令を実行しなければならない。

バイト一致（サーチ）： 一致成立による停止、または一致成立による停止および割り込みにおいて（WR3、WR4）、DMAを用いて他の動作を実行するには下記のコマンド・シーケンスを書き込む。

ロードまたはコンティニュー
ステータス・バイト初期状態再設定
DMAイネーブル

ステータス・バイト初期状態再設定コマンドには、他のコマンド（DMAイネーブル以外のコマンド）を先行させることが必要である（表4.1にバイト一致に対する停止時のいろいろなカウンタの内容を示す）。

エンド・オブ・ブロック： 停止後、またはエンド・オブ・ブロックによる停止および割り込み（WR4またはWR5）後、DMAを用いて他の動作を実行する必要がある場合、バイト一致（サーチ）に示したものと同じコマンド・シーケンスを書き込む。表4.1にエンド・オブ・ブロックによる停止時のいろいろなカウンタの内容を示す。

自動リスタート： はじめに入力したものと同じブロック長および開始アドレスを使って、繰り返し転送あるいはサーチを得るには、WR5のビット5＝1を含んでDMAを再設定する。アドレスのローディングおよびカウンタ・クリアは自動的に行われる。

バイト・モード、あるいはバースト・モード（RDYは場合によって解除されている）の場合も含めて、転送の期間中に（例：バス要求の後）その転送に影響することなく開始アドレスを変えることが可能である。これを行う転送の終了後、DMAは新しいアドレスをカウンタにロードし、中断することなく続行できる。

強制レディ条件： 強制レディ・コマンドが使われるのは、メモリーメモリ間転送またはサーチのみ動作で、入出力デバイスからのRDYが使われていない場合である。しかし、強制レディ・コマンドの書き込み後、この強制レディ条件を解放するDMAコマンドがいくつかある。そのためコマンドの入力シーケンスが重要で

ある。これについては、書き込みレジスタ6グループの強制レディ・コマンドの項で説明する。

パルス発生： 可変オフセット周期の後、256バイト間隔でパルスを発生するにはWR4グループのみを考えればよい。その$\overline{INT}$がパルス用として使われる。

可変タイミング： $\overline{RD}$、$\overline{WR}$、$\overline{MREQ}$、および$\overline{IORQ}$のタイミングは、WR1、WR2レジスタ・グループの片方または両方をプログラムすることによって各ポートに独立して変更できる。メモリー入出力、入出力ーメモリ、または入出力ー入出力間シーケンシャル転送、または転送／サーチのプログラミングでは特殊ケースが生じる。$\overline{IORQ}$のプログラミングには一定の方法が必要となる。WR1の可変サイクル（ポートA）の項を参照。

DMAイネーブル： 動作まえにDMAに最後に書き込むコマンドは、DMAイネーブル・コマンド、またはそれと同機能をもったWR3ビット6＝1である。DMAを動作させるコマンドはこれだけである。イネーブル時に他のすべての動作条件が満たされていれば（例：RDYアクティブ）、DMAは直ちに開始する。割り込みサービス・ルーチンにおいて、割り込み復帰命令のまえにDMAに最後に書き込まれるコマンドは、DMAイネーブル・コマンドである。サービス・ルーチンにおいて、RETI命令実行のまえに他の命令がDMAイネーブル・コマンドに続くこともあり、続くのが通常であるが、このようなコマンドはDMAに対するものではない。

読み出しステータス： CPUにDMAステータスを読み出させるには2つのコマンドを使用する。

ステータス・バイト読み出し： CPUの次のDMA読み出しにより、ステータス・バイトへのアクセスを行わせる。ステータス・バイトの読み出しにはその都度ステータス・バイト読み出しコマンドを書き込む。

読み出しシーケンスの起動： CPUは次のDMA読み出しにより、読み出しマスクによって読み出し可能と指定された最初のステータス・レジスタへのアクセスを行う。DMAはすべての指定された読み出し可能レジスタのシーケンスを実行しなければならない。レジスタのシーケンスの読み出しは、次のステータス・バイト読み出しまたは読み出しシーケンス起動コマンドのまえに完了しなければならない。

表5.4は、メモリ（ポートA）から周辺デバイス（ポートB）へのデータ転送のプログラムを示したものである。ここでポートAのメモリ開始アドレスは1050H、そしてポートBの周辺固定アドレスは05Hである。転送するデータ・バイト数は1001Hバイトである（ブロック長より1多い）。 DMAのコマンド・テーブルは連続したメモリ・ロケーションにストアし、Z-80 CPUのOTIR命令のような出力命令によってDMAへ転送することができる。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	16 進 数	説　　明
0	1 ブロック長(上位)の書き込み	1 ブロック長(下位)の書き込み	1 ポートA開始番地(上位)の書き込み	1 ポートA開始番地(下位)の書き込み	0 B→A*	0 転送のみ	1	79	WR0の設定、ブロック長、ポートA開始番地の書き込み指示、ポートBを一時的にソースとする
0 0 0 0	1 0 0 0	0 0 0 0	1 1 0 1	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0	50 10 00 10	ポートA下位番地 ポートA上位番地 ブロック長下位 ブロック長上位
0	0 タイミングの指定なし	0 アドレス可変	1 アドレスはインクリメント	0 ポートはメモリ	1 ポートAの指定	0	0	14	WR1の設定、ポートAは可変番地でインクリメント、メモリ
0	0 タイミングの指定なし	1 アドレス固定	0	1 ポートは入出力	0 ポートBの指定	0	0	28	WR2の設定、ポートBは固定番地
1	1　0 バースト・モード		0 割り込み制御の指定なし	0 ポートB上位番地書き込みなし	1 ポートB下位番地書き込み	0	1	C5	WR4の設定、バースト・モード、ポートBアドレス書き込みの指定
0	0	0	0	0	1	0	1	05	ポートB下位番地
1	0	0 オート・リスタートなし	0 WAIT 使用なし	1 RDYアクティブ"High"	0	1	0	8A	WR5の設定、RDYアクティブ"High"
1	1	0	0	1	1	1	1	CF	WR6の設定、ポートBアドレスのロード、バイト・カウンタのリセット
0	0　0 ブロック長の書き込みなし		0　0 ポートAの番地書き込みなし		1 A→B	0　1 転送のみ		05	WR0の設定、ポートAをソースとする
1	1	0	0 ロード	1	1	1	1	CF	WR6の設定、ポートAアドレスのロード、バイト・カウンタのリセット
1	0	0	0 DMAイネーブル	0	1	1	1	87	WR6の設定、DMAイネーブル

(注)　実際の転送バイト数はブロック長指定数より1多い。

＊　これらのエントリは固定デスティネーション・アドレスに限って必要。

表5.4　DMAプログラム例

第6章　アプリケーション

6.1　Z-80 DMA と Z-80 CPU

Z-80 ファミリとして Z-80 DMA の信号およびタイミングは、Z-80 CPU の信号およびタイミングと直接互換性を有するものである。バスを有している場合、DMA は Z-80 CPU と全く同一の読み出しおよび書き込みサイクル特性を示し、これによってシステム設計の単純化ができる。加えて可変タイミング仕様によって、システム設計者などはメモリおよび入出力デバイスを非標準的機能や要件などと、より容易にインターフェイスすることができる。DMA は性能を向上するためにその読み出しまたは書き込みサイクル・タイミングを短くしたり、より遅いデバイスの使用もできるように制御信号を合わせることができる。このような仕様はプログラムによって制御されるものであるから、サイクルおよび制御信号のタイミングの変更にもハードウェア構成は一定に保たれる。

DMA の接続：　小型システム、あるいは Z-80 DMA が CPU とボードを共用するシステムでは、DMA の端子の大部分は対応する CPU 端子に直接結線することができる。これにはアドレス・バス（A_0－A_{15}）、データ・バス（D_0－D_7）、$\overline{MREQ}$、$\overline{IORQ}$、$\overline{RD}$、および $\overline{WR}$ の制御信号が含まれる。$\overline{INT}$ および $\overline{BUSRQ}$ も他のデバイスからの対応するオープン・ドレイン出力と同様に、直接 CPU に結線できる。これらの機能の優先順位デージー・チェーンについては図4.4および図4.10に示してある。

電源、アース、クロック信号も同様に CPU と DMA に共通であるが、低インピーダンス・バスおよび適切なデカプリングには細心の注意が必要である。Z-80 ファミリのデバイスに要求される30 ns クロック遷移時間も考慮に値する。TTL クロック・ドライバ出力からの300Ωプルアップ抵抗は、2.5MHz で作動する小型システムの動作には充分であるが、大型の高性能システムで負荷およびスピードを増加するには、アクティブなプルアップが必要となる。Z-80／Z-8000 システム用のコンプリメンタリのトランジスタ・ドライブの一例を図6.1に示す。

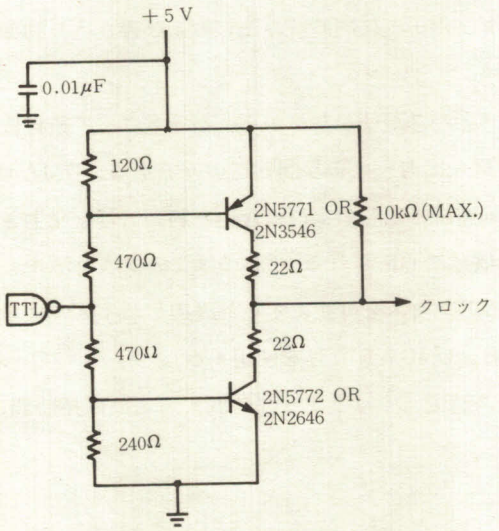


図6.1　Z-80／Z-8000 クロック・ドライバ

チップ選択およびイネーブル： Z-80の周辺は通常256の入出力アドレス空間内におかれる。各周辺チップは、$\overline{CE}$入力でイネーブルされる。アクティブな$\overline{IORQ}$信号がアドレス・バスの下位バイト上の周辺デバイスのアドレスと一致する場合、$\overline{CE}$入力はアクティブとなる。小型システムでは周辺デバイスも少ないため、アドレス線をこれらのデバイスに占用してデコーダ・ハードウェアを省くことができる。しかし、DMAを使用するシステムではより多くの周辺デバイスを有することが多いため、PROMまたはMSI TTLデコーダによるアドレス・デコーディング方式がとられるのが普通である。

図6.2に、3通りのチップ・イネーブルの回路図を示す。図6.2(a)において、小型システムではDMAは256の入出力アドレスの半分に応答する。図6.2(b)では、PROM（256×4）をプログラムして、DMAのアドレスが存在するときにのみそのO_1端子に"Low"出力が出るように設計されている。PROMはDMAの$\overline{CE}$セットアップ時間の条件に合うよう十分早いアクセスのものを使用しなければならない。

図6.2(c)は、8個の異なった周辺デバイスに、$\overline{CE}$信号を発するために使われるTTLデコーダを使った例を示す。アドレス・ビットA_0およびA_1は、Z-80 SIO、PIO、およびCTCなどの周辺デバイスで直接使用されることが多く、ここではデコードされない。周辺デバイスの数が多いときはさらにデコーダを追加できる。

Z-80周辺デバイスの場合、$\overline{CE}$は内部でゲートされており、$\overline{IORQ}$、$\overline{M1}$でゲートする必要はない。しかし、$\overline{CE}$信号をこれらの制御線とゲートさせても何の影響もなく、回路レベルのシーケンスはよりはっきりする。図6.2(c)にこれを示す。

$\overline{WAIT}$入力の用途： DMAがバスを有しているとき、$\overline{CE}$／$\overline{WAIT}$はメモリまたは入出力回路からの入力端子として機能し、待ち状態を要求することによって、読み出しあるいは書き込みサイクルを拡張する。CPUからの$\overline{BUSAK}$出力によってバス解除信号が発せられ、DMAがバスを有している場合、$\overline{WAIT}$をサンプリングしてこのような要求を判断する。図6.3に示すように、$\overline{CE}$／$\overline{WAIT}$は簡単な2入力マルチプレクサによって方向づけをされる。（$\overline{BUSAK}$はDMAが1個の場合を想定してある。バス・マスタが3個以上のときは$\overline{BAI}$アクティブ、$\overline{BAO}$非アクティブによってバス・マスタを識別する。）

同時転送： DMAの最も高速が得られる方法は、同時転送、すなわちフライバイ転送のときである。この場合、ソースおよびデスティネーション・ポートに同時読み出しおよび書き込み制御信号を発生するためのなんらかの外部ハードウェアを必要とする。

メモリ・アドレスにはアドレス・バスが使われるから、ハードウェア結線によって入出力ポートの選択が行われる場合、直接実行できるのは入出力ーメモリ間転送のみである。DMAはサーチ・モードになり、図6.4に示すような回路により個別の同時読み出しおよび書き込み制御信号を発生させ、メモリおよび入出力において読み出しおよび書き込み制御信号にORされる。図6.5はこの構成を示す。この構成によってCPUおよびDMAの入出力周辺デバイスへのアクセスを可能にする（周辺デバイスのコミュニケーションがDMAのみを通じて行われるときは、$\overline{IORD}$および$\overline{IOWR}$だけを使用する）。

