ＭＳＸテクニカルガイドブック

第　四　版

暫定版

ＡＳＣＡＴ編集部

ＭＳＸ　ＴＥＣＨＮＩＣＡＬ　ＧＵＩＤＥＢＯＯＫ
ＴＨＥ　ＦＯＵＲＴＨ　ＥＤＩＴＩＯＮ

１９９２

ＡＳＣＡＴ
Ａｓｈｉｋａｇａ，ＮＩＰＰＯＮ

装丁●著者自装

ＭＳＸテクニカルガイドブック　　序文

◆

ＭＳＸでプログラムを組む場合、主に「ＭＳＸ２テクニカルハンドブック」などを参考にします。しかし、この本には漢字ＲＯＭのアクセスの仕方や、ソフトウェアがフロッピーディスクドライブの存在を確かめる方法が書いてありません。「月刊ＭＳＸマガジン」などで、アクセスの仕方などが紹介されたのですが、あちこちの号でばらばらに取り上げられたため、検索に非常に手間がかかりました。また、古い号はすでに品切れになっており、新しいユーザーがバックナンバーをそろえることは、ほとんど不可能になっています。
そこで、あちこちで紹介されたそれらの技術的資料のうち、「ＭＳＸ２テクニカルハンドブック」には掲載されていないものを、検索しやすくまとめてみました。
なお、この資料は「ＭＳＸ２テクニカルハンドブック」を持っていることを前提にして書かれています。

お断わり

本書の内容は、当方が独自に編集・制作したもので、株式会社アスキーは一切関与していません。本書に対するご質問、お問い合わせは直接著者までお願いします。
なお、本書に関するご質問に対して返事を要求される場合は、必ず切手を貼った返信用封筒を同封してください。往復はがきですと、返事が書き切れない場合があります。
（※ PDF 版では住所を表記していませんので、奥付のアドレス宛に連絡をお願いします）

本書は「ＭＳＸ２テクニカルハンドブック」をすでに持っていることを前提にして書かれています。ＭＳＸ、ＭＳＸ２の基本的な情報については、本書ではページ数の関係から紹介していません。

同じく、ＭＳＸ２＋になってから拡張された漢字処理、ＹＪＫ方式の画像処理についても、本書では紹介するだけで扱いません。これらの機能については、「ＭＳＸ２＋パワフル活用法」（アスキー社刊）を参考にしてください。

文章中、特に断わりのない場合には、すべてのＭＳＸで有効です。

本書の内容によって生じたいかなる事態にも、当方は一切の責任をおいません。

本書の作成に当たっては青春という名の貴重な時間が大量に投資されています。本書を利用して開発したソフトウェアには、参考文献の欄にでも本書の題名を記して頂くよう、お願いします。

凡例

本書では次のような用語を使用します。

ＭＳＸ１（意味　ＭＳＸ－ＳＹＳＴＥＭ　ＶＥＲ．１．０）
ＭＳＸ－ＤＯＳ１またはＤＯＳ１（意味　ＭＳＸ－ＤＯＳ　ＶＥＲ．１．０３）
ＤＯＳ２（意味　日本語ＭＳＸ－ＤＯＳ２）
ＤＯＳ上（意味　ＭＳＸ－ＤＯＳでページ０と１がＲＡＭに切り換わっている状態）
ターボＲまたはturboR（意味　ＭＳＸ turboＲ）

ＭＳＸテクニカルガイドブック　　目次

参考文献一覧

本書の作成に当たっては、主に次のようなものを参考にさせていただきました。

「月刊ＭＳＸマガジン」（アスキー社）
「ＭＳＸ２テクニカルハンドブック」（アスキー社）
「月刊アスキー」（アスキー社）
「年刊ア・スキー」（ア・スキー社）
「ＭＳＸテクニカルデータブック増補改訂版（国立国会図書館蔵書版）」（アスキー社）
「バックアップ活用テクニック」（三才ブックス）
「ＶＪＥ－８０Ａ仕様書・第２版」（アスキー社）
「ＭＳＸ拡張ＢＩＯＳ仕様書」（アスキー社）
「ＭＳＸ－ＳＥＲＩＡＬ２３２ユーザーズマニュアル」（アスキー社）
「Ｒ８００ユーザーズマニュアル暫定版」（アスキー社）
「ＭＳＸ－ＭＵＳＩＣ　ＦＭ－ＢＩＯＳ仕様書　第１版」（アスキー社）
「日本語ＭＳＸ－ＤＯＳ２リファレンスマニュアル」（アスキー社）
「応用ＣＰ／Ｍ」（アスキー社）
「ディスク徹底活用術」（アスキー社）
「プリンタ徹底活用法」（アスキー社）
「ＭＳＸ－Datapack」（アスキー社）

第１部　　ＢＡＳＩＣ

第１章　ＢＡＳＩＣ

ここでは、ＭＳＸのＢＡＳＩＣで、あまり知られていない機能について説明します。

１．１．１　エスケープシーケンス

エスケープシーケンスは、主に通信などで使われますが、ＢＡＳＩＣのＰＲＩＮＴ命令、ＢＩＯＳコールなどでも有効です。大抵の機能はほかのコントロールコードで代用できるため、あまり使われていませんが、ＥＳＣ＋”Ｋ”，ＥＳＣ＋”Ｌ”などを使うと、テキスト画面を上下にスクロールさせることができます。巻末に一覧表を載せておきましたので、いろいろと試してみてください。なお、なぜだか知りませんが、「ＭＳＸ２テクニカルハンドブック」では、ＥＳＣ＋”Ｉ”の説明が抜け落ちています。

１．１．２　ＤＩＳＫ－ＢＡＳＩＣの隠しコマンド

ＭＳＸのＢＡＳＩＣでは、ＤＳＫＩ＄，ＤＳＫＯ＄というコマンドがサポートされています。これらを使うと、ＤＩＳＫ－ＢＡＳＩＣ上で、ディスクをセクター単位で直接読み書きできます。これらの機能は、余り説明されたことがありませんが、正式な仕様書にはきちんと機能が定められているので、普通に使っても問題はないでしょう。

●ＤＳＫＩ＄（＜ドライブ番号＞，＜論理セクター番号＞）
＜ドライブ番号＞で指定されたディスクドライブの＜論理セクター番号＞で指定されたセクターを読み込みます。

●ＤＳＫＯ＄　＜ドライブ番号＞，＜論理セクター番号＞
＜ドライブ番号＞で指定されたディスクドライブの＜論理セクター番号＞で指定されたセクターに書き込みます。ライトプロテクトされていても何も表示しません。

両方とも、Ｆ３５１Ｈから２バイトの番地に書かれているアドレスに、１セクター単位で読み書きします。この２バイトを書き換えると、違うアドレスに読み書きできるようです。このエリアは、ＯＰＥＮ，ＣＬＯＳＥ，ＦＩＬＥＳ，ＰＲＩＮＴ＃などのディスク関係のステートメントを実行したときに破壊されます。なお、ドライブ番号は、０がディフォルトドライブで、１がＡ，２がＢドライブといったような指定の仕方をします。
実際には、「　Ａ＄＝ＤＳＫＩ＄（０，１４）　」というふうに使います。（Ａ＄はダミー）

第２章　ＲＡＭディスクスイッチ

ＭＳＸ２では、ＲＡＭディスク機能が追加され、あらかじめ「ＣＡＬＬ　ＭＥＭＩＮＩ」命令で領域を確保しておけば、裏ＲＡＭ（ＲＡＭの、０００００Ｈ番地から７ＦＦＦＨ 番地までの、普段はＢＡＳＩＣ－ＲＯＭに隠れて直接読み書きできない領域）をＲＡＭディスクとして使うことができます。実際にはアスキーファイルしか読み書きできないうえ、フロッピーディスクよりもアクセスが遅くて、使い物になりませんでしたが、ディスクドライブを持っていないＭＳＸ２のユーザーなどは実際に使っているようです。
ところで、ソフトの中には、裏ＲＡＭを直接読み書きして、機械語サブルーチンを置いたり、データを置いたりしているものや、「ＭＳＸベーしっ君」（ログイン掲載版）などのように、拡張ステートメントをＲＡＭのページ１（裏ＲＡＭの、４０００Ｈから７ＦＦＦＨ番地まで）に置いて、ＢＡＳＩＣのＣＡＬＬ文で拡張ステートメントを呼び出すものがあります。これらのソフトとＲＡＭディスク機能を同時に使用すると、裏ＲＡＭに置かれたプログラムが書き換えられて暴走します。これを防ぐためには、次のような処理を行ないます。

●機械語プログラムが裏ＲＡＭを使用する場合はＭＡＩＮ－ＲＡＭのＦＤ０９Ｈ番地のビット５をセットします。

こうすると、「ＣＡＬＬ　ＭＥＭＩＮＩ」命令が実行できなくなります。
なお、ＦＤ０９Ｈ番地のビット６がセットされているときは、ＲＡＭディスクが裏ＲＡＭを使っているので、機械語プログラムが裏ＲＡＭを使うことはできません。
このほかにも、裏ＲＡＭを機械語プログラムとＲＡＭディスクが同時に使う方法もあるようです。

第３章　ＢＡＳＩＣの拡張

ＭＳＸでＢＡＳＩＣの機能を拡張する場合には、ＤＥＦＵＳＲ命令とＵＳＲ関数で機械語サブルーチンを呼び出す方法が一般的です。しかし、いくつもの機械語サブルーチンを使う場合や、いくつもの引数を引き渡す機械語サブルーチンでは、変数が一つしか使えないＵＳＲ関数では役不足です。
そこで、ＣＡＬＬ文によってステートメントを拡張することになります．ＣＭＤ命令を拡張しているアプリケーションソフトもいくつかありますが、後で面倒なことが起こらないように、ＣＡＬＬ文を使ったほうがいいようです。
ところで、「ＭＳＸ２テクニカルハンドブック」では、第５部７章３節で、ＣＡＬＬ文の拡張の方法が記されていますが、これはＲＯＭカートリッジがＣＡＬＬ文を拡張する方法です。実際には１．２で述べたように、裏ＲＡＭにＣＡＬＬ文拡張ルーチンを置くこともできます。もちろん、ＭＳＸ１では、裏ＲＡＭに拡張ステートメントを置く場合には、そのソフトウェアはＲＡＭ６４Ｋの機械でないと動作しないことになります。
裏ＲＡＭに拡張ステートメントを置くときには、次のように起動させます。

●裏ＲＡＭにＣＡＬＬ文拡張ステートメントを置く場合の起動方法
１．まず、裏ＲＡＭのあるスロットを探します。（裏ＲＡＭを探す方法は第２部第１章で詳しく述べます。
２．拡張ステートメントのプログラムを、９０００Ｈ－ＣＦＦＦＨ番地にロードします。（別に９０００Ｈ－ＣＦＦＦ番地でなくても、例えば８０００Ｈ－ＢＦＦＦ番地でも構いませんが、起動プログラムをＢＡＳＩＣで書けるので、このようにするのがよいでしょう）
３．拡張ステートメントのプログラムを、４０００Ｈ－７ＦＦＦＨ番地に転送します。（なぜ直接４０００Ｈ番地にロードしないのかというと、ＢＡＳＩＣのＢＬＯＡＤ命令では、直接裏ＲＡＭにロードできないからです）
４．スロットアトリビュート（ＦＣＣ９Ｈ＋裏ＲＡＭの基本スロット番号＊１６＋裏ＲＡＭの拡張スロット番号＊４＋ページ番号［必ず１になる］）に、２０Ｈを書き込みます。例えば、裏ＲＡＭが拡張スロット３－１にあった場合には、ＦＣＣ９Ｈ＋３＊１６＋１＊４＋１＝ＦＣＦＤＨ、つまり、ＦＣＦＤＨ番地に２０Ｈを書き込めばいいことになります。
５．ＢＡＳＩＣのコマンドレベルに戻ります。

これで、ＣＡＬＬ文拡張プログラムが裏ＲＡＭに置かれました。実際にＣＡＬＬ文を使えるようになっているはずです。
ＲＯＭカートリッジでＣＡＬＬ文を拡張する場合には、ＢＡＳＩＣが電源投入時にスロットアトリビュートに値を書き込んでくれるので、ユーザー側が値を書き込む必要はありません。しかし、裏ＲＡＭに拡張プログラムを置く場合には、ＢＡＳＩＣが値を書き込んだ後にプログラムをロードするわけですから､ユーザー側が値を書き込むことになります。
ＣＡＬＬ文拡張プログラムの作り方については、「ＭＳＸ２テクニカルハンドブック」の第５部７章に書かれています。スロットアトリビュートの詳しい意味についても、こちらを参考にしてください。

拡張されたＣＡＬＬ文中でＢＡＳＩＣのエラーを起こす方法は以下のとおりです。
Ｅレジスタにエラー番号、ＨＬレジスタに現在のテキストポインタの値を入れて、ＢＡＳＩＣの４０６ＦＨ番地をインタースロットコールする

エラーを起こす際、ＢＩＯＳのＣＡＬＢＡＳ（０１５９Ｈ／ＭＡＩＮ）を使うと安全です。

第２部　　プログラム開発上の注意

第１章　開発上の主な注意

ＭＳＸでは、ほかの機種よりもプログラムを開発するときに注意すべきことが多く、下手にプログラムを組むと、別の会社の機械では動かなかったというような現象がよく起こります。ここでは、基本的なシステムでの注意事項を述べます。特定のオプション機能については、別なところで述べます。

２．１．１　ＲＯＭカートリッジの暴走

ディスク内蔵機などで、ＲＯＭカートリッジが暴走する現象が知られています。これは主に、内蔵のディスクインターフェイスがスロット０の拡張スロットに置かれているＹＩＳ－８０５や、ＨＢ－Ｆ５００で起こります。シフトキーを押しながら立ち上げれば、ちゃんと立ち上がるのですが、はっきりいって不便です。これは、ＲＯＭカートリッジよりディスクインターフェイスが先に初期化を済ませてしまい、フリーエリアが減っているにもかかわらず、そのままプログラムを実行してしまうために起こります。これを回避するためには、ＲＯＭカートリッジが最初に

　　ＬＤ　ＳＰ，Ｆ３８０Ｈ

としてスタックを初期化すればよいわけです。ただし、これはカートリッジヘッダのＩＮＩＴから直接実行を開始する場合です。また、ディスクドライブなどの機能を全く使わないプログラムの場合だけ、この方法が使えます。このほかの場合にはスタックの場所を変更するなどの方法が必要になります。

２．１．２　裏ＲＡＭの検索

裏ＲＡＭをワークエリアに使ったり、機械語サブルーチンを置いたりする場合には、裏ＲＡＭのあるスロットを探さなくてはなりません。中には、ページ３のＲＡＭと同じスロットに裏ＲＡＭがあると思い込んでいて、裏ＲＡＭが別のスロットにある場合や、拡張ＲＡＭカートリッジを使っている場合に動かないソフトがありました。コンピューターの専門家がそんな間違いをするとはあまり思えないのですが、思い込みが強すぎるとこういうことになってしまうようです。ＭＳＸでは、理論的に、すべてのページのＲＡＭが、全く違うスロットにあることがありえますので、十分注意してください。

ところで、ディスクをつないでいる場合と、そうでない場合は、裏ＲＡＭの検索の仕方が変わります。順に説明しましょう。

１．ディスクがつながっている場合。

このときには、Ｆ３４１Ｈ番地からＦ３４４Ｈ番地までにＲＡＭのあるスロット番号が書かれています。Ｆ３４１Ｈ番地がページ０の、Ｆ３４４Ｈ番地がページ３のＲＡＭのあるスロットということになります。ただし、ディスクのブートセクターを書き換えて立ち上げた場合には、ゴミが入っている場合があります。また、ＭＳＸ１で、ＲＡＭ３２Ｋ以下の機械では、裏ＲＡＭが存在しないために裏ＲＡＭのスロットを調べても意味のある値は入っていません。

２．ディスクがない場合

ディスクがない場合や、本体のＲＡＭが６４Ｋであるかを調べるときは、次のような方法を使います。

調べたいスロットに値を書き込んで、読みだせるかどうかをチェックする。

かなり原始的な方法ですが、これ以外に有効な方法がありません。

具体的には、調べたいスロットから値を読んで、それを反転させて書き込み、再び読んで書き込んだ値が読み込めるかどうかをチェックします。そして、元の値を再び書き込みます。これは、ＤＯＳのワークエリアを書き換えたりすると異常な動作をするためです。

これを、００１０Ｈ番地から０００１Ｈ番地まで、０４１０Ｈ番地から０４０１Ｈ番地

まで、以下、３Ｃ１０Ｈ番地から３Ｃ０１Ｈ番地まで繰り返し、すべての番地で正常に読み書きできたときには、そのスロットのそのページには、ＲＡＭが存在することになります。ページ１のＲＡＭの有無を調べるときには４０１０Ｈ番地から同じように調べてください。これは、ＭＳＸマガジン１９８７年９月号に紹介されたアルゴリズムです。なぜ、検索するアドレスが飛び飛びなのかというと、例えば７ＦＦＤＨ番地などはディスクインターフェイスのメモリマップドＩ／Ｏなどが使っているためです。また、特定の１バイトにＲＡＭがあっても､そのＲＡＭはソフトが使う特殊なＳ－ＲＡＭである場合があります。そこで１６Ｋバイトほぼ全域を検索するようです。また、将来、Ｓ－ＲＡＭを３２Ｋバイト搭載したゲーム用データ保存カートリッジや音源カートリッジのようなものが、メモリマップドＩ／Ｏでない方式で出てくることも考えられますが、ソフトの説明書に「余分なＳ－ＲＡＭカートリッジやＲＯＭカートリッジは外しておいてください」と書いておけばいいでしょう。

２．１．３　ＶＤＰ

ここでは、ＶＤＰを使用する際、特に問題になりがちな機能について簡単に箇条がきします。

ＭＳＸ２になってから、ＶＤＰにビットブロックトランスファという、ＶＲＡＭからメモリやディスクに直接データを転送できる機能がつきました。ただ、ＶＲＡＭと裏ＲＡＭ間では転送が出来ません。（ＢＩＯＳを使った場合）これは、この機能が使われるときは、ＲＯＭが表に出ているためです。ＤＯＳ上でこの機能を使うときは注意してください。

ＭＳＸ２ではパレット機能がつきました。ＶＤＰの命令や、ＢＩＯＳのＳＥＴＰＬＴで、パレットを変えることが出来ます。ただ、ＢＩＯＳのＳＥＴＰＬＴでパレットを変更するときは、システムが自動的にパレットテーブルをＶＲＡＭ上の特定のアドレスに作成します。したがって、ハードウェアスクロールを使うソフトなどが、パレットをＢＩＯＳで書き換えようとすると､画面にゴミが出てしまいます｡カラーパレットを変えるソフトでは、パレット設定はＶＤＰを直接操作して行なってください。この方法については､「ＭＳＸ２テクニカルハンドブック」１６３ページに掲載されています。

ＭＳＸ２では、画面を切り替えても画面の初期化は行なわれません。画面の初期化はソフトウェア側で行なってください。特に、画面は消しても、スプライトを初期化していないために、画面にゴミスプライトが出るソフトがあります。スプライトの初期化は忘れずにやってください。

「ＭＳＸ２テクニカルハンドブック」のＶＤＰ関係の追加説明です。２１０ページの図４．６６で､「R#5　A14 A13 A12 A11 A10　1　1　1」と書いてありますが、本当はこうでなくて　「R#5　A14 A13 A12 A11 A10　A9　1　1」です。ただし、A9には必ず１を指定しなければなりません。それじゃあ、本と変わらないじゃないか、と思う人がいるかも知れませんが、全然違います。例えば、R#5に&B11110111を、R#11に&B00000011を代入したとすると、マニュアルの解釈では、アトリビュートテーブルに&H1F800を代入するという意味になってしまいますが、実際にはそのような場所にはアトリビュートテーブルは作成されず、&H1FA00に作成されます。このあたりはかなり苦しんだ人が大勢いて、このおかしな動作のために機械語からスプライトを扱うことをあきらめた人を著者は何人か知っています。もう一か所、この本にはとんでもない間違いがあるのですが、そちらのほうは第８部のほうを参照してください。

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

この本の初版本にはあまり意味のない情報だけをひたすら並べていたコーナーがありました。第２版では、そのような部分は真っ先に削除されてしまう予定でしたが、少しは役に立つ情報も載っていたので、このようなはみ出しコラムとして残ることができました。何といってもページの余った部分に陣取っているので、長い能書きをたれている暇がありません。とっとと本文に突入します。（とかいいながら５行も使ってしまった）

システムワークエリアのＮＥＷＫＥＹ（ＦＢＥ５Ｈ）にはキーボードマトリクスの状態がそのまま入っているのでキーの情報がリアルタイムで読み込めます。キーボードマトリクス表はＭＳＸ２テクニカルハンドブックの２８４ページに載っています。

ＢＡＳＩＣのプログラムをロードしてからＴＸＴＴＡＢ（Ｆ６７６Ｈ，２）の値＋２、＋３番地に２５５を書き込むとそのプログラムはＬＩＳＴが取れなくなりますが、普通に実行できます。ここの２バイトに０を書き込むとＬＩＳＴが取れるようになります。

第２章　システムのバグとその回避

ＭＳＸにはシステムにいくつかのバグがあり、プログラムにバグがないのに正常に動作しないことがあります。
ここでは、それらのバグの回避方法について説明します。
なお、ＲＳ－２３２Ｃのバグについては、第４部を参照してください。

２．２．１　サブＲＯＭコール

ＭＳＸ２で、ＤＯＳ上からサブＲＯＭをコールすると暴走することがあります。（しないこともあります）これは、ＤＯＳのスロット切り換えルーチンにバグがあるためです。
したがって、ＤＯＳからサブＲＯＭを呼び出すときは、次のような、大変に面倒な方法を使わなくてはなりません。

●ＤＯＳからサブＲＯＭを呼び出す方法
１．ページ０のＲＡＭがスロット０の拡張スロットにある場合と、ＭＳＸ２バージョンアップアダプターを使っている場合は、ＤＯＳのＣＡＬＳＬＴ（００１ＣＨ）ルーチンで直接呼び出します。

２．そのほかの場合は、スロット切り換えプログラムをページ２かページ３に置いて、サブＲＯＭを呼び出すときは直接サブＲＯＭを呼び出すのではなく、そのスロット切り換えプログラム（後述）を呼び出します。

ページ０のＲＡＭがスロット０の拡張スロットにある場合は､異常動作が起きないため、普通にＣＡＬＳＬＴでサブＲＯＭを呼び出せます。また、バージョンアップアダプターでは、ＤＯＳのスロット切り換えのバグの対策が立てられているので、これも普通にサブＲＯＭが呼び出せます。

切り換えプログラムの内容ですが、まずページ０をＤＯＳのＥＮＡＳＬＴでメインＲＯＭに切り換え、そこから値をいれてＥＸＴＲＯＭルーチンをコールします。サブＲＯＭから実行が帰ってきたら、（この時点ではページ０はメインＲＯＭになっている）メインＲＯＭのＣＡＬＳＬＴでページ０をＲＡＭに切り換えます。なぜＥＮＡＳＬＴを使わないかというと､メインＲＯＭのＥＮＡＳＬＴはページ０を切り換えることができないためです。（ＤＯＳのＥＮＡＳＬＴはページ０も切り換えることができます）
この、ページ２またはページ３にスロット切り換えプログラムを置く方法は、スロット０の拡張スロットにＲＡＭがある場合に暴走しますので、プログラムでは必ず二つの方法を別々に用意してください。

このような面倒なことをしなければ、ＤＯＳからサブＲＯＭは呼び出せません。サンプルプログラムに､ＤＯＳからサブＲＯＭを呼びだすプログラムを用意しておきましたので、そちらのほうも参考にしてください。なお、こんな面倒な方法は使っていられない、という方は、ＤＯＳからはサブＲＯＭは呼びださないか、Ｉ／ＯのＡ８Ｈ番地とＦＦＦＦＨ番地のメモリマップドＩ／Ｏを直接操作してサブＲＯＭを呼びだしてください。これらのＩ／Ｏポートは、将来にわたっても変更されることはないので、安心して直接操作することができます。
なお、ＭＳＸ２＋，turboRでは、異常動作が起きないようなスロット構成をするように決められましたので、安心してサブＲＯＭを直接インタースロットコールできます。

２．２．２　割り込みフック

ＭＳＸではキー入力やいろいろな処理を割り込みによって処理しています。そして、割り込みフックを書き変えれば、ユーザーが自分で割り込みを処理できるようにもなっています。（フックの使い方に関しては「ＭＳＸ２テクニカルハンドブック」第２部４章を参照）ところが、ＤＯＳ上で割り込みフックを使うと、一部の機種では暴走することがあります。これは、割り込みプログラムをＤＯＳが処理する場合に、一時的にページ０がメインＲＯＭに切り換わるために、スロット構成によってはうまく切り換えが出来ないためです。これを回避する方法は次のとおりです。

●ＤＯＳ上でプログラムを組む場合、割り込みプログラムはページ０には置かない。また、割り込みフックにＪＰ命令を置かない（必ずＲＳＴ　３０Ｈでインタースロットコールする）

この現象も、２．２．１と同じく、ＤＯＳのインタースロットコールルーチンに異常があるのが主な原因です。異常を起こす機種は、ソニーＨＢ－Ｆ５００、ヤマハＹＩＳ－８０５などの、ページ０のＲＡＭがスロット０の拡張スロットにある機種です。

２．２．３　メガＲＯＭ

メガＲＯＭは、ＲＯＭ内部のあるアドレスがメモリマップドＩ／Ｏになっていて、そこにＬＤ命令などで値を書き込んで、８Ｋバイトまたは１６Ｋバイトごとにメモリを切り替えることができるＲＯＭです。最大３２Ｍビットまでのメモリが一本のＲＯＭカートリッジの中に納められます。拡張スロットは使っていないため、拡張スロットに差すことも出来ます。切り換え方式が何種類かあって、会社ごとにメモリマップドＩ／Ｏのアドレスや切り換え方が微妙に違います。

この、メガＲＯＭですが、一部のＭＳＸ１では、動作不良を起こすことが知られています。この原因ははっきりと調べたわけではないのですが、本体からスロットに信号を送るときに、一部の機種では特殊な場合に信号をカットしてしまうことがあるためと、バスのタイミングが機種ごとにかなりばらつきがあるためのようです。これは、ハードをメーカーに修理に出すほかに回避する方法がありません。また、メガＲＯＭを差すと、動作はしてもかなり機嫌が悪くなる機種もあります。ＭＳＸ２ではバスのタイミングなどが統一されているので関係ありません。

２．２．４　Ｔ社の機種のみに現われる現象

Ｔ社のＭＳＸは、電源をいれてからＢＡＳＩＣが立ち上がるまでの間に、画面に女性の顔のデジタイズのようなものが一瞬だけ現われることがあります。この場合、その機械には悪霊が取り憑いていますので、近くの神社でお祓いを受けることが必要になります。（メーカーに修理に出しても、異常なしといわれてしまう）

これはＴ社の機械だけに現われる現象なので、そのほかの機種では関係ありません。著者の回りにはＴ社の機械を持っている人物がいないので、直接この現象を確かめたわけではないのですが、放っておくとワークエリアが壊されたり、ＶＤＰやＩ／Ｏポートに出るはずのないデータが出力されたりするようです。

２．２．５　ソフトウェアリセット

ＭＳＸでは、０番地ジャンプでリセットすると、（ソフトウェアリセットという）漢字ＲＯＭが使えなくなることがあります。これは、デバイスイネーブル機能が原因となっています。

デバイスイネーブルというのは、バス競合を防ぐための機能です。例えば、内蔵の漢字ＲＯＭがある機種に、外付けの漢字ＲＯＭを接続すると、同じＩ／Ｏポートに二つの機器が差さっていることになり、漢字ＲＯＭからの信号が競合して、最悪の場合ＭＳＸ本体を破壊することがあります。これを防ぐためにＩ／ＯポートのＦ５Ｈ番地に用意されたのがデバイスイネーブルで、これの働きは次のようになっています。

電源が投入されると、ＢＩＯＳは内蔵の漢字ＲＯＭを切り離した状態で漢字ＲＯＭの状態を調べます。内蔵の漢字ＲＯＭは切り離されているので、漢字ＲＯＭが存在すればそれは外付けの漢字ＲＯＭということになります。

外付けの漢字ＲＯＭがなかった場合、ＭＳＸはＩ／ＯポートのＦ５Ｈ番地のビット０に１を書き込んで、内蔵の漢字ＲＯＭを有効にします。（「ＭＳＸ２テクニカルハンドブック」３９１ページを参照）これによって、漢字ＲＯＭを内蔵した機種でも、外付けの漢字ＲＯＭを差すことが出来るわけです。ＲＳ－２３２Ｃにも同じような機能があります。

ところが、プログラムが“ＪＰ　０Ｈ”などで０番地に飛んだときは、内蔵漢字ＲＯＭが接続された状態で漢字ＲＯＭの状態を調べるルーチンにいってしまいます。すると、ＭＳＸは内蔵漢字ＲＯＭを外付けだと思って切り離してしまいます。こうして漢字ＲＯＭが使えなくなるわけです。電源が投入されたときや、リセットボタンが押されたときは内蔵漢字ＲＯＭが切り離された状態で外付けのものがあるかどうかを調べるルーチンに行くの

で問題はないのですが、プログラムがリセットをかけたいとき、すなわち０番地ジャンプをするときには、何らかの処置を施さなくてはなりません。具体的には次のようにします。

●ソフトウェアがリセットをかけるときは、次のようにする。

```
ＬＤ　　Ａ，００Ｈ
ＯＵＴ　（Ｆ５Ｈ），Ａ
ＪＰ　　０Ｈ
```

もともとＩ／ＯのＦ５Ｈ番地は「ＭＳＸ２テクニカルハンドブック」にも書かれているようにユーザーが読み書きしてはいけないことになっているので、あまりよい方法ではありませんが、これ以外には回避する方法がありません。ちなみに、この処理を行なっても動作不良が直らない可能性がありますが、それはその機種ではＩ／ＯのＦ５Ｈ番地の動作がほかの機種と違うためです。（もともと直接アクセスすることを考えていないのでこういうこともありうる）
なお、ＭＳＸ２＋では、ここに０を書き込めば必ず内蔵のものが切り離されるので、安心して０を書き込めます。

また、ＭＳＸ２＋では、リセットの状態を調べることができるようになりました。これには、増設されたＢＩＯＳを使います。

●ＭＳＸ２＋でリセットの状態を調べるには、次のようにする。

```
ＣＡＬＬ　０１７ＡＨ
```

このとき、返ってきたＡレジスタのビット７が立っていれば”ＪＰ　０Ｈ”でのリセット（ソフトウェアリセット）で起動されたと分かります。このビットが立っていなければハードウェアリセットです。ソフトのほうがハードウェアリセットかソフトウェアリセットか調べられないといろいろと不都合が生じるので、このような機能が増設されたようです。実際には、Ｉ／ＯのＦ４Ｈ番地にリセットを調べるハードが追加されて、このＢＩＯＳの内部では、そこを読み書きしています。
この機能を安全に使用するためには、リセットをかけるソフトウェア側であらかじめここに値を書き込んでおく必要があります。それにも、ＭＳＸ２＋になって新たに追加されたＢＩＯＳを使用します。

●ＭＳＸ２＋でリセットするときは、次のようにする。

```
ＣＡＬＬ　０１７ＡＨ　　’現在のリセットの状態をＡレジスタに書き込む
ＯＲ　　８０Ｈ　　’リセットフラグをセットする
ＣＡＬＬ　０１７ＤＨ　　’新たな値を書き込む
ＪＰ　　０Ｈ　　’０番地ジャンプ
```

ちなみに、ＭＳＸ２＋より前の機種ではソフトウェアリセットとハードウェアリセツトを区別する方法はありません。

２．２．６　汎用Ｉ／Ｏポート

ＭＳＸでは、汎用Ｉ／Ｏポート１にマウスを差したままＰＡＤ（０）関数を呼び出すか、ポート２にマウスを差してＰＡＤ（４）を呼び出すと、システムが停止してしまいます。これを回避する方法は以下のとおりです。