このモードの場合は、アクセス時間、セットアップ時間、および保持時間には特別の注意が必要である。

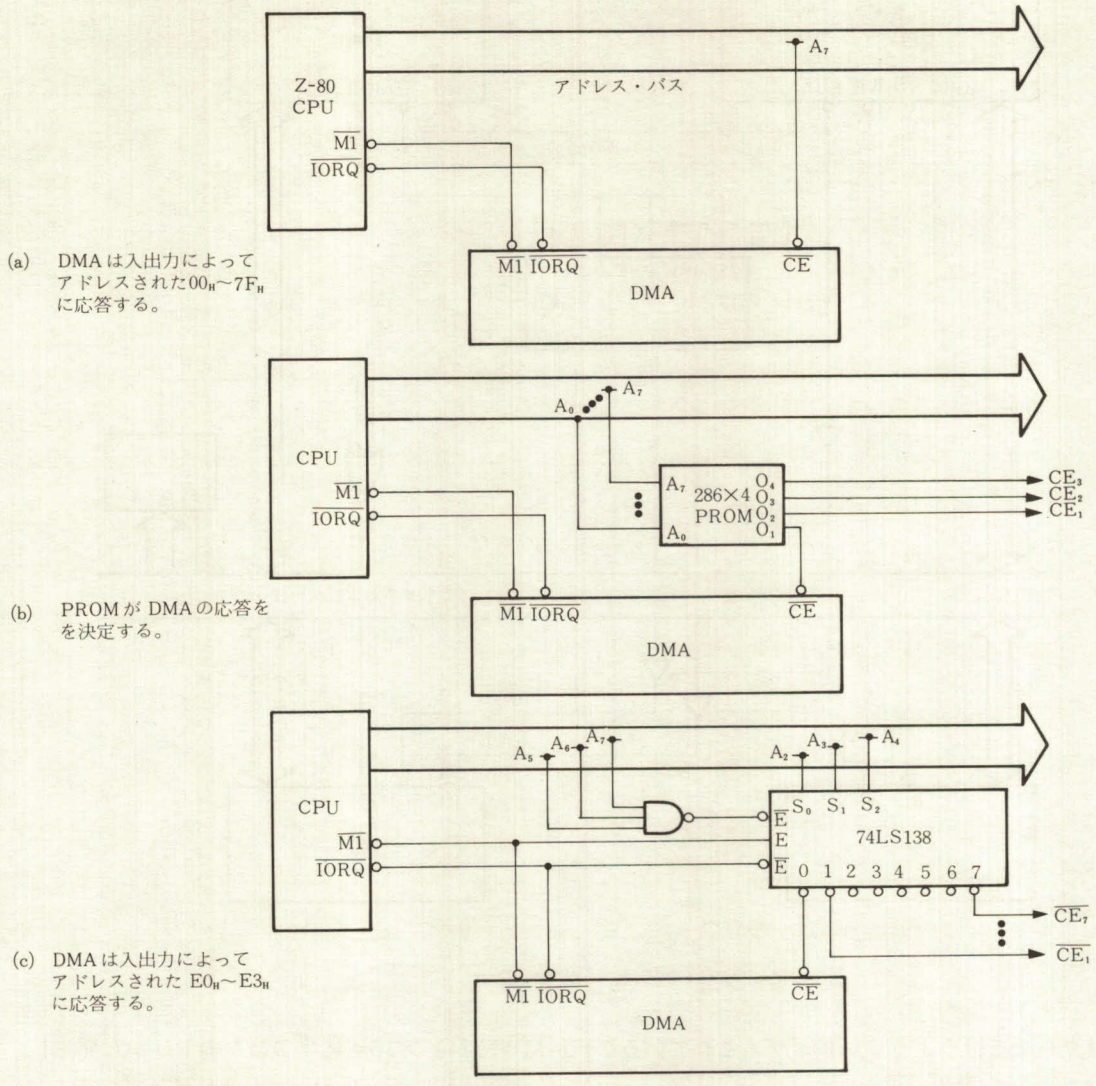


図6.2　Z-80 CPUを使ったチップ・イネーブル・デコーディング

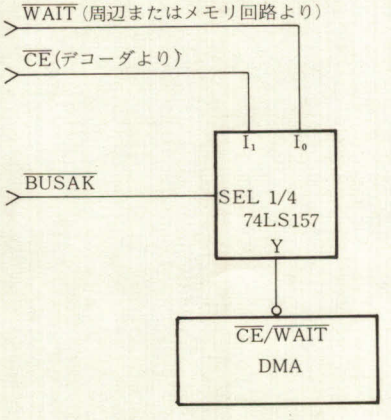


図6.3　CE／WAIT マルチプレクサ

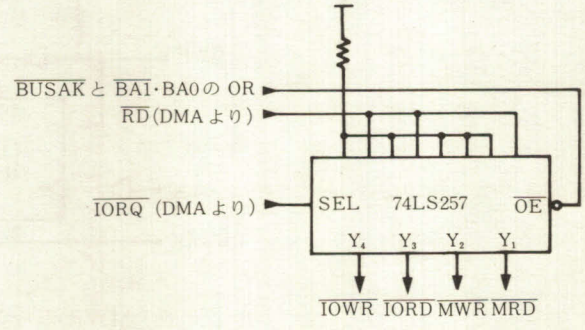


図6.4　同時転送マルチプレクサ

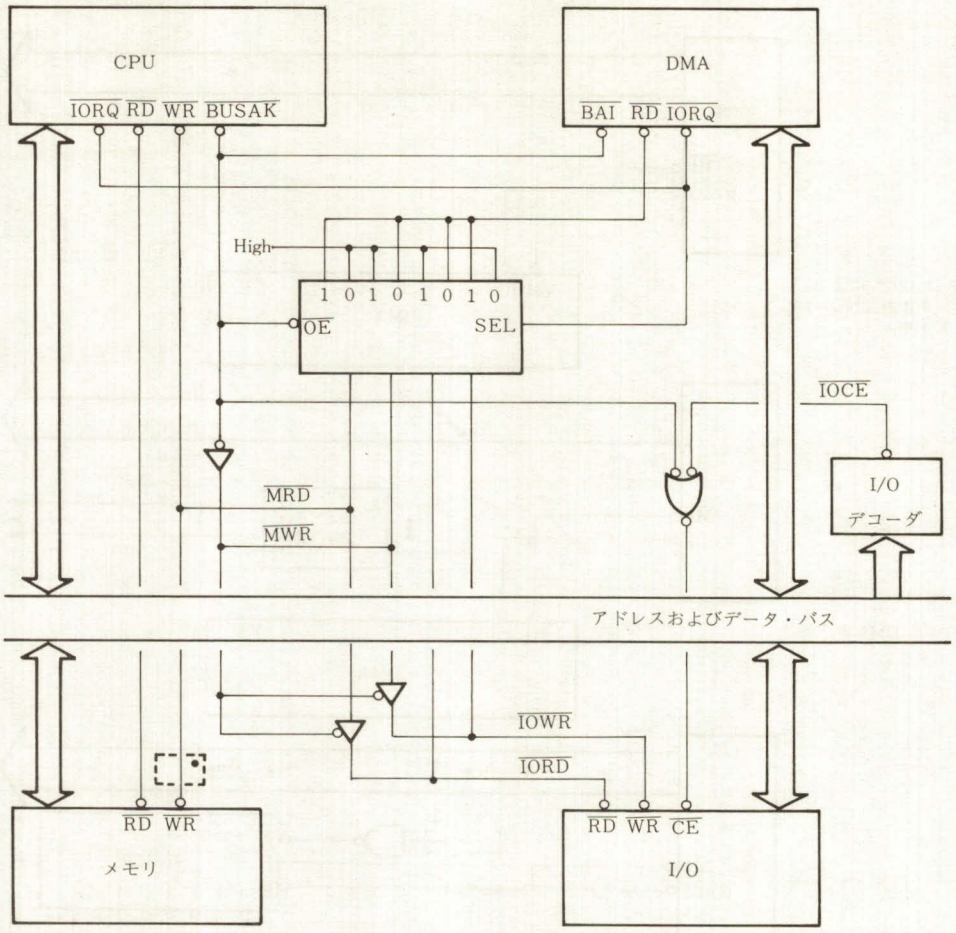


図6.5　同時転送

DMA はサーチを行うようにプログラムされているから、$\overline{\text{MWR}}$ および $\overline{\text{IOWR}}$ は DMA の $\overline{\text{RD}}$ から発生し、そのタイミングをそのまま使う。これは、立ち上がりアクティブとなっている書き込み動作に支障とはならない。$\overline{\text{MWR}}$ を CPU または DMA の書き込みサイクル信号により類似させるためには、図6.6に示した回路を使い、T_2の立ち下がりエッジの後まで $\overline{\text{MWR}}$ の立ち下がりエッジを遅らせる。DMA のプログラム可能な可変タイミング仕様も同様の目的に使用できる。

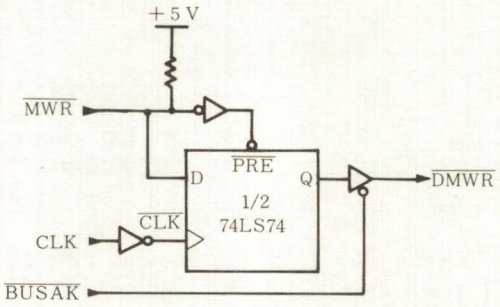


図6.6　$\overline{\text{MWR}}$ のち立ち下がりエッジの遅延

バス・バッファリング： DMAを使用するマイクロコンピュータ・システムには、大型メモリ、多くの周辺デバイスが含まれ、あるいはいくつかの回路を占有することが多い。このようなケースでは、駆動能力とノイズ・マージンを増大し、遅延時間を減少するために、システム・バスおよび制御信号をバッファする必要がある。

単一ボード内でのバッファの必要性は、バスを有するデバイス（CPUおよびDMA）の駆動能力を、バスに接続したすべての入力および出力によって生じる負荷を比較することによって推定できる。この場合、スタティック（DC電流）とダイナミック（キャパシティブ駆動）条件の両方を考慮しなければならない。マザーボードまたは他のボードを駆動するには、バッファが必要である。

バス・マスタ・デバイス（CPUおよびDMA）が同じボード上にあるときは、他のボードへのアドレス、データおよび制御バスについては、バッファを共用できる。その他の場合は、各ボードのバス・インターフェイスにバッファが必要である。

アドレス線は単方向で、74 LS 244および74 LS 367（ノン・インバーティング3ステート・バッファ／ドライバ）、あるいは74 LS 240および74 LS 366（インバーティング3ステート・バッファ／ドライバ）のような、多種の汎用デバイスによってバッファ可能である。このようなバッファ上の3ステート・イネーブル入力によって、CPUおよびDMAアドレス端子と同様な方法で、バスをフロートさせることができる。たとえば、CPUとDMAをそれぞれ1個使ったシステムでは、$\overline{\text{BUSAK}}$がアクティブのとき、この信号はCPUバッファをディセーブルし、DMAバッファをイネーブルできる。バス・マスタとなり得るデバイスが3以上の場合は、実際のバス・マスタに関連するバッファはつねにアクティブでなければならない。こうして、DMAの$\overline{\text{BAI}}$がアクティブでその$\overline{\text{BAO}}$が非アクティブのとき、DMAはバスを制御し、駆動をイネーブルできる。

データ・バスは多方向であるために、バッファ制御はより複雑である。CPUが読み出すことができるすべてのデバイスは選択され、$\overline{\text{RD}}$がアクティブのときデータ・バスを駆動できる。この意味で、$\overline{\text{RD}}$は主要な方向制御となる。CPU以外のデバイスも、割り込みアクノリッジ・サイクル（このときデバイスはバス上にベクトルを出力する）およびDMA書き込みサイクルの期間中、データ・バスを駆動できる。図6.7に双方向のデータ・バス・バッファとその制御を示す。ここでは、Z-80 SIO、PIO、CTC、およびDMA周辺デバイスが1つのボード上にある。これらの共通のボード上のデータ・バスとシステム（マザーボードまたはバックプレーン）バスとの間がバッファされている。ここに示した3通りの条件のいずれも、バッファがシステム・バスを駆動することができる。あるいは、データはボード内に向けてバッファされる。双方向バッファリングに適したデバイスには、74 LS 241（3ステート・バス・ドライバ）および74 LS 245（トランシーバ）が含まれる。

制御信号である$\overline{\text{MREQ}}$、$\overline{\text{IORQ}}$、$\overline{\text{RD}}$、および$\overline{\text{WR}}$は大型、またはマルチシステムにおいては単方向的にバッファできる。まえにも触れたように、これらの信号のバッファはその関連デバイスまたはボードがバスの制御を有しているときイネーブルされ、これらのバス線の制御が他のバス・マスタに移ると、強制的に高インピーダンス状態になる。バスが何によっても駆動されていない場合のバス転送期間中は、短いインターバルがあるため、$\overline{\text{MREQ}}$、$\overline{\text{IORQ}}$、$\overline{\text{RD}}$、および$\overline{\text{WR}}$は2.7kΩから4.7kΩの抵抗を使って＋5Vにプルアップし、これらの信号を非アクティブに維持しなければならない。CPUおよびDMA上の他の制御信号は恒久的に駆動できる。一般に、これにはCPUからの$\overline{\text{M1}}$、$\overline{\text{RFSH}}$および$\overline{\text{HALT}}$、およびDMAからの$\overline{\text{BAO}}$が含まれる。

$\overline{\text{BUSRQ}}$ は双方向であり、外部的にバッファすることは容易ではない。しかし、他の信号と比べると、DMA の $\overline{\text{BUSRQ}}$ は3.2 mA 引き込むことができる。電流を最大にするには、システムの $\overline{\text{BUSRQ}}$ プルアップ抵抗は1.8kΩまで低くしてもよい。

TTL バッファの容量負荷駆動能力の定格については興味深いが、多少不幸な状況がみられる。74 LS 367 のような標準バッファの DC 出力定格は一般に充分であるが、これらのバッファを通じての伝播時間の定格はわずか30 pF の容量負荷となっていて、実際面では簡単に超える数値である。このように、バス駆動条件の中ではこの容量負荷がネックとなるのが普通である（Z-80ファミリのパーツに容量負荷の範囲を示してある）。バスを駆動しているデバイスからその負荷をみる場合、配線およびランド容量、コネクタ容量、および信号に関係した入出力の容量をもみなければならない。ローパワー・ショットキー(LS)TTL 入力の容量負荷は約 6 pF、LS出力は約 8 pFとなる。他の入出力容量の大部分はそのデバイスのデータ・シートから推測できるが、相互結線に関する容量は大きく変わることがある。場合によっては、バッファされたラインの伝播時間と許容容量負荷は、測定が試行錯誤によって決めるしか方法がないこともある。

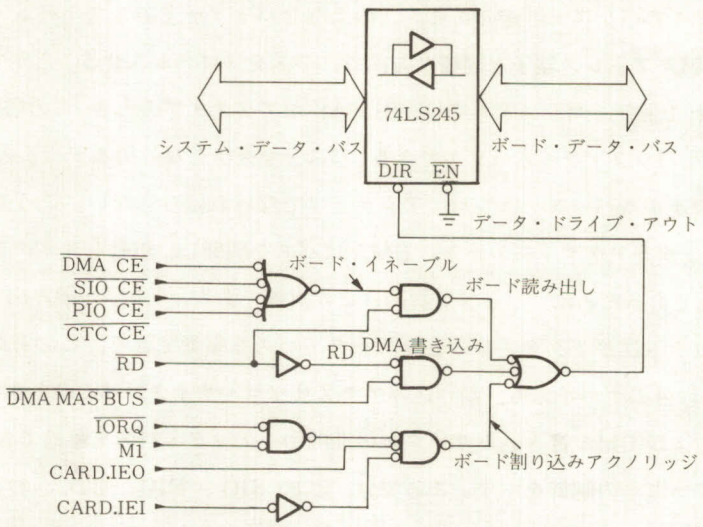


図6.7　データ・バス・バッファ制御例

6.2　Z-80 DMA と Z-80 SIO を使った例

DMA の一般的なアプリケーションは、シリアル・データ・リンク内でのデータ転送処理である。このリンクとのインターフェイスには、Z-80 SIO 周辺デバイスが使われ、これによってシリアルおよびパラレル・データ様式間での変換、同期、その他の機能を行わせる。

この場合、割り込みによって駆動されるデータ転送と DMA によるデータ転送の効率を比較するには、SIO がキャラクタ（バイト）転送を必要とする短時間のインターバルの間の事象シーケンスのチェックが必要である。もちろん、ほとんどの場合 SIO はメッセージ・ビットの送受信に忙しく、サービスを必要としない。

SIO は、その $\overline{\text{WAIT}}$ ／RDY を DMA への $\overline{\text{RDY}}$ として駆動するようにプログラムされなければならない。この $\overline{\text{RDY}}$ はバイト・モードにおいてアクティブとしてプログラムされる。

SIO － DMA 間転送のための事象シーケンスを表6.1および6.2に示す。

受信モード事象シーケンス

事象	事象によるディレイ	
SIO がキャラクタの最後のビットを受信		
SIO のRDYがアクティブとなる	10−13	待ち時間
DMA が $\overline{\text{BUSRQ}}$ を出す	2	
CPU の現在マシン・サイクルが終了	1 − 5	
CPU が $\overline{\text{BUSAK}}$ を帰す	1	
DMA の入出力サイクルが始まる	4	
DMA メモリ書き込みサイクルが始まる	4	バス占有
DMA は $\overline{\text{BUSRQ}}$ を“High”にする	2	
DMAメモリ書き込みサイクルの終了	1	
CPU は $\overline{\text{BUSAK}}$ を“High”にし,バスを制御する	1	

待ち時間(最後のデータ・ビット受信から受信したデータの読み出しまでの遅延)は22～29クロックである。システム・バスは転送1バイトあたり13クロックの間 DMA に占有される。

表6.1　受信モード事象シーケンス

送信モード事象シーケンス

事象	事象によるディレイ	
SIO がキャラクタの最後のビットを送信		
SIO のRDYがアクティブとなる	5 − 9	
DMA が $\overline{\text{BUSRQ}}$ を出す	2	
CPU の現在マシン・サイクルが終了	1 − 5	待ち時間
CPU が $\overline{\text{BUSAK}}$ を帰す	1	
DMA のメモリ読み出しサイクルが始まる	4	
DMA 入出力書き込みが始まる	3	
DMA は $\overline{\text{BUSRQ}}$ を“High”にする	3	バス占有
DMA の入出力書き込みが終了する	1	
CPU は $\overline{\text{BUSAK}}$ を“High”にし,バスを制御する	1	

待ち時間(最後のデータ・ビットの送信からもう1つのキャラクタのローディングまでの遅延)は20～28クロックである。システム・バスは転送バイトあたり13クロックの間 DMA に占有される。

表6.2　送信モード事象シーケンス

割り込みによるCPU転送では、SIOはキャラクタを受信した場合、あるいはさらにもう1つのキャラクタを送信する必要がある場合は、つねにCPUに割り込む。ここに短いベンチマーク・サービス・ルーチンを示す。このルーチンでは、SIO割り込み処理のためのZ-80 CPUの代替レジスタ・セットの排他的使用を想定してある（カッコ内の数字は1命令当たりのクロック数を示す）。

SIO SVC：

EXX	：転送パラメータを得る	(4)
OUTI	：バイト転送、パラメータ更新	(16)
JR Z,BLKEND	：エンド・オブ・ブロック・テスト	(7)
EXX	：パラメータ・セーブ	(4)
EI	：割り込み再イネーブル	(4)
RETI		(14)

サービス・ルーチン実行のまえに、CPUは割り込みをイネーブルとし、現在の命令を終わらせ、割り込みアクノリッジ・サイクル（19クロック）を実行しなければならない。このベンチマークは控え目にみたものであるが、転送バイト当たり少なくとも68クロックを要し、代替レジスタ・セットを占有することによってCPUの能力を大きく制限することになる。

これらの転送方法を比較するには、キロボー当たりの使用クロック数の毎秒使用可能なクロック数に対する割合を計算することができる。これによって転送キロボー当たりのCPUスループットの減少の割合が表わされる。

	Z-80（2.5MHz）	Z-80A（4MHz）
DMAシーケンシャル転送	0.065%	0.041%
DMAシーケンシャル転送／サーチ		
割り込み方式による転送	0.340%	0.213%

表6.3　転送キロボー当たりのCPUスループット減少率

この控え目な例でみても、DMAのときの待ち時間は短く、より予測可能なものであり、システム・オーバヘッドが少なくとも⅕軽減されている。

図6.8にZ-80 CPU、非同期ボーレート発生用Z-80 CTC、Z-80 SIO両チャネル、および2個のZ-80 DMA（各シリアル・チャネルに1個）を使ったZ-80システムの代表的な例を示す。このブロック図では、上述したシステム・メモリ（ROMおよびRAM）、バス、バッファ（必要に応じて使用）、およびチップ・イネーブル・デコーダは省略してある。

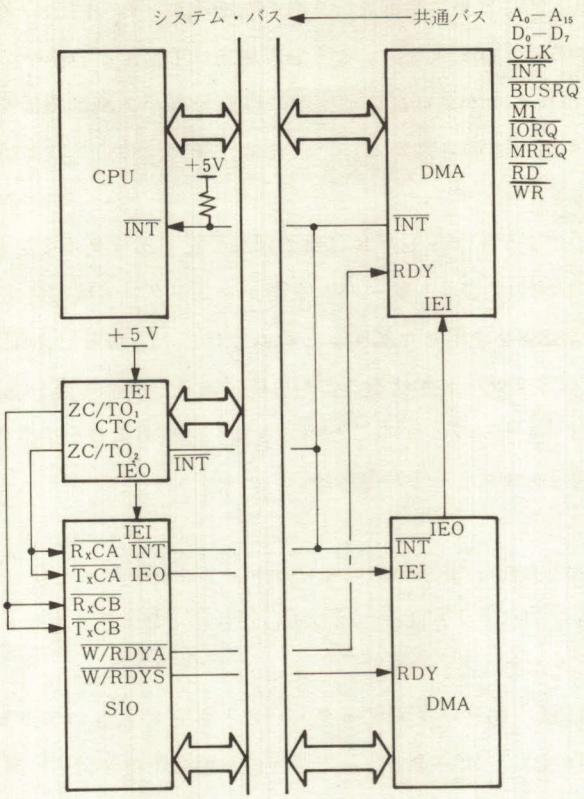


図6.8　Z-80 DMA-SIO システム接続例

6.3　Z-80 DMA と他のプロセッサとの組み合わせ

応用性の広さから他の CPU を使っているコンピュータ・システムの設計者が、アプリケーションにZ-80 DMA を利用することもあろう。この DMA は Z-80 ファミリとして設計されているため、充分に機能するためにはZ-80バスと同様の信号およびバス特性を必要とする。これには 3 つの条件グループがある。

- ○　バス要求／解放メカニズム
- ○　バス特性
- ○　割り込み要求、アクノリッジおよび復帰

ここでは、これらの条件に触れ、設計上のヒントを記す。もちろんすべての可能な組み合わせについて詳述することは不可能なので、各設計者の創造力によって使用できるように設計しなければならない。

バス要求／解放メカニズム：　Z-80 DMA と他のモノシリック DMA とを区別する最も基本的な特性は、そのアクティブな状態の期間中、Z-80 DMA がシステム・バスを完全に制御するということであろう。このことから、まず Z-80 DMA を使ったプロセッサは、アドレス、データ、制御線（$\overline{\text{MREQ}}$、$\overline{\text{IORQ}}$、$\overline{\text{RD}}$、および $\overline{\text{WR}}$ または同等のもの）を含み、システム・バスの制御を放棄しなければならないことを意味する。プロセッサの中には、バス解放のメカニズムをもっていないものもある。6800やそのファミリ・デバイスのよう

に、他のプロセッサには基本的にはバス制御を行うものもあるが、その内部のダイナミックな論理回路のために、無期限に制御を放棄することはできない。こういう場合、DMAとのインターフェイスは困難となる。

しかし、多くのよく使われるマイクロプロセッサには充分なバス制御機能を有するものもあり、Z-80の$\overline{\text{BUSRQ}}$、$\overline{\text{BUSAK}}$によく似たものさえある。たとえば、8080、8085、および8086のHOLDおよびHLDAなどは極めて近いものである。

HOLDおよびHLDAのアクティブ・レベルは負ではなく正でありタイミングも少し異なるが、HOLDおよびHLDAを使ってアドレスおよびデータ・バスを高インピーダンス状態にすることができる。コマンドのデマルチプレクスのために8238を使用した8080システムでは、$\overline{\text{MEMW}}$、$\overline{\text{MEMR}}$、$\overline{\text{IOW}}$、および$\overline{\text{IOR}}$制御線を$\overline{\text{BUSEN}}$入力を使ってフロートさせることが可能である。8085では、3ステート・デコーダを用いて対応信号をデコードまたはディセーブルすることができる。8086およびそのサポート・チップもHLDAがアクティブのときその制御信号を3ステートにできる。

バス特性： Z-80と同様に、8080および8085も8ビット・データ・バスと16ビット・アドレス・バスをもっている。DMAはこのようなデバイスとはよくマッチしており、全データをサーチし、メモリ内のいかなるバイトも直接アクセスすることができる。

8086およびZ-8000 CPUは、16ビット・データ・バスとさらに大きいアドレス空間をもっており、Z-80 DMAを使うのは多少むつかしい。データ・ワードの上位または下位半分の一致バイトのサーチングはできても、すべてのユニークなワードについてはできない。一般には、データ・ブロック内の特別なASCIIキャラクタの感知のように、バイトの一致だけで充分であるので、このこのとは特に問題とならない。より大きなアドレス空間の処理という問題では、DMAがバスを有するまえに適切な上位アドレスをラッチした外部セグメント、またはページ・レジスタを用いることにより、あるいはインデックスのような他の機構によって処理できる。このためには当然なんらかの外部ハードウェアを必要とする。

端子数の制限のため、8085、8086、Z-8001およびZ-8002は、アドレスおよびデータをマルチプレクスしている。アドレスとデータを区別するためのストローブはそのバス構成の一部となり、DMAとのインターフェイスにおいても考慮しなければならない。このような場合、DMAはプロセッサ自体の近くより、むしろデマルチプレクスされたアドレスおよびデータ線に接続しなければならない。図6.9に、この考え方を表した単純な図を示す。

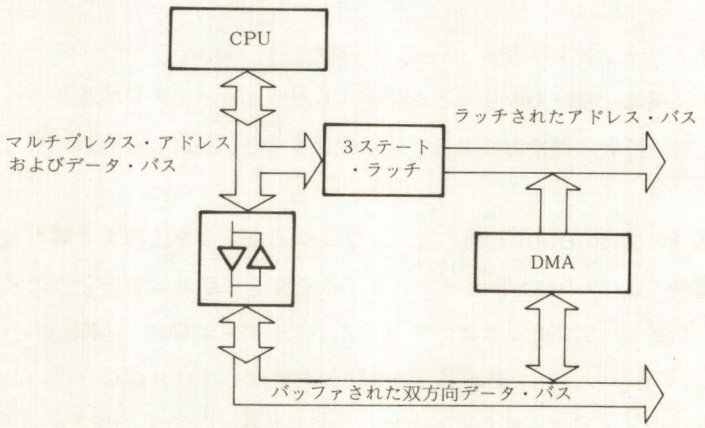


図6.9　DMAをデマルチプレクス・アドレス／データ・バスとの接続

プロセッサの多くはその制御信号をZ-80の $\overline{M1}$ 、 $\overline{MREQ}$ 、 $\overline{IORQ}$ 、 $\overline{RD}$ および $\overline{WR}$ のような類似の信号を作成し、メモリ、周辺デバイスなどへ分配されるまえにデマルチプレクスされるステータス・ワードに変える。前述したように、DMAがバス・マスタのとき出力をフロートする3ステート・デコーダ機能を利用して、DMAをこのようなデマルチプレクスされた信号にリンクする方が得策である。

DMAのZ-80に近似した制御信号は、Z-80以外のバスの条件を満たすためには、タイミングを検討し直すことが必要となろう。しかし、DMAのプログラム可能なタイミング仕様によってこれに要するハードウェア・コストを軽減することができよう。

割り込み要求、アクノリッジおよび復帰： これは多くの面で、DMAを他のプロセッサと使用するときに最も困難な問題となるものである。割り込みの信号、優先順位の決定、識別、応答および復帰は多様である。Z-80システム以外の割り込みでは、IEIおよびIEOは使用せず、ベクトル発生には個別の割り込みコントローラを用いることが多く、アクノリッジや復帰の処理も異なった方法による場合が多い(あるいは全く行わない)。

割り込み要求は一般に容易である。アクティブ・レベルは通常は“Low”であり、2つ以上の個別割り込み要求端子をもつ。割り込み要求のためのタイミング条件は多様で(パルス幅、ラッチなどを含む)、個々のケースについてチェックすることが必要である。

同時またはオーバラップした要求の優先順位は、いくつかの方法で処理される。プロセッサの中には（たとえば8085など）多重割り込み要求端子をもったものや、デージー・チェーンによる優先順位決定機構をもったものもあり、割り込み制御のためのICにもいくつかの種類がある。

アクノリッジや識別についてもその方法は多様である。いくつかの固定メモリ・ロケーションが異なった割り込み端子のサービス・ルーチンに対応する場合もある。他の場合では、割り込みをしているデバイス自体がデータ・バス上にベクトルまたは命令を乗せ、これをCPUに読ませて自己を識別させるものもある。割り込みコントローラがCPUに対して適切なベクトルを与え、多重の要求を区別して優先順位を決めることが多い。DMAは、Z-80割り込みアクノリッジ（$\overline{IORQ}$ および $\overline{M1}$ がともにアクティブ）を感知し、そのIEI入力がアクティブ（他の高位の優先順位のデバイスからの割り込みがない）な場合、任意のベクトル・バイトを供給する能力を内蔵している。このことから、$\overline{M1}$ 、$\overline{IORQ}$ 、およびIEIを適当にゲートすることによって、個別の割り込みコントローラを使わずに済ませることができる。$\overline{IORQ}$ にはもう1つの機能もあり、CPU－DMA間転送の期間中に現れ、アクティブ状態において入出力の読み出しまたは書き込み信号を発することに使われる。

サービス・ルーチンの終わりで、DMAはCPUがRETI命令（$ED4D_H$ は $\overline{M1}$ とともにデータ・バス上に現れる）をフェッチすることを前提にして動作している。Z-80 CPU以外のシステムにおいて、DMAのコマンドである割り込みセットおよびディセーブルはこの目的のために設計されている。あるいは、$\overline{M1}$ の再ゲートおよびCPUがバイト $ED4D_H$ を架空の周辺デバイスに書き込むようにプログラムすることによって、RETI命令をシミュレートすることができる。実在しない周辺デバイスのためのチップ選択は、このような場合に $\overline{M1}$ をシミュレートするために用いられる（図6.10）。

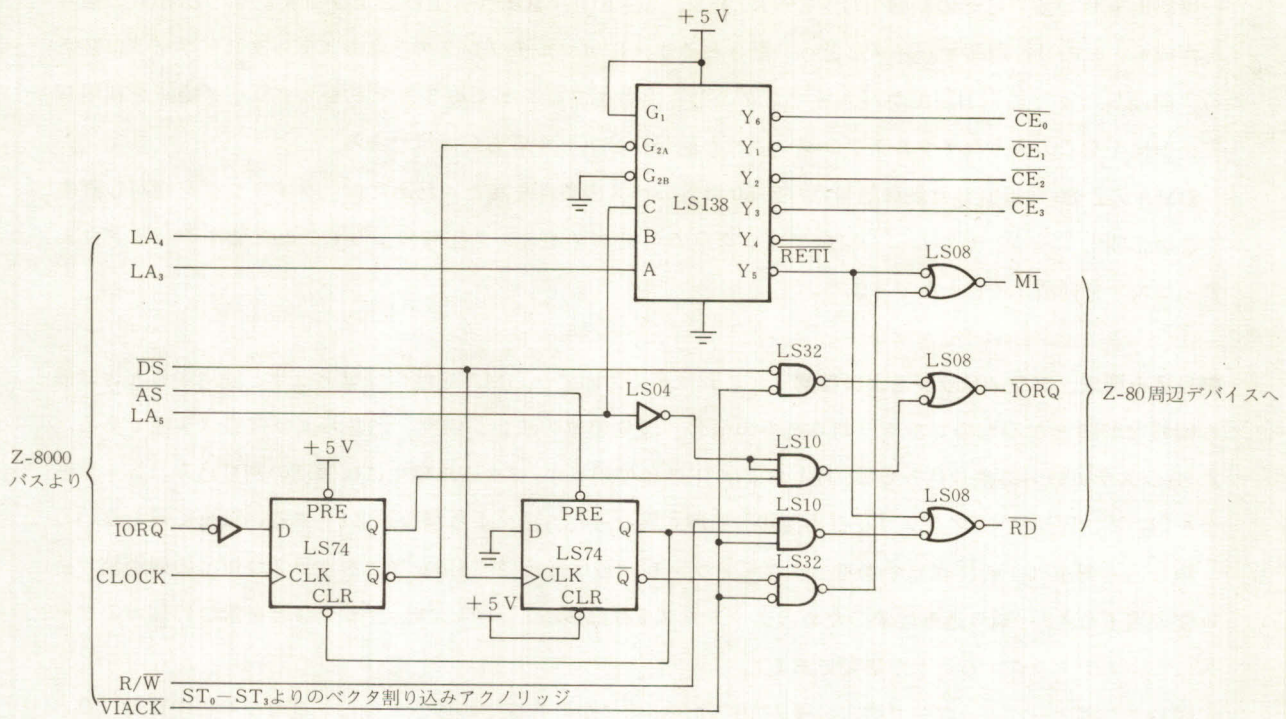


図6.10　Z-8000／Z-80周辺デバイス・インターフェイス

第7章　性能上の制限

7.1　バスの競合

DMAの使用に対する主な制限は、CPUの能力の低下である。DMAが動作するときはDMAがバス・マスタであり、CPUが命令をフェッチしこれを実行することはできない。このようにCPUを停止させることによって、下記を含むいくつかの問題が生じる。