●マウスをポートから抜く

２．２．７　算術演算ルーチンＭＡＴＨ－ＰＡＣＫ

算術演算ルーチン群MATH-PACKは、ページ０及び１がメインＲＯＭであることを期待してプログラムされているので、ＤＯＳ上から使用しようとすると動作しません。ＤＯＳ上からMATH-PACKを使用する場合は、次のルーチンを必ず組み込んでおいてください。

```
ORG  406FH
LD   IX,406FH
JP   0159H
ORG  4666H
LD   IX,4666H
JP   0159H
ORG  4EB8H
LD   IX,4EB8H
JP   0159H
ORG  5439H
LD   IX,5439H
JP   0159H
```

```
        ORG  5597H
GETYPR:LD   A,(0F663H)
       CP   8
       JR   NC,DOUBLE
       SUB  3
       OR   A
       SCF
       RET
DOUBLE:SUB  3
       OR   A
       RET
       ORG  66A7H
       POP  AF
       RET
```

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

　ＢＡＳＩＣのプログラム中に、０が２バイト続くようなデータを書き込むと、そこからあとの行はＬＩＳＴが出なくなります。

　ＰＣ－９８シリーズで２ＤＤフォーマットしたディスクをＭＳＸで読もうとすると、うまく読めないことがあります。９８とＭＳＸでディスクを共用するときにはＭＳＸでフォーマットしたほうが安全です。９８は通常８セクターフォーマットを、ＭＳＸは９セクターを使うのでうまく読めないと思う人もいるかも知れませんが、実はどちらの機種も両方のディスクを読み書きできるように設計されているので困りません。

　ディスクファイルのディレクトリーエリアのタイムスタンプの日付の部分の所を全部０で埋めると、ディレクトリーを取ったときタイムスタンプが出なくなります。

　ディスクのＦＡＴ中にＦＦ８Ｈ～ＦＦＦＨのいずれかの値があったら、そこがファイルの終了クラスタであるということを意味します。ＦＦ７Ｈならばそのクラスタは不良クラクタで、以後そのクラスタは使用されません。

　ＭＳＸ，ＭＳＸ２では、自分自身をページ１またはページ２のＲＡＭに転送し、あたかもそこがＲＯＭカートリッジであるかのように振る舞うソフトを起動すると、次に電源を入れたときに、システムがＲＡＭに残っている内容を見てそれがＲＯＭカートリッジであると誤認してしまうことがあります。そうなった場合、何度リセットしてもそのソフトが必ず起動してしまい、システムが起動しなくなることがあります。このような場合は、一旦電源を切り、コンセントを外して、さらに内蔵タイマー用の電池を外して２０分程度放っておくと、ＲＡＭの内容が消失して再びシステムが起動できるようになります。なお、ＭＳＸ２＋以降の機種は、システムが起動時にＲＡＭに書かれた内容をＲＡＭと間違えないように、ＲＡＭの内容を消去するので、この異常動作は起こりません。具体的には、ＲＡＭに書かれたＲＯＭ　ＩＤの内容を消去しています。

　インタースロットコールから戻ってきたとき、ＭＳＸ１では割り込みは禁止されていますが、ＭＳＸ２以降の機種では割り込みはコール前の状態になっていることになっています。しかし、まれにＥＩ状態でインタースロットコールしたのにＤＩ状態で戻ってくることがあります。これは、Ｚ８０ＣＰＵの特性によるものです。ＥＩである必要のあるソフトは、インタースロットコール直後にＥＩしておいたほうが安全です。

　ＭＳＸでは、今まで、Ｉ／Ｏポートは８ビットとして使用してきましたが、実は、仕様上、１６ビットＩ／Ｏポートを使用しても構わないということが判明しました。現在は存在しませんが、後々１６ビットＩ／Ｏを使用したハードが登場する可能性があります。

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

ＭＳＸ２，ＶＲＡＭ６４Ｋの機種は、ほとんどの市販ソフトが動かないのでとても淋しいのですが、日立と東芝の機械はメーカーが１２８Ｋ拡張サービスをやってくれていたので、ありがたや、市販ソフトが動くようになります。たまに、ＶＲＡＭ１２８Ｋ用のソフトが「日立ＭＢ－Ｈ３、東芝ＨＸ－２３では動作しません」などと書いているのをみて変だな、と思った鋭い人がいるかも知れませんが、そういうわけなのです。キヤノンや三洋もＶＲＡＭ６４Ｋの機械を出していましたが、こちらがバージョンアップサービスをしていたかどうかはわかりません。でも、半田ごてがあれば割りと簡単に拡張できます。

漢字コード２２２ＦＨ～２２７ＦＨ、２３２０Ｈ～２３２ＦＨまでのコード（特殊記号）は、機種によって違う場合があるので注意してください。ＭＺなどの機種では、全然違う記号がここのコードに割り当てられています。

ＢＡＳＩＣのＰＵＴ　ＫＡＮＪＩやＢＩＯＳのＫＮＪＰＲＴで漢字を表示する前に、ＰＳＥＴ命令で点を打てば、そこが始点になります。（ＯＰＥＮ”ＧＲＰ：”で文字を表示するときと同じです）

メモリマッパの説明がよく分からなかったかも知れないので､改めて詳しく解説します。メモリマッパ内蔵機にマッパＲＡＭ拡張カートリッジ（具体的にはＤＯＳ２カートリッジ）を差すと、Ｉ／ＯのＦＣＨからＦＦＨ番地には２つの機器が差さっていることになります。ここで、たとえばＦＤＨに値を書き込むと、内蔵、外付けの両方のメモリマッパに信号が伝わり、両方のマッパが切り換わります。もちろん、両方とも自分のスロットのＲＡＭだけを切り換えます。こういうことができるという恐ろしい仕様になっているのです。ＤＯＳ２のマッパは標準のマッパとは仕様が違う、というような報道がされたことがありますが、仕様自体は同じです。ただし、システムが自分用に３２Ｋの領域を確保することと、上に述べた２つの機器が同時に接続されている場合を予期してプログラムを組まないと、マッパ対応にしても、ＤＯＳ２では動作しません。ＦＣＨ～ＦＦＨのＩ／Ｏから値を読むことができないのも、一度に２つの機器が接続されていることがあるためです。（ＭＳＸでは、１つのポートに２つの機器が接続されている場合、そのポートから値を読むと異常な動作をする）なお、最高８つまでのマッパＲＡＭを同時に接続できることになっています。ちなみに、通常のコンピューターは２つの機器が同時に同じポートに接続されている、などということは絶対にありえないように設計するのが普通です。１つのポートに８つまでの機器が並列につながる、というのは尋常ではありません。

それで思い出しましたが、ＤＯＳ２の２５６ＫＲＡＭ内蔵版を２つ差すと、５１２Ｋで立ち上がります。マッパ内蔵機ではもっとＲＡＭが増えます。

ＭＳＸ２＋では､ＶＤＰがまだ前の処理を終えていないときにＶＤＰをアクセスすると、ＶＤＰからウェイト信号が来て、その間実行が停止します。turboRでは、８μ秒以内の間隔でＶＤＰをアクセスするとシステムのほうが自動的にＣＰＵにウェイトをかけるようになっています。なお、これらの保護機能はＭＳＸ，ＭＳＸ２には付いていません。あまりＶＤＰを酷使すると、異常動作を起こすことがあるので、少しばかり注意が必要です。

アスキーのＭＳＸ－Ｖｉｅｗ用のソフトは、Ｃ言語を使ってアプリケーションを組もうと思っても、ＭＳ－ＤＯＳ上で動くＣを使わなければ組めませんでしたが、ようやくＭＳＸ－Ｃで組むためのソフトが開発されたようです。ＭＳＸ－Ｄａｔａｐａｃｋ ｔｕｒｂｏＲ版のユーザーには無償でディスクの書き換えサービスを行なっているようです｡ので、興味のある方はアスキー社まで問い合わせるといいかもしれません。

ＭＳＸ－ＤＯＳ２　ＴＯＯＬＳのファイルの内容は、出荷時期により相当変わっています。これもアスキーに問い合わせないと最新版は手に入りません。

付属サンプルプログラムディスクに収録されていて、第７部では触れられていないソフトの紹介をちょっとします。これはＳＲＡＭＶＪＥという名前で、アスキー、松下のＭＳＸ－ＪＥ用の内蔵ＳＲＡＭをディスクファイルに保存するソフトです。松下の機種は、どうもＳＲＡＭが消えやすい機種があるようなので、そういう機種をお持ちの方はご利用ください。名前のとおりＳＲＡＭのアスキー、松下版です。

メインＲＯＭの０００４Ｈ番地から２バイトにフォントの先頭アドレスが入っています。ワークエリアのＣＧＰＮＴ（Ｆ９１ＦＨ，３）には、フォントのあるスロット番号と開始アドレスが入っていますが、ここを書き換えたあともとに戻すときはこの値を参考にすると面倒がなくていいです。ワークエリアを２バイト節約することができます。

ＤＯＳからＢＡＳＩＣにいって、ＣＡＬＬ　ＫＡＮＪＩで漢字ドライバを起動し、ＣＡＬＬ　ＳＹＳＴＥＭでＤＯＳに戻ります。すると、ＴＹＰＥコマンドで漢字ファイルが読めるようになります。ただ､この状態からＭＥＤなどを起動させると暴走してしまいます。たぶんＭＥＤがＶＲＡＭを直接いじっているのでしょう。

アスキー社製ハードディスクインターフェイスを接続した状態でマスタースロットの４０２９Ｈ番地をコールすると、ハードディスクのシッピングを行います。ソフトでシッピングを行いたいときに使用してください。

第３章　メモリマッパ

２．３．１　メモリマッパ概要

ＭＳＸ２では、Ｉ／ＯポートのＦＣＨ番地からＦＦＨ番地までをメモリマッパに割り当てています。これは、ＲＡＭをページごとに（１６Ｋごとに）一瞬にして切り換えてしまうという機能です。ただし、標準仕様ではないので、機種によってはこの機能がない場合もあります。メモリマッパがあるかどうかは、次の方法で調べられます。

ＢＡＳＩＣから　ＯＵＴ　２５５，１　を実行する
暴走すれば、メモリマッパがある。

本体ＲＡＭ１２８Ｋバイト以上の機種には、必ずついています。また、ＭＳＸ２＋ではすべての機種についているようです。
使い方としては、Ｉ／ＯのＦＣＨ番地がページ０の、ＦＤＨがページ１の、ＦＥＨがページ２の、ＦＦＨがページ３のＲＡＭの切り換えに使われています。
システムは、リセット時に、ＦＣＨ番地に３を、ＦＤＨに２を、ＦＥＨに１を、ＦＦＨに０を書き込みます。そして、ページ０に３番の、以下、ページ１に２番、ページ２に１番、ページ３に０番のＲＡＭがそれぞれ割り当てられます。（ＲＡＭはすべて１６Ｋごと）ここで、例えばＦＤＨに０を出力したとします。すると、ページ１のＲＡＭが一瞬にしてページ３と同じ０番のＲＡＭに切り換わってしまいます。ここで、ＲＡＭの５０００Ｈ番地に値を書き込むと、同時にＤ０００Ｈ番地の内容も書き変わってしまいます。先ほどの、メモリマッパがあるかどうかを調べる方法は、ページ３のＲＡＭを切り換えるということなので、その結果ワークエリアが消失して暴走するわけです。メモリマッパがない機種では、元々Ｉ／ＯのＦＦＨ番地には何もつながっていないので、何も変化がなく実行が返ります。
メインＲＡＭ６４Ｋの機種では、１６ＫのＲＡＭを４個しか持っていないので、このくらいしか利用価値がありませんが、ＲＡＭ１２８Ｋの機種では、ＲＡＭの番号に０から７までが指定できます。２５６Ｋの機種では０から１５までが指定できます。そこで、普段は隠れていて見えないＲＡＭをデータエリアに使ったりすることができます。
以下にスロット３－１にＲＡＭがある場合について例を図示します。

ＳＬＯＴ　３－１	３－１	３－１
ﾍﾟｰｼﾞ0　ＲＡＭ３番	ＲＡＭ３番	ＲＡＭ５番
ﾍﾟｰｼﾞ1　ＲＡＭ２番	ＲＡＭ０番	ＲＡＭ０番
ﾍﾟｰｼﾞ2　ＲＡＭ１番	ＲＡＭ１番	ＲＡＭ１番
ﾍﾟｰｼﾞ3　ＲＡＭ０番	ＲＡＭ０番	ＲＡＭ０番
電源投入時	OUT 0FDH,0を 実行したとき	RAM128K以上の機種でさらに OUT 0FCH,5を実行したとき

本体ＲＡＭがいくらあるのかを調べるにはひたすらＲＡＭを切り換えながらデータを書き込んで、ちゃんと読み込めるかどうかをチェックしていくという大変に原始的な方法を使ってください。（ＭＳＸ２＋の起動時にも、この方法が使われています）ＦＣＨ～ＦＦＨから値を読み込むという方法もありますが、この方法を使うと、ＤＯＳ２カートリッジなどのマッパＲＡＭ拡張カートリッジを接続した場合にバス競合が起こって本体を破壊する恐れがあるので、これらのＩ／Ｏポートから値を読み込むのは絶対に避けてください。（バス競合は読み込むときだけ起こるので、書き込みはしても大丈夫です）

以上がメモリマッパの使い方ですが、これではもともとメモリマッパがない機種ではマッパが使えません。また、ＲＡＭのＩ／ＯバスがＣＰＵに接続されている必要があります。つまり、６４Ｋ増設ＲＡＭカートリッジなどを接続しても、マッパＲＡＭは増えないのです。そこで、ＤＯＳ２カートリッジなどでは内蔵ＲＡＭに直接ＦＣＨ～ＦＦＨ番地のＩ／Ｏがつながっています。これは普通のマッパと同じ使い方もできますが、最後の２つのＲＡＭ（ＲＡＭ１２８Ｋの機種なら６番と７番、２５６Ｋの機種なら１４番と１５番）は、ＤＯＳ２システムが使っているので、ここをユーザープログラムが使おうとすると暴走します。また、そうやって拡張していくと、ユーザープログラムの負担がかなり重くなるので、ＤＯＳ２には拡張ＢＩＯＳでＲＡＭを切り替える方法が用意され、こちらを使用してマッパＲＡＭを操作することが推奨されています。この拡張ＢＩＯＳコールを用いた

マッパＲＡＭの使い方については、次の第２節を参照してください。

２．３．２　メモリマッパ拡張ＢＩＯＳ

ＭＳＸで６４Ｋバイトを越えるＲＡＭを使用するときは、Ｉ／ＯのＦＣＨ～ＦＦＨ番地のメモリマッパ用のＩ／Ｏを使って切り換えます。しかし、この方式を使うと、若干の問題が出てきます。具体的には、次のようなものです。

１．ユーザープログラムがＲＡＭの容量を調べるのが難しい

２．メモリマッパを使用したプログラムを二つ以上同時に使用しようとすると、使用するセグメントが衝突して動作しないことがある

ユーザープログラムがＲＡＭ容量を調べようとした場合、ひたすらセグメントを切り換えながらその番号のセグメントがあるかどうか調べるという原始的な方法を取らざるをえません。これはかなり面倒な方法であり、暴走の危険も生じます。

そこで、マッパＲＡＭを管理するための拡張ＢＩＯＳが用意されました。これを使ってＲＡＭを管理することにより､マッパを使うプログラムを同時に実行することができます。また、セグメント数の計算や、プログラムごとのセグメントの割り付けもマッパＢＩＯＳがやってくれますので、ユーザー側の負担が大幅に軽減されます。

現在、ＤＯＳ２がこのメモリマッパ拡張ＢＩＯＳを内蔵しています。ＤＯＳ２では動作するのに最低３２Ｋバイトのワークメモリを必要とするためです。ＤＯＳ２上で動かすことが明らかなソフトは、メモリマッパは直接Ｉ／Ｏ切り換えでなく、拡張ＢＩＯＳを使用してください。ＤＯＳ２の利用するセグメントをうっかり使用したりすると暴走します。

マッパテーブルの呼び出し方法

メモリマッパ拡張ＢＩＯＳは、常にページ３に置かれ、いくつのマッパＲＡＭが接続されていて、いくつのセグメントが使用できるのか、という内容が保存されているマッパテーブルも、やはりページ３に常に存在しています。マッパテーブルがどこにあるのかを調べるには、第２部第７章で紹介したＦＦＣＡＨ番地コールを使用します。ＦＢ２０Ｈ番地のビット０を見て拡張ＢＩＯＳコールが可能かどうかチェックしてから、次の方法で調べてください。

●マッパテーブルの先頭アドレスを得る方法

Ａレジスタに０、Ｄレジススタに４、Ｅレジスタに１を入れてＦＦＣＡＨ番地をコールする

戻ってきたＡにプライマリマッパのスロット番号（システムが使用しているメインＲＡＭのスロット番号)、ＨＬにマッパテーブルの先頭アドレスが返る。ＤＥは保存される。裏レジスタ及びＩＸ，ＩＹレジスタは破壊される

もし、戻ってきたＡの値が０だったら、メモリマッパ拡張ＢＩＯＳは存在しない

マッパテーブルの内容は以下のとおりです。

アドレス	内容
ＨＬ+0	マッパスロットのスロット番号
+1	16KバイトRAMセグメントの総数（1～255）
+2	未使用の16KRAMセグメントの数
+3	システムに割り当てられた16KRAMセグメントの数
+4	ユーザーに割り当てられた16KRAMセグメントの数
+5～+7	16KRAMセグメントの数現在未使用。予約
+8～	他のマッパのスロット番号。ない場合+8は０

（複数のマッパＲＡＭが存在する場合、先頭はプライマリマッパになる。セグメントの数は一つのマッパＲＡＭに1～255まで）

マッパサポートルーチンを使用するには、ページ３の特定のアドレスをコールします。その番地は不定なので、以下の方法で調べる必要があります。

●マッパサポートルーチンの先頭アドレスを得る方法

Ａに０、Ｄに４、Ｅに２を入れてＦＦＣＡＨ番地をコールする

戻ってきたＡにプライマリマッパの総セグメント数、Ｂにプライマリマッパのスロット

番号、Ｃにプライマリマッパの未使用セグメント数、ＨＬにジャンプテーブルの先頭アドレスが返る。ＤＥは保存される。裏レジスタ及びインデックスレジスタは破壊される
　マッパサポートルーチンはジャンプテーブルになっており、その内容は以下のとおりです。

アドレス	エントリ名	機能
ＨＬ+0H	ALL_SEG	16Kのセグメントを割り付ける
+3H	FRE_SEG	16Kのセグメントを開放する
+6H	RD_SEG	セグメント番号Ａの番地ＨＬの内容を読む
+9H	WR_SEG	セグメント番号Ａの番地ＨＬにＥの値を書く
+CH	CAL_SEG	インターセグメントコール（インデックスレジスタ）
+FH	CALLS	インターセグメントコール（インラインパラメーター）
+12H	PUT_PH	Ｈレジスタの上位２ビットのページを切り換える
+15H	GET_PH	Ｈレジスタの上位２ビットのページのセグメント番号を得る
+18H	PUT_P0	ページ０のセグメントを切り換える
+1BH	GET_P0	ページ０の現在のセグメント番号を得る
+1EH	PUT_P1	ページ１のセグメントを切り換える
+21H	GET_P1	ページ１の現在のセグメント番号を得る
+24H	PUT_P2	ページ２のセグメントを切り換える
+27H	GET_P2	ページ２の現在のセグメント番号を得る
+2AH	PUT_P3	何もせずに戻る
+2DH	GET_P3	ページ３の現在のセグメント番号を得る

マッパサポートルーチンの使用方法

　プログラムはマッパＲＡＭを使用する場合、まず、ＡＬＬ＿ＳＥＧを使って自分用のセグメントを確保しなければなりません。そうしないと、他のプログラムと使用するセグメントが衝突して、暴走する可能性があります。割り付けには、ユーザーセグメントとして割り付ける方法と、システムセグメントとして割り当てる方法があります。ユーザーセグメントは、プログラムが終了すると自動的に開放されますが、システムセグメントはＦＲＥ＿ＳＥＧで開放しないかぎり開放されません。通常のプログラムはユーザーセグメントとして割り付けるのが良いでしょう。システムセグメントは、プログラムが終了したとき次のプログラムにデータを引き渡す場合や、後で再び起動したときのために前のデータを保存しておきたい場合などに使用します。ＤＯＳ２が使うワークＲＡＭなどは、システムセグメントとして割り付けられています。プログラムは、解放したセグメントを使用し続けてはなりません。
　なお、これら二つのルーチンを使用する場合、スタックはページ１またはページ３に置かなければなりません。

●ＡＬＬ＿ＳＥＧ　（セグメントを割り付ける）
　　入力　Ａ：０ならユーザーセグメントとして割り付け
　　　　１ならシステムセグメントとして割り付け
　　　　Ｂ：０ならプライマリマッパに割り付ける
　　　　０以外は複数マッパの場合の割り付け
　　　　FxxxSSPPの形でスロット番号を指定する。bit6～4は以下の割り付け方法の指定に使用する
　　　　xxxが000のとき：指定スロットのみ割り付け
　　　　xxxが001のとき：指定のスロット以外で割り付け
　　　　xxxが010のとき：指定のスロットで割り付けを試み、
　　　　失敗の場合、あれば他のスロットで割り付ける
　　　　xxxが011のとき：指定のスロット以外で割り付けを試み、
　　　　失敗の場合は指定のスロットで割り付ける
　　出力　Ｃｙ，セットなら未使用セグメントがない
　　　　リセットなら割り付けに成功
　　　Ａに割り付けられたセグメント番号
　　　Ｂにマッパスロットのスロット番号（入力でＢ＝０だった場合は０）

●ＦＲＥ＿ＳＥＧ　（セグメントを開放する）
　　入力　Ａに開放するセグメント番号
　　　　Ｂ：０ならばプライマリマッパ
　　　　　０以外ならマッパのスロット番号
　　出力　Ｃｙ，セットなら失敗
　　　　　リセットなら成功

　マッパＲＡＭの値を読み書きするには、次のＲＤ＿ＳＥＧ，ＷＲ＿ＳＥＧを使用します。動作速度を維持するために、セグメント番号が有効かどうかのチェックは行われません。指定セグメントが存在するかどうかのチェックはユーザープログラムの責任になります。
　マッパサポートルーチンはページ２を使用して読み書きするため、スタックをページ２に置くことはできません。また、ページ２は読み書きするマッパスロットに切り換えておいてからコールしなければなりません。動作速度を維持するために、マッパサポートルーチンがスロット切り換えを行わないためです。ＡＦ以外のレジスタは保存されます。割り込みは禁止されて戻ります。ＨＬレジスタで指定されるアドレスの上位２ビットは無効です。セグメントが１６Ｋバイト単位で扱われるからです。

●ＲＤ＿ＳＥＧ　　（指定セグメントから値を読む）
　　入力　Ａに読み出すセグメント番号
　　　　ＨＬにセグメント内のアドレス（上位２ビットは無効）
　　出力　Ａに指定アドレスの値
　レジスタ　Ｆ？

●ＷＲ＿ＳＥＧ　　（指定セグメントに値を書く）
　　入力　Ａに書き込むセグメント番号
　　　　ＨＬにセグメント内のアドレス（上位２ビットは無効）
　　出力　なし
　レジスタ　ＡＦ

　セグメント内に機械語プログラムを置いて、ちょうどＭＳＸシステムのインタースロットコールのように、インターセグメントコールすることができます。使い方もインタースロットコールと似ています。インターセグメントコールには、以下に挙げるＣＡＬ＿ＳＥＧ，ＣＡＬＬＳの二つのルーチンが用意されています。
　これらのルーチンを使用する場合、指定セグメントが存在するかどうかのチェックは行われません。これは、動作速度を維持するためです。また、コールする前に指定ページを使用するマッパＲＡＭのスロットに切り換えておくのもユーザープログラムの責任になります。
　ページ３へのインターセグメントコールは実行することができません。もし指定した場合、単に指定アドレスがコールされます。ページ０をコールする場合割り込みルーチンなどのエントリーがあるので、コールは慎重に行う必要があります。
　この二つのルーチンは、割り込みの状態を変化させません。従って、コールされたルーチンが割り込みの状態を変化させない限り、呼び出し時と同じ状態で戻ります。
　この二つのルーチンは、裏レジスタ及びインデックスレジスタを内部で使用します。従って、これらのレジスタを使ってコール先にパラメーターを受け渡すことはできません。

●ＣＡＬ＿ＳＥＧ　（インターセグメントコール）
　　入力　ＩＹの上位８ビットでセグメント番号を指定
　　　　ＩＸでコールするアドレスを指定（上位２ビットでページを指定）
　　　　ＡＦ，ＢＣ，ＤＥ，ＨＬがコールされたルーチンに渡される
　　　　裏レジスタ及びインデックスレジスタは破壊される
　　出力　ＡＦ，ＢＣ，ＤＥ，ＨＬ，ＩＸ，ＩＹがコールされたルーチンから返される
　　　　裏レジスタは破壊される

●ＣＡＬＬＳ　　（インターセグメントコール・インラインパラメーター）
　　入力　ＡＦ，ＢＣ，ＤＥ，ＨＬがコールされたルーチンに渡される
　　　　裏レジスタ及びインデックスレジスタは破壊される
　コール手順：　ＣＡＬＬ　ＣＡＬＬＳ
　　　　ＤＢ　　セグメント番号
　　　　ＤＷ　　コールするアドレス
　　出力　ＡＦ，ＢＣ，ＤＥ，ＨＬ，ＩＸ，ＩＹがコールされたルーチンから返される

裏レジスタは破壊される

上に挙げたルーチンの他、直接セグメントの状態を切り換えてしまうルーチンが用意されています。連続してセグメントを読み書きしたり、他のセグメントにあるデータを転送せずにそのまま使用する場合などに利用します。これらのルーチンは高速なのでプログラムの効率に影響することはありません。(と、アスキーは言っている)

これらの直接セグメント切り換えルーチン(ダイレクトページングルーチン)は、指定されたセグメントが存在するかどうかのチェックは行いません。指定セグメントが存在するかどうかのチェックはユーザープログラムの責任になります。

ＰＵＴルーチンは、セグメントを切り換え、同時にメモリ上に書き込んだ値を保存します。ＧＥＴルーチンは、現在のマッパセグメントの状態を返すルーチンで、ＰＵＴルーチンでメモリ上に記録された値が返され、Ｉ／Ｏから値を読むことはしません。これは、マッパＲＡＭが複数存在する場合、Ｉ／Ｏから値を読むことによってバス競合が起こる可能性があるためです。従って、ＤＯＳ２上で動くプログラムや、メモリマッパ拡張ＢＩＯＳを使用しているプログラムは、Ｉ／ＯのＦＣＨ～ＦＦＨを直接使用してセグメントを切り換えてはなりません。

ＰＵＴ＿Ｐ３ルーチンは、存在はしますが、実際にはマッパを切り換えずに、何もせずに戻ります。これは、ページ３を切り換えると暴走するためです。

プログラムはこれらのルーチンでセグメントを切り換える前に、現在のセグメントの状態を保存しておき、プログラム終了時には必ず元の状態に戻す必要があります。プログラムが起動したときに、セグメントの状態が常に同じとは限らないからです。

ＰＵＴ＿ＰＨ，ＧＥＴ＿ＰＨは、ＨＬレジスタに何らかのアドレスを入れ、その上位２ビットでページを指定するとき有用なルーチンです。

●ＰＵＴ＿Ｐｎ　　(ページｎのセグメントを切り換える)
　入力　ｎはページ番号(０～３)
　　Ａにセグメント番号
　出力　なし
レジスタ　なし

●ＧＥＴ＿Ｐｎ　　(ページｎのセグメントの状態を得る)
　入力　ｎはページ番号(０～３)
　出力　Ａにセグメント番号
レジスタ　なし

●ＰＵＴ＿ＰＨ　　(セグメント切り換え・Ｈの上位２ビットでページを指定)
　入力　Ｈの上位２ビットでページを指定
　　Ａにセグメント番号
　出力　なし
レジスタ　なし

●ＧＥＴ＿ＰＨ　　(セグメントの状態を得る・Ｈの上位２ビットでページを指定)
　入力　Ｈの上位２ビットでページを指定(ページ３を指定しても無効になる)
　出力　Ａにセグメント番号
レジスタ　なし

ＤＯＳ２カートリッジに搭載されているものと、turboRに搭載されているものでは、マッパＢＩＯＳの動作が若干異なります。例えば、ＤＯＳ２カートリッジ版では最も容量が大きくて最もスロット番号の若いマッパがプライマリマッパになりますが、turboRでは必ず内蔵ＲＡＭがプライマリマッパになります。

第４章　漢字ＲＯＭ

ここでは、主に、漢字ＲＯＭのアクセスの仕方について説明します。

２．４．１　漢字ＲＯＭ入出力

ＭＳＸで漢字ＲＯＭを操作するときは、直接Ｉ／ＯのＤ８ＨからＤＢＨまでをアクセスします。
漢字のフォント・データを読みだすには、漢字コードと呼ばれる１２ビットのデータを使います。これは、ＪＩＳコードや区点コードとは別物ですので、まず目的の漢字コードを求める必要があります。

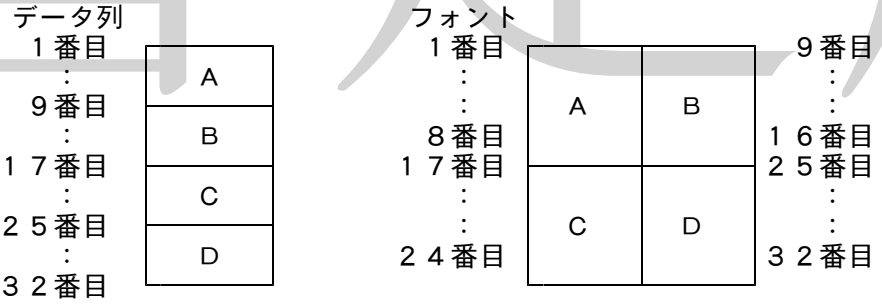
●漢字コードの求め方
漢字のＪＩＳコードの上位バイトから３２を引いた値を区、下位バイトから３２を引いた値を点と呼びます。（これが、いわゆる区点コードです）
第一水準の場合、区が１５以下なら、（区＊９６＋点）が漢字コードになります。区が１６以上なら、（区＊９６＋点－５１２）が漢字コードになります。
第二水準の場合、（（区－４８）＊９６＋点）が漢字コードになります。

次に、求めた漢字コードを元にＩ／Ｏポートをアクセスします。 第一水準と第二水準で読み出し方が違いますので、順に説明しましょう。

●第一水準漢字を読みだす場合
Ｉ／ＯポートのＤ８Ｈ番地に漢字コードの下位６ビットを、Ｄ９Ｈ番地に上位６ビットを出力します。そして、Ｄ９Ｈ番地から３２回連続して値を読み込むと、それがその漢字のフォントになります。

●第二水準の場合は、Ｉ／ＯポートのＤＡＨ番地に漢字コードの下位６ビットを、ＤＢＨ番地に上位６ビットを出力します。そして、ＤＢＨ番地から連続して３２回読み込みます。これが漢字フォントになります。

フォントの並びは、まず左上に８バイト並べて、次に右上に８バイト並べます。次に左下に８バイト、右下に８バイトという順番になります。



ＭＳＸでは、通常、Ｉ／Ｏポートへの直接入出力は禁じられています。しかし、漢字ＲＯＭではほかに方法がありません。ＭＳＸ２になってから追加されたＢＡＳＩＣのＰＵＴ　ＫＡＮＪＩ命令やＢＩＯＳのＫＮＪＰＲＴは、第一水準しかサポートしていない上に、（ＭＳＸ２＋では第二水準もサポートしています）画面に表示するだけなので、漢字フォントを直接いじりたいとき、例えば、１６＊１６ドットのフォントを１２＊１６ドットに圧縮したいときや、漢字フォントをプリンターに直接ビットイメージ印字したいときに使えません。また、スクリーン２や４のモードをサポートしていません。それにＭＳＸ１では、漢字ＲＯＭをアクセスするためのＢＩＯＳが用意されていません。そんなわけで、漢字ＲＯＭのアクセスについては、Ｉ／Ｏポートを直接操作しているわけです。
なお、御存知の方も多いと思いますが、漢字ＲＯＭの空白部分をアクセスすると、空白ではなく、関係ない文字のフォントが返ってきます。全角のスペースはＪＩＳコードの２１２１Ｈを使うようにしてください。半角文字については、ＪＩＳコードの２０２１Ｈ～２０７ＥＨ、２９２１Ｈ～２９５ＦＨのフォントの左半分だけを使います。ただし、これらの半角フォントは漢字ドライバやＰＵＴ　ＫＡＮＪＩ命令ではサポートしていません。

２．４．２　　漢字ＲＯＭの存在の有無

漢字ＲＯＭがないのに漢字入出力を行なおうとすると、画面に豆腐（１６＊１６ドットの四角のこと）が現われてとても悲しい目に合います。そこで、漢字ＲＯＭの存在の有無を確認することが必要になってきます。