- ○　割り込みサービスの不可（ノンマスカブル割り込みを含む）
- ○　ダイナミック・メモリのリフレッシュの不可（CPUに依存しているとき）
- ○　ポーリングの不可

DMAの動作中にCPUの時間中心の機能に及ぼす影響の程度は、DMAの動作モードによって異なる。

バイト・モード：　このモードの場合、転送されるバイトごとにCPU機能とDMA機能をインターリーブすることが可能であるから、バスの競合の見地からはこれは最も望ましいモードである。転送速度が遅いのが不利な点となる。

バースト・モード：　転送されるデータが時間的に幅をもっており、他のCPUに付属した機能が危うくなるまえに、DMAがCPUにバスを解放するような場合には有用なモードである。バースト・モードの利点は、必要時にのみバスを使用し、使用中は転送速度を最大にできることである。しかし、非常に長いデータの転送となる可能性（RDYがアクティブなときの長いデータ転送）があるときは使用できない。

連続モード：　このモードは、RDYの状態に関係なくエンド・オブ・ブロックまたはバイト一致の成立までバスを保持し、最もバスを悪用するものである。転送速度は最大であるが、CPUに付属した機能に時間的要素が少ないとき、あるいはブロックが比較的短いときにのみ使用できる。

大部分のアプリケーションにおいて、バイト・モードは一般に安全である。バースト・モードあるいは連続モードを使用する際の効果を計算するには、下記を考慮しなければならない。

- ○　最大ブロック長
- ○　最大DMA転送速度（表2.1参照）
- ○　RDYがアクティブ状態を維持できる最大時間

バイトまたはバースト・モードにおいて、DMAをバスから強制的に切り離す方法がある。この方法では外部ゲートを用いてDMAへのRDY入力を除去する。図7.1にこの方法を示す。この方法の利用を考える際、転送の途中でDMAを停止させることによって生じるマイナス面をも忘れてはならない。この方法はDMAが連続モードで動作中は使用できない。連続モードにおいてDMAにバスを解放させることができるのは、パワーダウンまたは通常のエンド・オブ・ブロックあるいはバイト一致成立による終了のみである。

図7.1　DMA バス・マスタ・ゲート（バイト・モードまたはバースト・モードのみ）

7.2　制御オーバヘッド

DMA のプログラムをスタートし、更新するために CPU に生じるソフトウェア・オーバヘッドも、システムのトータル効率に付与する DMA の貢献度を制限するものである。図5.8では、DMA の全機能をフルに発揮した場合、DMA 初期状態設定に要する制御バイト数は最大約35である。これに加えて、割り込みモードの使用には CPU によるサービスが必要で、このためにはさらに DMA へ書き込む制御バイトを増やさなければならない。そのため、システム・スループットの増大も、DMA を頻繁にプログラムし直したり、データとは独立した DMA 機能に対する広範囲に及ぶ割り込みサービスを要するようなアプリケーションでは、あまり重要でなくなる。転送バイト数に対する必要オーバヘッドの割合は、大きいブロックの繰り返し転送で最小限にすることが可能である。

第8章　タイミング

8.1　CPUがバス・マスタの場合

制御バイトの書き込み：　CPUがバスを有している場合、制御バイトを用いてDMAをプログラムできる。図5.1ではこれをディセーブル、イネーブル／非アクティブ、またはイネーブル／ストップの状態としてあらわしている（後の2つは同等）。

DMAはCPUの出力命令において、入出力周辺デバイスとしてアドレスすることによってプログラムされる（64Kの入出力アドレスのフルスペースにアドレス可能）。このためには、3本の線がクロックの立ち上がりエッジにおいて同時にアクティブ“Low”となる必要がある。

$\overline{CE}$　　チップ・イネーブル

$\overline{IORQ}$　入出力要求

$\overline{WR}$　　書き込み

図8.1にこのタイミングを示す。Z-80 CPUシステムにおいて、CPUとDMAが同一ボード上にあり、CPUおよびDMAに共通な端子と直列のバッファ、ドライバまたはその他のゲートを有していない場合、このタイミングは自動的に発生する。これはシーケンシャル転送、シーケンシャル転送／サーチ、およびサーチのみの動作クラスの場合である。これが同時転送または同時転送／サーチ動作にもいえるかどうかは使用する外部デバイスの速度によって決定する（アプリケーションの章を参照）。

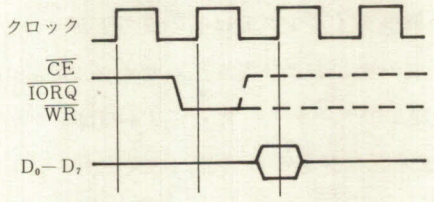


図8.1　CPUからDMAへの書き込みサイクル

DMAに制御バイトを書き込むための基本的な特性は、下記のとおりである。

- DMAの$\overline{CE}$が“Low”であること（通常、アドレス・バスの下位バイトをデコードすることによって生じる）。
- この場合、$\overline{IORQ}$および$\overline{WR}$は“Low”であること。
- 制御バイトがデータ・バス上に置かれ、立ち上がりクロック・エッジにおいて安定すること。このエッジは$\overline{CE}$、$\overline{IORQ}$、および$\overline{WR}$が安定してから1クロック後に発生する。

ステータス・バイトの読み出し： 図8.2は、CPUがバスを有しているとき、RR0からRR6までのDMAの読み出しレジスタを、CPUが読み出すのに必要なタイミングを示す。レジスタの読み出しには下記の条件が必要である。

- $\overline{CE}$、$\overline{IORQ}$および$\overline{RD}$が、クロックの2つの立ち上がりエッジにわたってアクティブ、および安定すること。
- 2つ目のクロックの立ち上がりエッジにおいて、ステータス・データはデータ・バス上に現れ、$\overline{CE}$、$\overline{IORQ}$、および$\overline{RD}$が同時にアクティブである期間中保存される。

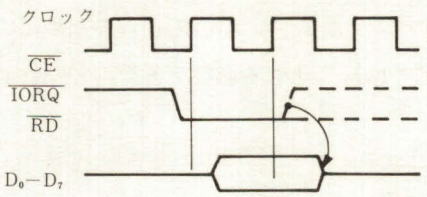


図8.2　CPUの読み出しサイクル

8.2　DMAがバス・マスタの場合

シーケンシャル転送： シーケンシャル転送および転送／サーチ動作において（ともにタイミングは同じ）、データは$\overline{RD}$の立ち上がりエッジ（標準タイミングではT_3の立ち下がりエッジ）でラッチされる。データは次の書き込みサイクルの終わりまで、読み出しおよび書き込みサイクルの間中、データ・バス上に保持される。$\overline{RD}$が非アクティブとなった後、DMAのデータ・バス・ドライバはアクティブとなる。

図8.3はメモリー入出力間転送のタイミングを示し、図8.4は入出力ーメモリ間転送を示す。メモリーメモリ、入出力ー入出力間転送のタイミングは、これらの図を単純に置き換えたものである。

標準タイミングでは、メモリ動作に3クロック・サイクル、入出力動作には4クロック・サイクルを使用するが、入出力動作ではT_2とT_3との間に待ちサイクルが自動的に挿入される。DMAがアクティブのとき$\overline{CE}$／$\overline{WAIT}$が$\overline{WAIT}$として動作するようにプログラムしてある場合、この信号線はメモリ動作ではT_2の立ち下がりエッジ、入出力動作ではT_Wの立ち下がりエッジでサンプルされる。この期間、$\overline{CE}$／$\overline{WAIT}$が“Low”のときは他のTサイクルが追加され、この期間に$\overline{CE}$／$\overline{WAIT}$は再びサンプルされる。動作の期間はこのように無限に拡張できる。

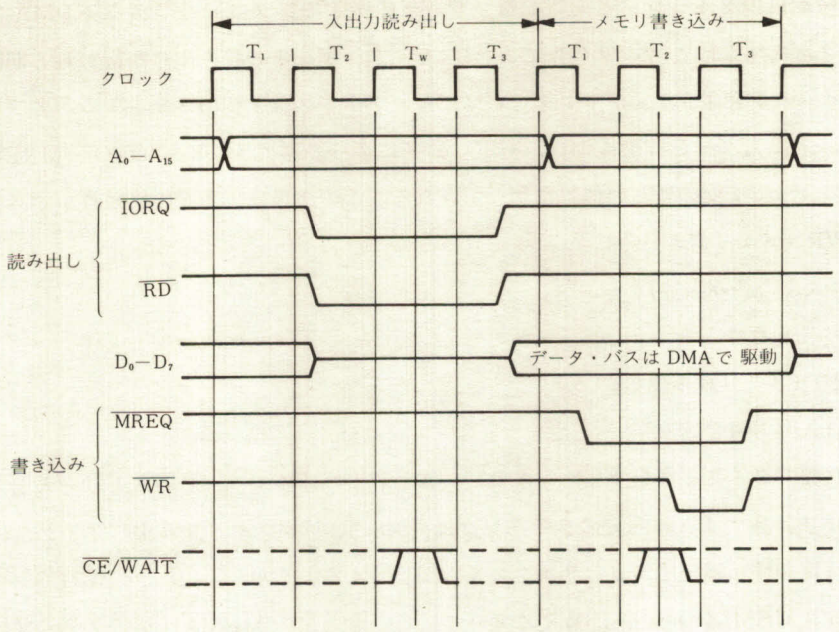


図8.3　メモリー入出力間シーケンシャル転送、標準タイミング（サーチングはオプション）

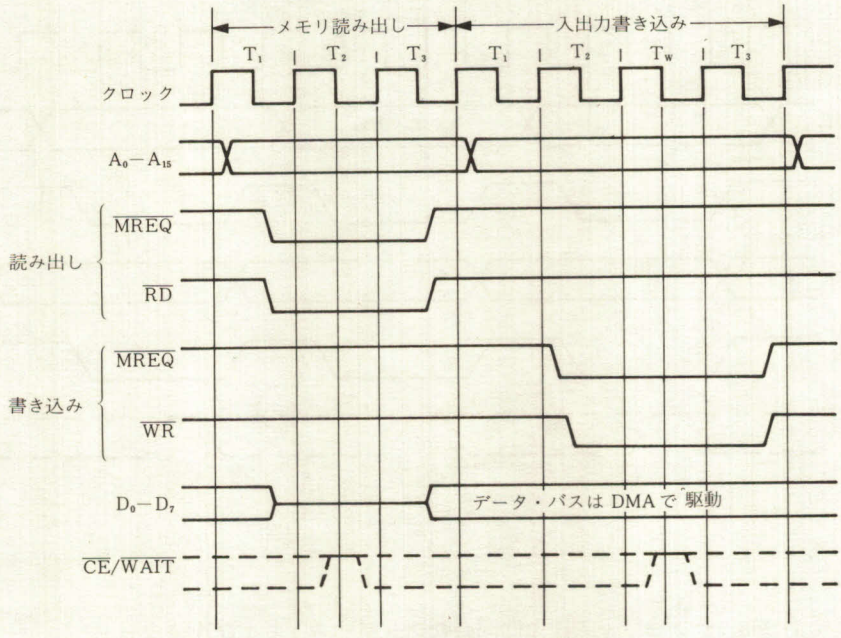


図8.4　入出力ーメモリ間シーケンシャル転送、標準タイミング（サーチングはオプション）

同時転送： 同時転送のタイミングと同時転送／サーチのタイミングは同じである。DMAはサーチのみのモードにプログラムされており、サーチのみであれば、ソース・ポートの読み出しが行われる期間に、読み出し、書き込みの両サイクルが発生する。アドレス・バスには、アドレスは1個のみ発生する。アプリケーションの章で示すように、入出力ポートは動作中はハードウェアによって選択される。$\overline{\text{IORQ}}$、$\overline{\text{MREQ}}$、$\overline{\text{RD}}$および$\overline{\text{WR}}$は外部回路によって2つの新しい信号にゲートされる。この信号は、下記のいずれかである。

- $\overline{\text{MEMWR}}$（メモリ書き込み）
- $\overline{\text{IORD}}$（入出力読み出し）

または

- $\overline{\text{MEMRD}}$（メモリ読み出し）
- $\overline{\text{IOWR}}$（入出力書き込み）

図8.5は、Z-80標準タイミングを使ったときのバーストおよび連続モードでのメモリー入出力間の同時転送のタイミングを示す。各サイクル内でのタイミングは、図8.3に示すメモリ読み出しサイクルに類似する。アドレス・バスの確定条件も同一であり、サイクル長も同じである。しかし、図8.3の$\overline{\text{MREQ}}$、$\overline{\text{RD}}$、$\overline{\text{IORQ}}$および$\overline{\text{WR}}$は図8.5で$\overline{\text{MEMRD}}$および$\overline{\text{IOWR}}$に変っている。さらに、$\overline{\text{MEMRD}}$がアクティブになるのに応じて、図8.5ではデータ・バスがアクティブになるのは早くなっている。データは$\overline{\text{IOWR}}$の立ち上がりエッジで入出力デバイスに取り込まれる。