●第一水準漢字ＲＯＭが存在するかどうかを確かめる方法
漢字ＲＯＭのＪＩＳコード２１４０Ｈ（１区３２点）のフォントの最初の８バイトは、必ず、順に００Ｈ，４０Ｈ，２０Ｈ，１０Ｈ，０８Ｈ，０４Ｈ，０２Ｈ，０１Ｈになっています。そこで、このコードのフォントを読みだして、最初の８バイトがこれに一致すれば、第一水準の漢字ＲＯＭが存在することになります。

●第二水準漢字ＲＯＭの存在の有無
漢字ＲＯＭのＪＩＳコード７３７ＥＨ（８３区９４点）のフォントを読みだして、最初の８バイトの値の合計を２５６で割った余りが１４９（９５Ｈ）ならば、第二水準漢字ＲＯＭが存在します。

ソフトウェアでは、必ず漢字ＲＯＭの存在の有無を調べ、漢字ＲＯＭがない機種でも使えるように、半角文字のメッセージを用意しておきましょう。第二水準漢字ＲＯＭがある機種では「國」とか「炬燵」とか「靨」とか難しい漢字を山のように使い、ない機種では「国」とか「こたつ」とか表示するソフトも考えられますが、今のところ見たことはありません。

２．４．３　　シフトＪＩＳコード

ＭＳＸでは、漢字ファイルはシフトＪＩＳコード（マイクロソフト漢字コード）を使います。これは、ＭＳ－ＤＯＳでは漢字ファイルはマイクロソフト漢字コードで扱うことになっているので、それと同じフォーマットを使っているＭＳＸでも、同じコードを使っているわけです。
シフトＪＩＳコードでは、漢字開始、終了のコードがなく、半角文字と全角の漢字を一緒に使えるようになっています。
ここでは、ＪＩＳコードからシフトＪＩＳコードに変換するアルゴリズムを紹介します。

●ＪＩＳコードからシフトＪＩＳコードへの変換
ＪＩＳコードの上位バイトをＪＩＨ、下位バイトをＪＩＬとします。シフトＪＩＳコードの上位バイトをＳＪＨ、下位バイトをＳＪＬとします。

１．ＪＩＨが５ＥＨ以下だったならば、 ＩＮＴ（（ＪＩＨ－１）／２＋７１Ｈ） がＳＪＨになります。そうでなかったら、 ＩＮＴ（（ＪＩＨ－１）／２＋Ｂ１Ｈ） がＳＪＨになります。

２．ＪＩＨが２で割りきれるなら、 （ＪＩＬ＋７ＥＨ） がＳＪＬになります。そうでないときには、まず、 （ＪＩＬ＋１ＦＨ） をＳＪＬにします。そして、このＳＪＬが７ＦＨ以上だったら、さらにＳＪＬに１を足します。７ＦＨより小さかったらそのままです。

●シフトＪＩＳからＪＩＳへの変換

１．ＳＪＨがＡ０Ｈより小さかったら、ＳＪＨから７１Ｈを引いたものをＨとします。Ａ０Ｈ以上だったら、ＳＪＨからＢ１Ｈを引いたものをＨとします。そして、どちらの場合でも、 Ｈ＊２＋１ をあらためてＨに代入します。

２．ＳＪＬが７ＦＨより大きかったら、 ＳＪＬ－１ を、そうでなかったらＳＪＬをＬとします。次に、Ｌが９ＥＨより小さかったら、 Ｌ－１ＦＨ をＬとします。Ｌが９ＥＨ以上だったならば、Ｌ－７ＤＨ をＬにして、 Ｈ＋１ をＨにします。

３．Ｈの値がＪＩＳコードの上位バイトに、Ｌの値が下位バイトになります。

第５章　プリンター

ここでは、プリンターの機能と、なるべく多くのプリンターでプログラムが動くようにするための注意点について述べます。なお、ここでは、一般的なドットプリンターについてだけ述べ、特殊なプリンター、例えば、活字プリンター、プロッタプリンター、点字プリンターなどについては触れません。

２．５．１　制御コード

ＭＳＸマークのついたプリンターに必ずついているはずの制御コード、ならびに大抵のＭＳＸ用漢字プリンターについていると思われる制御コードを一覧表にして巻末に載せておきました。参考にしてください。
ＭＳＸ用のプリンターは、日本電気ＰＣ－８０２３と同じ制御コードを使うことになっています。（第８部第９章の表で、＊マークのついているコード）漢字を使わないソフトでは、この＊印のあるコードだけを使うようにすると安全でしょう。ただし、プリンターによっては、いくつかのコードが存在しない場合があります。
第８部第９章の二つの表のコードをすべて持っている漢字プリンターは、「ＭＳＸ標準漢字プリンター」ということになっています。漢字を使うときはだいたいこのコードを使うようにしましょう。ただし、右側の表のコードは古いプリンター内蔵機などでは持っていないこともあります。

２．５．２　ひらがな、グラフィックキャラクター変換機能

ＭＳＸ用でないプリンターをつないで印字すると、ひらがなやグラフィックキャラクターが違う文字に化けてしまいます。そこで、ＲＡＭのＦ４１７Ｈ番地に０以外を書き込むか、ＢＡＳＩＣのＳＣＲＥＥＮ命令の第５パラメータを１にするかすると、ひらがなをかたかなに、グラフィックキャラクターをスペースに変換してくれます。しかし、ＢＩＯＳのＬＰＴＯＵＴや、ＤＯＳのシステムコールでは、この変換機能は働きません。ＢＩＯＳのＯＵＴＤＬＰを使えば変換が利くのですが、ＣＴＲＬ＋ＳＴＯＰが押されると、ＢＡＳＩＣの「Device I/O error」が出てしまうので、あまり使わないほうがいいでしょう。このようなときは、プログラム内部で変換するのが一番よいようです。

２．５．３　ＴＡＢ変換機能

ＭＳＸでは、プリンターに０９Ｈの信号が送られるときには、必ず２０Ｈを５個に変換する機能があります。しかし、この機能があるとうまく印字できない場合があるので、そういうときはＲＡＭのＦ４１８Ｈ番地に０以外の値を書き込む必要があります。具体的には次のような場合です。

１．ビットイメージ印字するとき。

２．ＭＳＸ２＋などで、漢字モードになっているとき、コントロールコードを出力するとき。

なお、漢字モードでコントロールコードを送ったら、次に漢字を印字する前に、Ｆ４１８Ｈ番地に０を書き込むようにしてください。ビットイメージ印字をした後も、同じように０を書き込むのがよいでしょう。これらは、ビットイメージデータ内に０９Ｈが入っていたときにうまく印字できないのを回避するためと、漢字モードで、コントロールコードを送るときに、ＢＡＳＩＣがコントロールコードを漢字データと間違えないための処理です。なお、Ｆ４１８Ｈ番地に０以外の値を書き込んでおくと、漢字ＢＡＳＩＣ上からも、ひらがななどが文字化けせずに印字できます。

２．５．４　ビットイメージ印字

ＭＳＸ用のプリンターには、必ず８ドットビットイメージ印字機能がついています。そこで、特殊な文字やグラフィックを印字しようというときは、この機能のお世話になるこ

とになります。ところが、８ドットビットイメージ印字しようとすると、縦方向の高さが３分の２になってしまうプリンターがあります。これは、ＮＥＣ系の２４ドット漢字プリンターや、ＭＳＸ用の古い２４ドット漢字プリンターなどで起こります。そこで、ソフトウェアでは、ビットイメージ印字する際に、２４ドットと８ドットの二つのフォントを用意しておき、ユーザーにどちらかを選ばせるようにすると安全です。

２．５．５　漢字プリンターで半角文字を出力する方法

漢字プリンターで半角文字を出力するには、まず、プリンターを漢字モードにして、（ＢＡＳＩＣ上からならば　LPRINT CHR$(&H1B)+CHR$(&H4B)　）それから、０と文字のアスキーコードを続けて出力します。(「ｱ」 と出力したいならば LPRINT CHR$(0)+"ｱ"　)

２．５．６　プリンター対応ソフト開発上の注意

このほかにも、例えば、１９８３年のいわゆる83ＪＩＳコードで罫線などがサポートされましたが、サポートされていないことが多いので、注意が必要です。ＭＳＸでは、基本的に78ＪＩＳを使っているようです。

また、書体で、エリートなどの幅が狭い文字を印字した後にビットイメージ印字をすると、ドットが詰まって印字される漢字プリンターがあるようなので、漢字プリンターでビットイメージ印字する直前には、字体を制御コードで普通のパイカ文字に指定しておくとよいようです。

もちろん、ソフトを作るときは回りにあるプリンターで全部の文字を印字して、それをいろいろなプリンターで試してみてください。ある特定の文字だけ異常な印字をするソフトが昔、あったそうです。また、漢字プリンターで漢字モードの状態から半角文字を印字するときはいったん漢字モードから半角モードに変えて印字したほうがいいでしょう。そのほかよく問題になるのは罫線です。つながらなかったり、文字の上に重ねて印字されてしまったりといったことがよくありました。それから、字体を変えたり、飾りをつけたりしたソフトを終了するときは、必ず字体などの設定を元に戻してからにしてください。

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

プリンターの1/144インチ改行では、半ドット単位で改行できるようです。昔は２４ドット漢字プリンターなどという豪華なものはとても高かったので、この機能を駆使して漢字ＲＯＭのデータを半ドットずらしで８ドットプリンターに直接ビットイメージ印字し、漢字を出力していました。大昔の話です。

ＭＳＸのプリンターの規格は本体に比べるとかなりゆるやかで、だいたいＰＣ－８０２３と同じコードを持っていればいいということになっています。だいたいということなので、プリンターによっては一部のコードを持っていないことがあります。例えば、ＰＣ－８０２３には平仮名印字、片仮名印字の制御コードがありますが、ＭＳＸのプリンターはそんな制御コードを使わなくても普通に平仮名が打ち出せるので、このコードは普通のＭＳＸ用プリンターでは削除されています。

「ＭＳＸ２テクニカルハンドブック」２９４ページにもプリンターコード表が載っていますが、これは、「ＭＳＸ用プリンターが最低限持っていなければならないコード」です。早い話がＭＳＸのプリンターの規格は２つあるのです。

プリンターによっては多少機能に差があると書きましたが、どういうふうに違うのか、というと、例えば０ＥＨの拡大印字をしているときに、改行すると文字が元の大きさに戻ってしまう機種とそうならない機種があります。ほかにも細かい違いが色々あります。

ＭＳＸ用の漢字プリンターに関しては実は規格がありません。そこで、巻末の表を参考にしてプログラムを組んでください。

実をいうと、説明書などに「ＰＣ－ＰＲ２０１／１０１準拠」と書いてある漢字プリンターは大抵ＭＳＸにも使えます。ＬＩＳＴを取るときなどに平仮名が化けてしまいますが、ワープロ用としてなら全く問題なく利用できるものがあります。

本書では触れませんでしたが、ＭＳＸ用にはこのほかにもプロッタプリンターや活字プリンターが接続可能です。もっとも今どき活字プリンターなんか誰も使わないでしょう。

第６章　マウス、トラックボール

ＭＳＸ２では、ＢＩＯＳにマウス、トラックボールの値を読みだす機能が追加されたので、これを使えばなんの心配もせずにマウスやトラックボールを使うことができます。しかし、ＭＳＸ１では、マウスやトラックボールの読み込みを行なうためのＢＩＯＳが用意されていないので、直接Ｉ／Ｏポートをアクセスすることになります。（正確には、マウスを読むＢＩＯＳならありますが、トラックボールが読めません）ＭＳＸ１用のソフトがマウスやトラックボールを使うときには、次のようにしてください。なお、ＭＳＸ２用のソフトでは、必ずＢＩＯＳを使ってください。ＨＣ－９５のターボモードで動作しなくなってしまいます。当然、turboRでも動作しません。

まず、ジョイスティックポートにアクセスするのですが、ＢＩＯＳのＲＤＰＳＧを使うと割り込みが許可されてしまうので、（アクセスするときには、割り込みは禁止されている必要がある）ＢＩＯＳを通さずに直接Ｉ／Ｏポートをアクセスしてください。もちろん、割り込みは禁止しておいてください。

ピン８に、ＬＯＷの信号を送りながら、ピンの１～４番の値を読み込み（これを値１とします）終わったらすぐにピン８の信号をＨＩＧＨにします。

３３μ秒ＨＩＧＨの状態を継続して、また１～４番の信号を読み取り、（値２）すぐに８番ピンの信号をＬＯＷにします。

２５μ秒ＬＯＷの状態を継続し、また値を読み込んで（値３）信号をＨＩＧＨにします。

２５μ秒状態を継続し、値を読み込んで（値４）信号をＬＯＷにします。

９５μ秒状態を継続し、値を読み込み、（値５）信号をＨＩＧＨにします。

３３μ秒状態を継続し、値を読み込み（値６）信号をＬＯＷにします。

１７μ秒状態を継続し、信号をＨＩＧＨにします。

１７μ秒状態を継続し、信号をＬＯＷにします。

これで読み込みは終わりです。

なお、読み込んだ信号の内容は、ピン１がビット０、ピン４がビット３です。要するに、４ビットの信号を６回読み込むわけです。

信号の意味は次のとおりです。信号の意味はマウスとトラックボールで異なります。昔はマウスかトラックボールの一方しかサポートしていなかったソフトが多かったのですが、必ず両方に対応するようにプログラムを組んでください。

値	マウス	トラックボール
１	無意味	Ｘ座標
２	Ｘ座標上位	Ｙ座標
３	Ｘ座標下位	無意味
４	Ｙ座標上位	無意味
５	Ｙ座標下位	無意味
６	右記の値以外	1000B,1001B,0111Bのどれか

マウスは２の補数表現の８ビットでＸ，Ｙの値が出てきます。これに対し、トラックボールでは、0000B が－８、0111B が－１、1000B が０、1111B が７ということになっています。

マウス、トラックボールの値は、読み込むたびに前回との差を返します。そこで、割り込みを使って一定時間ごとに値を読み込むと、速さに比例した値が得られます。なお、入力の間隔が長すぎると、無意味な値を返すので、そういうときは一回読み込んだ値を捨てて、もう一度読み込むようにします。例えば、ディスクアクセスをした直後など、割り込みが禁止されたときや、初めて値を読むときなどがこれに当たります。

マウスは、製品によってタイミングが少しずつ違うようです。マウス対応ソフトを作るときは、何種類かのマウスで動作確認をしておいたほうがよいようです。

どうしてＭＳＸ１ではマウスなどを使うときに直接Ｉ／Ｏポートをアクセスしなければならないかというと、ＭＳＸ１が発売されたときにはまだマウスやトラックボールの仕様が決まっていなくて、後から仕様が決められたためです。ＭＳＸ用マウスの仕様については、月刊アスキー１９８５年３月号を参考にしてください。もっとも、自分でＭＳＸ用のマウスを作ろうという人でなければ、読んでもあまり意味はないでしょう。

第７章　拡張ＢＩＯＳコール

ＭＳＸでは、ＲＳ－２３２Ｃなどの、新しく拡張された機能を利用するときは、拡張ＢＩＯＳコールで処理します。拡張ＢＩＯＳコールで機能を使うものには、例えば、漢字ドライバ、ＭＳＸ－ＡＵＤＩＯなどがあります。ここでは、拡張ＢＩＯＳの仕組みと、簡単な使い方について述べます。

２．７．１　拡張ＢＩＯＳの利用方法

ＭＳＸでは、拡張されたオプション機能を利用するときには、拡張ＢＩＯＳコールを使います。拡張ＢＩＯＳを呼びだすには、Ｄレジスタにデバイス番号、そのほかのレジスタに必要な値をいれて、ＲＡＭのＦＦＣＡＨ番地をコールします。ＦＦＣＡＨ番地からの５バイトでは、ＲＳＴ３０命令で拡張ＢＩＯＳ処理のあるスロットにジャンプしています。デバイス番号は、拡張された機能ごとに決まっています。例えば、ＲＳ－２３２ＣやモデムはＯ８Ｈ、漢字ドライバは１１Ｈです。
拡張ＢＩＯＳコールができるかどうかを調べるには、次の方法を使います。

●ＦＢ２０Ｈ番地のビット０が立っていれば、拡張ＢＩＯＳコールが可能。立っていなければ不可能
なお、このチェックは、ディスク上で動作することが明らかなソフト（ＤＯＳ用のソフトなど）は省略してかまいません

例えば、漢字ドライバがあるときに、（ＭＳＸ２＋では標準でついています。ＭＳＸ２ではＤＯＳ２かソニーのＨＢＩ－Ｊ１を差すと使えるようになります）Ｄレジスタに１１Ｈ（漢字ドライバのデバイス番号)、Ｅレジスタに０１Ｈ（機能番号)、Ａレジスタに０１ＨをいれてＦＦＣＡＨ番地をコールすると、画面が漢字モードになります。この状態から、Ａレジスタを０にして、ほかの値をさっきと同じ値にして再びＦＦＣＡＨ番地をコールすると、普通のＡＮＫモードに戻ります。拡張ＢＩＯＳコールがどんなものか、だいたい分かってもらえたでしょうか。

２．７．２　拡張ＢＩＯＳの動作

先ほどの説明だと、どの拡張デバイスも同じＦＦＣＡＨ番地をコールするので、一度に２個以上の拡張デバイスを接続することはできないような気がしますが、もちろんちゃんとモデムとＭＳＸ－ＪＥを同時に使ったりすることができるのはよく知られたとおりです。どうやってこんなことをしているのかというと、それぞれの機器の初期化に秘密があるのです。
ＭＳＸをリセットすると、拡張デバイス（例えば、ＭＳＸ－ＪＥ）は、初期化のさいに、ＦＦＣＡＨ番地から５バイトに、自分をインタースロットコールするように値を書き込みます。初期化が終わって、システムが立ち上がると、ＦＦＣＡＨ番地が書き換えられているので、ＭＳＸ－ＪＥが使えるわけです。
それでは、このとき同時にモデムカートリッジが差されているとどうなるかというと、モデムはやはりＦＦＣＡＨ番地から５バイトに値を書き込もうとしますが、この場合、ＦＦＣＡＨ番地はすでにＭＳＸ－ＪＥによって書き換えられているので、モデムは、ＦＦＣＡＨ番地から５バイトの内容を、自分用に用意されたワークエリアにコピーし、それからモデムをインタースロットコールするようにＦＦＣＡＨ番地を書き換えるわけです。
そして、ユーザーが拡張ＢＩＯＳを使ってモデムを使おうとすると、ＦＦＣＡＨ番地にはモデムを呼びだすように書き換えられているので、普通にモデムが使えます。ＭＳＸ－ＪＥを使おうとしたときはどうなるかというと、実行はいったんモデムに移りますが、Ｄレジスタの値が自分が呼びだされるときと違うので、(呼びだすとき、モデムなら０８Ｈ、ＭＳＸ－ＪＥなら１０Ｈが入っている）そのまま、先ほどコピーした自分用のワークエリアにジャンプさせます。するとそこにはＭＳＸ－ＪＥをインタースロットコールするように書いてありますから、ちゃんとＭＳＸ－ＪＥの機能が使えるというわけです。
したがって、例えば、ＭＳＸ－ＡＵＤＩＯがないのにＭＳＸ－ＡＵＤＩＯを使おうとすると、レジスタの値が全く同じままで実行が帰ってくることになります。
と、拡張ＢＩＯＳは、このように使われるわけですが、モデムなどのスピードを要求される機器では、これでは遅くて使えません。それに、拡張デバイスがたくさんつながっているとどんどん実行が遅くなるので、タイミングがうまく取れなくなる恐れがあります。

そこで、モデムなどでは、最初に拡張ＢＩＯＳコールするときに、モデムがいるスロットと、それぞれの機能の飛び先が書いてあるジャンプテーブルの先頭アドレスを読みだすことになっています。そうすれば、拡張ＢＩＯＳコールは最初の一回で済み、あとはモデムカートリッジを直接インタースロットコールすればいいので、スピードも上がるしタイミングも取りやすくなります。

２．７．３　デバイス番号

各拡張デバイスに割り当てられたデバイス番号（ＦＦＣＡＨ番地コールするときに、Ｄレジスタにいれておく値）は、次のとおりです。

００Ｈ　すべてのデバイス
０４Ｈ　メモリマッパ
０８Ｈ　ＲＳ－２３２Ｃ及びモデム
０ＡＨ　ＭＳＸ－ＡＵＤＩＯ
０ＢＨ　ＭＩＤＩ
１０Ｈ　ＭＳＸ標準日本語入力フロントエンドプロセッサ（ＭＳＸ－ＪＥ）
１１Ｈ　漢字ドライバ
ＦＦＨ　システムエクスクルーシブ

メモリマッパの仕様については、第２部第３章のメモリマッパ拡張ＢＩＯＳの項を、ＭＳＸ－ＡＵＤＩＯの仕様については、「ＭＳＸ－ＡＵＤＩＯ技術資料」（アスキー社・通信販売のみ）、漢字ドライバの仕様については、「ＭＳＸ２＋パワフル活用法」（アスキー社）を参考にしてください。

ＲＳ－２３２Ｃ，ＭＳＸ－ＪＥに関しては、本書第４部、第５部で説明します。基本的な拡張ＢＩＯＳコールの仕方についても、こちらを参考にしてください。

システムエクスクルーシブというのは、ハードウェアメーカーが会社ごとに独自に機能を拡張する場合に使用するもので、取りあえず現在のところは関係ないので説明は省略します。

すべてのデバイスというのは、例えば、いくつの拡張機器が接続されているかを調べたいときなど、接続されているすべての機器に命令を与えたいときに使います。これについては次の節で説明します。

２．７．４　すべてのデバイスに対する命令

すべてのデバイスに命令を与えるには、Ｄレジスタに０、Ｅレジスタに機能番号をいれてＦＦＣＡＨ番地をコールします。スタックは必ずページ３に置き、幾らか余裕を持っておいてください。通常はＣ１００Ｈ番地以降に置きます。

機能番号０・デバイス番号の取得
　入力、Ｄに０
　　Ｅに０
　　ＢにＲＡＭのスロット番号
　　ＨＬにワークエリアのアドレス
　出力、ＨＬに書かれたアドレスにデバイス番号が入り、ＨＬがインクルメントされる。
　レジスタ、すべて
　接続されている拡張デバイスのデバイス番号が返ります。例えば、返ってきたときにＨＬが４だけインクルメントされていて、そこのアドレスに１０Ｈ，００Ｈ，０４Ｈ，００Ｈが入っていたら、ＭＳＸ－ＪＥ、メモリマッパの２つの機器が接続されているということになります。（下の図を参考のこと）

もとのＨＬ→

１０Ｈ	（ＭＳＸ－ＪＥのデバイス番号）
００Ｈ	（すべてのデバイスを意味する）
０４Ｈ	（メモリマッパのデバイス番号）
００Ｈ	（すべてのデバイスを意味する）

新しいＨＬ→

なお、このファンクションでは、漢字ドライバがあるかどうかを調べることができません。（著者もサンプルプログラムを組むまで知らなかった）恐らく、漢字ドライバだけはほかの拡張デバイスとは少し違った動作をするためだと思われます。
また、すべてのデバイスを意味する００Ｈは、拡張デバイス１つについて１つ出てきます。これは、すべてのデバイスが、「すべてのデバイス」としても動作するためです。

機能番号１・トラップ使用数の取得
入力、Ｄに０
Ｅに１
Ａに０
出力、Ａに拡張デバイスが使用するトラップ数
レジスタ、Ｆ，ＤＥ
ＢＡＳＩＣに「ON STRIG GOSUB」などの、強制的に実行をジャンプさせる命令がありますが、拡張デバイスがこのような機能（トラップ機能）を拡張するときに使用します。システムが使う機能なので、ユーザーが使うことはないでしょう。詳しい説明は省略します。

機能番号２・割り込み禁止宣言
入力、Ｄに０
Ｅに２
出力、なし
レジスタ、なし
拡張デバイスを使っているときに、プログラムが割り込みを禁止すると不都合が生じることがあります。例えば、ＲＳ－２３２がデータを受信中に割り込みを禁止されると受信データを取りこぼしてしまいます。このようなことが起こるのを避けるために、プログラムが割り込みを禁止する直前にこのファンクションを呼ぶようにしてください。１ｍ秒以上割り込みを禁止するときは必ず呼んでください。（１ｍ秒以上割り込みを禁止されると、ＲＳ－２３２Ｃは９６００ｂｐｓでの受信ができなくなります）各拡張デバイスは、割り込みが禁止されても異常動作が起きないよう内部で処理をしてから、（ＲＳ－２３２Ｃの場合はＸＯＦＦを送信してから）戻ります。

機能番号３・割り込み許可の宣言
入力、Ｄに０
Ｅに３
出力、なし
レジスタ、なし
機能番号２で禁止された割り込みを再び許可するときに呼んでください。

以上ですべての拡張ＢＩＯＳに与える命令について説明を終わります。このように、拡張デバイスに対する命令には、Ｄレジスタにデバイス番号、Ｅレジスタに機能番号をいれてＦＦＣＡＨ番地をコールすることによって与えます。ちなみに、機能番号０はどのデバイスでも、そのデバイスがどこのスロットのどの番地にあるのかを調べるための命令に使われています。４バイトの値を返すのですが、内容はデバイスによって少しずつ違います。詳しくは第４部、第５部を参考にしてください。

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆
ＭＳＸ－ＢＡＳＩＣで、０行に何かプログラムをいれてＬＩＳＴを取り、さらに０行の上にカーソルを合わせてＲＥＴＵＲＮキーを押します。そしてまたＬＩＳＴを取るとあら不思議、０行のスペースが１つ増えています。もう一度やるとスペースがまた増えます。ポケットのビスケットがたたくたびに増えるみたいで気分がいいのですが、これが有名なＢＡＳＩＣのバグです。
ＢＡＳＩＣのＢＡＳＥ変数の値を書き換えようとすると、画面が異常な表示をしてしまうことがあります。仕様書にはＢＡＳＥ関数の内容は書き換えることができると書いてあるのですが、ＢＡＳＩＣにバグがあって書き換えがうまく行かないようです。しかし、ワークエリアのＢＡＳＥ変数セーブエリアに直接値を書き込めば、ちゃんと異常動作なしに書き換えられます。
システムワークエリアのＪＩＦＦＹ（ＦＣ９ＥＨ，２）はタイマー割り込みのたびにインクルメントされるカウンターです。乱数の種にでも使ってください。

第８章　漢字ドライバ使用上の注意

漢字ドライバの使い方については、「ＭＳＸ２＋パワフル活用法」（アスキー社）を参考にしてください。ここでは、それには載っていない情報を主に取り上げます。

漢字ドライバが起動していると、画面はＳＣＲＥＥＮ５や７の状態になります。つまり、漢字ドライバが起動していると、ＳＣＲＥＥＮ０などとやっても画面の文字数が変わるだけで、やっぱり画面はグラフィックモードのままなのです。ここでＳＣＲＥＥＮ５などとやると、ＬＩＮＥ文やＰＡＩＮＴ文が普通に使えて、なおかつＰＲＩＮＴ文で漢字が表示できるという漢字グラフィックモードと呼ばれるモードに突入します。ただ、このモードでは漢字の入力はできません。（機械語プログラムを自分で組めば可能）

その他注意すべき点としては、漢字ドライバを起動すると、フリーエリアが４Ｋバイト減ってしまいます。（組み込まれたＭＳＸ－ＪＥの種類によっては、４Ｋバイト以内のこともありますが、プログラムは安全のため、必ず４Ｋバイトを漢字ドライバ用に確保してください）そこで、メモリをぎりぎりまで使っているソフトでは、漢字ドライバを起動すると暴走することがあります。また、最初に「ＣＡＬＬ　ＫＡＮＪＩ」命令で漢字ドライバを起動したときには、プログラムスタックの内容がすべてクリアされてしまいます。つまり、ＧＯＳＵＢ命令の飛んできた元の行番号や、ＦＯＲ～ＮＥＸＴループなどの状態、変数の内容がすべて消えてしまいます。漢字ドライバを使うプログラムでは、一番最初の行で「ＣＡＬＬ　ＫＡＮＪＩ」を実行しておいたほうがいいようです。

それから、漢字ドライバが起動していると、ファイル名にも漢字が使えるようになってしまうのですが、これをやると普通のモードでは絶対に呼びだせない名前のファイルができてしまうので、ファイル名に漢字を使うのは避けたほうがいいです。試しに、一文字目が「世」のファイルを普通のモードで指定しようとすると、「Bad file name 」のエラーになります。（ＤＯＳカーネルVer.2.31以降ではエラーになりません）

念のために漢字ドライバが書き換えるフックを紹介しておきます。プログラムがこれらのフックを使うときは注意してください。

ＦＤＡ４Ｈ（Ｈ．ＣＨＰＵ），ＦＤＡ９Ｈ（Ｈ．ＤＳＰＣ）
ＦＤＡＥＨ（Ｈ．ＥＲＡＣ），ＦＤＢ３Ｈ（Ｈ．ＤＳＰＦ）
ＦＤＢ８Ｈ（Ｈ．ＥＲＡＦ），ＦＤＢＤＨ（Ｈ．ＴＯＴＥ）
ＦＤＣ２Ｈ（Ｈ．ＣＨＧＥ），ＦＤＤＢＨ（Ｈ．ＰＩＮＬ）
ＦＤＥ５Ｈ（Ｈ．ＩＮＬＩ），ＦＦＢ６Ｈ（Ｈ．ＬＰＴＯ）

これらのフックが呼ばれるとき、裏レジスタ及びＩＸ，ＩＹレジスタの内容が破壊されます。したがって、それらのレジスタの内容が保持されると期待しているソフトウェアは、漢字ドライバを起動すると正常に動作しません。

なお、漢字ドライバが書き換えるフックは上記のほかにさらに次のようなものがあります。これらのフックが呼ばれるとき、裏レジスタ及びインデックスレジスタが保存されるかどうかは不明です。多分保存されるんじゃないかなあ。アスキーの人間じゃないから分からないや。

ＦＤ９ＦＨ（Ｈ．ＴＩＭＩ），ＦＦ８４Ｈ（Ｈ．ＷＩＤＴ）
ＦＦＣ０Ｈ（Ｈ．ＳＣＲＥ），ＦＦＣＡＨ（ＦＣＡＬＬ）

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

漢字ドライバ使用中のパターンネームテーブルの場所について。パターンネームテーブルは、以下のＶＲＡＭ上のアドレスに置かれます。

```
SCREEN2      1E20H+40*Y+X            （SCREEN2,3は、ＭＳＸ１に漢字ドライバを
SCREEN3      0B00H+16*Y+X             接続したときに使用されます）
SCREEN4      1B00H+40*Y+X     (0<=Y<=2)
             1B08H+40*Y+X     (3<=Y<=11)
SCREEN5      (6A00H+40*Y+X+RG23SA*128) AND 7FFFH
SCREEN6      (6A00H+80*Y+X+RG23SA*128) AND 7FFFH
SCREEN7      D400H+80*Y+X+RG23SA*256 AND FFFFH
SCREEN8      D400H+40*Y+X+RG23SA*256 AND FFFFH
```

通常は、ＳＣＲＥＥＮ５～７のいずれかが使われます。（漢字ＢＡＳＩＣのマニュアルも参照のこと）漢字ドライバは、ハードウェアスクロールを使っているので、パターンネームテーブルは、スクロールするたびにねちねちと移動します。

SCREEN2及び3は、ＭＳＸ１で漢字ドライバを起動させたときに使用されます。

第９章　ＭＳＸ－ＭＵＳＩＣ

２．９．１　ＭＳＸ－ＭＵＳＩＣ利用上の注意

ＭＳＸ－ＭＵＳＩＣは、豪華な機能を満載したＭＳＸ－ＡＵＤＩＯが全然売れなかったので、そこからサンプリングなどの機能を削った、いわば、廉価版ＭＳＸ－ＡＵＤＩＯです。しかし、ＭＳＸ－ＡＵＤＩＯが拡張ＢＩＯＳを使ってアクセスしたのに対し、ＭＳＸ－ＭＵＳＩＣでは直接ＲＯＭをインタースロットコールするなど､使い方は全然違います。音源チップも違いますが、どこがどう違うのかはよく知りません。