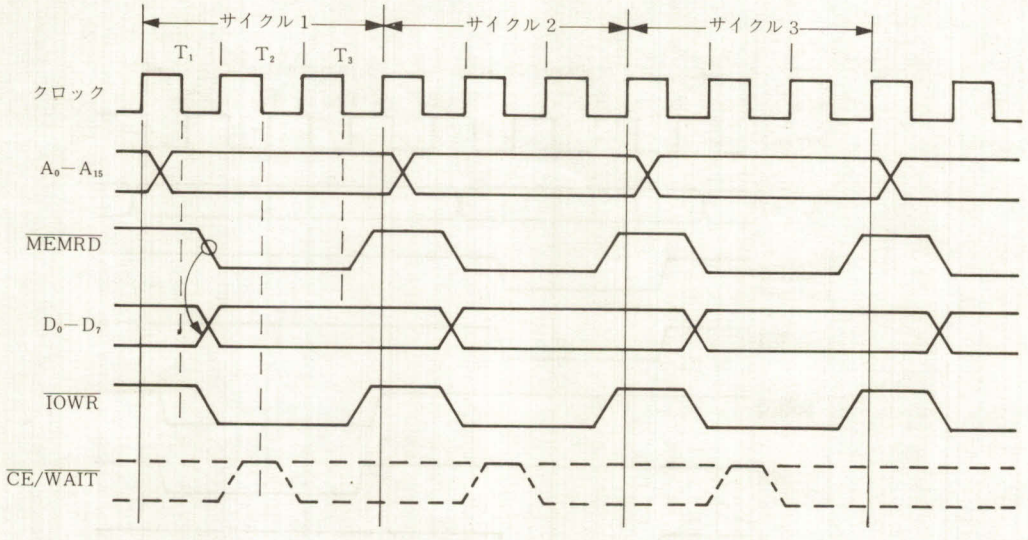


図8.5　メモリー入出力間同時転送（バーストおよび連続モード）

図8.6はバイト・モードのタイミングを示す。各サイクルについては図8.5と同じである。サイクル間の切れ目でアドレスおよびデータ・バスが3ステートとなり、$\overline{\text{MEMRD}}$ および $\overline{\text{IOWR}}$ が引き続き非アクティブとなっている部分は、$\overline{\text{BUSRQ}}$ および $\overline{\text{BAI}}$ 上のアクティブによって生じるものであり、これについては後述する。

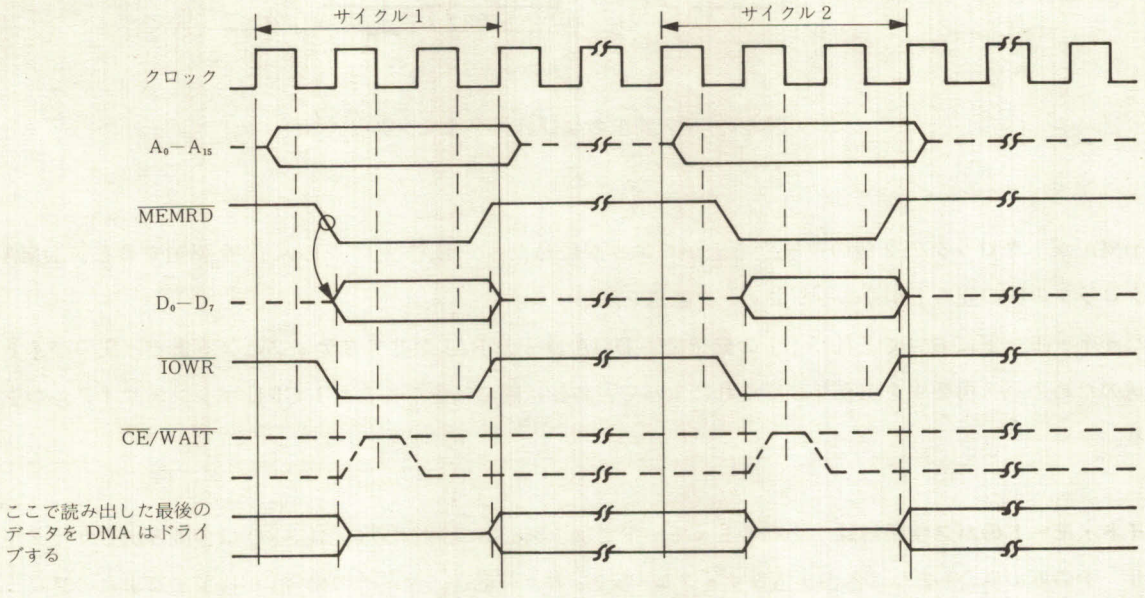


図8.6　メモリー入出力間同時転送（バイト・モード）

サーチのみ：　サーチ動作の標準タイミングは、図8.3および図8.4の読み出しサイクルと同一である。サーチのみ動作は読み出しのみと同じであり、データは一致バイトとの比較のためにDMAレジスタに読み込まれるだけである。

バス要求：　図8.7はバス要求およびその許可タイミングを示す。RDYはアクティブ"High"あるいは"Low"のいずれにプログラムしてもよく、クロックのすべての立ち上がりエッジでサンプルされる。アクティブの場合、バスが他のデバイスによって使用されていない場合、クロックの立ち上がりエッジで $\overline{\text{BUSRQ}}$ を"Low"にする。$\overline{\text{BUSRQ}}$ 受信後、CPUはその $\overline{\text{BUSAK}}$ を"Low"にする（$\overline{\text{BUSAK}}$ は、直接あるいは多重DMAのデージー・チェーンを通して、DMAの $\overline{\text{BAI}}$ 入力に接続されている）。

CPUはその各マシン・サイクルの終了の1クロック・サイクルまえに、その $\overline{\text{BUSRQ}}$ 入力をみて要求があればそのマシン・サイクルの終了時にバスを解放する。そのため、CPUが $\overline{\text{BUSRQ}}$ を受けてから $\overline{\text{BUSAK}}$ に反応するまでの間の最大遅延は、1マシン・サイクルに1クロック・サイクル弱を加えた時間である。

CPUはその $\overline{\text{BUSAK}}$ でアクノリッジする場合、すべてのバス制御線を高インピーダンスにする（$\overline{\text{M1}}$ は高インピーダンスにならない）。

RDYは、クロックの立ち上がりエッジに関して特定のセットアップ時間をもっており、バイトあるいはバースト・モードにおいて、DMAがバス・マスタになるまでアクティブ状態を保持しなければならない。（RDYはエッジとしてではなく、レベルとして感知される。）RDY上のパルスによってDMAがバス・マスタとなり得るのは、連続モードのときだけである。この場合、DMAはバス・マスタとなるが動作は開始しない。

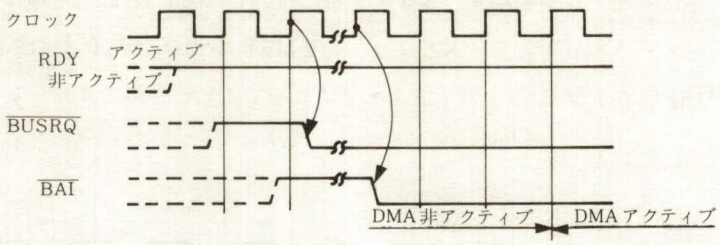


図8.7　バス要求および許可タイミング

DMAが、クロックの2個の連続立ち上がりエッジにわたって$\overline{BAI}$上に“Low”を感知すると、DMAはクロックの次の立ち上がりエッジでデータ転送を開始する。

バイト・モードにおいて、各バイトの転送後、DMAはその$\overline{BAI}$が非アクティブとなるまで、次のバイト転送のためのバス再要求を出さない。これによって、各転送間には少なくとも1 CPUマシン・サイクルが生じる。

バイト・モードのバス使用解除：　バイト・モードでは、図8.8に示すように$\overline{BUSRQ}$は各読み出しサイクル（サーチのみの場合）または各書き込みサイクル（転送、および転送／サーチの場合）が終了するまえのクロックの立ち上がりエッジにおいて、“High”になる。これはRDYの状態に関係なく行われるが、Z-80 CPUを使用する場合、次のクロック・サイクルまでCPUは動作を開始できないため、混乱を起こす可能性はない。また、他のCPUに支障を来たすこともほとんどない。ただし、1クロック・サイクルによるバイト転送に要する時間を少なくするためには、この点に留意が必要である。

次のバイトのための次のバス要求は、$\overline{BUSRQ}$および$\overline{BAI}$がともに“High”に復帰した後で行う。Z-80では$\overline{BAI}$が“High”になるのは、$\overline{BUSRQ}$が“High”に復帰して1クロック・サイクル後である。

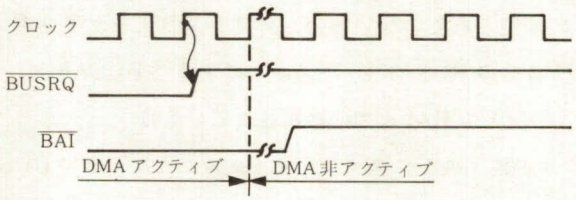


図8.8　バス使用解除（バイト・モード）

エンド・オブ・ブロック時のバス使用解除： バーストあるいは連続モードにおいて、DMA がエンド・オブ・ブロックによって停止するようにプログラムされている場合、エンド・オブ・ブロックによって $\overline{\text{BUSRQ}}$ は“High”（非アクティブ）になる。そのタイミングは、DMA がデータ・ブロックの転送を完了するのと同じクロックの立ち上がりエッジである（図8.9参照）。最後のバイト転送が完了するまえに RDY が非アクティブになっても、ブロックの最後のバイトは転送される。

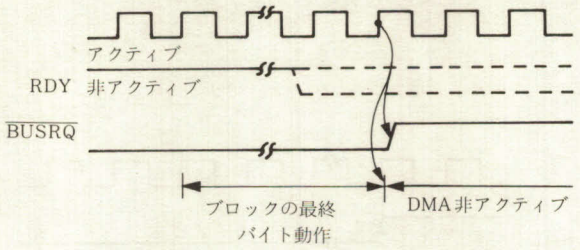


図8.9　エンド・オブ・ブロック時のバス使用解除（バーストおよび連続モード）

一致成立時のバス使用解除： バーストまたは連続モードにおいて、一致成立によって DMA が停止（バスを解放）するようプログラムされている場合、一致成立により次の DMA 動作において $\overline{\text{BUSRQ}}$ は非アクティブになる。たとえば、サーチのみあるいは同時転送／サーチの次の読み出しの終了時、あるいはシーケンシャル転送または転送／サーチの次の書き込みの終了時（図8.10を参照）である。

パイプライン方式により、一致の成立は次の DMA 読み出しあるいは書き込み中に決定する。表4.2は、すべてのクラスまたはモードにおける転送バイト数について完全に記したものである。

RDY は一致動作の開始後、このバス使用解除のタイミングに影響を与えることなく、非アクティブにできる。しかし、表4.1および図4.5に示すように、RDY がいつ非アクティブになるかによって、転送バイト数が変わることがある。

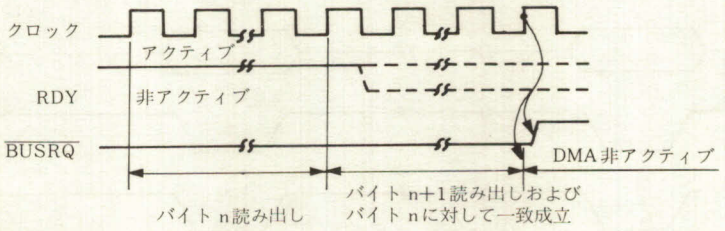


図8.10　一致成立時のバス使用解除（バーストおよび連続モード）

レディでなくなった時のバス使用解除： バースト・モードにおいてRDYが非アクティブになると、$\overline{\text{BUSRQ}}$は現在のバイト動作完了後、次のクロックの立ち上がりエッジにおいて“High”になる。たとえば、サーチのみまたは同時転送／サーチの現在の読み出し終了時、あるいはシーケンシャル転送／サーチの次の書き込みの終了時（図8.11）である。したがって$\overline{\text{BUSRQ}}$に対する動作は、RDYに対する動作により若干遅れる。DMAはつねに現在のバイト動作を完了するまでバスを解放しない。

これとは対照的に、連続モードではRDYが非アクティブになっても$\overline{\text{BUSRQ}}$は解除されない。代わりにDMAは現在のバイト動作の完了後、RDYが再びアクティブになるまでアイドル状態となる。

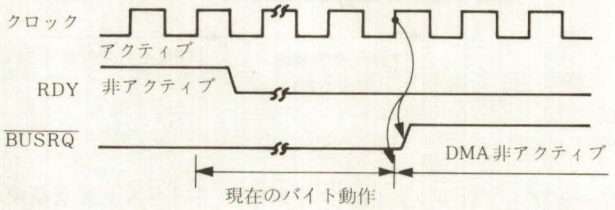


図8.11　レディでない時のバス使用解除（バースト・モード）

図8.12、8.13および8.14は、RDYの非アクティブと他の線の状態との関係を各動作モードについて示したもので、Z-80標準タイミングによるメモリ・サーチのみ動作を想定している（RDYがアクティブになるタイミングについてはバス要求に記す）。RDYは、各読み出しまたは書き込みサイクルの最後のクロック・サイクルの立ち上がりエッジでサンプルされる。これはレベル・サンプルであり、エッジ・サンプルではない。

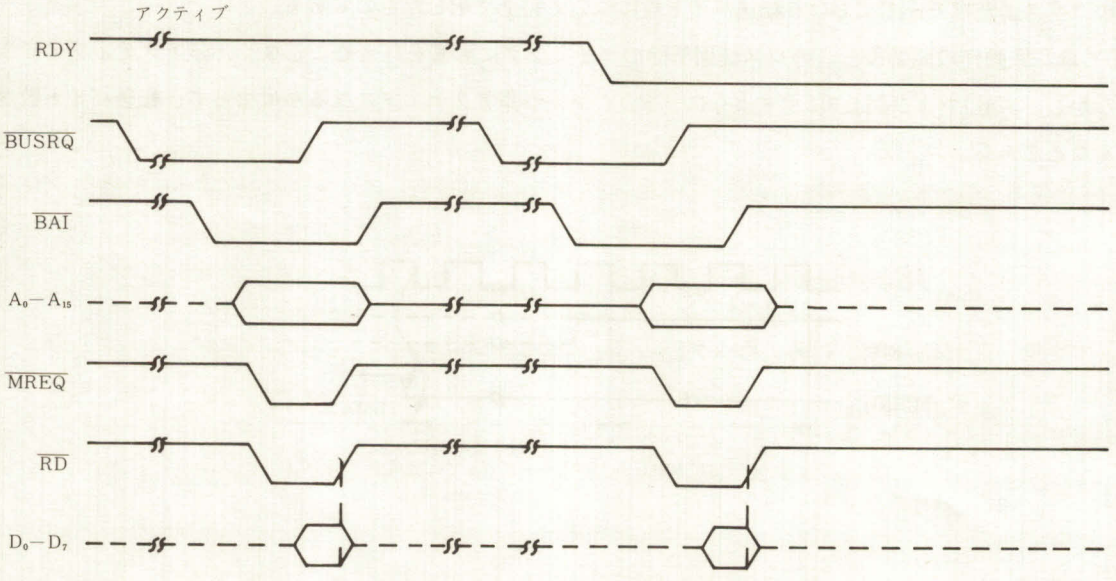


図8.12　バイト・モードにおけるRDY

RDY は最後のバイト動作の完了まえに、その動作に影響することなく非アクティブになることが可能である。その動作の終わりに、図8.11および8.13に従い、バイトまたはバースト・モードにおいて、$\overline{\text{BUSRQ}}$ および $\overline{\text{BAI}}$ は“High”になる。RDY が非アクティブの間、バイトおよびバースト・モードにおいて、バス制御線（$\overline{\text{MREQ}}$、$\overline{\text{IORQ}}$、$\overline{\text{RD}}$、および $\overline{\text{WR}}$）も同じく“High”を維持し、アドレスおよびデータ・バスは3ステートとなっている。