ＭＳＸ－ＭＵＳＩＣの音源チップはＹＭ２４１３（ＯＰＬＬ）です。チップの具体的な使用方法については､「ＭＳＸ２＋パワフル活用法」などを参考にしてください。

ＭＳＸ－ＡＵＤＩＯの音源チップはＹ８９５０（ＯＰＬ）です。チップの使い方については「ＭＳＸ－ＡＵＤＩＯ技術資料」（アスキー社）などを参考にしてください。

機械語でプログラムを組むときは、ＭＳＸ－ＡＵＤＩＯとＭＳＸ－ＭＵＳＩＣでは、操作の仕方が全然違いますから、一つのプログラムで両方に対応させることはできなくて、両方に対応したプログラムを別々に組むか、あるいは片一方にしか対応しなくするかのどちらかしかありません。つまり､「ＭＳＸ－ＡＵＤＩＯ対応」としか書いていないソフトはＭＳＸ－ＭＵＳＩＣでは使えません。逆もそうです。ゼビウスなどは両方に対応していますが、これは内部で処理を別々にやっているのでしょう。音のデータなどは、同じものが使えるのだろうと思いますが。

ＢＡＳＩＣからは、呼びだす方式が､「ＣＡＬＬ　ＡＵＤＩＯ」と「ＣＡＬＬ　ＭＵＳＩＣ」なだけで、ほかはほとんど同じなので、ちょっと書き換えれば、基本的に同じプログラムが使えますが、サンプリングなどＭＳＸ－ＡＵＤＩＯにはあってＭＳＸ－ＭＵＳＩＣのほうにない機能を使う命令は削除されています。

機械語からＭＳＸ－ＭＵＳＩＣを使うときには､「ＭＳＸ２＋パワフル活用法」に、ＦＭ　ＢＩＯＳというＭＳＸ－ＭＵＳＩＣを扱うための拡張されたＢＩＯＳの使い方が書いてあるので、これを参考にしてください。ただ、この本にはＭＳＸ－ＭＵＳＩＣがあるスロットを探す方法が書いてないので、ここで紹介すると、

●４０１ＣＨ番地から４バイトの内容が､”ＯＰＬＬ”の四文字だったら、そのスロットにはＭＳＸ－ＭＵＳＩＣがあります。

ちなみに、４０１８Ｈからの４バイトには”ＰＡＣ２”などの製品名を表す文字が入っています。内蔵の機種ではここが”ＡＰＲＬ”だそうです。間違って、４０１８Ｈ番地から８バイトが”ＰＡＣ２ＯＰＬＬ”ならＭＳＸ－ＭＵＳＩＣがあるのだろうと勘違いして、内蔵の機種ではＦＭ音源が使えなかった大間抜けなソフトがありましたが、名前を出すのはかわいそうなのでやめておきます。

ＭＳＸ－ＭＵＳＩＣを使用するときには､「ＣＡＬＬ　ＭＵＳＩＣ」命令でＭＳＸ－ＭＵＳＩＣを初期化しますが、このとき変数がクリアされ、ワークエリアに８０７バイトの領域が確保されます。機械語プログラムを併用するときは注意してください。

ＭＳＸ－ＭＵＳＩＣのＦＭ　ＢＩＯＳを起動するときは、Ａ０Ｈバイトのワークエリアが必要とされます。使用スタックサイズは２０Ｈバイトです。

ＦＭ　ＢＩＯＳのワークエリアはページ１に置くことはできません。また、ページ０に置いた場合、機種によっては暴走する可能性がありますので、ワークエリアはページ０には置かないほうがいいでしょう。

ＩＮＩＯＰＬを呼ぶ場合、ＦＭ　ＢＩＯＳがページ１のスロットを切り換えます。切り換えるだけなら別に困らないのですが、切り換えた後、もとに戻さずに帰ってきます。つまり、ＩＮＩＯＰＬを呼ぶとページ１のスロットが切り換わってしまうことがあるので、ちゃんとプログラムを組んだつもりでも、動作しないことがあるのです。

これを回避するために、ＩＮＩＯＰＬをコールする直前にページ１のスロットの状態を保存しておき、帰ってきたらページ１を切り換えてください。（ＦＭ　ＢＩＯＳは拡張スロット選択レジスタの状態だけを換えるので、そこだけを保存するのでも大丈夫です）

２．９．２　ＦＭ　ＢＩＯＳの演奏データ

ＦＭ　ＢＩＯＳのＭＳＴＡＲＴ，ＯＰＬＤＲＶを利用すれば、ＯＰＬＬに直接値を書き込まなくても、データを与えることによって音を鳴らすことができます。ここでは、ＯＰＬＬミュージックドライバの演奏データ、通称ＭＭＬについて解説します。

ＭＭＬのヘッダー

ＭＬＬのヘッダは、それぞれのチャンネルのデータの先頭オフセット値を順に並べていくような構造になっています。次の表を見てください。

ＦＭ６音＋リズム音源の場合

先頭アドレス→		
	0EH　00H	リズム部データの先頭オフセット値
ｵﾌｾｯﾄ +0002H→	２バイト	ＦＭ１ＣＨデータの先頭オフセット値
+0004H→	２バイト	ＦＭ２ＣＨ
+0006H→	２バイト	ＦＭ３ＣＨ
+0008H→	２バイト	ＦＭ４ＣＨ
+000AH→	２バイト	ＦＭ５ＣＨ
+000CH→	２バイト	ＦＭ６ＣＨ
+000EH→	演奏データ	

ＦＭ９音の場合

先頭アドレス→		
	12H　00H	ＦＭ１ＣＨデータ先頭オフセット値
ｵﾌｾｯﾄ +0002H→	２バイト	ＦＭ２ＣＨ
+0004H→	２バイト	ＦＭ３ＣＨ
+0006H→	２バイト	ＦＭ４ＣＨ
+0008H→	２バイト	ＦＭ５ＣＨ
+000AH	２バイト	ＦＭ６ＣＨ
+000CH	２バイト	ＦＭ７ＣＨ
+000EH	２バイト	ＦＭ８ＣＨ
+0010H→	２バイト	ＦＭ９ＣＨ
+0012H→	演奏データ	

１つのチャンネルのデータが終わったら、その次の番地から次のチャンネルのデータが始まり、その開始番地が先頭番地からのオフセットとして「２バイト」で与えられます。たとえば、ＦＭ９音の場合、第１チャンネルのデータは先頭から0012H番地後から始まりますから、最初のデータは12H,00Hになります。（これで意味が分かるかなあ）使用しないチャンネルの先頭オフセットは00H,00Hになります。

メロディ部のデータ

00H～5FH　音階の指定です。00Hが休符で、01Hがオクターブ１のＣの音、02HがＣ＃、03HがＤ、と、半音上がるごとにデータが１つ上がります。（次の図を参照）
さらに、続く１バイトが音長データになります。この音長データがFFHの場合は、次の１バイトも音長データとなります。これが、データがFFH以外になるまで続けられます。音長データの単位は1/60秒です。例えば［18H,FFH,10H］という値があったら、オクターブ３のＣの音を(255+16)*1/60秒鳴らすという意味になります。

60H～6FH　音量指定です。この値から60Hを引いた値が実際の音量となります。

70H～7FH　音色指定です。実際の音色データはこの値から70Hを引いた値になります。

80H　サスティン解除です。

81H　サスティン設定です。

82H　拡張音色設定です。続く１バイトの値（０～６３）がＲＯＭ内の音色番号になります。音色番号の最上位ビットは無視されます。

83H　ユーザー音色指定です。続く２バイトの値（下位、上位の順）が、音色データの先頭アドレスになります。

84H　レガートオフ（音を音符ごとに切る)。

85H　レガートオン（音を切らずにつなげる)。

86H　Ｑ指定。続く１バイト（１～８）で指定されます。レガートオンのときは、Ｑ指定は実行されません。

87H～FEH　現在未使用。

FFH　そのチャンネルのデータの終了コードです。

ユーザー音色指定のデータは、指定アドレスから８バイトにＯＰＬＬレジスタの００～０７の値が並ぶという、きわめて単純な形をしています。ＦＭ　ＢＩＯＳのＲＤＤＡＴＡでもこの形が使われます。

音階データの形（00Hは休符です）

オクターブ＼音階	C	#	D	D#	E	F	F#	G	G#	A	A#	B
1	01H	02H	03H	04H	05H	06H	07H	08H	09H	0AH	0BH	0CH
2	0DH	0EH	0FH	10H	11H	12H	13H	14H	15H	16H	17H	18H
3	19H	0AH	1BH	1CH	1DH	1EH	1FH	20H	21H	22H	23H	24H
4	25H	26H	27H	28H	29H	2AH	2BH	2CH	2DH	2EH	2FH	30H
5	31H	32H	33H	34H	35H	36H	37H	38H	39H	3AH	3BH	3CH
6	3DH	3EH	3FH	40H	41H	42H	43H	44H	45H	46H	47H	48H
7	49H	4AH	4BH	4CH	4DH	4EH	4FH	50H	51H	52H	53H	54H
8	55H	56H	57H	58H	59H	5AH	5BH	5CH	5DH	5EH	5FH	---

リズム部のデータ

リズム部のデータは１バイトで表され、ビット単位で機能の指定を行ないます。以下の表は１バイトをビット単位で示したものです。

V	0	1	B	S	T	C	H

Ｂ，Ｓ，Ｔ，Ｃ，Ｈの各ビットが立っていれば、その楽器が選択されます。Ｂはバスドラム、Ｓはスネアドラム、Ｔはタムタム、Ｃはシンバル、Ｈはハイハットです。

Ｖ＝１なら、リズム発生指定で、続く１バイトが音長データになります。音長データの形はメロディ部の場合と同じです。

Ｖ＝０なら、音量指定で、続く１バイトが音量データ（０～１５）になります。値は下位４ビットのみが有効です。

終了コードはメロディ部と同じでFFHです。

２．９．３　ＯＰＬＬを直接操作する場合の注意

Ｉ／Ｏポートを直接操作してＭＳＸ－ＭＵＳＩＣを使用する場合、以下のような処理を行う必要があります。これは、ＭＳＸ－ＭＵＳＩＣ内蔵機に外付けのＭＳＸ－ＭＵＳＩＣカートリッジを接続した場合に両方の音源が動作してしまうのを防ぐための機能があるからです。

１．まず、４０１８Ｈ番地からの８バイトの内容が”ＡＰＰＲＯＰＬＬ”の８文字であるスロットを探します。
２．もしあれば、ＭＳＸ－ＭＵＳＩＣが内蔵されているので、そのままＩ／Ｏの７ＣＨ，７ＤＨを使用してＭＳＸ－ＭＵＳＩＣを操作できます。
３．もしなかった場合は４０１ＣＨ番地からの４バイトの内容が”ＯＰＬＬ”の４文字であるスロットを探します。
４．もしなければＭＳＸ－ＭＵＳＩＣは存在しないので、当然、使用もできません。
５．あれば、そのスロットの７ＦＦ６Ｈ番地の内容を読み、ビット０を１にして書き込めば、そのスロットに搭載されているＭＳＸ－ＭＵＳＩＣが使用できるようになります。

なお、ＦＭ　ＢＩＯＳを使って操作する場合は上のようなチェックは全く必要ありません。本文どおりのチェックを行ってＭＳＸ－ＭＵＳＩＣを操作してください。
Ｉ／Ｏポートを直接操作してＭＳＸ－ＭＵＳＩＣをアクセスする場合、必ず以下のウェイト時間を守ってＯＰＬＬに値を書き込んでください。このウェイト時間を無視した場合、そのときの設定は正常に行われないことがあります。
アドレス指定の場合、３．３６μ秒。データ書き込みの場合２３．５２μ秒。
ＦＭ　ＢＩＯＳを使用する場合はＦＭ　ＢＩＯＳがこの時間だけウェイトをかけてくれます。ただし、ＦＭ　ＢＩＯＳは大変にありがたいことにこの時間だけ本当に待ってくれますので、高速化を必要とするソフトではＦＭ　ＢＩＯＳを使用せずにＩ／Ｏポートを直接操作したほうが効率的です。ただし、アスキーは直接操作されることはあまり好んでいないようです。

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆
ＦＭ音源を利用する場合は、マニュアルや仕様書だけを読んでいても意味がよくわからないことが多いので、ＦＭ音源の解説書などを参考にしてみてください。ＯＰＬＬの解説書なんかないじゃないか、と思われるかもしれませんが、他の音源チップの解説本でも案外参考になります。
音源内蔵カートリッジ（コナミのＳＣＣやＦＭ－ＰＡＣなど）を差すと、音が出なかったり、ＰＳＧと音源カートリッジの音のバランスが悪くなる機種があります。これは、カートリッジスロットの音源端子に信号がつながっていなかったり、音のボリューム調節のために使われている抵抗の値が機種ごとに違ったりするためのようです。音が鳴らない機種は、メーカーで修理してくれます。
カセットインターフェイスのタイミングは、機種ごとに微妙に異なります。これは、カセット出力するときに、Ｉ／Ｏポートに直接垂れ流ししてしまう機種と、一旦バッファにためて、そこから出力する機種があるためです。カセット入出力を行なうソフトは注意してください。
ＭＳＸ－ＭＵＳＩＣのＭＭＬマクロで、音量は０から１２７まで、すなわち７ビットが指定できますが、実際には下位６ビットのみが有効です。

第10章　動作チェック

ここでは、よく起きる動作不良の原因について簡単に述べます。

１．ＭＳＸ１で動くのに、ＭＳＸ２で動かない。
　メインＲＡＭが基本スロットにあるとプログラムが思い込んでいる、Ｉ／Ｏポートを直接いじっている、メインＲＯＭが基本スロットにあると思い込んでいる。
２．ＭＳＸ２では動くのに、ＭＳＸ１にＭＳＸ２バージョンアップアダプターをつないだときに動かない。
　Ｉ／Ｏポート、特にＶＤＰのポートの番地を調べずに直接いじっている、メインＲＯＭがスロット０にあると思い込んでいる。ＲＡＭがすべてページ３と同じスロットにあると思い込んでいる。
３．ヤマハＹＩＳ－８０５，ビクターＨＣ－９０，ＨＣ－９５，ソニーＨＢ－Ｆ５００の製造番号５５００以下の機械で暴走する。
　サブＲＯＭの切り替えがうまくいっていない、タイマ割り込みルーチンをページ０に置いている。
４．東芝ＨＸ－２３で暴走する。
　ＲＡＭがすべて同じスロットにあると思い込んでいる。
５．日立Ｈ２，Ｈ３，三洋ＭＰＣ－１０／ＭＫ２，ヤマハの機械など内部でハードウェアを拡張している機種で暴走する。
　拡張されていることを予期していない。内部拡張のＩ／Ｏポートが衝突している。
６．ソニーＨＢ－７０１ＦＤ，パイオニアＰＸ－７で動作しない。
　スロット２が拡張されていて、そこに内蔵ソフトがあることを予期していない、スロット３にカートリッジを差されることを予期していない。
７．拡張スロットに差されると動作しない。スロット２や３に差すと動作しない。
　拡張スロットに差されることを予期していない。スロット２や３に差されることを予期していない。
８．１９８５年以前に製造されたディスクドライブ（ヤマハＦＤ－０５など）をＭＳＸ２に差したとき動作しない
　１９８５年以前に製造されたディスクドライブに、問題があって、サブＲＯＭが呼びだせないことがある。第２部第２章を参考にしてプログラムを組むことによって、動作不良を回避することができる。
９．turboRのＲ８００モードで動作しない
　クロック数でソフトのタイミングを取っている。

このほかにも、ＲＡＭが特定のスロットにあると動かないなど、動作不良の型はいろいろあります。上に名前をあげた機種では動作不良が起きやすいようなので、これらの機種では重点的にチェックをしたほうがいいようです。特に、ヤマハＦＤ－０５を接続したときについて重点的にチェックしてください。（ただ、現在ではこのドライブは幻の製品になっています）ＭＳＸ２バージョンアップアダプターなどは忘れられがちですが、これをつないだときに動かないソフトが多いようなので注意してください。

ＭＳＸ用のソフトは、すべての機種で動作確認をすることになっているのですが、中には、誰も売っているところを見たことがない日立Ｈ５０などという機種もあるので、動作チェックもできるところまで、ということになるのでしょう。

動作チェックのこつとして、何社かの別の会社の機種でチェックするというのがあります。例えば、ソニーの機種などは、スロット構成などはどの機種もほとんど変わりません。そこで、ソニーで動くからと安心していると、松下の機械ではスロット構成が全然違うので動かない、ということがたまにあります。自分の機種のメインＲＯＭはスロット０にあるので、つい、調べずにプログラムを書いてしまった、というようなのがもっともありがちな動作不良です。おまけに、自分の機種ではちゃんと動くので、もっと始末が悪いわけです。ＭＳＸで多いバグは、うっかり自分のいるスロットを切り換えてしまった。データエリアをＲＡＭに置いたら、読み書きするときにそこが裏になっていることに気づかなかった、スタックのあるページをなにげなく切り換えてしまった、というふうに、たいていスロットが絡んでいることが多い、というのも覚えておいてください。

なお、ヤマハＹＩＳ－８０５にはＲＡＭがＳＬＯＴ３－１にあるバージョンも存在します。ご注意ください。

第３部　　ディスク操作

◆

ここでは、ＭＳＸのディスク操作方法のうち、あまり知られていない機能について説明します。

第１章　ディスクの存在の有無

ディスク版のソフトがディスクの存在を確かめてもナンセンスですが、ＲＯＭ版やテープ版のソフトでは、ディスクが接続されているかどうかを確かめたいときがあります。確かめる方法は、次のとおりです。

●ＲＡＭのＦＦＡ７Ｈ番地の内容がＣ９Ｈならば、ディスクは接続されていない。それ以外の値なら、ディスクは接続されている。

第２章　フォーマット

グラフィックエディターなどで絵を描いていて、いざセーブしようと思ったときにフォーマットしたディスクがなくて、さらにグラフィックエディターにフォーマット機能がついていなくて苦心して描いた絵をなくなく消去しなければならないときなどは、とても悲しいものです。ほかにも、ゲームで途中経過をセーブしようとしたら、やはりフォーマットしたブランクディスクがなくてセーブできなかったりすることもあるでしょう。ところが、ＭＳＸでは、ソフトウェアがディスクをフォーマットすることは出来るのです。しかし、その方法があまり知られていなかったために、そのようなソフトはほとんどありませんでした。ソフトの内部でフォーマットが出来るソフトには、例えば「文書作左衛門」や、「Ｆ１ツールディスク付属グラフィックエディター」などがあります。

●ソフトがディスクをフォーマットする方法
マススロット（ドライブＡを制御しているインターフェイスのスロット）の４０２５Ｈ番地をコールします。
入力・ＨＬレジスタにワークエリアの開始アドレス
　　ＢＣレジスタにワークエリアの長さ
出力・エラーが起きなければＣｙをリセット、エラーが起きればＣｙをセットしさらにＡにエラー番号（エラー番号は第３章の表と同じ）
破壊されるレジスタ・すべて

なお、ワークエリアは先頭番地が８０００Ｈ番地よりも大きく、ＴＰＡに全部が収まっている必要があります。長さは、８Ｋバイトあれば十分のようです。（一番たくさんワークエリアを必要とするディスクインターフェイスが、約７Ｋバイトのワークエリアを使っている）また、マススロットのスロット番号は、Ｆ３４８Ｈ番地に入っています。
ここをコールすると、ＢＡＳＩＣやＤＯＳでフォーマットしたときと同じようにドライブ番号、メディアを指定してフォーマットします。つまり、完全オートのフォーマットルーチンは出来なくて、必ずキーボードからＡとか２とか入力することになります。また、文字を表示するので、グラフィック画面でフォーマットするとメッセージが表示されませんが、これはフックのＨ．ＣＨＰＨを書き換えて、グラフィック画面に文字が表示出来るようにすればいいわけです。実際、「文書作左衛門」などではそうやっています。
メインＲＯＭの０１４７Ｈ番地や、ＲＡＭのＦＦＡＣＨ番地をコールしてもフォーマット出来るようですが、こちらは使いません。

第３章　ディスクエラー

例えば、ディスクの読み書きで、読み込みや書き込みに失敗すると、帰ってくるレジスタの内容で、それを知ることができます。しかしながら、ディスクが入っていないときにディスクをアクセスしたときなどは、ＤＯＳ上では「Not ready error」などと表示が出

て実行が止まってしまいます。（このように、ＤＯＳがメッセージを表示して実行が一旦停止するエラーをハードウェアエラーと呼びます）そこで、これを回避するために、ディスクエラーをユーザーが直接処理する方法が必要になります。
　ＲＡＭのＦ３２３Ｈ番地からの２バイトは、ＤＩＳＫＶＥと呼ばれ、ディスクエラー（ハードウェアエラー）が起きたときに飛ぶ番地のアドレスが書いてあるアドレスが入っています。つまり、Ｆ３２３Ｈ番地から２バイトにＣ３ＤＥＨと書いてあって、Ｃ３ＤＥＨ番地から２バイトにＢＥ３２Ｈと書いてあったら、ディスクエラーが起きたときにはＢＥ３２Ｈ番地に実行が移ることになります。何だかややこしいことをしているのですが、ＭＳ－ＤＯＳを元にしてＭＳＸ－ＤＯＳが作られたので、このような面倒臭いことをしているようです。とにかく、このＦ３２３Ｈ番地からの２バイトを書き換えれば、ユーザーがディスクエラーを処理することができるわけです。なお、このＦ３２３Ｈ番地に書いてあるアドレスに書いてある番地にジャンプしたときには、Ｃレジスタに下の表にあるようなエラー番号が、Ａレジスタにドライブ番号（Ａ：０，Ｂ：１…）が入っています。そこで、例えば「書き込みに失敗しました」とか、「ディスクが異常です」とか、エラーによってメッセージの内容を変えることができるわけです。ただし、ディスクエラー処理ルーチンはページ１には置くことができません。ディスクドライブの機種によっては、通常とは違うエラーを出すことがあるので注意してください。例えば、あるドライブはCRC ERRORを出すのに、別のドライブではRECORD NOT FOUNDを出したりすることがあります。

b7	b6	b5	b4	b3	b2	b1	b0	エラーの種類
1	X	X	X	X	X	X	X	BAD FAT
0	0	0	0	0	0	0	W	WRITE PROTECT
0	0	0	0	0	0	1	W	NOT READY
0	0	0	0	0	1	0	W	CRC ERROR
0	0	0	0	0	1	1	W	SEEK ERROR
0	0	0	0	1	0	0	W	RECORD NOT FOUND
0	0	0	0	1	0	1	0	UNSUPPORTED MEDIA
0	0	0	0	1	0	1	1	WRITE ERROR
0	0	0	0	1	1	0	W	OTHER ERROR

unsupported mediaは、機種によってはない場合もある

　なお、Ｘは不定、Ｗは書き込み時に１、読み込み時には０になります。つまり、Ｃレジスタに０３Ｈが入っていたら、書き込み時にNOT READYのエラーが出たということが分かります。ＣＲＣエラーというのは、読み込めたけれどもデータのチェックサムがおかしいというエラーです。（正確にはチェックサムとはちょっと違う）ディスクの中には、データのほかに、ＣＲＣという、データを全部つなげて一つの大きな数にして（１６バイトのデータだったら、１２８ビットからなる巨大な２進数にする）それを特定の数（10001000000100001B）でＸＯＲをとった結果が入っています。読み込み時にこの値を計算してディスク上に書いてあるＣＲＣの値と比べて、違っていたらエラーを出すというわけです。これで、ディスクが壊れたときには異常なデータを読み込む前に事前にチェックできます。
　実際にＤＩＳＫＶＥを書き換えるときは、ディスクアクセスの直前にＤＩＳＫＶＥを書き換え、スタックを保存してからＢＤＯＳコールすることが多いようです。そして、エラー処理ルーチンの中で、スタックを元に戻すわけです。（エラー処理プログラムは、ＢＤＯＳコールルーチンからジャンプする）そして、ＢＤＯＳコールから帰ってきたり、エラー処理ルーチンから帰ってきたときにＤＩＳＫＶＥの内容を元に戻します。なお、ソフトが実行を終了するときには、必ずＤＩＳＫＶＥの内容を元に戻してください。そうでないと、次に違うプログラムでディスクエラーが起きたときに暴走します。ディスクＢＡＳＩＣ上でこのＤＩＳＫＶＥを書き換えるときは、ここに書いてあるアドレスに書いてある番地に書いてある番地に実行が移ったときに、ページ１がディスクインターフェイスＲＯＭに切り換えられているので、エラー処理プログラムは必ずページ２か３に置かれ、かつ、エラー処理が終わったらページ１をメインＲＯＭに戻す必要があります。
　参考までに、エラーが起きたときにＤＯＳはどのような処理をしているかといいますと、まず、Ｃレジスタを見て、エラーメッセージを表示し、 A,R,Iのいずれかのキーを押すようにというメッセージを出して、一旦実行を停止します。Aが押されたらＣレジスタに２を、Rなら１を、Iなら０をいれて、エラー処理ルーチンからリターンします。なお、Ｃ以外のレジスタには何が入っていてもかまいません。
　ソフトウェアの中には、わざとＣＲＣエラーを出すようにディスクのデータを書き換えていて、ディスクエラーが起きたら本物だと識別するようなプロテクトを掛けている物があります。ＭＳＸでは、よっぽど特殊なことをしなければわざとエラーが起きるようなデータをディスクに書き込むことはできませんから、エラーの起きないディスクでは立ち上

がらないようにしておけば、コピーされることを防ぐことができます。もっとも、プログラムを直接解析して書き換えれば、コピーした物を立ち上げることができます。
　ＤＩＳＫＶＥは、文章で説明されてもよく分からないことが多いので、サンプルディスクも参考にしてください。

第４章　ＣＴＲＬ＋Ｃの処理

　ＤＯＳ上で、ＣＴＲＬ＋Ｃを自分で処理したいときがあります。
　ＭＳＸ－ＤＯＳのＣＴＲＬ＋Ｃの処理は、ディスクエラーの場合と似ていて、Ｆ３２５Ｈ番地から２バイトに書いてあるアドレスに書いてあるアドレスがコールされます。そこで、ここを書き換えればＣＴＲＬ＋Ｃをプログラムで処理することができます。このほかのことはディスクエラーとほとんど同じですので、ディスクエラーの処理のところも参考にしてください。もちろん、プログラムが実行を終了するときは、必ず値を元に戻しておかないと暴走します。

第５章　ディスクが止まらないとき

　ディスクアクセスが終わってすぐにプログラムで割り込みを禁止すると、ディスクドライブのモーターが回転したまま止まらなくなるドライブがあります。これは、割り込みの数が一定回数になったらドライブを止めるようになっている機種でおこります。
　止まらなくなったディスクドライブを停止させるには、マスタースロット（ドライブＡをコントロールしているディスクインターフェイスが入っているスロット。Ｆ３４８Ｈ番地にスロット番号が書いてある）の４０２９Ｈ番地をコールしてください。
　ただし、これは後になってから付け足された機能なので、とても古いドライブではこのファンクションが存在しないことがあります。そこで、この番地の内容が０でないことを確かめた上でコールしてください。０のときは、ドライブを停止させることはできません。

第６章　ＣＯＭＭＡＮＤ．ＣＯＭ

　ユーザーがＣＯＭＭＡＮＤ．ＣＯＭという名前のファイルを作ってＤＯＳから直接プログラムを起動しようとすると、大抵暴走します。ＤＯＳのコマンドレベルに戻ってくることがないプログラムなら、ＣＯＭＭＡＮＤ．ＣＯＭという名前であっても構わないのですが、暴走を起こす原因は大抵スタックの初期化にあります。ＤＯＳがＣＯＭＭＡＮＤ．ＣＯＭという名前のファイルを実行するときは、スタックポインタがＤＯＳのワークエリアの内部にあります。つまり、あまり大量にスタックを使うプログラムでは、スタックがＤＯＳそのものを破壊して暴走してしまうわけです。メインＲＡＭの０００６Ｈ番地から２バイトには、ＤＯＳのシステムの先頭番地が入っています。ここから上の番地をプログラムが使うと暴走してしまうわけですから、プログラムの最初で、　ＬＤ　ＳＰ，（０００６Ｈ）　とスタックを初期化すれば、暴走しなくなるはずです。

第７章　２ドライブシミュレーション

　ＭＳＸ－ＤＯＳには２ドライブシミュレーションという便利な機能があり、１ドライブでも疑似的に２ドライブとして使うことができます。２ドライブシミュレーションでは、ディスクを入れ替えるとき、例えば「Insert diskette for drive B: and strike a key when ready 」というようなメッセージがＤＯＳによって表示されますが、この処理を書き換えることができます。このメッセージが表示される前に、フックのＦ２４ＦＨ番地がコールされます。そこで、ここを書き換えれば、２ドライブシミュレーションのメッセージを書き換えたり、別の処理を行なったりすることができます。このフックが呼ばれたときにはＡレジスタにドライブ番号（Ａ：４１Ｈ，Ｂ：４２Ｈなど）が入っています。なお、スタックを２レベル（２バイト）増やしてからリターンすると、ＤＯＳのメッセージ表示とキー入力待ちが行なわれません。このフックがコールされたときはページ１がディスクインターフェイスに切り換わっているので注意してください。Ａ以外のレジスタは破壊してかまいません。また、このフックを書き換えている間は、ＤＯＳのファンクションコールでメッセージ表示をすることはできません。もちろん、このフックを書き換えたプログラムを終了するときには、フックの内容をちゃんと元に戻してください。

第８章　物理ドライブ

ＤＯＳのＢＤＯＳコールの１８Ｈを使えば、論理ドライブの数が分かりますが、２ドライブシミュレーションが行なわれているときは、これは実際の物理ドライブ数とは違います。例えば、Ａ，Ｃ，Ｄが物理ドライブで、Ｂドライブが仮想ドライブというようなこともあります。今まで、物理ドライブの数をソフトウェアが数えることはできないとされていたのですが、ＭＳＸマガジン１９８８年２月号で、物理ドライブの数を数えるかなり過激な方法が紹介されました。こんなことまでして物理ドライブの数を数える必要はないと思うのですが、一応その方法を紹介します。

まず、Ｆ２４ＦＨ番地からの２ドライブシミュレーションのフックを書き換えて、ＤＯＳの２ドライブシミュレーションのメッセージとキー入力を利かなくします。（ＤＩＳＫＶＥも書き換えて、ディスクが入ってなくてもエラーが起きないように細工もしなければなりません）

次に、論理ドライブを後のほうから順々にアクセスします。（アクセスするだけ）そして、次に、Ａドライブから順々にアクセスしていきます。そして、Ｆ２４ＦＨ番地が呼ばれたら、そのドライブは仮想ドライブと分かります。もし仮想ドライブだったら、一つ前のドライブをアクセスし、（元の状態に戻す）また次のドライブをアクセスします。

これで、どのドライブが物理ドライブで、どのドライブが仮想ドライブか分かります。最初にドライブを逆にアクセスするのは、場合によっては物理ドライブと仮想ドライブが入れ替わっていることがあるためです。こういう大変に面倒な操作をすれば、物理ドライブの数を調べることは可能ですが、はっきりいってあまり意味はないと思います。

第９章　ＴＰＡの上限

ディスクをつないでいない状態だと、ユーザーはＦ３７ＦＨ番地まで目一杯使ってプログラムを組むことができます。しかし、ディスクＢＡＳＩＣでは、ディスクのワークエリアが使うので、ユーザーはＤＥ３ＦＨ番地までしか使えなくなってしまいます。ＤＯＳ上ではもう少しユーザーエリアが減って、Ｄ４０５Ｈ番地までです。本当は、メインＲＡＭの０００６Ｈ番地から２バイトに入っているのがＤＯＳシステムの先頭番地なので、これの１バイト前までが使えることになるはずなのですが、（この値はＤ６０５Ｈ）安全のため、ここまでを使うのが良いだろうということになっているようです。なお、これは、ＤＯＳ１及びＤＯＳ２に２ＤＤのディスクを２台つなげた場合のエリアです。３台以上の場合は、１台につき約１５５８～１５９０バイト弱減ります。（機種によって多少異なります）ディスク版のソフトは、２台ディスクがつないである状態で動けばいいことになっているので、その環境で動くようにソフトを作れば良いでしょう。なお、漢字ドライバを積んでいる場合に、ＣＡＬＬ　ＫＡＪＩなどで漢字ドライバを有効にすると、ＢＡＳＩＣ上でも、ＤＯＳ上でも、さらにユーザーエリアが約４Ｋバイト減るのでこれにも注意してください。また、ＭＳＸ－ＭＵＳＩＣを初期化すると、やはりユーザーエリアは８０７バイト減少します。プログラムの中でユーザーエリアの上限を調べて、足りない場合にはメッセージを出して実行を止めるのが良いプログラミングとされているそうです。

第10章　ＰＨＹＤＩＯモード

例えば、「イース」のように、ディレクトリを取ろうとすると、「Disk I/O error」が出て読めないディスクソフトがあります。これは、ブートセクタのメディアＩＤを書き換えていて、普通とは別のディスク管理をしているためです。ところで、ＭＳＸでは、ディスクアクセスで、ディスクにメディアＩＤが書いていないと、先ほどのようにエラーが出てしまいます。それでは、このようなソフトでは、どうやってディスクアクセスをしているのでしょうか。

種を明かせば、ＰＨＹＤＩＯと呼ばれる隠しＢＩＯＳを使っているのです。

０１４４Ｈ番地をコールすることで、直接ディスクの読み書きを行う事が出来ます。

このルーチンでは、メディアＩＤを指定してから呼ぶので、ディスクのブートセクターにメディアＩＤが書いてないディスクでも読み書きできます。主に、高速でディスクを読み込むことが必要なソフトで使われています。０１４４Ｈ番地はすぐにフックのＦＦＡ７Ｈ番地（Ｈ．ＰＨＹＤ）にジャンプするので、もっと高速化したいソフトでは、こちらをコールしている場合もあります。隠しＢＩＯＳなので、あまり使わないほうがいいような気もするのですが、市販ソフトがみんなこれを使っているのですから、たぶん使っても大丈夫なのでしょう。メディアＩＤは、１ＤＤ．９セクターフォーマットのときはＦ８Ｈ，

２ＤＤ．９セクターのときはＦ９Ｈ、１ＤＤ．８セクターのときはＦＡＨ，２ＤＤ．８セクターのときはＦＢＨになります。ほかにも、２Ｄや１ＤのときなどにもメディアＩＤはついているのですが、関係ないと思われるので省略します。
なお、ＭＳＸ－ＤＯＳ２でＰＨＹＤＩＯを使うと、読み書きは普通にできるのですが、ドライブをＰＨＹＤＩＯで変更しようとするとうまく動きません。注意してください。
レジスタの設定など詳細は、第８章第１部をご覧下さい。

第11章　バッチの停止

例えば、ＤＯＳ上でとても長いバッチコマンドを実行すると、実行中に異常が起きても、そのままバッチコマンドの実行が継続されるので、その後の処理がめちゃめちゃになってしまいます。そこで、ＣＯＭファイル実行中などに、それ以後のバッチの処理の停止をしたいときは、ＲＡＭの０００６Ｈ番地から２バイトに書かれているアドレスに１３Ｈを足したアドレスに、０を書き込んでください。以後、バッチの実行が停止します。
ただし、ＭＳＸ－ＤＯＳ２を差していて、ＣＯＭＭＡＮＤ２．ＣＯＭが起動中には正常に動作しません。ＤＯＳ２を差していても、ＣＯＭＭＡＮＤ．ＣＯＭが起動しているときは、正常に動作します。

第12章　ＤＯＳのファンクションコール

ＭＳＸ－ＤＯＳのファンクションコールを使うときは、ＤＯＳ上では０００５Ｈ，ＢＡＳＩＣ上ではＦ３７ＤＨをコールしますが、Ｆ３７ＤＨ番地コールでは、ページ１からのコールと、ページ１への読み込み、書き込みはできません。ＦＣＢをページ１に置くこともできません。しかし、ＤＯＳの０００５Ｈコールのほうはそのような制限はありません。

第13章　日本語ＭＳＸ－ＤＯＳ２利用上の注意

ＤＯＳ１上で動作するソフトはほとんどそのままＤＯＳ２上でも動作するのですが、一部非互換性があって、正常動作をしないことがあります。
まず第一に、ファイルアクセスの方式がが少し違います。そこで、ＤＯＳ２で作成したファイルがＤＯＳ１で読めなくなるようなことがまれにあります。次に、ＣＯＰＹ命令に互換性がありません。ＣＯＰＹ命令や、ディスク－ＶＲＡＭ間直接転送を使っているソフトは、ＤＯＳが変わると動作不良を起こすことがあります。これはＢＡＳＩＣでも機械語でも同じです。それから、いくつかのファンクションの機能が若干変更になっているので、これらのファンクションを使用しているソフトは、自分には問題がないのにＤＯＳ２で動かないことが可能性としてはありえます。
その他として、システムコールの１４Ｈ（ＣＰ／Ｍ互換のファイルアクセス）を使ったとき、ＣＰ／ＭやＤＯＳ１は、最後に半端が出るとそこに１ＡＨを詰めるのですが、ＤＯＳ２では００Ｈを書き込むので、たまに読み書きに不都合が起きる場合があるようです。もっとも、この機能はあまり使われないので、困ることはほとんどないでしょう。
また、ＤＯＳ２では裏レジスタが保存されますが、ＤＯＳ１では保存されません。また、ＤＯＳ１では００８０Ｈ番地からのＤＴＡアドレスに保存されるコマンドパラメーターの末尾が0DHで終わりますが、ＤＯＳ２では00Hで終わるので、そのへんも注意してプログラムを組んでください。これ以外にも若干あまり問題にならない小さなバグ、の・ようなものがいくらかあるようです。

第14章　デバイスドライバの接続

デバイスドライバなどを常駐させる場合は、カートリッジのＩＮＩＴルーチン実行の際に、ＨＩＭＥＭ（ＦＣ４ＡＨ）を自分が使用する大きさだけ減少させてください。この新しいＨＩＭＥＭと古いＨＩＭＥＭの間が、自分用のワークエリアになります。この新しいＨＩＭＥＭの値はＳＬＴＷＲＫ（ＦＤ０９Ｈ）の自分用の２バイトに保存しておいてください。

第15章　ディスクの起動手順

「ＭＳＸテクニカルハンドブック」９７ページ、『ＭＳＸ－ＤＯＳの起動手順』の追加説明をします。５．の時点でＣ０１ＥＨ番地がコールされるとき、メモリは以下のような状態になっています。

ページ０・メインＲＯＭ　ページ１・ディスクＲＯＭ　ページ２及び３・メインＲＡＭ

この時点ではＢＩＯＳなどでページ１をメインＲＡＭに切り換えることができません。また、ＤＯＳのＢＤＯＳコールも行えません。したがってユーザープログラムの起動には不適当です。

７．の時点でＣ０１ＥＨ番地がコールされるときは、メモリは以下の状態になっています。

ページ０、２及び３・メインＲＡＭ　ページ１・ディスクＲＯＭ

ここで、ＤＥレジスタにはコールするとページ１がＲＡＭに切り換わる番地（具体的にはＦ３６ＢＨ）が、ＨＬレジスタにはディスクエラー処理ルーチンの番地が書いてある番地が書いてある番地（具体的にはＦ３２３Ｈ）が入っています。Ａレジスタが０なら起動直後を意味します。（ここまでは公開情報）この時点ではまだ０００００Ｈ番地のＤＯＳのリブートと０００５Ｈ番地のＢＤＯＳコールは使えないのでＢＤＯＳコールはＦ３７ＤＨ番地を使います。（未公開情報）

第16章　ハードディスク補足説明

アスキー社のハードディスクインターフェイスを買えば､技術資料がついてきますので､それを参考にできるのですが、ハードディスク（ＨＤＤ）を持っていない方がディスクを操作するソフトを作った場合、ハードディスクのことを想定していないためにソフトが動作しなかったり､ハードディスクの中身を破壊するソフトを作ってしまうことがあります。ここではそういうことが起きないよう、ハードディスクをお持ちでない方のために最低限の情報を記します。

ます、容量が違います。単純ですがＨＤＤではＭＳＸ－ＤＯＳ（２）が使用できる最大容量の３２Ｍバイトまでのディスクが存在します。ディレクトリーエントリーの数も違います。例えばＤＯＳ１フォーマットしたＨＤＤには、最高２５４個までのファイルが作成できます。（ＤＯＳ２ではルートディレクトリーに最大２５５８）従ってＤＰＢを参照しているソフトはＤＯＳ２で動作しないことがあります（ＤＰＢではディレクトリーエントリーに１バイトの値しか返さない)。またクラスターサイズも違います。通常のＭＳＸ用のフロッピーディスクは１Ｋバイト（２セクター）で１クラスターですが、ＨＤＤでは容量などにもよりますが、最大６４Ｋバイトで１クラスターという設定まで行うことができます。もっともそんな設定をしたら１バイトファイルでも６４Ｋの領域をとってしまうので通常はそんな値を指定しませんが。例えば３２Ｍバイトでフォーマットすると、最低でも８Ｋバイトで１クラスターになります。１セクターの大きさは５１２バイトで変わりません。

ドライブがフロッピーディスクなのかハードディスクなのかを知る方法は以下のとおりです。ＤＯＳ２でしか調べられませんが、本文第８部第３章のＣＨＯＩＣＥを使えば恐らく（メーカー非公開ですが）ＤＯＳ２と同じ方法で調べられると思います。

Ｂレジスタに調べたいドライブ番号（Ａ：＝１…)、Ａに００Ｈ、Ｃに６７Ｈを入れ、ＢＤＯＳコール（ＤＯＳ上では０００５Ｈ番地コール）します。Ａにエラー番号（ＨＤＤの場合必ずＦ０Ｈ)、Ｂに文字列のスロット番号、ＨＬに文字列の長さが入って返ってきます。この文字列は、ディスクをフォーマットするときに画面に表示される文字列で、ＨＤＤの場合ＨＬで指定される番地には００Ｈが入っています。（つまり、フォーマットできないということです）そして、その００Ｈのすぐあとに、”ＳＣＳＩ７ＦＦＣ”という８文字の文字列が入っています。最初の４文字がＩＤで、あとの４文字はＨＤＤが使用するメモリマップドＩ／Ｏの番地です。

ＨＤＤで使用するＩ／ＯはメモリマップドＩ／Ｏの７ＦＦＣＨ～７ＦＦＥＨで、７ＦＦＣＨがデータ（読み書き)、７ＦＦＤＨがコマンドポート（書き込み)、７ＦＦＥＨがステータスポート（読み込み）です。ただしページ１に転送データがあるときはこの番地がＢＦＦＣＨ～ＢＦＦＥＨに変わります。

第４部　　ＲＳ－２３２Ｃ・モデム

◆

ここでは、ＲＳ－２３２Ｃ及びモデムを機械語から利用するための拡張ＢＩＯＳコールの方法と、それらの機能についての説明を行ないます。ＢＡＳＩＣからＲＳ－２３２Ｃやモデムを使う場合には、ＲＳ－２３２Ｃやモデムの取扱説明書を参考にしてください。ここでは、紙面の都合から、拡張ＢＡＳＩＣの使用方法については触れません。

第１章　ＲＳ－２３２Ｃとモデム

ＭＳＸのシステムは通常ＲＳ－２３２Ｃとモデムを区別しません。従って、プログラム上はどちらも同じものとして利用するわけです。ただし、モデムの拡張ＢＡＳＩＣにはオートダイアル機能があるのに、ＲＳ－２３２Ｃにはない、ＲＳ－２３２Ｃはボーレートを１９２００ボーまでどんな値でも指定できるのに対し、モデムでは一般にボーレートは３００ボーや１２００ボーといった値に固定されているといった違いがあります。
なお、どちらの拡張ＢＡＳＩＣも持っている命令は、次のとおりです。

新たに拡張されたステートメント
「CALL COMINI,CALL COM GOSUB,CALL COMON,CALL COMOFF,CALL COMSTOP,CALL COMBREAK,CALL COMDTR,CALL COMSTAT,CALL COMTERM」

ＲＳ－２３２Ｃ機能が追加されたコマンド及びステートメント
「SAVE,LOAD,RUN,MERGE,OPEN,CLOSE,PRINT#,PRINT# USING,INPUT#,LINEINPUT#」

ＲＳ－２３２Ｃ機能が追加された関数
「EOF,INPUT$,LOC,LOF」

これらの機能は拡張されて、新たに使えるようになります。元々ある機能はデバイスにＲＳ－２３２Ｃやモデムを使えるようになります。これらの機能の具体的な使い方については、ＲＳ－２３２Ｃ及びモデムカートリッジの説明書を参考にしてください。
ただ、モデムカートリッジのユーザーはボーレートを指定する方法が分からないと思われるので、ここに記しておきます。
「CALL　COMINI」命令でボーレートを指定する際、[50,75,110,300,600,1200,1800,2000,2400,3600,4800,7200,9600,19200]ボーを指定するときには数字をそのまま受信、送信ボーレートの部分に代入します。これ以外の値を指定するときには[－１８４３２００／(求めるボーレート＊１６)] を指定します。(値は負になります) 端数は四捨五入します。ただし、モデムでは固定したボーレートしかサポートしていないので、このような値を代入してもおそらく無視されてしまうでしょう。

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆
ディスクのマスタースロットの４０１０Ｈ番地から４０３０Ｈ番地までをコールすることによりいくつかのディスク操作が行なえます。ほとんどの機能が未公開なのが残念ですが。(これらのルーチンに関しては第８部第３章を参照のこと) また、ＤＯＳ２を差すとこれらの機能のほとんどが動作しません。ところが、我々の必死の解析の結果、ワークエリアのＦＢ２２Ｈ番地に、真のマスタースロットのスロット番号が書いてあるという事実が発覚しました。つまり、ＤＯＳ２を差してしまうと、マスタースロットがＤＯＳ２の差さっているスロットになってしまいます。したがって、マスタースロットをコールするつもりでＦ３４８Ｈ番地に書いてあるスロットをインタースロットコールすると、ＤＯＳ２のカートリッジを呼び出してしまうことになります。ＤＯＳ２内部ではこれらのルーチンが呼ばれるとたいていエラーが出て帰るか、または何もせずに帰るようにプログラムされているので、ＤＯＳ２ではこれらの機能が使えなかったわけです。(一部使える機能もあります) しかし、ＦＢ２２Ｈ番地に真のマスタースロットのスロット番号が書いてあるのならば、Ｆ３４８Ｈ番地のほうではなくてこちらのスロットを呼びだせば、今までどおりこれら未公開ルーチンを使えるわけです。このワークエリアも多分未公開だと思いますが、まあ何とかなるでしょう。どうせ未公開だし。

第２章　ＲＳ－２３２Ｃ・モデムの拡張ＢＩＯＳコール

ＲＳ－２３２Ｃの拡張ＢＩＯＳコールの方法について説明します。なお、これらの方法はモデムカートリッジの場合も同じです。拡張ＢＩＯＳコールがどんなものかについては第２部第７章を参考にしてください。

まず、Ｄレジスタに０８Ｈ（ＲＳ－２３２Ｃのデバイス番号)、Ｅレジスタに０、ＢレジスタにＲＡＭのスロット番号、ＨＬレジスタにワークエリアの先頭番地をいれ、(ワークエリアは必ず６４バイト用意します）ＦＦＣＡＨ番地をコールします。このとき、スタックポインタは必ずページ３に置いてください。また、ＨＬレジスタで指定されるワークエリアは、必ずページ２か３に置いてください。

もしＲＳ－２３２Ｃインターフェイスが存在すれば、ＨＬレジスタの値が４だけインクルメントされ、ワークエリアに図１のようなＲＳ－２３２Ｃのスロットとジャンプテーブルの先頭アドレスが返ってきます。2チャンネルのＲＳ－２３２Ｃを持っている場合には、ＨＬレジスタは８だけインクルメントされ、図１のテーブルと同じものが２つできます。そして、最初のものがチャンネル０、後のものがチャンネル１になります。プログラム上は最大、チャンネル３までサポートできるようです。(ただし、古い型のＲＳ－２３２Ｃカートリッジには、２個以上接続すると動作しないものもあります）なお、ＲＳ－２３２Ｃが存在しなければ、ＨＬレジスタの値が変化せずに戻ってきます。

ジャンプテーブルの内容は図２のようになっています。プログラムがＲＳ－２３２Ｃを使うときは、このジャンプテーブルを直接インタースロットコールするわけです。これらの機能については、次の章で詳しく述べます。

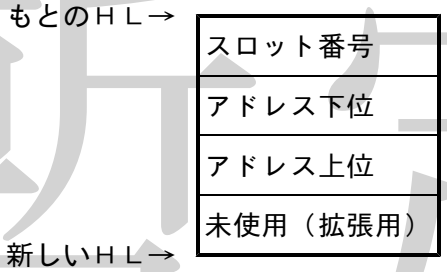


図１・ワークエリアの内容

```
DB DVINFB
DB DVTYPE
DB 0
JP INIT
JP OPEN
JP STAT
JP GETCHR
JP SNDCHR
JP CLOSE
JP EOF
JP LOC
JP LOF
JP BACKUP
JP SNDBRK
JP DTR
JP SETCHN
JP NCUSTA    *   (*印はモデムのみが
JP SPKCNT    *    使用する)
JP LINSEL    *
JP DIALST    *
JP DIALCH    *
JP DTMFST    *
JP RDDTMF    *
JP HOKCNT    *
JP CONFIG    *
JP SPCIAL    *
  (NOENT)
  (NOENT)
```

図２・ジャンプテーブルの内容

第３章　ＲＳ－２３２Ｃジャンプテーブルの使用方法

ここでは、第２章で紹介したジャンプテーブルのそれぞれの機能について説明します。

●ＤＶＩＮＦＢ　（オプション機能の有無）

これはデバイスインフォメーションバイトと呼ばれ、ＲＳ－２３２Ｃのオプション機能の有無についての情報が入っています。これはデータが入っているだけで、ジャンプはする必要はありません。

ＢＩＴ５はＲＩ信号検出、ＢＩＴ４はＣＤ信号検出、ＢＩＴ３はタイマー割り込み、ＢＩＴ２はシンク／ブレーク信号検出、ＢＩＴ１は割り込みを使う送信の、それぞれの機能の有無についての情報が入っています。各ビットが立っていれば、その機能が存在します。これ以外のビットは拡張用で、現在は使用されていません。

●ＤＶＴＹＰＥ　（複数チャンネル対応かどうか）

ここが０でなければ複数チャンネル対応です。あまり使うことはないと思います。

●ＩＮＩＴ　　（ＲＳ－２３２Ｃのイニシャライズ）

入力　　Ｂ，パラメータテーブルのスロットアドレス
　　ＨＬ，パラメータテーブルの先頭アドレス
出力　　Ｃｙ，パラメータに異常があって、イニシャライズできなかったときセット
レジスタ　ＡＦ

ＲＳ－２３２Ｃをイニシャライズします。パラメータテーブルの内容は以下のとおりです。（ＢＡＳＩＣのCALL COMINI命令とほとんど同じ形式です）

ＨＬの値→

キャラクター長	”5”～”8”
パリティ	”E”or”O”or”I”or”N”
ストップビット	”1”or”2”or”3”
ＸＯＮ／ＸＯＦＦ	”X”or”N”
ＲＳ／ＣＳ制御	”H”or”N”
受信時ＬＦ挿入	”A”or”N”
送信時ＬＦ削除	”A”or”N”
ＳＩ／ＳＯ	”S”or”N”
受信ボーレート	（下位）
	（上位）
送信ボーレート	（下位）
	（上位）
ﾀｲﾑｱｳﾄｶｳﾝﾄ	0～255

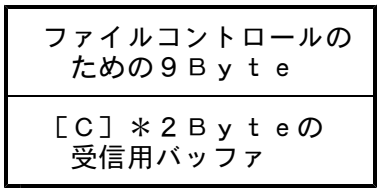
受信ボーレート、送信ボーレート、タイムアウトカウントはバイナリ形式、それ以外のデータはアスキー形式で代入します。

●ＯＰＥＮ　　（ＲＳ－２３２Ｃをオープン）

入力　　ＨＬ，ＦＣＢのアドレス
　　Ｃ，バッファ長　範囲はＲＳ－２３２Ｃの場合３２～１２７
　　モデムカートリッジ（含内蔵）の場合３２～２５４
　　Ｅ，ＩＮＰＵＴモードなら１
　　ＯＵＴＰＵＴモードなら２
　　ＩＮＰＵＴ／ＯＵＴＰＵＴモードなら４
出力　　Ｃｙ，パラメータが異常で設定を行なえなかったときセット
レジスタ　ＡＦ

なお、ＦＣＢの内容については次のとおりです。

ＨＬで指定されたアドレス→



●ＳＴＡＴ　　（ステータス情報を得る）
　入力　　なし
　出力　　ＨＬ，ステータスデータ
　レジスタ　なし
　ＲＳ－２３２Ｃポートのステータスを読み込みます。データは１６ビットで、各ビットの意味は拡張ＢＡＳＩＣのＣＡＬＬ　ＣＯＭＳＴＡＴ命令と同じです。

●ＧＥＴＣＨＲ　（１文字受信）
　入力　　なし
　出力　　Ａ，受信されたキャラクター
　　Ｓ，エラー発生時にセット
　　Ｃｙ，ＩＮＰＵＴモードでＥＯＦが受信されたときセット
　レジスタ　Ｆ
　受信用バッファから１文字を取ってきます。ＢＡＣＫＵＰでバックアップされた文字があれば、それを返します。必ず、ＬＯＣを使用して受信バッファに文字があるかどうかを調べてから呼んでください。文字がない場合、動作は保証されません。

●ＳＮＤＣＨＲ　（１文字送信）
　入力　　Ａ，送信キャラクター
　出力　　Ｃｙ，ＣＴＲＬ＋ＳＴＯＰが押されたときセット
　　Ｚ，ＸＯＮまたはＣＳ信号がＯＮになっていなかったときにタイムアウトエラーが発生したときセット
　レジスタ　Ｆ
　ＲＳ－２３２Ｃポートに１文字出力します。タイムアウトエラーが発生したとき、ＣＴＲＬ＋ＳＴＯＰが押されたときは出力は行なわれません。

●ＣＬＯＳＥ　（ＲＳ－２３２Ｃのクローズ）
　入力　　なし
　出力　　ＯＵＴＰＵＴモードでオープンされていたとき
　　Ｃｙ，ＸＯＦＦを受信していたとき、ＣＳがＯＮになっていなかったときにＣＴＲＬ＋ＳＴＯＰが押されるとセット
　　Ｚ，ＸＯＦＦを受信していたとき、ＣＳがＯＮになっていなかったときにタイムアウトエラーが発生したときセット
　　ＩＮＰＵＴモードでオープンされていたとき
　　なし
　レジスタ　ＡＦ
　ＲＳ－２３２Ｃポートをクローズして、ＲＳ信号をインアクティブにします。ＯＵＴＰＵＴモードでオープンされていたときにはＥＯＦを出力します。

●ＥＯＦ　　（ＥＯＦ検出）
　入力　　なし
　出力　　ＨＬ，－１なら次のキャラクターはＥＯＦ
　　０ならＥＯＦでない
　　Ｃｙ，セットなら次のキャラクターはＥＯＦ
　　リセットならＥＯＦでない
　レジスタ　ＡＦ
　受信バッファから次に取るキャラクターが、ＥＯＦかどうかを調べます。結果はＨＬとＣｙの両方で返されます。

●ＬＯＣ　　（受信バッファ内のキャラクター数を得る）
　入力　　なし
　出力　　ＨＬ，キャラクター数
　レジスタ　ＡＦ
　ＲＳ－２３２Ｃの受信用バッファ内にある受信されたキャラクター数を調べます。これにはバックアップされたキャラクターも含まれます。ＩＮＰＵＴモードでオープンされているときにはＥＯＦより後ろのキャラクターは数えません。

●ＬＯＦ　　（受信バッファの空き領域の大きさを得る）
　入力　　なし
　出力　　ＨＬ，空き領域の大きさ
　レジスタ　ＡＦ
　受信バッファの空き領域の大きさを調べます。実際の空き領域なので、ＥＯＦの後ろのキャラクターは空き領域になりません。

●ＢＡＣＫＵＰ　（キャラクターをバックアップしておく）
　入力　　Ｃ，バックアップするキャラクター
　出力　　なし
　レジスタ　　ＡＦ
　特殊バッファに文字をバックアップしておきます。以前にバックアップした文字は失われます。

●ＳＮＤＢＲＫ　（ブレークキャラクターの出力）
　入力　　ＤＥ，送信する回数
　出力　　Ｃｙ，ＣＴＲＬ＋ＳＴＯＰが押されたときセット
　レジスタ　ＡＦ，ＤＥ
　ブレークキャラクターをＤＥで指定された回数だけ送信します。途中でＣＴＲＬ＋ＳＴＯＰが押されたときはＣｙをセットして戻ります。

●ＤＴＲ　　（ＤＴＲ信号の状態を変える）
　入力　　Ａ，０ならばＤＴＲをＯＦＦ
　　０以外ならＯＮにする
　出力　　なし
　レジスタ　　Ｆ
　ＤＴＲ信号の状態を変化させます。ＤＴＲは電源ＯＮ時、リセット時及びＩＮＩＴが実行されたときＯＮになります。

●ＳＥＴＣＨＮ　（チャンネル変更）
　入力　　Ｅ，チャンネル番号
　出力　　Ｃｙ，チャンネル番号に異常があったときセット
　レジスタ　ＡＦ，ＢＣ
　複数チャンネル対応ＲＳ－２３２Ｃのチャンネルを切り換えるためのエントリーです。当然ながら、複数対応でないＲＳ－２３２Ｃではこの機能は使えません。複数チャンネル対応ＲＳ－２３２Ｃと、そうでないＲＳ－２３２Ｃが同時に接続されている場合、チャンネル番号は複数チャンネル対応のものだけに付けられます。つまり、このエントリーでは複数対応でないＲＳ－２３２Ｃに切り換えることはできません。また、ＢＡＳＩＣのチャンネル番号とこのエントリーのチャンネル番号は対応しません。チャンネル番号はリセット時に０に設定されます。

　ＢＡＣＫＵＰというのは一旦読み込んだデータをバッファに返してまた読むという、ちょっと変わったことをしています。これは、コントロールキャラクターの制御に使われます。　以上でＲＳ－２３２Ｃのジャンプテーブルは終わりですが、モデムの場合はこのあとにダイアル機能などのためのジャンプテーブルが用意されています。これらの使い方については次の章で解説します。

第４章　モデムＢＩＯＳジャンプテーブルの使用方法

モデムＢＩＯＳのジャンプテーブルの使用方法はＲＳ－２３２Ｃの場合と全く同じです。ただし、JP NCUSTAからJP SPCIALまでの１０個のエントリーがモデム用に新たに追加されました。なお、JP DTRはモデムでは使用しません。

そのほか変更があったのはJP INITです。モデムのために、電話回線の指定などの設定が必要になります。

以下で、変更のあったエントリー、モデム専用のエントリーについて説明します。そのほかのエントリーの使用方法は第３章を参照してください。

なおこのほかに、モデムの場合はＯＰＥＮでＣレジスタによって指定されるバッファの大きさは３２～２５４バイトになります。

●ＩＮＩＴ　　（ＲＳ２３２Ｃ，モデムポートのイニシャライズ）

入力　　Ａ，モデムの種類
　　０ならBELL 103　　300bps 全二重
　　１ならBELL 212A　1200bps 全二重
　　２ならCCITT V21　　300bps 全二重
　　３ならCCITT V22　　1200bps 全二重
　　４ならCCITT V22bis 2400bps 全二重
　　５ならCCITT V23　　1200bps 半二重
　　６ならCCITT V27ter 4800bps 半二重
　　７ならCCITT V29　　9600bps 半二重
　　８ならCCITT V32　　9600bps 全二重
　　９～２５４・未定義。将来拡張用
　　２５５ならシステムディフォルト
　　Ｂ，パラメータテーブルのスロット番号
　　ＨＬ，パラメータテーブルの先頭番地（パラメーターテーブルの内容は第３章と同じ）
　　Ｃ，ダイアラのモード
　　０ならＤＴＭＦ
　　２ならパルス（20pps）
　　３ならパルス（10pps）
　　４ならオートマチック
　　１または５～２５４は未定義。将来拡張用
　　２５５ならシステムディフォルト

出力　　Ｃｙ，パラメータに異常があってイニシャライズできなかったときセット

レジスタ　ＡＦ

モデムポートをイニシャライズします。モデムの種類が指定されているときはパラメータテーブルの受信・送信ボーレートはチェックされません。モデムの指定がディフォルトの場合だけ、受信・送信ボーレートがチェックされます。この指定に異常がある場合、ディフォルトのモデムが選択されます。システムディフォルトが指定されたときは、カートリッジのスイッチやメモリスイッチ、システムであらかじめ決められているものが選択されます。

オートマチックは、まず回線がＤＴＭＦかどうか調べ、そうでなければパルス10ppsでダイアルします。

●ＮＣＵＳＴＡ　（ＮＣＵのハードウェアの状態を返す）

入力　　なし

出力　　ＨＬ，ステータス

レジスタ　ＨＬ

ＮＣＵの状態を返します。ステータスの意味は拡張ＢＡＳＩＣのCALL NETSTATと同じで、以下のようになります。

ＨＬのbit15～9は拡張用で現在はすべて０です。bit8はＤＴＭＦデータ検出で、ＤＴＭＦデータが来ていると１になります。bit7は外部電話機のHOOK検出で、０ならON HOOK,１ならOFF HOOKです。bit6はCall progressで400Hzのトーンを検出すると１になります。bit5は課金パルス検出で極性反転をラッチします。リードセットです。bit4,3は回線極性検出でbit4,3がそれぞれ0,0ならLoop off, 0,1ならDC loop(LB), 1,0ならDC loop(LA), 1,1は未定義です。bit2,1はダイアラモードでbit2,1がそれぞれ0,0ならＤＴＭＦ、 0,1ならパルス（10pps)、 1,0ならパルス（20pps)、 1,1ならオートマチックです。bit0はリング信号ラッチで、リング信号がある場合１になります。

サポートされていない機能は常に０を返します。

●ＳＰＫＣＮＴ　（スピーカーをＯＮ／ＯＦＦする）
入力　　Ａ，０ならスピーカーＯＦＦ
　　０以外ならスピーカーＯＮ
出力　　Ｃｙ，この機能がサポートされていないときセット
レジスタ　なし
スピーカーをＯＮ／ＯＦＦします。拡張ＢＡＳＩＣのCALL NETSPKと同等の機能です。

●ＬＩＮＳＥＬ（回線の切り換え）
入力　　Ａ，bit5～7は拡張用で現在は０です。bit4,3が立っていれば内蔵ハンズフリーフォン（bit4・スピーカー、bit3・マイク）を回線に接続します。bit2,1が立っていれば内蔵ハンドセット（bit2・スピーカー、bit1・マイク）を回線に接続します。bit0が０なら内蔵モデムに、１なら外部電話機に回線を接続します。
出力　　Ｃｙ，パラメータにエラーがある場合セット
レジスタ　なし
回線の接続を切り換えます。拡張ＢＡＳＩＣのCALL LINESELと同等の機能です。

●ＤＩＡＬＳＴ　（電話番号の出力）
入力　　Ｂ，ダイアルデータのスロット番号
　　ＨＬ，ダイアルデータの先頭番地
　　Ｃ，ダイアラのモード。ＩＮＩＴのＣレジスタと同じ
出力　　Ｃｙ，パラメータにエラーがある場合セット
レジスタ　なし
電話回線をダイアラに接続し、ダイアルデータを出力します。終了後は回線はモデムに接続されたままになります。ダイアルデータは"0"～"9", "A"～"D", "H", "#", "*", "<", ":", "T"が有効です。"H"は１秒間ON HOOK, "<"は３秒間のポーズ、"T"はトーンダイアルへの切り換えです。":"は第２ダイアルトーンを待ちますが、この機能のない場合、"<"と同じになります。サポートされていないデータが指定された場合エラーになります。また、上にあげたデータ以外は無視されます。データ列の最後は00Hで終了してください。
このファンクションは拡張ＢＡＳＩＣのCALL DIALと同等の機能です。

●ＤＩＡＬＣＨ　（１桁だけの電話番号出力）
入力　　Ａ，ダイアル文字
　　Ｃ，ダイアラのモード。ＩＮＩＴのＣレジスタと同じ
出力　　Ｃｙ，パラメータにエラーがある場合セット
レジスタ　なし
１桁だけダイアル信号を出力します。データはＤＩＡＬＳＴと同じです。これは、拡張ＢＡＳＩＣのCALL DIALCと同等の機能です。

●ＤＴＭＦＳＴ　（ＤＴＭＦデコーダのステータスを調べる）
入力　　なし
出力　　Ｚ，ＤＴＭＦコードが入力されていればセット
　　Ｃｙ，この機能がサポートされていなければセット
レジスタ　ＡＦ
ＤＴＭＦデコーダのステータスを調べます

●ＲＤＤＴＭＦ　（ＤＴＭＦデコーダからデータを読み込む）
入力　　なし
出力　　Ａ，ＤＴＭＦコード（アスキーコード）
　　Ｃｙ，ＣＴＲＬ＋ＳＴＯＰが押されたとき、または、この機能がサポートされていないときセット
レジスタ　ＡＦ
ＤＴＭＦデコーダからデータを読み込みます。データがない場合は新しい信号が来るまで待ちます。拡張ＢＡＳＩＣのCALL DTMFと似た機能です