連続モード（図8.14）は、RDY が非アクティブとなっている期間中、アドレス・バスは次のバイトに対して、あらかじめインクリメントされているアドレスを保持する点で異なっている。このアドレスは RDY が再びアクティブになると即使用可能となる。

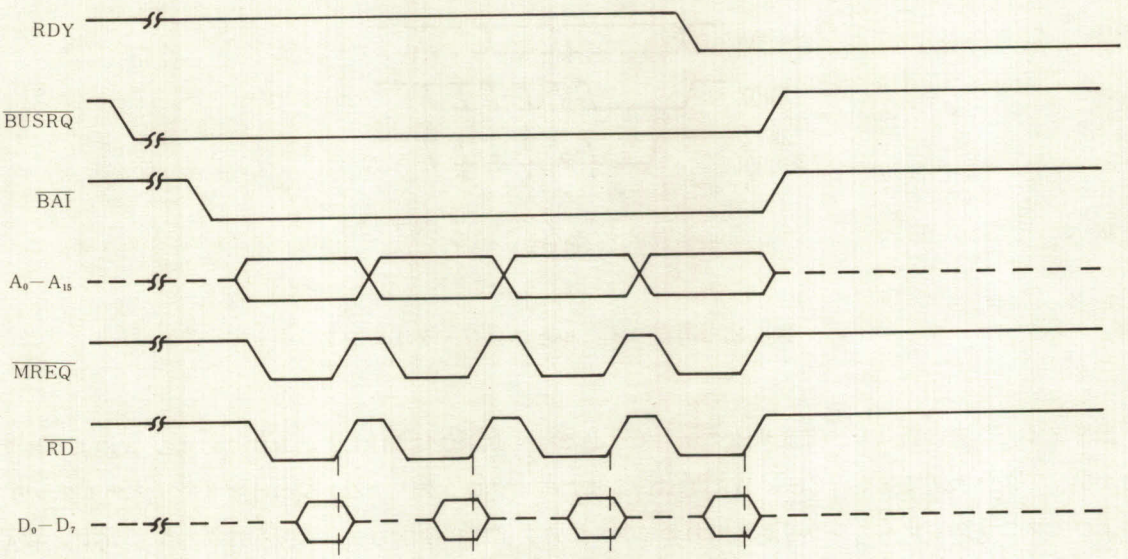


図8.13　バースト・モードにおける RDY

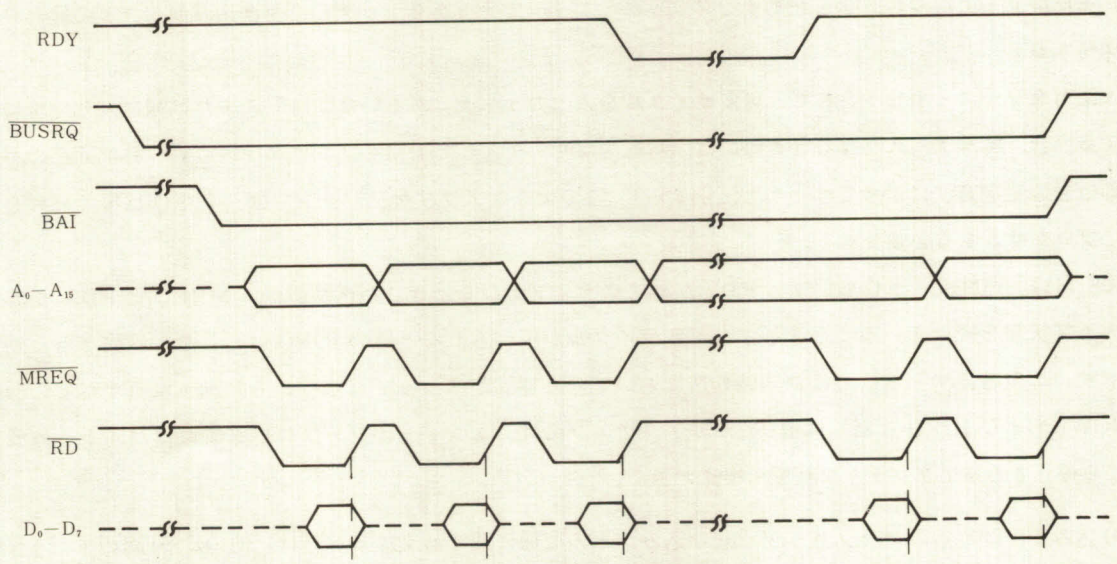


図8.14　連続モードにおける RDY

可変サイクルとエッジ・タイミング： Z-80 DMA の動作サイクル長は、待ち状態なしでソース（読み出し）およびデスティネーション（書き込み）ポートに対して独立してプログラムできる。この可変サイクル機能によって、2、3、あるいは4クロック・サイクル（待ちサイクルの挿入があるときはさらに多いクロック・サイクル）の読み出しまたは書き込みサイクルが可能となり、これにより、DMA が発生するすべての信号のパルス幅を増減できる。さらに、$\overline{\text{IORQ}}$、$\overline{\text{MREQ}}$、$\overline{\text{RD}}$、および $\overline{\text{WR}}$ の立ち上がりエッジは独立して½サイクル早く終わらせることができる。図8.15にこれを示す。

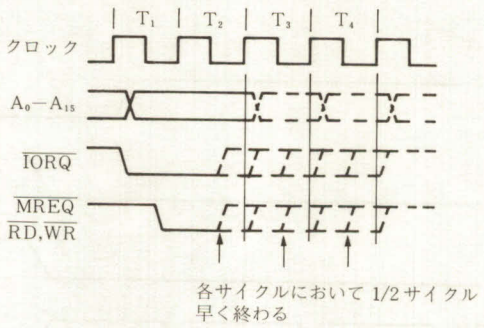


図8.15　可変サイクルとエッジ・タイミング

標準タイミングと異なり、可変サイクル・モードでは、$\overline{\text{IORQ}}$ は $\overline{\text{MREQ}}$、$\overline{\text{RD}}$、および $\overline{\text{WR}}$ より½サイクル早くアクティブになる。$\overline{\text{CE}}$／$\overline{\text{WAIT}}$ は、3または4クロック・サイクルの可変メモリ・サイクルのみ、および4クロック・サイクルの可変入出力サイクルのみの拡張に使用できる（図8.15を参照）。$\overline{\text{CE}}$／$\overline{\text{WAIT}}$ は、3または4サイクル・メモリ動作の場合はT_2の立ち下がりエッジにおいて4サイクル、入出力動作の場合はT_3の立ち下がりエッジにおいてサンプルされる。2サイクル動作の場合はサンプルされない。転送の期間中、$\overline{\text{RD}}$ の立ち上がりエッジを起動するクロック・エッジでデータをラッチし、書き込みサイクルの終わりまで保持する。

入出力をソース・ポートとして、可変サイクルを入出力サーチ、あるいは同時転送または転送／サーチに使用する場合、特殊ケースとなる（実際には同時転送はサーチとして DMA にプログラムされ、外部回路のバス制御信号の処理方法によってサーチと区別する）。このようなアプリケーションでは、$\overline{\text{IORQ}}$ は早く終わるようにプログラムする必要がある。

図8.14は、連続モードにおいて、RDY が非アクティブとなるとき、バス制御線（$\overline{\text{MREQ}}$ と $\overline{\text{RD}}$）が非アクティブ状態を続けていることを示す。可変サイクルが用いられるときの $\overline{\text{IORQ}}$ は、これとは異なった動作を示す。この場合、$\overline{\text{IORQ}}$ および同時転送において、外部回路によって $\overline{\text{IORQ}}$ から始動されるすべての機能（$\overline{\text{IOWR}}$および $\overline{\text{IORD}}$ など）は RDY が非アクティブの間、エンド・オブ・ブロックまたはバイト一致成立により停止するまでアクティブ状態のままである。

割り込み： 割り込みアクノリッジや割り込みからの復帰に関するタイミングは、他のZ-80周辺デバイスの場合と同じである。図8.17にこのタイミングを示す。$\overline{\text{INT}}$ は、すべての命令の最後のクロック・サイクルの立ち上がりエッジにおいて、CPU によってサンプルされる。内部の CPU ソフトウェアによる割り込みイネー

ブル・フリップ・フロップがセットされていない場合、または $\overline{\text{BUSRQ}}$ がアクティブの場合、この信号は許可されない。$\overline{\text{INT}}$ が許可されると、特別な $\overline{\text{M1}}$ サイクルが作成される。

この特別な $\overline{\text{M1}}$ サイクルの期間中、$\overline{\text{IORQ}}$ も同時にアクティブとなり（通常の場合は $\overline{\text{MREQ}}$ ）、割り込み

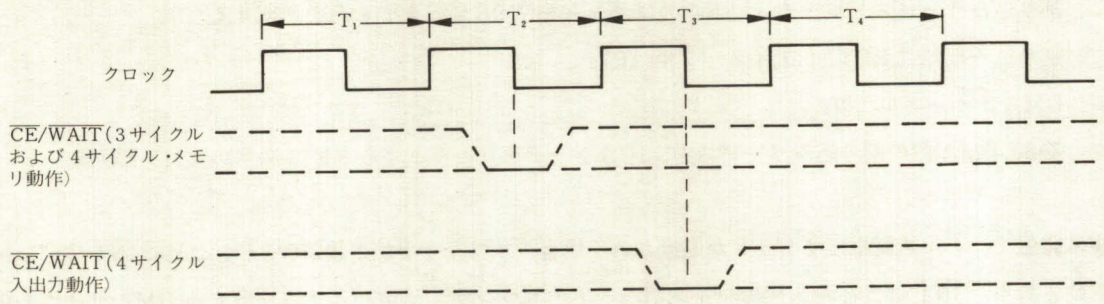


図8.16　可変サイクル・タイミングにおける $\overline{\text{WAIT}}$ サンプリング

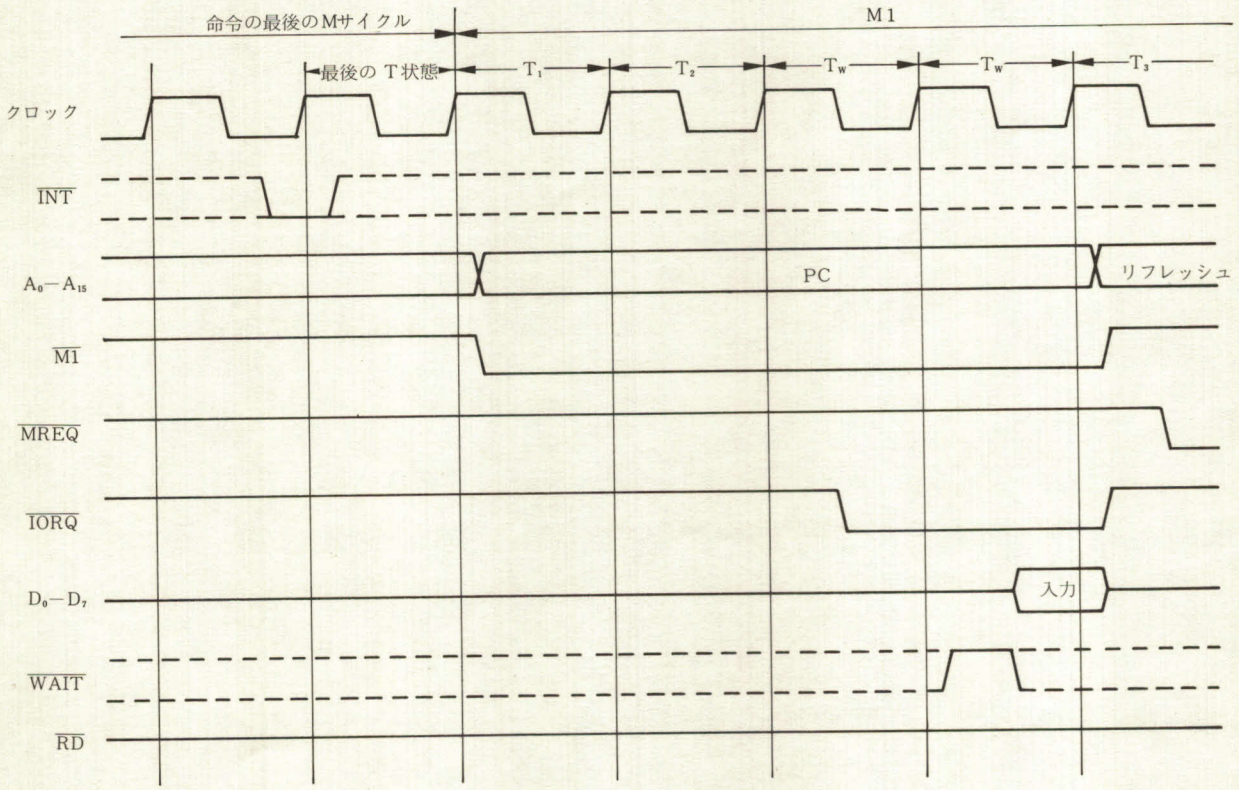


図8.17　割り込みアクノリッジ

中のデバイスがデータ・バス上にその8ビット・ベクトルを乗せられることを示す。2つの待ち状態が自動的にこのサイクルに加えられる。これは優先順位割り込み機構の実行を容易にするためである。この2つの待ち状態によって、IEI-IEO信号が安定する時間が与えられ、どの入出力デバイスが反応するかを識別できる。これについての詳細は、シャープ マニュアル Z-80 CPUの割り込み No.453および、Z-80 テクニカルマニュアル No.451を参照のこと。

RDYに対する割り込み（バス要求まえの割り込み）は、$\overline{\text{BUSRQ}}$に直接影響を与えることはない。代わりに、割り込みサービス・ルーチンで下記のコマンドをWR6に与えてこれを処理する。

- 割り込み復帰（RETI）後イネーブル　$B7_H$
- DMAイネーブル　87_H
- Z-80 DMA内の割り込みサービス中、IUSラッチをリセットするRETI命令の実行　$ED4D_H$

パルス発生： パルス発生オプションが選択される場合、オフセット値の後256バイトごとに$\overline{\text{INT}}$は“Low”に駆動される。$\overline{\text{INT}}$は、パルス制御バイトがバイト・カウンタの下位バイトに一致するDMAサイクルの期間中“Low”となり、完全な転送サイクルの期間中“Low”を保持する。ここでは、転送サイクルとは、読み出しサイクル（サーチのみまたは同時転送動作）または読み出しと書き込みサイクルを意味し、読み出しおよび書き込みサイクルの長さは、可変サイクル・オプションによって独立してプログラムできる。

第9章　用語解説

アクティブ：　DMAがアクティブの場合、DMAはバス・マスタであり、動作中か中断しているかのどちらかである。回路によってアクティブと認知される場合、信号はアクティブである。たとえば、電圧レベルが論理1（“High”）または論理0（“Low”）のいずれかで、ハードウェアまたはソフトウェアがそのいずれかの状態でこれをアクティブと認知する場合である。アクティブ“High”の信号は、RDYのようにバーなしで表され、アクティブ“Low”の信号は$\overline{\text{RDY}}$のようにバーを付加して表される。