●ＨＯＫＣＮＴ　（回線の切断・接続を行なう）
入力　　Ａ，０ならON HOOK
　　１ならOFF HOOK

出力　　Ｃｙ，この機能がサポートされていないときセット
レジスタ　なし
回線の切断・接続を行ないます．拡張ＢＡＳＩＣのCALL NETHOOKと同等の機能です

●ＣＯＮＦＩＧ　（ハードウェアの仕様を返す）
入力　　Ａ，０のとき
出力　　ＨＬにモデム機能のステータス．　bit15～9　拡張用。現在は０
bit8:CCITT V32　9600bps 全二重，bit7:CCITT V29　9600bps 半二重
bit6:CCITT V27ter 4800bps 半二重，bit5:CCITT V23　1200bps 半二重
bit4:CCITT V22bis 2400bps 全二重，bit3:CCITT V22　1200bps 全二重
bit2:CCITT V21　300bps 全二重，bit1:BELL　212A　1200bps 全二重
bit0:BELL　103　300bps 全二重
入力　　Ａ，１のとき
出力　　ＨＬにダイアラー機能のステータス．　bit15～8拡張用。現在は０
bit7:10pps-20ppsのソフト切り換え　bit6:DTMF-ﾊﾟﾙｽのソフト切り換え
bit5:"H"のサポート　bit4:"A"～"D"のサポート
bit3:オートマチック　bit2:パルス（20pps）
bit1:パルス（10pps）　bit0:ＤＴＭＦ
入力　　Ａ，２のとき
出力　　ＨＬにライン機能のステータス、　bit15～4　拡張用。現在は０
bit3:内蔵ハンズフリーフォン　bit2:内蔵ハンドセット
bit1:内蔵モデム　bit0:外部電話機
入力　　Ａ，３のとき
出力　　ＨＬにその他の機能のステータス、　bit15～13　拡張用。現在は０
bit12:Long loop検出機能　bit11:キャリア制御機能
bit10:送出電力切り換え機能　bit 9:ＲＳ－２３２Ｃ
bit 8:ＭＳＸ標準カートリッジ　bit 7:外部電話機ON/OFF HOOK検出
bit 6:ON/OFF HOOK機能　bit 5:スピーカー
bit 4:ＤＴＭＦデコーダ　bit 3:課金パルス検出
bit 2:回線極性検出　bit 1:Call progress 検出
bit 0:リング信号検出
入力　　Ａ，４～２５５のとき
出力　　ＨＬ，0000H
レジスタ　ＨＬ
ハードウェアの仕様を返します。機能のある場合はそのビットが立ちます。これは拡張ＢＡＳＩＣのCALL NETCONFIGと同等の機能なのでそちらも参考にしてください。ＲＳ－２３２Ｃは、モデムとしてもＲＳ－２３２Ｃとしても使用できるようなカートリッジの場合１になります。

●ＳＰＣＩＡＬ　（メーカー別、機種別の特別処理を行なう）
入力　　Ａ，０：モデムの送出電力切り換え機能
Ｃ，－送出電力値（ｄＢｍ）
２５５ならディフォルト
出力　　Ｃｙ，この機能がサポートされていなかったときセット
入力　　Ａ，１：キャリア制御
Ｃ，０ならキャリアＯＦＦ
１ならキャリアＯＮ
Ｈ，RS ONまでのディレタイム（ｎ＊１０ｍＳ）
Ｌ，CS ONからRETURNまでのディレタイム（ｎ＊１０ｍＳ）
出力　　Ｃｙ，この機能がサポートされていなかったときセット
入力　　Ａ，２：イコライザの設定
Ｃ？，０ならイコライザを使用しない
１ならイコライザを使用する
２なら自動的にイコライザを調節する
２５５ならディフォルト
出力　　Ｃｙ，この機能がサポートされていなければセット
レジスタ　不明
メーカー別、機種別の特別処理を行ないます。

第５章　ＲＳ－２３２Ｃ・モデム使用上の注意

有名ですが、ＭＳＸのＲＳ－２３２Ｃカートリッジの内蔵プログラムにはバグがあって正常に動作しません。モデムカートリッジではバグが直してあるので問題ないのですが、ＲＳ－２３２Ｃを使ったソフトでは、自分でＲＳ－２３２Ｃのバグを直してから使わなくてはなりません。また、このことを知らない人が、モデム対応のソフトを作ると、そのソフトはモデムを使えば正常に動作するのに、ＲＳ－２３２Ｃにモデムをつないで使おうとすると、暴走してしまうことになります。そこで、モデム、ＲＳ－２３２Ｃの両方に対応するように、必ず次のパッチプログラムをプログラム内に組み込んでください。なお、機種によってはこのバグが直っているＲＳ－２３２Ｃもあります。

その１．

```
ＰＵＳＨ　ＢＣ
ＣＡＬＬ　ＦＦＣＡＨ
ＰＯＰ　　ＢＣ
ＲＥＴ
```

上の６バイトのプログラムをページ３に置き、その先頭番地をＦＦＤＥＨ番地に書き込んでください。ＲＳ－２３２Ｃを使うプログラムが終了するときには、ＦＦＤＥＨ番地から２バイトを元に戻しておいてください。

これで、ディスクアクセスのときにＲＳ－２３２ＣがＸＯＦＦを送信する機能が異常を起こすバグが回避できます。

その２．　ＤＯＳ上でＲＳ－２３２Ｃを使う場合

```
ＪＰ　００１ＣＨ
```

上の３バイトを必ず０２１７Ｈ番地から３バイトに置いてください。したがって、ＲＳ－２３２Ｃを使うときは０２１７Ｈ番地から３バイトは使うことができません。これで、ＲＳ－２３２Ｃが、インタースロットコールルーチンが０２１７Ｈ番地から始まると思い込んでいるために、ＤＯＳ上からＲＳ－２３２Ｃを使うときに暴走するのを回避することができます。

このほかに、バグではありませんが、プログラムが長時間割り込みを禁止すると、受信時にデータを取りこぼすことがあるので、そのようなときには第２部第７章を参考にして取りこぼしを防いでください。

ＲＳ－２３２Ｃ、モデムを接続していると、ＭＳＸの動作速度が若干低下します。使用しないときはＲＳ－２３２Ｃ、モデムカートリッジは外しておいたほうがいいでしょう。

実際にＲＳ－２３２Ｃを使ったプログラムを作ろうとしている人は、ＭＳＸマガジン１９８８年３月号から５月号までにＲＳ－２３２Ｃを使ったプログラムの作り方が連載されているので、そちらを参考にしてください。ここでは、紙面の関係から実際の使い方については触れません。

なお、モデムとＲＳ－２３２Ｃを識別する方法は以下のとおりです。

●モデム､またはＲＳ－２３２Ｃのジャンプテーブルのスタートアドレス＋２ＡＨ番地を、Ｈレジスタに０ＦＦＨを入れて、インタースロットコールする
戻ってきたＨがＦＦＨならＲＳ－２３２Ｃ、それ以外ならモデム

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆

昔、ＢＡＳＩＣで動いていたソフトの中に、終了する直前にキーバッファに直接値を書き込んで、ＥＮＤで実行を終了させるとすぐにキーバッファに書いてあった内容を実行するという結構変なことをやっていたソフトがありました。

ＭＳＸのＲＳ－２３２Ｃは、最大４チャンネルまで同時に接続できるような仕様になっています。しかし、ＲＳ－２３２Ｃの種類によっては、２つ以上同時に接続すると動作しないものもあります。特に、古い型のものに多いようです。

ＭＳＸはＣＰＵがＺ８０Ａで、さらにウェイトが非常に多いので、ＲＳ－２３２Ｃ用のソフトは、ＢＡＳＩＣでは１２００ボー、機械語では４８００ボー程度が限界とされています。アスキーのＳＥＲＩＡＬ２３２は内部にバッファを持っていて、９６００ボーまでは問題なく動作するのですが、ソフトによってはこのモードでは動作しなかったりするのがちょっと困りものです。もちろん、バッファを使わないモードもあります。

第５部　　ＭＳＸ－ＪＥ

◆

ここでは、「ＭＳＸ標準日本語入力フロントエンドプロセッサ」、通称「ＭＳＸ－ＪＥ」の説明をします。

なお、これらの資料を作成するに当たっては、「ＶＪＥ－８０Ａ仕様書・第２版」（アスキー社）を参考にさせていただきました。

第１章　ＭＳＸ－ＪＥとは何か

日本語入力フロントエンドプロセッサというのは、ワープロなどの日本語入力の仕方や使い方やそれらをコントロールしているプログラムのことです。ＭＳＸ－ＪＥが規格になる前は、ワープロソフトがそれぞれ日本語入力フロントエンドプロセッサを持っていました。昔はそれで良かったのですが、この方法だと、通信ソフトが日本語入力をサポートしようとしたようなとき、自分自身で日本語入力フロントエンドプロセッサを作らなくてはなりません。しかし、はっきりいってこれはかなり大変な作業です。さらに、ユーザーの方もソフトが変わるたびに違う日本語フロントエンドプロセッサの使い方を覚えなければなりません。通信ソフトの日本語入力の時に、いつも使っているワープロの時と同じ入力方法が使えたほうが便利なのは当然です。そこで、日本語入力の仕方とそれらの機能をプログラムから呼びだす方法について規格ができました。これが「ＭＳＸ標準日本語入力フロントエンドプロセッサ」、「ＭＳＸ－ＪＥ」（略称ＭＪＥ）です。

例えば、ソニーのＨＢ－Ｆ１ＸＤＪでワープロソフト「文書作左衛門」を使ったときに内蔵Ｓ－ＲＡＭ辞書に登録した単語は、漢字ＢＡＳＩＣ上から漢字入力を使ったときにも有効です。キヤノンの通信カートリッジＶＭ－３００をつないで通信をするときも同じように使えます。もちろん、漢字変換の方法なども統一されています。

説明が非常に長くなりましたが、第２章以降でＭＳＸ－ＪＥをソフトウェア側から使う方法について説明していきます。

第２章　ＭＳＸ－ＪＥの起動方法

ＭＳＸ－ＪＥを使うには、例によって拡張ＢＩＯＳコールを利用します。

まず、拡張ＢＩＯＳコールが可能かどうかを調べなくてはなりません。ワークエリアのＦＢ２０Ｈ番地の最下位ビットが立っていて、ＦＦＣＡＨ番地の値がＣ９Ｈ以外なら拡張ＢＩＯＳコールが可能です。それ以外の場合には拡張ＢＩＯＳコールを利用する機器は接続されていないということなので、当然、ＭＳＸ－ＪＥも存在しません。

前に挙げた条件が満足されているときは、Ｄレジスタに１０Ｈ、Ｅレジスタに０、ＨＬレジスタにワークエリアの先頭番地、Ｂレジスタにワークエリアのあるスロット番号をいれ、ＦＦＣＡＨ番地をコールしてください。なお、ワークエリアは必ずページ２か３に置き、スタックは必ずページ３に置いてください。ワークエリアの大きさは６４バイトです。

もし日本語フロントエンドプロセッサが存在すれば、ＨＬの値が４だけインクルメントされ、次のようなテーブルができます。２個以上の日本語フロントエンドプロセッサが接続されているときには、このテーブルがその数だけできます。

もとのＨＬ→

ｹｰﾊﾞﾋﾞﾘﾃｨｰﾍﾞｸﾄﾙ
スロット番号
アドレス下位
アドレス上位

スロット番号はＭＳＸ－ＪＥの存在するスロット番号、アドレスはソフトウェアがＭＳＸ－ＪＥを利用するときにコールするアドレスです。ＭＳＸ－ＪＥを利用するときには、この番地をインタースロットコールするわけです。

ケーバビリティーベクトルというのは、日本語入力フロントエンドプロセッサのオプション機能の有無などが入っています。それぞれのビットが立っているか立っていないかでそれらの機能の有無を調べます。

ビット０が０なら、その日本語入力フロントエンドプロセッサはＭＳＸ－ＪＥ互換、１なら非互換を意味します。
ビット１が０ならば、仮想端末インターフェイスあり、１ならなしです。
ビット２が０なら辞書インターフェイスあり、１ならなしです。
ビット３が０なら辞書登録及び削除機能あり、１ならなしです。
ビット４から７までは拡張用で、現在はすべて１になっています。
ＭＳＸ－ＪＥを使用するときには、これらのビットの状態を調べて、利用したい機能があるかどうかチェックしてください。例えば、VJE-80（アスキーのＭＳＸ－ＪＥ）のケーバビリティーベクトルはＦ８Ｈ，VJE-80A（アスキーのＭＳＸ－ＪＥのニューバージョン）はＦ０Ｈです。
ソフトウェアは、ＭＳＸ－ＪＥが二つ以上ある場合にはメニューを出してユーザーにどちらのＭＳＸ－ＪＥを使うか選ばせるようにしてください。機種によっては、本体内蔵ＭＳＸ－ＪＥが必ず先に立ち上がってしまい、外付けのＭＳＸ－ＪＥを利用できなくなってしまうことがあります。例えば、松下ＦＳ－４６００ＦにＭＳＸ－ＷＲＩＴＥを差すと、必ず内蔵ＭＳＸ－ＪＥが起動してしまいます。

第３章　ＭＳＸ－ＪＥの利用方法

ＭＳＸ－ＪＥを利用するときには、レジスタに値をいれて、先ほど調べたルーチンをインタースロットコールします。
ただ、文字列の代入の仕方がちょっと変わっているので先に説明します。

５．３．１　文字列のフォーマット

文字列のフォーマットは
［文字数］＋”［文字列］”という形になります。文字列はＳＨＩＦＴ－ＪＩＳ形式です。例えば、「　０７Ｈ＋”ｱｶｲﾌｳｾﾝ”　」といった具合です。そして、その先頭番地をＢＣまたはＤＥレジスタで指定します。全角文字は２バイトの扱いです。以下にいくつか例を挙げます。
「　０８Ｈ＋”赤い風船”　」
「　０ＤＨ＋”納豆ｽﾊﾟｹﾞｯﾃｨｰ”　」
「　０７Ｈ＋”ｸﾗﾌﾞﾊｳｽ”　」
「　０ＥＨ＋”あかいふうせん”　」
「　０７Ｈ＋”TOUKYOU”　」
半角の濁点、半濁点は１文字の扱いです。半角平仮名及び「ヰ」、「ヱ」、「ヴ」、「ヵ」、「ヶ」などの文字は半角では使用できません。文字列の長さは最大６４バイトです。

５．３．２　ＭＳＸ－ＪＥのファンクションコール

●Ｉｎｑｕｉｔｙ　（Ｆｕｎｃｔｉｏｎ０１Ｈ）
入力・Ａに１
出力・ＨＬにＭＡＸ
　　ＤＥにＭＩＮ
　　ＢＣにＭＩＮ２
ＭＳＸ－ＪＥが動作するのに必要なワークエリアの大きさを返します。ＭＡＸはワークエリアの最大値でこれ以上の大きさが与えられてもＭＳＸ－ＪＥはこれだけしか使いません。ＤＥには最低限必要な大きさが、ＢＣには学習機能を利用する場合に最低限必要な大きさが入ります。値はバイト数です。例として挙げると、VJE-80の場合はＭＡＸ＝ＭＩＮ２＝３３２８バイト、ＭＩＮ＝２３０４です。VJE-80A の場合はＭＡＸ＝ＭＩＮ２＝５２０，ＭＩＮ＝５１２、SONY JFEPはＭＡＸ＝ＭＩＮ＝ＭＩＮ２＝２５６０です。

●Ｉｎｖｏｋｅ　（０２Ｈ）
入力・Ａに２
　　ＨＬにＢＷの先頭番地
　　ＤＥにＢＷの長さ
出力・なし

ＭＳＸ－ＪＥのワークエリアを確保します。起動するときに必ず呼んでください。ＢＷとはＭＳＸ－ＪＥのワークエリアのことです。必ずページ２か３に置いてください。ページ２に置いたときには、アスキーのVJE-80を使ったときに多少の制限ができるようですが、余り気にしなくてもいいでしょう。必ずファンクション１で必要なワークエリアの長さを調べてからコールしてください。ＭＳＸ－ＪＥが起動するのに最低限必要なワークエリアは２５６０バイトを越えることはないので、最低それだけ用意すれば大丈夫です。

●Ｒｅｌｅａｓｅ　（０３Ｈ）
入力・Ａに３
　　ＨＬにＢＷの先頭アドレス
出力・なし
ＭＳＸ－ＪＥ用のワークエリアを開放します。ＭＳＸ－ＪＥを使ったプログラムが終了するときに、必ず呼んでください。そのままにすると学習辞書が破壊される場合があります。

●ｈａｎ＿ｚｅｎ　（４０Ｈ）
入力・Ａに４０Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥにｿｰｽﾊﾞｯﾌｧ（半角文字列）のアドレス
　　ＢＣにﾃﾞｽﾃｨﾈｰｼｮﾝ（全角文字列）のアドレス
出力・Ａ：０なら成功
　　０以外なら失敗、変換できなかった文字のバッファ上での位置
バッファの半角文字列を全角文字列に変換します。濁点、半濁点も１文字として変換します。
例、入力にＤＥ→０５Ｈ＋”ｲﾊﾞﾗｷ”
　　出力　ＢＣ→０ＡＨ＋”イハ゛ラキ”

●ｚｅｎ＿ｈａｎ　（４１Ｈ）
入力・Ａに４１Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥにｿｰｽﾊﾞｯﾌｧ（全角文字列）のアドレス
　　ＢＣにﾃﾞｽﾃｨﾈｰｼｮﾝ（半角文字列）のアドレス
出力・Ａ：０ならば成功
　　０以外なら失敗、変換できなかった文字のバッファ上での位置
バッファの全角文字列を半角文字列に変換します。平仮名、片仮名の濁点、半濁点は１文字となります。
例、入力にＤＥ→０８Ｈ＋”あけぼの”
　　出力　ＢＣ→０５Ｈ＋”ｱｹﾎﾞﾉ”

●ｈａｎ＿ｋａｔａ　（４２Ｈ）
入力・Ａに４２Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥにｿｰｽﾊﾞｯﾌｧ（半角文字列）のアドレス
　　ＢＣにﾃﾞｽﾃｨﾈｰｼｮﾝ（ｶﾀｶﾅ文字列）のアドレス
出力・Ａ：０ならば成功
　　０以外なら失敗、変換できなかった文字のバッファ上での位置
半角文字列を全角片仮名文字列に変換します。入力は片仮名、またはローマ字の文字列です。
例、入力にＤＥ→０７Ｈ＋”AKATUKA”
　　出力　ＢＣ→０８Ｈ＋”アカツカ”

●ｈａｎ＿ｈｉｒａ　（４３Ｈ）
入力・Ａに４３Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥにｿｰｽﾊﾞｯﾌｧ（半角文字列）のアドレス
　　ＢＣにﾃﾞｨｽﾃｨﾈｰｼｮﾝ（平仮名文字列）のアドレス
出力・Ａ：０ならば成功
　　０以外なら失敗、変換できなかった文字のバッファ上での位置
半角文字列を全角平仮名文字列に変換します。入力は片仮名、またはローマ字の文字列です。

例、入力にＤＥ→０ＡＨ＋”KAIRAKUENN”
　　出力　ＢＣ→０ＣＨ＋”かいらくえん”

●ｋａｔａ＿ｈｉｒａ　（４４Ｈ）
入力・Ａに４４Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥにｿｰｽﾊﾞｯﾌｧ（片仮名文字列）のアドレス
　　ＢＣにﾃﾞｽﾃｨﾈｰｼｮﾝ（平仮名文字列）のアドレス
出力・なし
全角片仮名文字列を全角平仮名文字列に変換します。入力に全角片仮名以外の文字があったときは、その文字は変換しません。入力に半角文字があったときは、変換結果は保証されません。
例、入力にＤＥ→０ＡＨ＋”ダイクマチ”
　　出力　ＢＣ→０ＡＨ＋”だいくまち”

●ｈｉｒａ＿ｋａｔａ　（４５Ｈ）
入力・Ａに４５Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥにｿｰｽﾊﾞｯﾌｧ（片仮名文字列）のアドレス
　　ＢＣにﾃﾞｽﾃｨﾈｰｼｮﾝ（片仮名文字列）のアドレス
全角平仮名文字列を全角片仮名文字列に変換します。それ以外はｋａｔａ＿ｈｉｒａと同じです。

●ｏｐｅｎ＿ｄｉｃ　（４６Ｈ）
ユーザーが呼ぶ必要はありません。

●ｈｅｎｋａｎ１　（４７Ｈ）
入力・Ａに４７Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥに読みデータ（全角片仮名文字列）のアドレス
出力・Ａ：０なら候補なし
　　０以外なら先頭文節の候補数
全角片仮名文字列を漢字仮名交じり文に変換します。変換結果はｊｉ＿ｋｏｈｏなどで取りだします。
例、入力にＤＥ→１０Ｈ＋”カワノナカ”
　　出力　Ａに「カワノ」の候補数

●ｊｉ＿ｋｏｈｏ　（４８Ｈ）
入力・Ａに４８Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥに候補バッファ
　　ＢＣに後続文節バッファ
出力・Ａ：０なら候補なし
　　０以外なら候補ブロック内通し番号
先頭文節の次の候補を取り出します。このファンクションを呼ぶ前に、ｈｅｎｋａｎ１で読みデータをセットしておく必要があります。
例、入力　「カワノナカノサカナ」をｈｅｎｋａｎ１で変換した後このファンクションを呼んだものとします。
　　出力・Ａに候補ブロック内通し番号
　　ＤＥ→０４Ｈ＋”川の”
　　ＢＣ→０６Ｈ＋”中の魚”

●ｚｅｎ＿ｋｏｈｏ　（４９Ｈ）
入力・Ａに４９Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥに候補バッファ
　　ＢＣに後続文節バッファ
出力・Ａ：０なら候補なし
　　０以外なら候補ブロック内通し番号
先頭文節の１つ前の候補を取りだします。このファンクションを呼ぶ前に、ｈｅｎｋａ

ｎ１で読みデータをセットしておく必要があります。
例、入力　「カワノナカ」をｈｅｎｋａｎ１で変換しておいたとします。

出力の例　最初のｊｉ＿ｋｏｈｏの結果
ＤＥ→０４Ｈ＋”川の”
ＢＣ→０２Ｈ＋”中”
２回目のｊｉ＿ｋｏｈｏの結果
ＤＥ→０４Ｈ＋”川野”
ＢＣ→０２Ｈ＋”中”
ｚｅｎ＿ｋｏｈｏの結果
Ａに候補ブロック内通し番号
ＤＥ→０４Ｈ＋”川の”
ＢＣ→０２Ｈ＋”中”

●ｊｉ＿ｂｌｏｃｋ　（４ＡＨ）
入力・Ａに４ＡＨ
ＨＬにＢＷのアドレス
出力・Ａ：０ならブロックなし
０以外なら候補の数
先頭文節の長さを短くしたり品詞レベルを変えたりして候補群ブロックを次のレベルに移動し、候補群を作成します。このファンクションを呼ぶ前に、ｈｅｎｋａｎ１で読みデータをセットしておく必要があります。結果はｊｉ＿ｋｏｈｏなどで取りだします。
例、入力　「ヨミカキソロバン」をｈｅｎｋａｎ１で変換しておいたとします。
出力の例　最初のｊｉ＿ｋｏｈｏの結果
ＤＥ→０６Ｈ＋”読みか”
ＢＣ→０８Ｈ＋”基礎路盤”
ｊｉ＿ｂｌｏｃｋを行なった後のｊｉ＿ｋｏｈｏの結果
ＤＥ→０４Ｈ＋”読み”
ＢＣ→０８Ｈ＋”書き算盤”

●ｚｅｎ＿ｂｌｏｃｋ　（４ＢＨ）
入力・Ａに４ＢＨ
ＨＬにＢＷのアドレス
出力・Ａ：０ならブロックなし
０以外なら候補の数
先頭文節の候補ブロックを一番前のレベルに戻して、候補群を作成します。このファンクションを呼ぶ前に、ｈｅｎｋａｎ１で読みデータをセットしておく必要があります。ｊｉ＿ｂｌｏｃｋを呼んでおく必要は必ずしもありません。
例、入力　「ヨミカキソロバン」をｈｅｎｋａｎ１で変換しておいたとします。
出力　最初のｊｉ＿ｋｏｈｏの結果
ＤＥ→０６Ｈ＋”読みか”
ＢＣ→０８Ｈ＋”基礎路盤”
ｊｉ＿ｂｌｏｃｋを行なった後のｊｉ＿ｋｏｈｏの結果
ＤＥ→０４Ｈ＋”読み”
ＢＣ→０８Ｈ＋”書き算盤”
その後ｚｅｎ＿ｂｌｏｃｋを行なった後のｊｉ＿ｋｏｈｏの結果
ＤＥ→０６Ｈ＋”読みか”
ＢＣ→０８Ｈ＋”基礎路盤”

●ｋａｋｕｔｅｉ１　（４ＣＨ）
入力・Ａに４ＣＨ
ＨＬにＢＷのアドレス
Ｅに候補ブロック内通し番号
ＢＣに変換結果格納バッファアドレス
出力・ＢＣで指定されたバッファに変換結果が格納される
Ｅレジスタで指定された候補で変換結果を確定します。結果はＢＣで指定されたアドレスに入ります。このファンクションを呼ぶ前に、ｈｅｎｋａｎ１で読みデータをセットしておく必要があります。ｊｉ＿ｋｏｈｏなどは必ずしも必要ではありません。
例、「ナットウカレー」をｈｅｎｋａｎ１で変換しておいたとします。

入力・Ｅに候補ブロック内通し番号
出力・ＢＣ→０ＡＨ＋”納豆カレー”

●ｋａｋｕｔｅｉ２　（４ＤＨ）
入力・Ａに４ＤＨ
ＨＬにＢＷのアドレス
Ｅに候補ブロック内通し番号
ＢＣに変換結果格納バッファアドレス
出力・Ａに先頭文節の読みの長さ（バイト単位）
ＢＣで指定されたバッファに変換結果が収納される。
Ｅレジスタで指定された候補で先頭文節を確定します。Ａに確定した先頭文節の読みの長さが、ＢＣで指定されたアドレスに変換結果が入ります。このファンクションを呼ぶ前に、ｈｅｎｋａｎ１で読みデータをセットしておく必要があります。ｊｉ＿ｋｏｈｏなどは必ずしも必要ではありません。
例、「ナットウカレー」をｈｅｎｋａｎ１で変換しておいたとします。
入力・Ｅに候補ブロック内通し番号
出力・Ａに先頭文節の読みの長さ（０８Ｈ）
ＢＣ→０４Ｈ＋”納豆”

●ｃｌｏｓｅ＿ｄｉｃ　（４ＥＨ）
ユーザーが呼ぶ必要はありません。

●ｔｏｕｒｏｋｕ　（４ＦＨ）
入力・Ａに４ＦＨ
ＨＬにＢＷのアドレス
ＤＥに読みバッファのアドレス
ＢＣに単語登録バッファのアドレス
出力・Ａ：０ならば正常に登録された。
１なら空き領域がないため登録できなかった
２なら同音語が多すぎるため登録できなかった
４なら読みデータが不適切のため登録できなかった
８なら単語データが不適切のため登録できなかった
１６なら文法（品詞）が不適切のため登録できなかった
ユーザー辞書に単語を登録します
読みデータ、単語データの形は次のとおりです。
ＤＥ→［文字数］＋”［読みデータ（全角片仮名６４バイト以内)］”
ＢＣ→［文字数］＋”［単語データ（全角文字６４バイト以内)］”＋［品詞コード］
読みデータは全角片仮名で２バイト以上６４バイト以内（つまり、１文字から３２文字まで）で、「ヮ」「ヰ」「ヱ」「ヴ」「ヵ」「ヶ」は使用できません。単語データはＳＨＩＦＴ－ＪＩＳコードで、文字数はやはり２バイトから６４バイトまでです。同音語の登録は２５５個まで可能です。品詞は以下の５種類のうちから１個選びます。
１が名詞（納豆など)、２が人名の姓（徳川など)、３が人名の名（光圀など)、４が地名（水戸など)、５がサ変動詞（審判する、科学するなど）
単語登録をサポートしていないＭＳＸ－ＪＥ（例えばアスキーのVJE-80など）では、ＡにＦＦＨがセットされて返ってきます。
例・読みが「アリスガワ」で品詞が人名の姓の単語「有栖川」を登録した場合
入力・ＤＥ→０ＡＨ＋”アリスガワ”
ＢＣ→０６Ｈ＋”有栖川”＋０２Ｈ
出力・Ａに０（正常に登録された）

●ｓａｋｕｊｏ　（５０Ｈ）
入力・Ａに５０Ｈ
ＨＬにＢＷのアドレス
ＤＥに読みバッファのアドレス
ＢＣに登録単語バッファのアドレス
出力・Ａ：０ならば正常に削除された
１ならば削除されるべき単語が見つからなかった
４ならば読みデータが不適切で削除できなかった
８ならば単語データが不適切で削除できなかった

１６ならば品詞が不適切で削除できなかった
ユーザー辞書から単語を削除します。読みデータ、単語データなどの形はｔｏｕｒｏｋｕと同じです。この機能をサポートしていないＭＳＸ－ＪＥでは、ＡにＦＦＨをいれて返ってきます。

以上がＭＳＸ－ＪＥで定義されているファンクションコールです。通常はこれらの機能だけを使っていればよいのですが、VJE-80A にはこのほかにもいくつか拡張されたファンクションコールが定義されているので、ついでにそれらも説明します。

●ｈｅｎｋａｎ２　（５８Ｈ）
入力・Ａに５８Ｈ
ＨＬにＢＷのアドレス
ＤＥに全角片仮名文字列のアドレス
出力・Ａに候補数
ｈｅｎｋａｎ１と全く同じ動作をします。VJE-80では候補の有無(１：有)が返ります。

●Number of Candidates　（５９Ｈ）
入力・Ａに５９Ｈ
ＨＬにＢＷのアドレス
出力・Ａに現在の候補ブロックの候補数
現在の候補ブロックの候補数を返します。

●ｈｅｎｋａｎ３　（５ＡＨ）
入力・Ａに５ＡＨ
ＨＬにＢＷのアドレス
出力・Ａに後続文節の先頭文節の候補数
後続文節になっている部分を再度変換します。
例・「カワノナカノサカナ」をｈｅｎｋａｎ１で変換しておいたとします。
最初のｊｉ＿ｋｏｈｏの結果
ＤＥ→０４Ｈ＋”川の”
ＢＣ→０６Ｈ＋”中の魚”
ｈｅｎｋａｎ３の後のｊｉ＿ｋｏｈｏの結果
ＤＥ→０４Ｈ＋”中の”
ＢＣ→０２Ｈ＋”魚”

●ｈｉｒａｇａｎａ　（６０Ｈ）
入力・Ａに６０Ｈ
ＨＬにＢＷのアドレス
ＤＥに候補バッファ
ＢＣに後続文節バッファ
出力・Ａ：０なら正常終了
ＦＦＨなら異常終了（エラー）
直前に変換した先頭文節の読みを全角平仮名文字列に変換します。あらかじめｊｉ＿ｋｏｈｏ、ｚｅｎ＿ｋｏｈｏなどで候補を獲得してから使用します。
例・「カワノナカノサカナ」をｈｅｎｋａｎ１で変換しておいたとします。
最初のｊｉ＿ｋｏｈｏの結果
ＤＥ→０４Ｈ＋”川の”
ＢＣ→０６Ｈ＋”中の魚”
その後のｈｉｒａｇａｎａの結果
ＤＥ→０６Ｈ＋”かわの”
ＢＣ→０６Ｈ＋”中の魚”

●ｋａｔａｋａｎａ　（６１Ｈ）
入力・Ａに６１Ｈ
ＨＬにＢＷのアドレス
ＤＥに候補バッファ
ＢＣに後続文節バッファ
直前に変換した先頭文節の読みを全角片仮名文字列に変換します。あらかじめｊｉ＿ｋｏｈｏ、ｚｅｎ＿ｋｏｈｏなどで候補を獲得してから使用します。それ以外はｈｉｒａｇａｎａとほぼ同じです。

以上で拡張ファンクションコールの説明を終わります。なお、これらの拡張されたファンクションコールをサポートしていないＭＳＸ－ＪＥではＡにＦＦＨをいれて帰ってきます。例えば、ｈｉｒａｇａｎａ、ｋａｔａｋａｎａはVJE-80ではサポートされていないので、ＡにＦＦＨをいれて帰ってきます。
なお、今後、規格で定義されていない機能を拡張するときには、ファンクション６２Ｈ以降を順に使用することになっています。

第４章　仮想端末インターフェイス

５．４．１　仮想端末の仕組み

第３章までではＭＳＸ－ＪＥの基本的な使い方について述べましたが、これはワープロソフトなどが漢字変換などを行なうときに利用するものです。ユーザーがちょっとプログラム中でＭＳＸ－ＪＥの機能を使いたいな、というときなどにはこれでもまだ負担が多すぎます。そこで，ＭＳＸ－ＪＥには仮想端末インターフェイスという機能があって、割と簡単にＭＳＸ－ＪＥの機能を利用することができます。これは、漢字変換の処理は仮想端末が請け負い、プログラムは仮想端末が返す文字列を表示するだけ、というものです。仮想端末を使った場合のやり取りは、次のようになります。

プログラムが日本語入力を行ないたいときは、画面にウィンドーを開いて仮想端末を呼びだす。
↓
ユーザーはそれ（ウィンドー）を見て例えば、「ＹＵＵＧＵＲＥ」と入力する。
↓
仮想端末はそれを「ユウグレ」と変換し、プログラムに一旦実行を返す。
↓
プログラムは先ほど開いたウィンドーに「ユウグレ」と表示し、再び実行を仮想端末に渡す。
↓
ユーザーはそれを見て変換キーを押す。
↓
仮想端末は変換キーが押されたので「ユウグレ」を「夕暮れ」に変換し、プログラムに実行を返す。
↓
プログラムはウィンドーに「夕暮れ」と表示し、実行を仮想端末に渡す。
↓
ユーザーはそれを見て確定キーを押す。
↓
仮想端末は確定キーが押されたので「夕暮れ」で確定したという内容をプログラムに返す。
↓
「夕暮れ」が入力された。

ＭＳＸ２＋などの漢字ＢＡＳＩＣなどがこの機能を使っているので、上の説明で分からなかった人は漢字ＢＡＳＩＣなどを立ち上げて漢字変換してみてください。プログラム（ＢＡＳＩＣインタープリタ）が仮想端末を呼びだすのはＣＴＲＬ＋ＳＰＡＣＥが押されたときで、このとき一番下の行にウィンドーが開くのでユーザーは漢字変換モードに入ったことが分かります。文字が入力されたときに変換しているのは仮想端末、ウィンドーに表示しているのはプログラム（ＢＡＳＩＣインタープリタ）です。確定キー（リターン）が押されると、そのときウィンドーにあった文字列が入力されたことになります。そしてＣＴＲＬ＋ＳＰＡＣＥでプログラムは漢字入力モードから抜け出ます。
ＭＳＸ－ＪＥの機能を直接使おうとすると、変換キーや確定キーが押されたかどうかまでいちいちプログラムが処理しなければなりません。これに対し、仮想端末を使うときはウィンドーを用意して仮想端末をコールすれば、あとは変換の途中経過と結果だけを表示するだけでいいので漢字入力がかなり楽になります。そのかわり、ウィンドーの大きさや形に制限があったり、確定、変換などは仮想端末で処理するのでそれらの機能キーが必ず決まっています。
ワープロソフトなどでは直接ＭＳＸ－ＪＥを使い、通信ソフトがちょっと漢字入力したい、などというときは仮想端末を使うのがいいでしょう。

５．４．２　仮想端末の起動

仮想端末の起動方法はＭＳＸ－ＪＥを起動する場合と全く同じです。ＦＦＣＡＨ番地コールでＭＳＸ－ＪＥ（仮想端末）のインタースロットコールアドレスを調べ、ファンクション１で必要なワークエリアの大きさを調べ、ファンクション２でＭＳＸ－ＪＥ（仮想端末）を起動します。仮想端末機能を使ったあとは、ファンクション３をコールして仮想端末を閉じます。

５．４．３　仮想端末が出力する文字列

第１節での説明にもあったように仮想端末は変換した文字列は表示せずに、プログラムに文字列を返してプログラムがそれを表示するという方法を取ります。なぜこうなっているかというと、ウィンドーの場所をプログラムが選べるようにするためと、どのスクリーンモードでも仮想端末が使えるようにするためです。このおかげでスクリーン６のインターレスモードやスクリーン２でも仮想端末が使えるわけです。ウィンドーは普通最下行に取りますが、別に画面の真ん中でもいいわけです。

このデータ文字列は仮想端末がカーソルを消したいときなどのためにちょっと変わった形式になっています。ＳＴＢ形式という名前です。プログラムはこのＳＴＢ形式の文字列をウィンドーに表示するわけです。

ＳＴＢのフォーマット

１：ＳＨＩＦＴ－ＪＩＳ文字列
　　カーソル位置に一文字表示し、カーソルを進める。半角英数片仮名を含む。
２：ＮＵＬＬ（０）
　　テキストの終わり
３：＾Ｌ（１２）
　　ウィンドーの消去。カーソルはトップに戻る。
４：＾Ｅ（５）
　　カーソルよりあとの部分を消去。
５：＾Ｈ（８）
　　カーソルの前の文字を消去し、カーソルを１文字後退させる。
６：＾Ｒ（１８）
　　カーソル位置から次の１バイトで指定される長さだけ文字をリバースする。
７：＾Ｚ（２６）
　　次の１バイトの上位４ビットを文字の色、下位４ビットを背景色とする。
８：＾Ｘ（２４）
　　次の１バイトをＸ座標としてカーソルを移動。
９：＾Ｙ（２５）
　　次の１バイトをＹ座標としてカーソルを移動（オプション）
10：＾Ｗ（２３）
　　次の１バイトで指定されるウィンドーを開く（オプション）
11：上記以外の０～15はスキップ。上記以外の16～31はスキップし次の１バイトもスキップ。

文字の位置はすべて半角単位で指定します。ただしカーソルのＸ座標指定は１が開始値で、０の場合はカーソルを表示しないという意味です。

仮想端末のサポートするウィンドーの大きさは１行＊２８文字（全角１４文字）ですが、プログラムがこれ以外の大きさのウィンドーを使うこともできます。そのときはあとで述べるファンクションコール９を利用することになりますが、ディフォルトの１行＊２８文字のウィンドーを使う、というときは＾Ｙ、＾Ｗはサポートする必要はありません。

ウィンドーを開くと当然そこの部分のグラフィックは破壊されます。破壊されると困るときにはプログラムがそこの内容を保存してください。

仮想端末はウィンドー内の内容、カーソル位置などは保存されているものとして実行するので、これらを破壊しないようにしてください。例えば、文字列の最後に１文字追加するときなど、仮想端末は１文字だけのＳＴＢを返します。

５．４．４　仮想端末のファンクションコール

仮想端末のファンクションコールの仕方はＭＳＸ－ＪＥを普通に使う場合と同じです。

ＦＦＣＡＨ番地コールで得られた番地をインタースロットコールします。ファンクション０５Ｈ，０８Ｈ，０９Ｈ，０ＡＨはオプションなので、ないＭＳＸ－ＪＥもあります。

●ｃｌｅａｒ　（Ｆｕｎｃｔｉｏｎ　０４Ｈ）
　入力・Ａに０４Ｈ
　　ＨＬにＢＷのアドレス
　出力・なし
　漢字変換用のバッファをクリアします。今までの入力テキストをクリアするときに使います。

●ｓｅｔ　ＴＴＢ　（０５Ｈ、オプション。この機能のないＭＳＸ－ＪＥもある）
　入力・Ａに０５Ｈ
　　ＨＬにＢＷのアドレス
　　ＤＥに再変換するテキストデータのアドレス
　　ＢＣにＴＴＢのアドレス
　出力・Ａが０なら再変換可能
　　ＦＦＨなら再変換不可能
　特定の文字列を変換させたいとき使います。このファンクションをサポートしないＭＳＸ－ＪＥ（VJE-80Aなど）はＡにＦＦＨをいれて戻ります。

●ｄｉｓｐａｔｃｈ　（０６Ｈ）
　入力・Ａに０６Ｈ
　　ＨＬにＢＷのアドレス
　出力・Ａにステータス
　　ＨＬにＳＴＢのアドレス
　アプリケーションから仮想端末にプロセッサ資源をディスパッチする、とありますが、要するに漢字変換の開始です。仮想端末はウィンドーに文字を表示するときなどにパラメータをセットして実行を一旦返します。プログラムは返ってきたＳＴＢをウィンドーに表示し、ステータスを見てそのあとどうするかを決めます。ＳＴＢはＢＷの内部にあります。
　ステータスの意味は以下のとおりです。
　ｂｉｔ０：１ならプログラムがＳＴＢを表示する
　ｂｉｔ１：１ならプログラムが変換結果を得られる
　ｂｉｔ２：１なら仮想端末の変換が終了した
　それぞれのステータスの意味
　000　仮想端末がキーを無視した場合、また、キー入力がなかった場合
　001　入力中、または変換中
　010　部分確定をした場合
　011　部分確定をした場合
　100　変換が中断されて終了した場合
　101　変換が中断されて終了した場合
　110　全確定した場合
　111　全確定した場合
　プログラムは３つのビットをそれぞれ別に見てください。例えば、ビットが３つとも立っていた場合、プログラムはまずＳＴＢを表示し、ファンクション０７Ｈで変換結果を得、ｄｉｓｐａｔｃｈを中止します。プログラムはbit2が立つまで画面表示と０６Ｈ番のファンクションコールをひたすら繰り返すわけです。

●ｇｅｔ＿ｒｅｓｕｌｔ　（０７Ｈ）
　入力・Ａに０７Ｈ
　　ＨＬにＢＷのアドレス
　出力・ＨＬに得られた変換結果の先頭アドレス
　変換結果を返します。結果は００Ｈで終わるＳＨＩＦＴ－ＪＩＳ形式で返ります。

●ｇｅｔ＿ＴＴＢ　（０８Ｈ、オプション。この機能のないＭＳＸ－ＪＥもある）
　入力・Ａに０８Ｈ
　　ＨＬにＢＷのアドレス
　出力・ＨＬにＴＴＢのアドレス
　ｇｅｔ＿ｒｅｓｕｌｔで得られたテキストの読みデータを得ます。結果はＴＴＢ形式で返ります。このファンクションをサポートしないＭＳＸ－ＪＥ（VJE-80Aなど）はＨＬに０へのポインタを返します。

●inquere_window_size　（０９Ｈ、オプション。この機能のないＭＳＸ－ＪＥもある）
　入力・Ａに０９Ｈ
　　ＨＬにＢＷのアドレス
　　Ｅにテール形式の最大長さ
　　Ｂにウィンドー形式の最大高さ
　　Ｃにウィンドー形式の最大長さ
　出力・ＨＬにウィンドー指定データ
　仮想端末のウィンドーはディフォルトでは１行＊２８文字（全角１４文字）ですが、これ以外のウィンドーを使いたいときに利用します。プログラムが用意できるウィンドーの大きさをいれてコールします。ディフォルトの大きさのウィンドーを使うときは呼ぶ必要はありません。
　ウィンドー指定データは３バイトからなるウィンドー指定子が幾つか並び、最後に０で終わります。
　ウィンドー指定子の３バイトの意味は次のとおりです。
　１バイト目、ウィンドータイプ
　２バイト目、ウィンドー長さ
　３バイト目、ウィンドー高さ
　ウィンドータイプは、画面右はじまで入力されたときに次の行の左はじに折りかえすタイプならば２（このタイプのウィンドーをテール形式といいます)、折りかえさないタイプなら１（このタイプをインデペンデント形式といいます）です。ディフォルトのウィンドーはインデペンデント形式です。
　仮想端末は、指示された形のウィンドーをサポートしないときには次のデータの先頭アドレスを返します。つまり、ディフォルトのウィンドーを再度指定します。　［０１Ｈ＋１ＣＨ＋０１Ｈ＋００Ｈ］
　すべての仮想端末が、使いたい大きさのウィンドーをサポートしているという保証はないので、このファンクションは通常は使わないほうがいいでしょう。

●conflict_detect　（０ＡＨ、オプション。この機能のないＭＳＸ－ＪＥもある))
　入力・Ａに０ＡＨ
　　ＨＬにＢＷのアドレス
　出力・Ａ：０ならばNOT DETECTED
　　ＦＦＨならばDETECTED
　プログラムが取りださなければならないキーを取りださずに仮想端末に制御を移すときに使います。プログラムで利用するキーと仮想端末で利用するキーが衝突するのを防ぐためのファンクションです。通常利用することはないでしょう。

　以上で仮想端末のファンクションコールの説明を終わります。

５．４．５　仮想端末のキー制御の決まり

　仮想端末はキー入力を自分自身で処理します。そこで、同じキーをプログラムと仮想端末が機能キーに割り当てると、プログラムがハングアップ（暴走の一種）してしまいます。例えば、プログラムが、スペースキーで仮想端末に制御を移すようにしておいたとします。そして、仮想端末でもスペースでプログラムに実行を返すようになっていたとすると、スペースが押されたときにそのスペースがキーバッファに残ったまま実行がバドミントンし続けて返ってこなくなります。これを防ぐため、仮想端末およびアプリケーションプログラムがバッファから取りださなければならないキーについての決まりがあります。

　仮想端末
　　キーバッファから取り除かずに、変換を中断させるキー
　　：　Ｆ６～Ｆ１０，ＣＴＲＬ＋ＳＴＯＰ
　　キーバッファから取り除き、変換を中断させるキー
　　：　ＥＳＣ
　　キーバッファから必ず取り除くキー
　　：　Ｆ１～Ｆ５，ＳＰＡＣＥ，グラフィック文字を含むすべての文字キー

　アプリケーションプログラム
　　キーバッファから必ず取りだすキー

：　Ｆ６～Ｆ１０，ＣＴＲＬ＋ＳＴＯＰ，文字キー以外のすべてのキー
キーバッファから取りださず、変換を開始させるキー
：Ｆ１～Ｆ５，ＳＰＡＣＥ，グラフィック文字を含むすべての文字キー
プログラムがこのキーのチェックを行なう必要があるのはｃｌｅａｒしたあと最初にｄｉｓｐａｔｃｈするまでの間です。それ以降は仮想端末が処理します。
ところで、この決まりを守るためには、キーバッファの先頭の文字を、キーバッファから取りださずに調べなければなりません。それには次のような方法を使います。
まず、ＢＩＯＳのＣＨＳＮＳ（００９ＣＨ）でキーバッファに内容があるかどうかを調べます。
キーバッファが空でなかったら、Ｆ３ＦＡＨ番地からの２バイトで指定される番地に、次に取りだされるキーコードが入っています。

また、ＭＳＸはファンクションキーが押されたことを調べることができないので、仮想端末はファンクションキーに次のようなコードをセットし、ファンクションキーの識別に使います。これはｃｌｅａｒをコールしたときにセットされます。元の内容を破壊されたくないときには､プログラムで元のファンクションキーの内容を保存しておいてください。
Ｆ１　０１Ｈ＋３１Ｈ，　Ｆ２　０１Ｈ＋３２Ｈ，　Ｆ３　０１Ｈ＋３３Ｈ，　Ｆ４　０１Ｈ＋３４Ｈ，　Ｆ５　０１Ｈ＋３５Ｈ，

また、Ｆ６～Ｆ１０に次のような値をセットしておくと、仮想端末はそれらをＦ６～Ｆ１０として処理をします。そのほかの値が入っていると、その文字列が入力されたとして実行されます。
Ｆ６　ＦＦＨ＋３６Ｈ，　Ｆ７　ＦＦＨ＋３７Ｈ，　Ｆ８　ＦＦＨ＋３８Ｈ，　Ｆ９　ＦＦＨ＋３９Ｈ，　Ｆ１０　ＦＦＨ＋３ＡＨ，

５．４．６　仮想端末のキー割り当て

編集中
[F1]　　入力モード切り替え（ローマ、英数）
[F2]　　平仮名変換
[F3]　　片仮名変換
[F4]　　半角変換
[F5]　　ＪＩＳコード変換
[CTRL]+[F5] 全確定
[BS]　　後退削除（バックスペース）
[DEL]　　カーソル上の文字を削除
[ESC]　　編集中断（キーはバッファに残らない）
[RETURN]　部分確定（編集中はCTRL+F5と同じ）
[UP]　　変換開始（SPACEと同じ）
[DOWN]　　変換開始（SPACEと同じ）
[RIGHT]　　カーソル右移動
[SHIFT]+[RIGHT] カーソル右端移動
[LEFT]　　カーソル左移動
[SHIFT]+[LEFT] カーソル左端移動
[SPACE]　　変換開始
[SHIFT]+[SPACE] 全角スペースの入力
[CTRL]+[STOP] 編集中断（キーはバッファに残る）
[F6]～[F10] 編集中断（キーはバッファに残る）

編集中の文字列がある場合で、上に挙げたキーでない場合にはそのキーを無視してバッファから取り除きます。そのときにも仮想端末は一旦プログラムに実行を返します（このときのステータスは000Bになります)。キー入力がなかった場合も実行を返します。これは、プログラムが日本語入力と平行して別の処理を行なえるようにするためです。平行処理を行なわないときはそのままｄｉｓｐａｔｃｈをコールしてください。編集中、文字列が何もないときに00H～1FHのキーが入力されると、それをバッファに残したまま変換を終了します。(ステータスは100B)
編集中､文字列がないときにＳＰＡＣＥが押されると全角スペースを確定して戻ります。
文字として編集文字列に加えられるのは英数字および平仮名です。片仮名は平仮名として処理されます。

変換中
[F1]～[F5] 編集中と同じ
[CTRL]+[F5] 全確定（カーソルまでを現在の表示の内容で確定）
[ESC]　　変換中断（キーはバッファに残らない）
[RETURN]　部分確定（先頭文節を確定）
[UP]　　前ブロック
[DOWN]　　次ブロック
[RIGHT]　　変換中断
[LEFT]　　変換中断
[SPACE]　　次候補
[SHIFT]+[SPACE] 前候補
[CTRL]+[STOP] 編集中断（キーはバッファに残る）
[F6]～[F10] 編集中断（キーはバッファに残る）

前ブロックと次ブロックを実行したときはキーバッファがクリアされます。
変換・確定の対象になるのはカーソルの直前の文字までです。
なお、この節のキー割当は、あくまでもアスキーのVJE-80,80Aのものです。第５節の決まりさえ守ればこのとおりでなくてもいいので、メーカーによっては一部の機能キーがアスキーのものと異なる場合があります。例えば、ソニーでは前候補は[SHIFT]+[SPACE]ではなくて[GRAPH]です。[F2]は片仮名変換で、[F3]は半角変換です。ほかにも若干の違いがあります。

第５章　ＭＳＸ－ＪＥ使用上の注意

ＭＳＸ－ＪＥとVJE-80の違いについて
ＭＳＸ－ＪＥとVJE-80を混同している人が多いらしいので念のため解説すると、ＭＳＸ－ＪＥはＭＳＸで日本語入力するときの規格の名前で、VJE-80は「ＭＳＸ－ＪＥ規格を満足した日本語フロントエンド」の名前です。

アスキーの「ＭＳＸ－ＷＲＩＴＥII」は内蔵Ｓ－ＲＡＭ辞書のバックアップ機能を持っています。便利な機能だな、と思ったのですが、仕様書を見てもこの機能の説明はありませんでした。要するに、ＭＳＸ－ＷＲＩＴＥIIの場合はワープロソフトとＳ－ＲＡＭが同じカートリッジに入っているためにこのような機能を付加することができただけのようです。現に、ソニーのＭＳＸ－ＪＥカートリッジとＭＳＸ－ＷＲＩＴＥIIを一緒に差すと、ワープロソフトはＭＳＸ－ＷＲＩＴＥIIが、ＭＳＸ－ＪＥはソニーのものが立ち上がります。ここでＳ－ＲＡＭのバックアップをすると、残念ながらバックアップされるのはＭＳＸ－ＷＲＩＴＥIIの内蔵Ｓ－ＲＡＭのほうでした。辞書のバックアップや辞書の一覧などのファンクションが用意されていると思ったのですが、ちょっと残念です。だってどんな単語を登録したかなんていちいち覚えてないもん。

ＭＳＸ－ＪＥが起動するために最低限必要なワークエリアの大きさは２５６０バイトを越えることはありません。従って、メモリが限られているソフトはＭＳＸ－ＪＥ用に２５６０バイトの領域を確保して起動するとよいでしょう。

STBの指定する文字色について。STBでは＾Ｚで文字色を指定してきます。これは、下のような設定になっています。

文字列の属性	文字色／背景色
編集文字列	１ＦＨ
変換対象文字列	Ｆ５Ｈ
後続文節	１ＦＨ
未変換文節	１ＦＨ

パレット設定を変えているソフトや、SCREEN6を利用しているソフトが、正直にこのとおりの色で表示すると変な色になるので、そういうソフトはこの指定を勝手に解釈して、自分に合った色で表示してください。

第６部　ＭＳＸ　ｔｕｒｂｏ　Ｒ

◆

ＭＳＸの新規格，ＭＳＸターボＲが発表になりました。この機種は、新開発Ｒ８００ＣＰＵを搭載した高速機です。以下で、システムの概要と変更点について説明していきます。

第１章　turboRシステム概要

ＣＰＵ	Ｒ８００（７．１６ＭＨｚ）＋Ｚ８０（３．５８ＭＨｚ）切り換え
ＲＯＭ	ＢＡＳＩＣＲＯＭ＋ＤＩＳＫＲＯＭ＋漢字ドライバ＋ＦＭ音源ドライバ＋第一、第二水準漢字ＲＯＭ標準搭載
ＲＡＭ	２５６Ｋｂｙｔｅ以上
ＢＡＳＩＣ	ＭＳＸターボＢＡＳＩＣ　VERSION ４．０
ＤＩＳＫ	２ＤＤ一台以上標準実装
ＤＯＳ	ＭＳＸ－ＤＯＳ１＋日本語ＭＳＸ－ＤＯＳ２標準実装
ＶＤＰ	Ｖ９９５８（ＭＳＸ２＋と同じ）
ＶＲＡＭ	１２８Ｋｂｙｔｅ以上
ＳＯＵＮＤ	ＰＳＧ３声＋ＦＭ音源（ＭＳＸ－ＭＵＳＩＣ）＋８ｂｉｔサンプリング
Ｉ／Ｏ	ＭＳＸ標準スロット２ プリンターインターフェイス　セントロニクス社準拠 ビデオ出力　ＲＦ＋ビデオ＋Ｓ映像＋アナログＲＧＢ 汎用入出力ポート（ジョイスティックポート）２

主なスペックは以上のとおりです。Ｒ８００ＣＰＵは、Ｚ８０とオブジェクトコンパチブルな１６ｂｉｔＣＰＵで、１命令当たりのクロック数が大幅に減っているので、かなりの高速化が計れます。Ｚ８０の未定義命令や乗算命令が新設されたほかは命令はＺ８０と全く同じです。ＲＡＭは２５６Ｋ以上になりました。ＭＳＸ－ＤＯＳ２も標準実装になりました。ＶＤＰ及びＶＲＡＭは、ＭＳＸ２＋と全く変わりません。第二水準漢字、ＦＭ音源、ディスクドライブ、Ｓ映像出力端子が標準実装になりました。新たに付け加わったのは８ｂｉｔサンプリング機能ですが、ゲーム中にほかの音源と一緒に鳴らすことが非常に困難なので用途は限られるでしょう。

今回は削除された機能も多くあります。まず、カセットインターフェイスがなくなりました。ライトペン、パドル、などの機器がシステムではサポートしなくなりました。これらの機能を使うときはＩ／Ｏポートを直接いじらなければなりません。そのほか規格書だけにあって、実在しない機能も削除されましたが、あまり意味がないと思われますので省略します。

Ｒ８００自体は、メモリを６４Ｋバイトしか扱えないので、それ以上の領域はバンク切り換えでアクセスします。Ｉ／ＯポートのＦＣＨ～ＦＦＨを使うか、ＤＯＳ２に内蔵されているメモリマッパ拡張ＢＩＯＳを使います。１６Ｋごとにメモリを切り換える機構になっています。詳しい使い方は第２部第３章を参照してください。今までは、Ｉ／Ｏを必死に切り換えながら、自分でメモリマッパを切り換えるのが普通でしたが、ＤＯＳ２を標準装備したturboRでは、拡張ＢＩＯＳ方式で切り換えたほうが楽な上に安全なのでそうしたほうがいいでしょう。

２つのＣＰＵは切り換えて使うもので、２つをいっぺんに動作させることはできません。Ｚ８０Ａのほうは、今までのＭＳＸ，ＭＳＸ２，ＭＳＸ２＋用のソフトを走らせるときに速くなりすぎないように付けられたものです。ＲＯＭ版のソフトは、原則としてすべてＺ８０モードで立ち上がるようです。Ｒ８００モードで起動したいときは、ＲＯＭの先頭にＢＩＯＳのＣＨＧＣＰＵで切り換えるプログラムを書いてください。ディスクソフトの場合、ＤＯＳ１フォーマットのディスクが起動時に差されていれば、Ｚ８０モードで立ち上がります。識別不可能な場合はＲ８００モードで動いてしまいますが、”１”キーを押しながら立ち上げればＺ８０モードで立ち上がります。ＤＯＳ２フォーマットのディスクはどうしてもＲ８００モードで起動してしまうので、このような機構になっているのだろうと思われます。なお、ＣＰＵの切り換えはソフトウェアでも行なえますので、今までのディスクソフトをＲ８００モードで動かすのは比較的簡単です。

ＤＯＳ１ではＣＰＵをＲ８００にすることはできません。これは、ＤＯＳ１がＲ８００の速度についていけないことがあるためです。どうしてもＤＯＳ１でＲ８００を動作させたいときは、ディスクアクセスの直前で必ずＣＰＵをＺ８０に切り換えてください。

第２章　Ｒ８００ＣＰＵ

Ｒ８００は、以下の表を見てもらえばわかるとおり、１命令当たりのクロック数がＺ８０に比べて非常に少なくなっています。したがって、ＣＰＵのクロックは２倍でも、動作速度は平均８～１０倍になります。

このほかに、Ｒ８００では特殊なウェイトがかかります。自分のいるアドレスと上位バイトが違うアドレスのＲＡＭをアクセスしたときは、1クロックのウェイトが発生します。つまり、「 LD（C055H),A 」という命令がＣ０Ｅ０Ｈ番地にあった場合と、Ｃ１１０Ｈ番地にあった場合では、動作速度が違うのです。このように、アドレスの上位バイトが違うアドレスをアクセスしたとき起きるウェイトのことをページブレークと呼びます。どうしてこのようなウェイトがかかるのかというと、Ｒ８００はメモリアクセスの仕方がＺ８０と異なるからです。より一層動作速度を上げたいときは、なるべくページブレークを起こさないように、アドレスに注意しながらプログラムを組んでください。このほかに、turboRでは、外部スロット（カートリッジスロット）をアクセスするときは３ウェイト、内部ＲＯＭ（内蔵ＲＯＭ）をアクセスするときは２ウェイトがかかります。

Ｒ８００では、このほかにも、メモリのリフレッシュがクロックとは非同期で行なわれるという特性があります。Ｚ８０では、メモリのクロックの中に、リフレッシュのための時間も含まれていました。しかし、Ｒ８００では、リフレッシュがクロックとは無関係に行なわれます。具体的には、３１マイクロ秒ごとに２８０ナノ秒かけてメモリリフレッシュが行なわれます。つまり、今までのＺ８０用ソフトのように、クロック数を数えて動作にかかる時間を正確に計ることはできないのです。

そこで、turboRでは、動作速度を調節するために、システムタイマーという、Ｒ８００とは無関係に動作するタイマーが付きました。これの使い方については、あとで紹介します。

また、Ｒ８００のように速いＣＰＵでは、ＲＯＭをアクセスするたびに２ウェイトもかかっていたのでは遅すぎます。そこで、turboRでは、メインＲＯＭ、サブＲＯＭ、漢字ドライバをＲＡＭに転送して実行速度を上げる機能が付加されました。もちろん、このモードではメインＲＡＭが６４Ｋバイト減ってしまいます。実行速度を上げるか、メモリ空間を取るかで、プログラムの組み方も変わってくるでしょう。

Ｒ８００はセグメント方式で大容量のメモリ空間を扱うことが可能ですが、turboRでのメモリ切り換えは従来どおりメモリマッパ方式で行なう仕様になっているので、この機能は使用されていません。

Ｒ８００の新設命令

アスキー社の資料では、命令表がアスキーニーモニックで書かれていますが、ここではなじみの深いザイログニーモニックで書いていきます。

新設命令中、MULUBはＡレジスタと他の８ｂｉｔレジスタを掛けてＨＬレジスタに格納する命令で、MULUWはＨＬレジスタと他の１６ｂｉｔレジスタを掛けて、ＤＥ：ＨＬレジスタを二つ合わせて３２ｂｉｔとしてみてそこに値をいれる命令です。

そのほかはほとんどＺ８０の未定義命令としてあったもので、主にインデックスレジスタを８ｂｉｔレジスタとして使用する命令です。 LD IXH,A などとして使います。

IN F,(C) は、Ｉ／Ｏポートから値を読んで、どこにも格納せずにフラグのみを変化させる命令です。レジスタを壊さずにＩ／Ｏポートの状態を検出したいときなどに使用します。

なお、Ｚ８０の未定義命令の SLL（シフト命令）はＲ８００でも定義されていないので使えません。１６ｂｉｔＩ／Ｏポート関係の命令（ IN A,(BC)など）は、実はＭＳＸではＺ８０時代から使用して良いことになっていました。もちろんＲ８００でも使えます。

表について若干の説明をしておきます。新設命令表中、[Ｂ]は命令長（バイト数)、[Ｃ]はクロック数です。フラグの変化の中で、[･]は変化なし、[!]は実行結果により変化することを意味します。

クロック表の中で、クロック数[Ｃ]の値が二つ書いてあるものは、上が条件が成立しないとき、下が成立したときを意味します。

なお、１クロックにかかる時間は［１／クロック周波数］秒です。turboRの場合は［1/7159090（±２)］秒となります。

ﾆｰﾓﾆｯｸ	動作	FLAG SZHPNC	オペコード 2進 16	B	C	
LD t,u	t←u	・・・・・・	11011101＊DD 01 t u	2	2	
LD v,w	v←w	・・・・・・	11111101 FD 01 v w	2	2	
LD t,n	t←n	・・・・・・	11011101 DD 00 t 110 ← n →	3	3	
LD v,n	v←n	・・・・・・	11111101 FD 00 v 110 ← n →	3	3	
ADD A,p	A←A+p	!!!VO!	11011101 DD 10000 p	2	2	
ADD A,q	A←A+q	!!!VO!	11111101 FD 10000 q	2	2	
ADC A,p	A←A+p+Cy	!!!VO!	11011101 DD 10001 p	2	2	
ADC A,q	A←A+q+Cy	!!!VO!	11111101 FD 10001 q	2	2	
INC p	p←p+1	!!!VO・	11011101 DD 00 p 100	2	2	
INC q	q←q+1	!!!VO・	11111101 FD 00 q 100	2	2	
SUB p	A←A-p	!!!V1!	11011101 DD 10010 p	2	2	
SUB q	A←A-q	!!!V1!	11111101 FD 10010 q	2	2	
SBC A,p	A←A-p-Cy	!!!V1!	11011101 DD 10011 p	2	2	
SBC A,q	A←a-q-Cy	!!!V1!	11111101 FD 10011 q	2	2	
DEC p	p←p-1	!!!V1・	11011101 DD 00 p 101	2	2	
DEC q	q←q-1	!!!V1・	11111101 FD 00 q 101	2	2	
AND p	A←A and p	!!1POO	11011101 DD 10100 p	2	2	
AND q	A←A and q	!!1POO	11111101 FD 10100 q	2	2	
OR p	A←A or p	!!OPOO	11011101 DD 10110 p	2	2	
OR q	A←A or q	!!OPOO	11111101 FD 10110 q	2	2	
XOR p	A←A xor p	!!OPOO	11011101 DD 10101 p	2	2	
XOR q	A←A xor q	!!OPOO	11111101 FD 10101 q	2	2	
CP p	A-p	!!!V1!	11011101 DD 10111 p	2	2	
CP q	A-q	!!!V1!	11111101 FD 10111 q	2	2	
MULUB A,r	HL←A*r	O!・O・!	11101101 ED 11 r 001	2	14	rはB,C,D,Eのみ
MULUW HL,ss	DE:HL←HL*ss	O!・O・!	11101101 ED 11ss0011	2	36	ssはBC,SPのみ
IN F,(C)	(C)	!!OPO・	11101101 ED 01110000 70	2	3	