アドレス・カウンタ：　ソースまたはデスティネーション・アドレスの計数を行う（2個のレジスタ使用）。ソース・アドレスが可変の場合、計数バイトの読み出しの直前にインクリメントまたはデクリメントする。デスティネーション・アドレスが可変の場合、計数バイトの書き込みの直前にインクリメントまたはデクリメントする。1つの動作の最初のバイトに対して、インクリメントまたはデクリメントは生じない。最初のバイトに対してはプログラムされた開始アドレスが使われる。

バッファ：　論理状態の保持、信号の増幅、あるいは信号の隔離のための手段。

バースト・モード：　入出力デバイスのRDY（または内部の強制レディ条件）がアクティブの期間中、DMAが動作を継続して行う転送またはサーチ・モード。RDYが非アクティブになるとDMAはバスを解除する。

バス：　アドレス・バス、データ・バス、制御バス、あるいはこれら3つのバス（システム・バス）すべてを表す。DMAがバス・マスタになるときは、DMAがアドレス・バス、データ・バス、およびバス制御線である$\overline{\text{RD}}$、$\overline{\text{WR}}$、$\overline{\text{IORQ}}$および$\overline{\text{MREQ}}$を制御することを意味する。

バス制御：　DMAの$\overline{\text{BUSRQ}}$および$\overline{\text{BAI}}$が同時にアクティブの場合、DMAはバスを制御する。

バイト・カウンタ：　読み出されるバイト数を計数する。カウンタは0から開始し、計数バイトの読み出し後インクリメントする。バイト・カウンタの内容がブロック長レジスタの内容と等しくなるとき、エンド・オブ・ブロックとなる。

バイトの一致：　データ・バイトとマスクされた一致バイトの一致（または比較）。

バイト・モード：　DMAがバス解放まで1バイトのみについて動作する転送またはサーチ・モード。次のバイト動作に対しては再びバスを要求する。

チャネル：　データの流れを追跡する2つのポート間の制御されたリンク。チャネルにはアドレス・カウンタ、バイト・カウンタ、および制御回路が含まれる。

動作クラス： 転送、サーチ、およびサーチ中転送（または転送／サーチ）クラス。動作モードおよび動作の方法を参照。

クリア： 論理0にすること。

クロック： システム・クロック。

クロック・サイクル： システム・クロック（CLK）の1サイクルで、慣例的には立ち上がりエッジにはじまり、次の立ち上がりで終わると定義されている。クロック・サイクル当たりTサイクル（タイム・サイクル）は1つ。2.5MHzクロックでは、クロック・サイクルの長さは400 ns。4 MHzクロックでは、クロック・サイクルの長さは250ns。

コマンド： CPUによってDMAに書き込まれる制御バイト。将来の動作機能のためにモード設定を行う制御バイトに比べ、コマンドは即時的な動作を発生させる。モードの設定と即時的動作の組み合わせが、1つの制御バイト数に実行される場合もある。

連続モード： DMAがシステム・バスを解放するまえにブロック転送を完了する転送またはサーチ・モード。転送の期間中に入出力ポートのRDYが非アクティブになるとDMAは中止するがバス制御は保持する。

制御バイト： CPUがバス・マスタで、かつデコードされた$\overline{CE}$信号を介して、DMAを周辺入出力デバイスとしてアドレスするとき、DMAに書き込まれるバイト。

デスティネーション・ポート： 転送においてデータが書き込まれるポート。ポートA、ポートBのいずれもデスティネーション・ポートとしてプログラムできる。ポート参照。

ディセーブル： DMAがディセーブルの状態のとき、DMAはシステム・バスを要求することができない。DMAイネーブル・コマンドを除いて、DMAに書き込まれるすべての制御バイトによってDMAはディセーブルされる。

DMAC： ダイレクト・メモリ・アクセス・コントローラ（またはチップ）。単にDMAと書くことも多い。

イネーブル： DMAはイネーブルされたときシステム・バスの要求ができる。現在この状態にあってバス・マスタである場合もある。イネーブル状態には、アクティブおよび非アクティブの両方が含まれる。

フロースルー： シーケンシャル転送を参照。

フライバイ： 同時転送を参照。

“High”： 論理1（高電圧電位）。

非アクティブ： DMAのイネーブルの一状態。DMAが非アクティブの場合、DMAはバス要求ができる。バス・マスタとなるとDMAはアクティブになる。信号線がそのアクティブ状態として反対の論理レベルにあるとき、信号線は非アクティブである。

割り込みベクトル： 周辺デバイスの割り込みをCPUがアクノリッジした後、そのデバイスからCPUに送られる8ビットのベクトル。Z-80 CPUシステムでは、ベクトルによって割り込みを行っているデバイスを識別し、割り込みサービス・ルーチンの開始アドレスの下位バイトを形成する。このアドレスの上位バイトは、CPU内で供給される。

ランド容量： 信号用接地に関して基板上の回路の各部の容量。

“Low”： 論理0（低電圧電位）。

Mサイクル： マシン・サイクル参照。

マシン・サイクル： DMAマシン・サイクルとは、シーケンシャル転送において、1つの読み出しまたは書き込み動作を行うに要する時間である。同時転送においては、読み出しと書き込み両方に要する時間である。CPUマシン・サイクルとは、OPコードのフェッチ、メモリ読み出し、またはメモリ書き込みなどの1つの基本動作を意味する（1つの命令が複数のマシン・サイクルを有することもある）。待ち状態の付加がない場合、DMAのマシン・サイクルの長さは、2、3、または4クロック・サイクル（Tサイクル）である。Z-80 CPUのマシン・サイクルは、待ち状態の付加がない場合、3から10クロックの間である。

動作の方法： シーケンシャルおよび同時。それぞれ、フロースルーおよびフライバイとも呼ぶ。動作モードおよび動作クラス参照。

動作モード： バイト・バースト、および連続モード。それぞれ、シングル、デマンド、およびブロック・モードとも呼ぶ。動作クラスおよび動作の方法参照。

動作： DMAが動作している場合、DMAはデータのバイトを転送、サーチのいずれかあるいは両方を行っている。この意味では、CPUとDMAのステータスの制御の転送は動作とはならない。

ポート： データのソースまたはデスティネーション。周辺入出力デバイスまたはメモリがポートとなり得る。Z-80 DMAはデータのバイトが転送されるごとに、ソースおよびデスティネーションのアドレスを発生させる。データがサーチされているだけ（転送なし）の場合は、ソース・ポートのみ使われる。

レース状態： 複数の論理回路信号の変位で、その相対的なタイミングに応じて、システムに各種の状態を発生させる。レースの結果をつねに正確に予測することはできない。

リセット： ディフォルト状態を初期値設定すること。これには論理1と論理0が含まれる。クリア参照。

RR： 読み出しレジスタ。DMAがバスを解放したとき、DMAがステータス・バイトを読み出すことができる読み出しレジスタは7個である。

シーケンシャル転送： バイトが1つの読み出しサイクルの中で、ソース・ポートから読み出され、次に別の書き込みサイクルの中で、デスティネーション・ポートに書き込まれる転送。この動作クラス中にサーチも同時に実行可能で、入出力およびメモリ・ポートのいずれの間でも転送できる。外部回路を必要としないが、同時転送に比べると速度は½となる。

セット： 開始状態に設定すること。しばしば論理1に設定する意味に用いられる。

同時転送： バイトのソース・ポートからの読み出しと、デスティネーション・ポートへの書き込みが同時に実行される転送。この動作クラスでは、同時にサーチも行えるが、少なくとも1個の外部回路ゲートを必要とし、メモリー入出力、または入出力ーメモリ間転送に限定される（メモリーメモリ、あるいは入出力ー入出力間転送は不可）。シーケンシャル転送に比べ速度は2倍となる。同時転送の実行には、DMAをサーチのみの動作クラスにプログラムし、適切な制御信号を発生させるために外部回路を使用する。

ソース・ポート： データが読み出されるポート。ポートA、ポートBのいずれもソース・ポートとしてプログラムできる。ポート参照。

ステータス・バイト： 読み出しレジスタ0（RR0）。

ストップ（停止）： DMAが停止した場合、DMAはバスを解放する。この状態になるとDMAの転送、サーチは終了する。

比較による停止： バイト一致成立による停止。

中断： 中断状態のとき、DMAはバス・マスタであっても現在は動作していない（データの転送、サーチを行っていない）。

システム・バス： アドレス・バス、データ・バス、および制御バスの組み合わせ。

Tサイクル： クロック・サイクル参照。

WR： 書き込みレジスタまたは書き込み線（$\overline{WR}$）。DMAが制御バイトの書き込みができる書き込みレジスタの数は21であるが、これらへのアクセスはその7個のサブセット、WR0－WR6によって得られる。

上記の用語の他に、本マニュアルには下記のシンボルが使われている。

アドレス・バス、A_0－A_{15}： 0から15までの並列アドレス線。

バー表記： $\overline{RDY}$。アクティブ“Low”信号（例：低電圧または論理0でアクティブ）。RDY。アクティブ“High”信号（例：高電圧または論理1でアクティブ）。

データ・バイト中のビット： D_0—D_7。データ・バス上に送られるバイト中のビット。

データ・バス： D_0—D_7。0から7までの並列データ線。

第10章　規　　格

10．1　絶対最大定格

項　　目	記号	定　格　値	単位
入　力　電　圧	V_{IN}	−0.3～＋7	V
出　力　電　圧	V_{OUT}	−0.3～＋7	V
動　作　温　度	T_{opr}	0　～＋70	℃
保　存　温　度	T_{stg}	−65～＋150	℃

10．2　電気的特性

DC 特性

（Ta＝0℃～＋70℃、V_{CC}＝＋5 V±5%）

項　　目		記　号	測　定　条　件	最小値	最大値	単　位
クロック“Low”入力電圧		V_{ILC}		−0.3	0.45	V
クロック“High”入力電圧		V_{IHC}		V_{CC}−0.6	5.5	V
“Low”入力電圧		V_{IL}		−0.3	0.8	V
“High”入力電圧		V_{IH}		2.0	5.5	V
“Low”出力電圧		V_{OL}	$\overline{BUSRQ}$ 出力　I_{OL}＝3.2 mA その他出力　I_{OL}＝2.0 mA		0.4	V
“High”出力電圧		V_{OH}	I_{OH}＝−250μA	2.4		V
消費電流	Z-80 DMA	I_{CC}	t_C＝400ns		150	mA
	Z-80A DMA		t_C＝250ns		200	mA
入力リーク電流		I_{LI}	V_{IN}＝0～V_{CC}		10	μA
3ステート出力リーク電流		I_{LOH}	V_{OUT}＝2.4 V～V_{CC}		10	μA
3ステート出力リーク電流		I_{LOL}	V_{OUT}＝0.4 V		−10	μA
入力時のデータ・バスのリーク電流		I_{LD}	0 V≦V_{IN}≦V_{CC}		±10	μA

端子容量

（Ta＝＋25℃、f＝1 MHz）

項　　目	記　号	測　定　条　件	最大値	単　位
クロック入力容量	C_Φ	被測定端子以外のすべての端子は接地	35	pF
入　力　容　量	C_{IN}		5	pF
出　力　容　量	C_{OUT}		10	pF

AC 特性(I)

ペリフェラルとして動作するとき(非アクティブ状態)

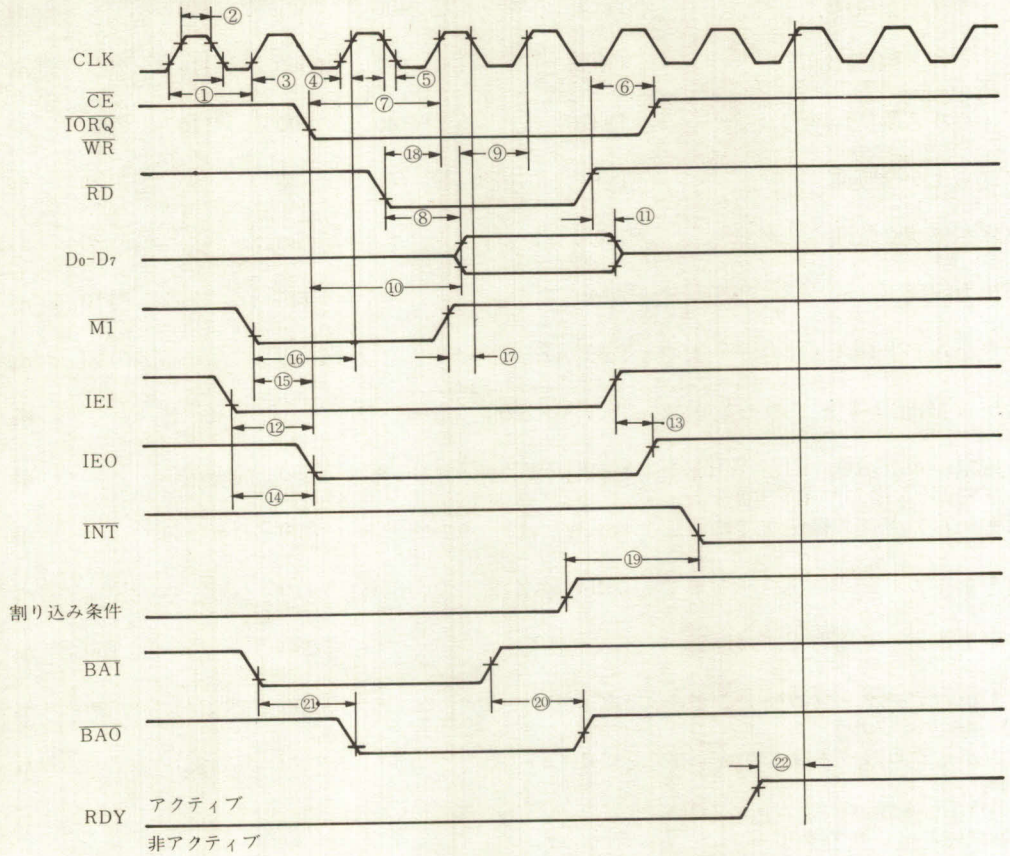
番号	項目	記号	Z-80 DMA		Z-80A DMA		単位
			最小値	最大値	最小値	最大値	
1	クロック・サイクル時間	TcC	400	4000	250	4000	ns
2	クロック・パルス幅("High")	TwCh	170	2000	110	2000	ns
3	クロック・パルス幅("Low")	TwCe	170	2000	110	2000	ns
4	クロック立ち上がり時間	TrC		30		30	ns
5	クロック立ち下がり時間	TfC		30		30	ns
6	保持時間	Th	0		0		ns
7	$\overline{IORQ}$, $\overline{WR}$, $\overline{CE}$ ↓からクロック↑までのセットアップ時間	TsC(Cr)	280		145		ns
8	$\overline{RD}$ ↓からデータ出力までの遅延時間	TdDO(RDf)		500		380	ns
9	データ入力からクロック↑までのセットアップ時間($\overline{WR}$ または $\overline{M1}$)	TsWM(Cr)	50		50		ns
10	$\overline{IORQ}$ ↓からデータ出力までの遅延時間(INT A サイクル)	TdCf(DO)		340		160	ns
11	$\overline{RD}$ ↑からデータ・フロートまでの遅延時間(出力バッファ・ディセーブル)	TdRD(DZ)		160		110	ns
12	IEI ↓から $\overline{IORQ}$ ↓までのセットアップ時間(INT A サイクル)	TsIEI(IORQ)	140		140		ns
13	IEI ↑から IEO ↑までの遅延時間	TdIEOr(IEIr)		210		160	ns
14	IEI ↓から IEO ↓までの遅延時間	TdIEOf(IEIf)		190		130	ns
15	$\overline{M1}$ ↓から IEO ↓までの遅延時間($\overline{M1}$ ↓のまえに割り込みが発生)	TdM1(IEO)		300		190	ns
16	$\overline{M1}$ ↓からクロック↑までのセットアップ時間	TsM1f(Cr)	210		90		ns
17	$\overline{M1}$ ↑からクロック↓までのセットアップ時間	TsM1r(Cf)	20		10		ns
18	$\overline{RD}$ ↓からクロックの↑までのセットアップ時間($\overline{M1}$ サイクル)	TsRD(Cr)	240		115		ns
19	割り込み発生から $\overline{INT}$ ↓までの遅延時間($\overline{INT}$ は DMA が非アクティブのときのみ生じる)	TdI(INT)		500		500	ns
20	$\overline{BAI}$ ↑から $\overline{BAO}$ ↑までの遅延時間	TdBAIr(BAOr)		200		150	ns
21	$\overline{BAI}$ ↓から $\overline{BAO}$ ↓までの遅延時間	TdBAIf(BAOf)		200		150	ns
22	RDY アクティブからクロック↑セットアップ時間	TsRDY(Cr)	150		100		ns

注1　負の最小のセットアップ値は初めの事象が2番目の事象より後に来ることができることを意味する。

ACタイミング図(Ⅰ)

ペリフェラルとして動作するとき(非アクティブ状態)

	"High"	"Low"	
CLOCK	4.2 V	0.8 V	FLOAT
OUTPUT	2.0 V	0.8 V	V=+0.5 V
INPUT	2.0 V	0.8 V	



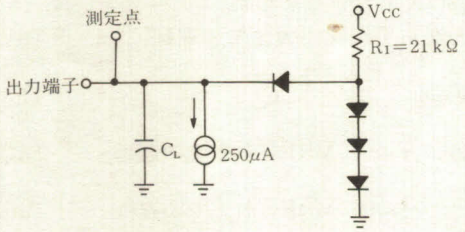
(注) この図は各信号の立ち上がり・立ち下がりのタイミングの相関を表したもので実際のタイム・シーケンスとは異なる。

ACパラメータの測定回路を右図に示す

C_L=100pF

負荷容量は最大200pFまで許されるが、

50pFの増加で遅延は10ns増加する。



AC 特性(II)

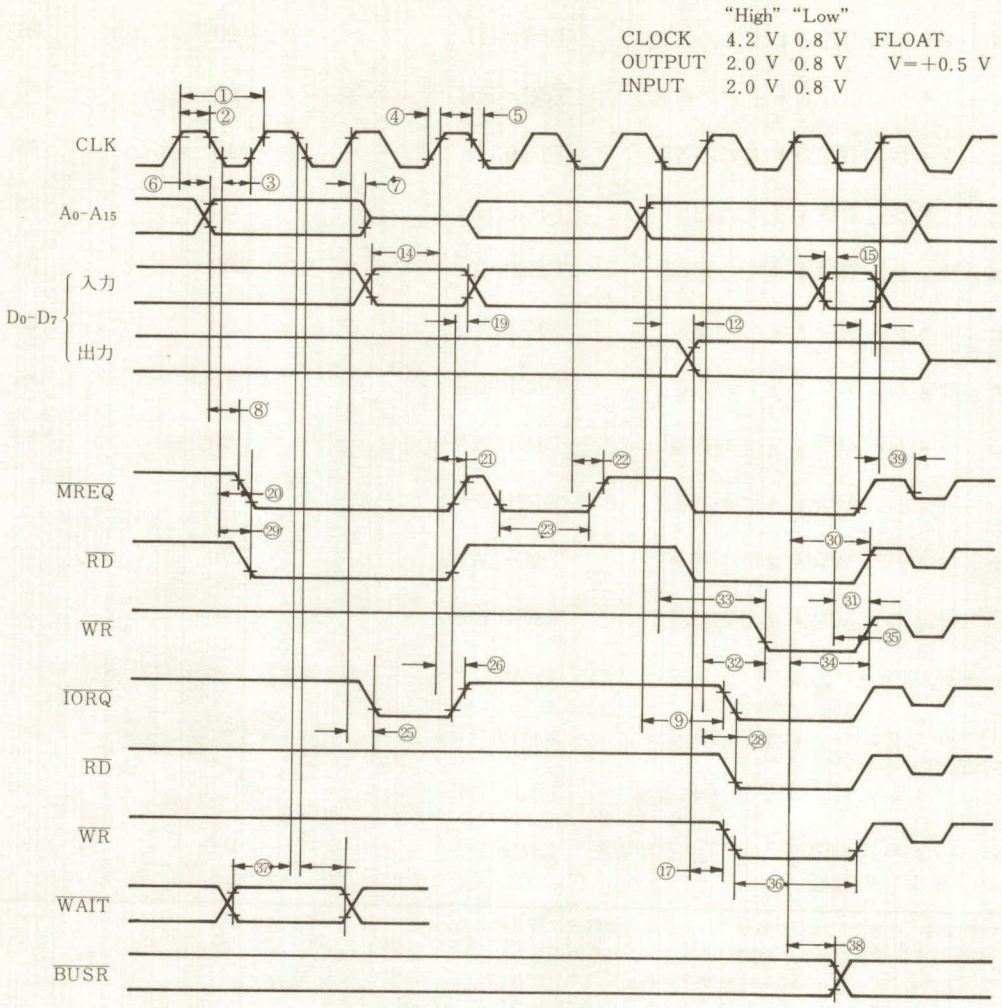
バス・コントローラとして動作するとき(アクティブ状態)

番号	項目	記号	Z-80 DMA		Z-80A DMA		単位
			最小値	最大値	最小値	最大値	
1	クロック・サイクル時間	TcC	400		250		ns
2	クロック・パルス幅("High")	TwCh	180	2000	110	2000	ns
3	クロック・パルス幅("Low")	TwCl	180	2000	110	2000	ns
4	クロック立ち上がり時間	TrC		30		30	ns
5	クロック立ち下がり時間	TfC		30		30	ns
6	アドレス出力遅延	TdA		145		110	ns
7	クロック↑からアドレス・フロートまでの遅延	TdC(AZ)		110		90	ns
8	アドレスから $\overline{MREQ}$ ↓までのセットアップ時間(メモリ・サイクル)	TsA(MREQ)	(2)+(5)−75		(2)+(5)−75		ns
9	アドレス確定から $\overline{IORQ}$, $\overline{RD}$, $\overline{WR}$↓セットアップ時間(入出力サイクル)	TsA(IRW)	(1)−80	90	(1)−70		ns
*10	$\overline{RD}$, $\overline{WR}$↑からアドレス確定までの遅延	TdRW(A)	(3)+(4)−40	100	(3)+(4)−50		ns
*11	$\overline{RD}$, $\overline{WR}$↑からアドレス・フロートまでの時間	TdRW(AZ)	(3)+(4)−60	110	(3)+(4)−45		ns
12	クロック↓からデータ出力までの遅延	TdCf(DO)		230		150	ns
*13	クロック↑からデータ・フロートまでの遅延(書き込みサイクル)	TdCr(Dz)		90		90	ns
14	データ入力からクロック↑までのセットアップ時間(クロックの立ち上がりで読み込みが終わる場合の読み出しサイクル)	TsDI(Cf)	50		35		ns
15	データ入力からクロック↓までのセットアップ時間(クロックの立ち下がりで読み込みが終わる場合の読み出しサイクル)	TsDO(WfM)	60		50		ns
*16	データ出力から $\overline{WR}$↓までのセットアップ時間(メモリ・サイクル)	TsDO(WPI)	(1)−210		(1)−170		ns
17	データ出力から $\overline{WR}$↓までのセットアップ時間(入出力サイクル)	TsDO(WPI)	100		100		ns
*18	$\overline{WR}$↑からデータ出力までの遅延	TdWr(DO)	(3)+(4)−80		(3)+(4)−70		ns
19	保持時間	Th	0		0		ns
20	クロック↓から $\overline{MREQ}$↓までの遅延	TdCf(Mf)		100		85	ns
21	クロック↑から $\overline{MREQ}$↑までの遅延	TdCr(Mr)		100		85	ns
22	クロック↓から $\overline{MREQ}$↑までの遅延	TdCf(Mr)		100		85	ns

23	パルス幅($\overline{MREQ}$ が"Low")	TwMl	(1)−40		(1)−30		ns
*24	パルス幅($\overline{MREQ}$ が"High")	TwMh	(2)+(5)−30		(2)+(3)−20		ns
25	クロック↑から$\overline{IORQ}$↓までの遅延	TdCr(If)		90		75	ns
26	クロック↑から$\overline{IORQ}$↑までの遅延	TdCr(Rf)		100		85	ns
*27	クロック↓から$\overline{IORQ}$↑までの遅延	TdCf(Ir)		110		85	ns
28	クロック↑から$\overline{RD}$↓までの遅延	TdCr(Rf)		100		85	ns
29	クロック↓から$\overline{RD}$↓までの遅延	TdCf(Rf)		130		95	ns
30	クロック↑から$\overline{RD}$↑までの遅延	TdCr(Rr)		100		85	ns
31	クロック↓から$\overline{RD}$↑までの遅延	TdCf(Rr)		110		85	ns
32	クロック↑から$\overline{WR}$↓までの遅延	TdCr(Wf)		80		65	ns
33	クロック↓から$\overline{WR}$↓までの遅延	TdCf(Wf)		90		80	ns
34	クロック↑から$\overline{WR}$↑までの遅延	TdCr(Wr)		100		80	ns
35	クロック↓から$\overline{WR}$↑までの遅延	TdCf(Wr)		100		80	ns
36	パルス幅($\overline{WR}$ が"Low")	TwWl	(1)−40		(1)−30		ns
37	$\overline{WAIT}$からクロック↓までのセットアップ時間	TsWA(Cf)	70		70		ns
38	クロック↑から$\overline{BUSRQ}$までの遅延	TdCr(B)		150		100	ns
39	クロック↑から$\overline{IORQ}$, $\overline{MREQ}$, $\overline{RD}$, $\overline{WR}$フロートまでの遅延	TdCr(Iz)		100		80	ns

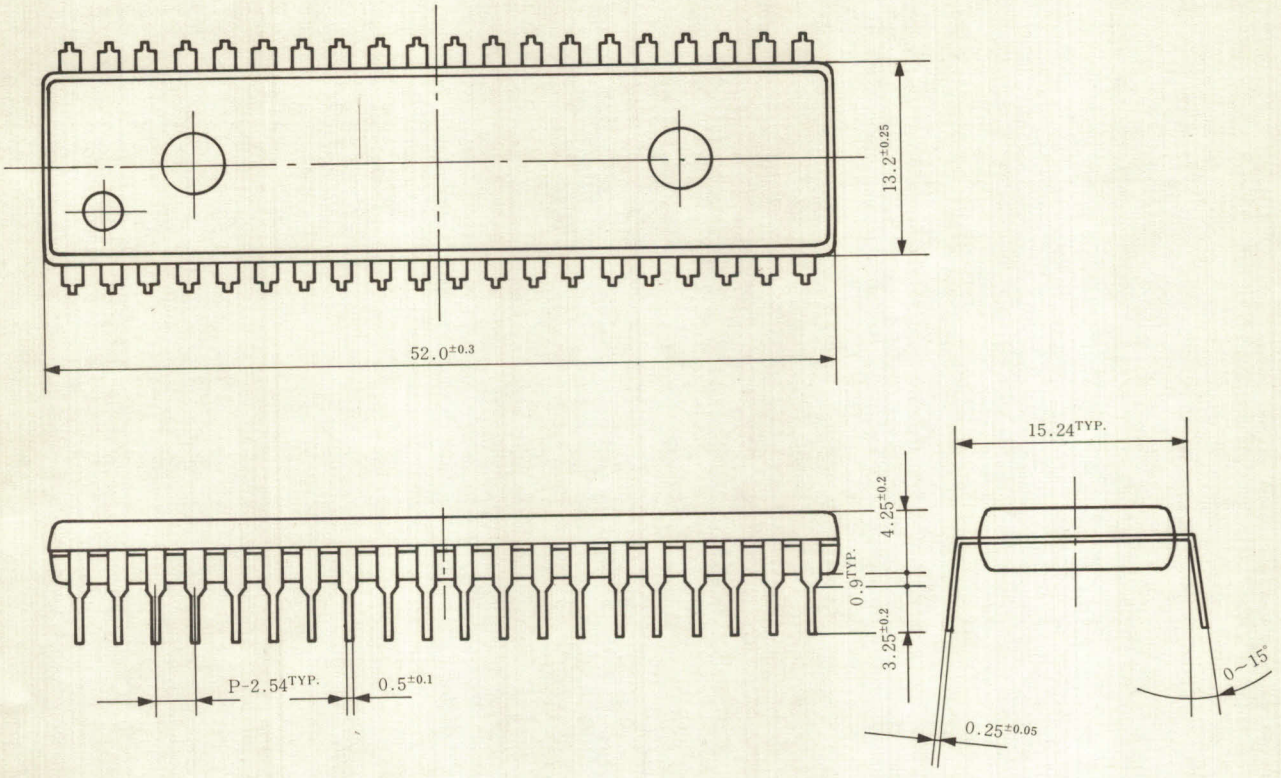
注1　括弧内の数字はこの表の他の項目番号で、それらの値は式に置き換える。
2　すべての式はDMAディフォルト(標準)タイミングを意味する。
3　データは$\overline{RD}$がアクティブの場合にデータ・バス上でイネーブルになる。
4　項目番号のまえのアスタリスク(*)はACタイミング図には表されていない項目である。

ACタイミング図(バス・コントローラとして動作するとき)



(注) この図は各信号の立ち上がり、立ち下がりのタイミングの相関を表したもので実際のタイム・シーケンスと異なる。

10．3　外形寸法図



単位：mm

（おことわり）

○本書に記載された説明事項、特性、回路例などは、シャープZ-80マイクロコンピュータファミリLSIの使用上の参考として示されたものであり、別途用意の製品仕様書の記載内容が本書の内容に優先します。LSIご使用に当たっては、必ず製品仕様書をご用命のうえ内容をご確認ください。また規格契約を必要とする場合には、製品仕様書をもってご契約ください。

○製品の改良のため予告なしに内容の一部を変更することがあります。

●落丁・乱丁本はお取り替え致します。

SHARP

シャープ株式会社

本　　社	〒545	大阪市阿倍野区長池町22番22号	☎ (06)621－1221 (大代表)
電子部品事業本部	〒632	奈良県天理市櫟本町2613番地の1	☎ (07436)5-1321 (大代表)
営業本部	〒545	大阪市阿倍野区長池町22番22号	☎ (06)621－1221 (大代表)
東部地区営業	〒162	東京都新宿区市谷八幡町8番地	☎ (03)260－1161 (大代表)
長野駐在	〒399-65	長野県松本市芳野8番14号	☎ (0263)27-1677
北関東駐在	〒361	埼玉県行田市門井町2丁目5番地	☎ (0485)53-3127
中部地区営業	〒454	名古屋市中川区山王3丁目5番5号	☎ (052)332-2681 (代表)
西部地区営業	〒545	大阪市阿倍野区長池町22番22号	☎ (06)621－1221 (大代表)

昭和57年8月作成　電部-963 Z1