レジスタ表

	000	001	010	011	100	101	110	111
r,s	B	C	D	E	H	L		A
t,u	B	C	D	E	IXH	IXL		A
v,w	B	C	D	E	IYH	IYL		A
p					IXH	IXL		
q					IYH	IYL		

	00	01	10	11
dd	BC	DE	HL	SP
ss	BC	DE	HL	SP
pp	BC	DE	IX	SP
rr	BC	DE	IY	SP
qq	BC	DE	HL	AF

クロック表

ニーモニック	＊	B	C
LD r,s	1	1	
LD r,n	2	2	
LD t,u	2	2	
LD t,n	3	3	
LD v,w	2	2	
LD v,n	3	3	
LD r,(HL)	1	2	
LD r,(IX+d)	3	5	
LD r,(IY+d)	3	5	
LD (HL),r	1	2	
LD (IX+d),r	3	5	
LD (IY+d),r	3	5	
LD (HL),n	2	3	
LD (IX+d),n	4	5	
LD (IY+d),n	4	5	
LD A,(BC)	1	2	
LD A,(DE)	1	2	
LD A,(nn)	3	4	
LD (BC),A	1	2	
LD (DE),A	1	2	
LD (nn),A	3	4	
LD A,I	2	2	
LD A,R	2	2	
LD I,A	2	2	
LD R,A	2	2	
CCF	1	1	
SCF	1	1	
LD dd,nn	3	3	
LD IX,nn	4	4	
LD IY,nn	4	4	
LD HL,(nn)	3	5	
LD dd,(nn)	4	6	
LD IX,(nn)	4	6	
LD IY,(nn)	4	6	
LD (nn),HL	3	5	
LD (nn),dd	4	6	
LD (nn),IX	4	6	
LD (nn),IY	4	6	
LD SP,HL	1	1	
LD SP,IX	2	2	
LD SP,IY	2	2	

ニーモニック	＊	B	C
PUSH qq	1	4	
PUSH IX	2	5	
PUSH IY	2	5	
POP qq	1	3	
POP IX	2	4	
POP IY	2	4	
EX DE,HL	1	1	
EX AF,AF'	1	1	
EXX	1	1	
EX (SP),HL	1	5	
EX (SP),IX	2	6	
EX (SP),IY	2	6	
RLCA	1	1	
RLA	1	1	
RRCA	1	1	
RRA	1	1	
RLC r	2	2	
RLC (HL)	2	5	
RLC (IX+d)	4	7	
RLC (IY+d)	4	7	
RL ［RLCと同様］			
RRC［RLCと同様］			
RR ［RLCと同様］			
SLA［RLCと同様］			
SRA［RLCと同様］			
SRL［RLCと同様］			
RLD	2	5	
RRD	2	5	
BIT b,r	2	2	
BIT b,(HL)	2	3	
BIT b,(IX+d)	4	5	
BIT b,(IY+d)	4	5	
SET b,r	2	2	
SET b,(HL)	2	5	
SET b,(IX+d)	4	7	
SET b,(IY+d)	4	7	
RES［SETと同様］			
NOP	1	1	
HALT	1	2	
DI	1	2	
EI	1	1	
IM 0	2	3	
IM 1	2	3	
IM 2	2	3	

ニーモニック	＊	B	C
LDI	2	4	
LDIR	2	4	
LDD	2	4	
LDDR	2	4	
CPI	2	4	
CPIR	2	5	
CPD	2	4	
CPDR	2	5	
ADD A,r	1	1	
ADD A,n	2	2	
ADD A,p	2	2	
ADD A,q	2	2	
ADD A,(HL)	1	2	
ADD A,(IX+d)	3	5	
ADD A,(IY+d)	3	5	
ADC［ADDと同様］			
SUB［ADDと同様］			
SBC［ADDと同様］			
AND［ADDと同様］			
OR ［ADDと同様］			
XOR［ADDと同様］			
CP ［ADDと同様］			
INC r	1	1	
INC p	2	2	
INC q	2	2	
INC (HL)	1	2	
INC (IX+d)	3	7	
INC (IY+d)	3	7	
DEC［INCと同様］			
DAA	1	1	
CPL	1	1	
NEG	2	2	
ADD HL,ss	1	1	
ADC HL,ss	2	2	
SBC HL,ss	2	2	
ADD IX,pp	2	2	
ADD IY,rr	2	2	
INC ss	1	1	
INC IX	2	2	
INC IY	2	2	
DEC ss	1	1	
DEC IX	2	2	
DEC IY	2	2	

ニーモニック	＊	B	C
IN A,(n)	2	3	
IN r,(C)	2	3	
IN F,(C)	2	3	
INI	2	4	
INIR	2	4	
		3	
IND	2	4	
INDR	2	4	
		3	
OUT (n),A	2	3	
OUT (C),r	2	3	
OUTI	2	4	
OTIR	2	4	
		3	
OUTD	2	4	
OTDR	2	4	
		3	
CALL nn	3	5	
CALL cc,nn	3	3	
		5	
RET	1	3	
RET cc	1	1	
		3	
RETI	2	5	
RETN	2	5	
RST	1	4	
JP nn	3	3	
JP cc,nn	3	3	
JR e	2	3	
JR C,e	2	2	
		3	
JR NC,e	2	2	
		3	
JR Z,e	2	2	
		3	
JR NZ,e	2	2	
		3	
JP (HL)	1	1	
JP (IX)	2	2	
JP (IY)	2	2	
DJNZ e	2	2	
MULUB A,r	2	14	
MULUW HL,ss	2	36	

第３章　turboRの変更点

ここでは、turboRになってから変更のあった機能について説明します。

６．３．１　ＢＩＯＳの変更

ここでは、turboRになってからの変更のあったＢＩＯＳについて説明します

新設ＢＩＯＳ
turboRでは以下のＢＩＯＳが新設されました。これらの使い方については第８部を参照してください。
　ＣＨＧＣＰＵ（００１８０Ｈ／ＭＡＩＮ）　ＣＰＵのモードを切り換える
　ＧＥＴＣＰＵ（００１８３Ｈ／ＭＡＩＮ）　ＣＰＵのモードを得る
　ＰＣＭＰＬＹ（００１８６Ｈ／ＭＡＩＮ）　ＰＣＭの再生
　ＰＣＭＲＥＣ（００１８９Ｈ／ＭＡＩＮ）　ＰＣＭの録音

削除されたＢＩＯＳ
以下のＢＩＯＳは、turboRでは削除されました。
　ＴＡＰＩＯＮ（００Ｅ１Ｈ／ＭＡＩＮ），ＴＡＰＩＮ　（００Ｅ４Ｈ／ＭＡＩＮ），
　ＴＡＰＩＯＦ（００Ｅ７Ｈ／ＭＡＩＮ），ＴＡＰＯＯＮ（００ＥＡＨ／ＭＡＩＮ），
　ＴＡＰＯＵＴ（００ＥＤＨ／ＭＡＩＮ），ＴＡＰＯＦＦ（００Ｆ０Ｈ／ＭＡＩＮ），
上の６つのＢＩＯＳをコールするとＣｙフラグを立ててＲＥＴします。
　ＳＴＭＯＴＲ（００Ｆ３Ｈ／ＭＡＩＮ）は、何もせずに戻ります。
　ＧＴＰＤＬ　（００ＤＥＨ／ＭＡＩＮ）は、Ａレジスタに０を入れて返ります
　ＧＥＴＰＡＤ（００ＤＢＨ／ＭＡＩＮ），ＮＥＷＰＡＤ（０１ＡＤＨ／ＳＵＢ）
上の２つのＢＩＯＳは、ライトペンの値を読むと０を返します

６．３．２　ＢＡＳＩＣの変更

turboRでは、以下の命令が追加、変更されました。

追加されたもの
　ＣＡＬＬ　ＰＣＭＰＬＡＹ　ＰＣＭを再生する。詳しくは取扱説明書を参照
　ＣＡＬＬ　ＰＣＭＲＥＣ　ＰＣＭを録音する。詳しくは取扱説明書を参照
　ＣＡＬＬ　ＰＡＵＳＥ（ｎ）　プログラムの実行をｎミリ秒間停止する
ＤＯＳ２関連の命令も増設されました。詳しくはＢＡＳＩＣのマニュアルを参照のこと

削除されたもの
　ＣＬＯＡＤ，ＣＳＡＶＥ，ＭＯＴＯＲ。これらの命令を実行しようとするとSyntax errorになります
　なお、デバイス名のＣＡＳ：も使用できなくなりました
　ＰＤＬ関数は、常に０を返します。ＰＡＤ関数は、ライトペンの値を読もうとすると０を返します

変更されたもの
　ＣＯＰＹ，ＦＩＬＥＳなどディスク関係の命令。ＤＯＳ２の階層化ディレクトリーをサポートするようになったほか、いくつか変更が加えられました。

６．３．３　ワークエリアの変更

turboRでは、以下のワークエリアが追加、削除されました。

追加されたもの
　？？？？？？（ＦＣＢ１Ｈ，１）　　Ｉ／ＯポートＡ７Ｈ番地の値の保存場所

削除されたもの
　ＣＳ１２０　（Ｆ３ＦＣＨ，５＊２）　カセットテープ用パラメータ

ＬＯＷ　　（Ｆ４０６Ｈ，２）　　カセットテープのデータ
ＨＩＧＨ　　（Ｆ４０８Ｈ，２）　　カセットテープのデータ
ＨＥＡＤＥＲ（Ｆ４０ＡＨ，１）　　カセットテープのデータ
ＣＡＳＰＲＶ（ＦＣＢ１Ｈ，１）　　”ＣＡＳ：”用文字保存場所

６．３．４　Ｉ／Ｏポートの変更

turboRでは、以下のようなＩ／Ｏポートが増設されました。
ＡＡＨ　　Ｄ／Ａコンバーター。ＰＣＭ用
ＡＢＨ　　Ｄ／Ａコンバーター。ＰＣＭ用
ＡＤＨ　　ポーズランプ、ＣＰＵモードを示すランプの点滅に使用
ＥＡＨ　　ＣＰＵモード変更の際に使用する
ＥＢＨ　　ＣＰＵモード変更の際に使用する
ＥＣＨ　　システムタイマー（下位８ビット）
ＥＤＨ　　システムタイマー（上位８ビット）

なお、ＶＤＰのＩ／Ｏポートアドレスが９８Ｈ～９ＢＨに固定になりました。メインＲＯＭの６番地と７番地を使ってＶＤＰのアドレスを探す手間が省けます。
システムタイマーは３．９９１マイクロ秒ごとにカウントアップするカウンターです。turboRでは、ＣＰＵのクロック数を数えて実行速度を計ることが困難なので、ＥＣＨ，ＥＤＨ番地の内容を読んで実行速度を調節してください。なお、２つのポートを読み込んでいる間にカウンターの値が変化すると都合が悪いので、必ずＥＣＨ，ＥＤＨのいずれか一方だけを使用するようにしてください。ＥＤＨ番地を使用すると１．００１２ミリ秒ごとにカウントアップするカウンターになります。なお、この２つのポートは書き込みも可能です。
ＡＤＨは、ポースランプやＣＰＵのモードを示すランプの点滅に使用します。bit7を立てれば高速モードランプが、bit0を立てればポーズランプが点灯します。bit1を立てると、低速ランプが点灯するようですが、今のところ低速ランプを搭載した機種が存在しませんので、調べられませんでした。例えば、ＢＡＳＩＣ上から「ＯＵＴ＆ＨＡ７，１」とやると高速ランプが消え、ポーズランプが点灯します。
今回は仕様から削除されたＩ／Ｏポートも存在します。９８Ｈ～９ＢＨのバージョンアップアダプター用ＶＤＰポート、Ｂ０Ｈ～Ｂ３ＨのソニーＨＢ－５５付属Ｓ－ＲＡＭカートリッジ用ポート、Ｂ８Ｈ～ＢＢＨの三洋仕様のライトペンのポート、ＢＣＨ～ＢＦＨのビクター仕様ＶＨＤインターフェイスのポート、Ｃ０Ｈ～Ｃ３ＨのＭＳＸ－ＡＵＤＩＯ用ポート、Ｃ８Ｈ～ＣＣＨのＭＳＸインターフェイス用ポート、Ｄ０Ｈ～Ｄ７Ｈのフロッピーディスク用ポートです。なお、フロッピーディスクはこのポートは使用していませんので、なくなっても全く困りません。バージョンアップアダプター用のＶＤＰ用ポートは、ＭＳＸ１をＭＳＸ２にバージョンアップするためのアダプターで実際に使用されていたポートです。

６．３．５　turboRのスロット構成

turboRでは、メインＲＯＭのスロットがＳＬＯＴ０－０に、ＦＭ音源ドライバがＳＬＯＴ０－２に、サブＲＯＭと漢字ドライバがＳＬＯＴ３－１に統一されました。なお、サブＲＯＭはページ０に、漢字ドライバはページ１及び２に置かれます。メインＲＡＭはＳＬＯＴ３－０に、ＭＳＸ－ＤＯＳ２及びディスクインターフェイスＲＯＭはＳＬＯＴ３－２に置かれます。
turboRでは、増設ＲＡＭカートリッジなどを差しても、必ず本体内蔵ＲＡＭがメインＲＡＭに割り当てられます。これは、外付けのＲＡＭは、１バイトアクセスするたびに３ウェイトがかかるので、実用的でないからです。
ディスクドライブの場合は、やはり本体内蔵ドライブが必ずマスタースロットになるようですが、のちのちになって例えばＭＳＸ－ＤＯＳ３カートリッジなどが発売された場合はどうなるか全く分かりませんので、今までのようにマスタースロットを探したほうが安全かもしれません。付け加えるならば、ＭＳＸ－ＤＯＳ２カートリッジをturboRに差すと、本体内蔵のＤＯＳ２が優先的に立ち上がります。これは、本体内蔵版のほうがバージョンが上だからです。（カートリッジ版のバージョンは2.20、turboR内蔵版は2.30）
turboRでは、ＤＯＳ１及びＤＯＳ２が同じスロット、すなわちＳＬＯＴ３－２におかれます。この二つのＤＯＳは、必要に応じて切り換えられます。なお、Ｒ８００モードで

ＯＳ１を使用することはできません。ＤＯＳ１がＲ８００の速度についていけないことがあるためです。どうしても使用したい場合は、ディスクアクセスの直前にＺ８０モードに切り換えるようにしてください。

turboRでは、システムＲＯＭをＲＡＭに転送して高速化するＤＲＡＭモードがつきました。この機能を使えば、ＢＡＳＩＣ－ＲＯＭやＢＩＯＳを書き換えることも可能です。

システムは、起動時にシステムＲＯＭをメインＲＡＭの最後の４つのセグメントに転送します。このセグメントは、ＲＡＭ２５６Ｋの機種の場合は１２番から１５番まで、ＲＡＭ５１２Ｋの機種の場合は２８番から３１番までのセグメントが割り当てられます。例えばＲＡＭ２５６Ｋの機種の場合、１５番のセグメントにメインＲＯＭのページ０が、１４番にメインＲＯＭのページ１が、１３番にサブＲＯＭが、１２番に漢字ドライバがそれぞれ転送されています。

Ｒ８００ ＤＲＡＭ モードで動いている場合、この部分のＲＡＭは同時にシステムＲＯＭとしてもアクセスが可能になります。すなわち、スロット０－０のＲＯＭとしても、スロット３－０のマッパＲＡＭとしてもアクセスできるのです。しかし、ＲＯＭとしてアクセスする場合は書き込みはできません。スロット０－０に値を書き込んでも書き込めません。しかし、スロット３－０のＲＡＭとしてなら書き換えも可能なのです。これをうまく使えば、メインＲＯＭを書き換えてＢＩＯＳにトラップをかけたりリセット時に特殊な処理をさせたりすることが可能です。実際、そういったプログラムはあちこちで発表されています。実際に書き換えるときは、まずＣＰＵをＲ８００ＲＯＭモードにして、拡張ＢＩＯＳ方式のメモリマッパで最後の４セグメントを開放して、書き換えた後ＣＰＵをＲ８００ ＤＲＡＭモードに戻してください。

turboRでは、ＣＰＵがＲ８００のときには外部スロット、つまり、カートリッジスロットをアクセスするときには、３ウェイトがかかります。外に出ていないスロット、具体的にはシステムＲＯＭや内蔵ディスクインターフェイスＲＯＭなどはこのウェイトはかかりません。従って、外付けの拡張ＲＡＭカートリッジなどを差した場合、１バイトアクセスするたびに３ウェイトがかかってしまいます。計測したところ、メインＲＡＭにプログラムを置いた場合と外付けのＲＡＭに置いた場合は実行時間が４～５倍違うという結果が得られました。外付けの拡張ＲＡＭカートリッジはＲＡＭディスクとしてだけ使うのが賢い使い方のようです。外部スロットにつながっているＩ／Ｏにも同様なウェイトがかかるようです。

turboRでは、Ｒ８００ ＤＲＡＭモードで使うための６４Ｋのメモリを確保しつつ実行します。つまり、ＲＯＭモードで動かしてもその分の領域は開放されないのです。この領域を使用するには､メモリマッパ拡張ＢＩＯＳを使って領域を開放しなければなりません。

ＣＰＵのモードを切り換えるハードについて。これは、松下ＦＳ－Ａ１ＳＴ専用かもしれませんが、Ｉ／ＯポートのＥ４Ｈ, Ｅ５Ｈを利用して切り換えます。

切り換えるにはまずＥ４Ｈに０６Ｈを出力して切り換え用のハードを使用できる状態にして、それから、Ｚ８０モードにしたいときはＥ５Ｈに６０Ｈを、Ｒ８００ ＲＯＭ モードにしたいときは４０Ｈを、Ｒ８００ ＤＲＡＭ モードにしたいときは００Ｈを書き込みます。切り換える前がＺ８０モードかＲ８００ ＲＯＭ モードだったときはさらにＥ５Ｈに６０Ｈを出力します。例えば、Ｚ８０モードから Ｒ８００ ＲＯＭ モードにするときは、「OUT E4H,06H　OUT E5H,40H　OUT E5H,60H」とこのようにすればいいわけです。Ｅ５Ｈに２０Ｈを出力すればＺ８０ ＤＲＡＭ モードというあまり役に立たないモードにも切り換えることもできます。ただしＢＡＳＩＣ上のＯＵＴ命令でこれをやると、レジスタの値が完璧に破壊されるので暴走します。

６．３．６ ＰＣＭ音源

turboRのＰＣＭは、Ｉ／Ｏポートを使ってユーザーが操作することが許可されています。自分で操作すれば、サンプリングレートを自由な周波数に変更したり、また、録音したデータを違う周波数で再生して音を変化させることができます。ここでは、Ｉ／Ｏポートを直接操作してＰＣＭを利用する方法について説明します。ＢＩＯＳの使い方については第８部のアペンディックスを参照してください。

Ｉ／Ｏポートの仕様

I/Oポート番号	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Ａ４Ｈ(READ)	0	0	0	0	0	0	CT1	CT0
Ａ４Ｈ(WRITE)	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
Ａ５Ｈ(READ)	COMP	0	0	SMPL	SEL	FILT	MUTE	ADDA
Ａ５Ｈ(WRITE)	0	0	0	SMPL	SEL	FILT	MUTE	ADDA

解説

CT1, CT0　（カウンターデータ）

63.5μ秒ごとにカウントアップするカウンターです。ＢＩＯＳでは、このカウンターに同期して値の読み書きをしています。Ｉ／ＯのＡ４Ｈに値を書き込むとカウンターはクリアされます。

DA7～DA0　（Ｄ／Ａ出力データ）

ここに8bitの値を書き込むことによってＰＣＭデータを再生します。データの形式はアブソリュートバイナリで１２７が０レベルに相当します。録音の際は、ここに値を出力して入ってきた値との大小で録音データを調べます。

COMP　　（コンパレーターの出力信号）

サンプルホールドの出力信号とＤ／Ａコンバーターの出力信号を比べます。Ｄ／Ａ出力のほうがサンプルホールド出力より小さければ、このビットが立ちます。Ｄ／Ａ信号のほうが大きいか、同じならばこのビットは立ちません。録音の際に使用します。

SMPL　　（サンプルホールド信号）

入力信号をサンプルするか、ホールドするかを選択します。入力信号の値を調べている最中に値が変化するとまずいので、その間はここを１にしておきます。

このビットが０ならサンプル（リセット時の状態)、１ならホールドです。

SEL　　（フィルター入力信号の選択）

ローパスフィルターに入力する信号を、Ｄ／Ａコンバーターの出力信号にするか、マイクアンプの出力信号にするかを選択します。０ならＤ／Ａコンバーター出力信号、１ならマイクアンプ出力信号です

FILT　　（サンプルホールド回路入力信号の選択）

Ａ／Ｄ時にサンプルホールド回路に入力する信号を、フィルターの出力信号にするか、基準信号にするかを選択します。０なら基準信号（リセット時の状態)、１ならフィルタ出力信号になります。

MUTE　　（ミューティング制御）

システム全体の音声出力をONにしたりOFFにしたりします。０なら音声出力OFF（リセット時の状態)、１なら音声出力ONです。

ADDA　　（バッファモード）

Ｄ／Ａコンバーターの出力を指定します。Ｄ／Ａ時には０（ダブルバッファ、リセット時の状態)、Ａ／Ｄ時には１（シングルバッファ）を指定します。

ＰＣＭを再生するには、Ａ５Ｈ番地に０３Ｈを出力して、メモリ上からＡ４Ｈ番地にデータを次々と出力していけば大丈夫です。システムタイマーなどを使って同期をとって出力してください。Ａ４Ｈ番地のカウンターを使うと、ＢＩＯＳでサポートしている４種類の周波数で再生ができます。できれば、このカウンターは使わずに、システムタイマーを使って周波数を微妙に変えて再生すると音の高さを変化させることができます。

ＰＣＭを録音するには、Ａ５Ｈ番地にまず０６Ｈを出力します。(これは準備です）次に、《Ａ５Ｈに１６Ｈを出力して、Ａ４Ｈに８０Ｈを出力して、すぐにＡ５Ｈの値を読みます。もし、この値の第７ビットが立っていたならば、(COMPのビットが立っていたならば）求める信号は８０Ｈ以上ということですから、出力値８０Ｈに４０Ｈを足して再びこの値をＡ４Ｈに出力します。もし、第７ビットが立っていなかったなら、求める値は８０Ｈ未満ということですから、出力値８０Ｈから４０Ｈを引いて再びＡ４Ｈに出力します。そしてまたＡ５Ｈの値を読み、ビット７が立っているかどうかを見て出力値に２０Ｈを足すか引くかして、その値をＡ４Ｈに出力してＡ５Ｈを読んでビット７を見て出力値から１０Ｈを足すか引くかして（中略）こうしてこれを８回繰り返して、やっと１バイトの読み込みが終了です。この値をメモリに保存して、Ａ５Ｈに０３Ｈを出力して､》さらにシステムタイマーなどで同期をとって《》内を繰り返して次の１バイトを読んで、また読んで（中略）こうして録音するわけです。サンプルプログラムも参考にしてください。

ＶＲＡＭにサンプリングデータを置く場合、録音時と再生時のスクリーンモードが違っているとうまく再生できないことがあります。これはＭＳＸのＶＲＡＭの配置がスクリーンモードによって違うためです。紙面の関係で詳しい説明はできませんが簡単にいうとＳＣＲＥＥＮ５の０００１Ｈ番地とＳＣＲＥＥＮ７の０００１Ｈ番地は違う場所なのです。

６．３．７　ＭＳＸ－ＭＩＤＩ拡張ＢＡＳＩＣ

turboR になってから規格になったＭＳＸ－ＭＩＤＩ用の拡張ＢＡＳＩＣコマンドについて解説します。

ＣＡＬＬ　ＭＵＳＩＣ
モード１で使用できるチャンネル数が「内蔵ＦＭ（６音＋１リズム）＋ＭＩＤＩ（８音＋１リズム）に変更になりました。使用できるチャンネル数の合計は８音（＋１リズム）でＦＭ音源のときと比べて２音増えました。また、ＦＭ音源だけのときと同じく、この命令を使用したとき８０７バイトの領域がＭＳＸ－ＭＵＳＩＣ用に確保され、変数その他の値がすべてクリアされます。ＭＩＤＩを接続した状態でこの命令を使用すると、ＭＩＤＩのない機種より動作が「重く」なりますのでご注意ください。

ＣＡＬＬ　ＭＤＲ（<ＢのＭＩＤＩノート番号>，<ＳのＭＩＤＩノート番号>，<ＭのＭＩＤＩノート番号>，<ＣのＭＩＤＩノート番号>，<ＨのＭＩＤＩノート番号>）
リズム音用ＭＭＬで使用するＢ，Ｓ，Ｍ，Ｃ，Ｈコマンドに割り当てるＭＩＤＩノート番号を指定します。ノート番号は０～１２７を指定します。省略はできません。

ＰＬＡＹ
書式は今までと変わりませんが、「ＰＬＡＹ　＃１，…」という書式を使用することによってＭＩＤＩ楽器を使用できるようになります。文字列との関連は、「ＰＬＡＹ　＃１，<ＭＩＤＩ機器用文字列１>，…<ＭＩＤＩ機器用文字列ｎ>，<ＭＩＤＩ機器リズム音用文字列>，<ＰＳＧ用文字列１>，<ＰＳＧ用文字列２>，<ＰＳＧ用文字列３>」です。

なお、ＭＳＸ－ＭＩＤＩに対してはＣＡＬＬ　ＡＵＤＲＥＧ、ＣＡＬＬ　ＰＩＴＣＨ、ＣＡＬＬ　ＴＥＭＰＥＲ、ＣＡＬＬ　ＴＲＡＮＳＰＯＳＥ、ＣＡＬＬ　ＶＯＩＣＥは無効です。音色設定はＣＡＬＬ　ＶＯＩＣＥではなくＭＭＬの＠ｎを使用します。
また拡張ＢＡＳＩＣではＭＩＤＩ受信は行えません。

またＭＩＤＩ用に追加・変更されたＭＭＬは以下のとおりです。

（追加）

文字	意味	値の範囲
＠Ｈｎ	使用するＭＩＤＩチャンネル番号	１≦ｎ≦１６
＠Ｃｍ、ｎ	コントロールチェンジを出力する	０≦ｍ≦１２７
	ｍはコントロール番号、ｎはコントロール番号に対する値	０≦ｎ≦１２７
＠Ｓｎ	ＭＩＤＩリアルタイムクロックに関する命令	
	ｎ＝０　ＦＣＨ（ＳＴＯＰ）を出力し、クロック出力停止	
	ｎ＝１　ＦＡＨ（ＳＴＡＲＴ）を出力し、クロック出力開始	
	ｎ＝２　ＦＢＨ（ＣＯＮＴＩＮＵＥ）を出力し、クロック出力開始	
	クロックのテンポはＰＬＡＹ文の第１文字列と同じになる	

（変更）

Ｌｎ	長さを指定する	１≦ｎ≦９６
Ｒｎ	休符を設定する	１≦ｎ≦９６
＠Ｗｎ	ｎで指定した長さだけ状態を継続する	１≦ｎ≦９６
Ｖｎ	ＭＩＤＩ機器に対し８倍した値をノートオン・ベロシティーとして出力する	１≦ｎ≦９６
＠Ｖｎ	ボリューム（コントロールレンジ＃７）を出力する	１≦ｎ≦１２７
＠ｎ	内蔵ＦＭ音源に対するときと意味が違う。ＭＩＤＩ機器に	０≦ｎ≦６３
	対してはプログラムチェンジを出力する	０≦ｎ≦１２７
Ｚｎ	１バイトＭＩＤＩデータを出力する	１≦ｎ≦２５５

（リズム用、追加）

＠ｎ	ＭＩＤＩ機器に対してプログラムチェンジを出力する

（リズム用、変更）

＠Ａｎ	ＭＩＤＩ機器に対してはＶｎと同じくベロシティーを設定する

ＭＳＸ－ＭＩＤＩ拡張ＢＡＳＩＣのＭＩＤＩデータフォーマットは以下のとおりです。

ノート送信
●ノートｏｎ
　ステータス　　1001nnnn(9nH)　　n=0～15（ＭＩＤＩチャンネル番号）
　ノートナンバー　0kkkkkkk　　K=24～119
　ベロシティ　　0vvvvvvv　　v=8*I　（I=0～15）
　ベロシティ値はＶｎ、@Ａｎで指定された値の８倍
●ノートｏｆｆ
　ステータス　　1001nnnn(9nH)　　n=0～15（ＭＩＤＩチャンネル番号）
　ノートナンバー　0kkkkkkk　　k=24～119
　ベロシティ　　00000000　　０：ノートｏｆｆ
　リズム音を複数音同時発音または発音停止するときは、ランニングステータスを使用します。

コントロールチェンジ
●@Ｖｖ（ボリューム）
　ステータス　　1011nnnn(BnH)　　n=0～15（ＭＩＤＩチャンネル番号）
　コントロール番号　00000111　　c=7
　コントロール値　0vvvvvvv　　v=0～127
●@Ｃｃ、ｖ（コントロールレンジ）
　ステータス　　1011nnnn(BnH)　　n=0～15（ＭＩＤＩチャンネル番号）
　コントロール番号　0ccccccc　　c=0～127
　コントロール値　0vvvvvvv　　v=0～127

プログラムチェンジ
●@ｐ（プログラムチェンジ）
　ステータス　　1100nnnn(CnH)　　n=0～15（ＭＩＤＩチャンネル番号）
　プログラム番号　0ppppppp　　p=0～127

シリアルリアルタイムメッセージ
●タイミングクロック
　ステータス　　11110000(F8H)
　@Ｓ１か@Ｓ２でクロックが開始されていてＰＬＡＹ文が実行中のみ送信されます。ＰＬＡＹ文が終了するとクロックも停止しますがストップ（ＦＣＨ）は送信されません。
●@Ｓ１（スタート）
　ステータス　　11111010(FAH)
　クロック出力開始
●@Ｓ２（コンティニュー）
　ステータス　　11111011(FBH)
　クロック出力開始
●@Ｓ０（ストップ）
　ステータス　　11111100(FCH)
　クロック出力停止

ＭＳＸ－ＭＩＤＩ使用上の注意

　ＭＡＩＮ－ＲＯＭの００２ＥＨ番地のビット０が１ならばＭＳＸ－ＭＩＤＩが本体に内蔵されています。内蔵ＭＩＤＩが存在しない場合は、４０１ＣＨ番地から４バイトの内容が”ＭＩＤＩ”の４文字なら、そのスロットに外付けＭＳＸ－ＭＩＤＩが存在します。ＭＳＸ－ＭＩＤＩはターボＲ以降でないと動作しません。ＭＩＤＩは最大２系統まで接続できるようです。
　また、以下のフックは内蔵ＭＩＤＩ用に割り当てられています。
　０ＦＦ７５Ｈ，５（Ｈ．ＭＤＩＮ）　旧名称Ｈ．ＯＬＮＯＲＮ　ＭＩＤＩ　ＩＮ割込
　０ＦＦ９３Ｈ，５（Ｈ．ＭＤＴＭ）　旧名称Ｈ．ＦＲＱＩＮＴ　8253タイマー割込

　外付けＭＩＤＩの場合、次のフックが今までの機能と兼用して使われます。
　０ＦＤ９ＡＨ，５（Ｈ．ＫＥＹＩ）

ＭＳＸ－ＭＩＤＩが使用するＩ／Ｏポートは以下のとおりです。
E0H　８２５１送受信データ（読み書き）
E1H　外付　Ｒ／８２５３割り込みフラグ、８２５１ステータス
　　Ｗ／モード、コマンド
E2H　外付　ＭＩＤＩ及び８２５１のポート指定
E8H　内蔵、外付兼用。E1Hと機能は同じ
E9H　内蔵、外付兼用。E2Hと機能は同じ
EAH　ＯＵＴ２端子の信号ラッチ（書き込み専用）
EBH　EAHのイメージ
ECH～EFH　ＭＩＤＩインターフェイス（読み書き、EFHのみ書き込み専用）

E2Hに値を書き込むことにより、外付けＭＩＤＩのＩ／Ｏ番地が変わります。これによって複数のＭＩＤＩインターフェイスを同時に使用することができるようです。
Ｉ／Ｏを用いて機械語からＭＩＤＩを操作する方法については本書では触れません。

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆
turboRにＭＳＸ－ＤＯＳ２カートリッジ１２８ＫＲＡＭ内蔵版を差した場合、マッパＲＡＭとして動作しませんが、これは、ひょっとすると、ＲＡＭがＲ８００の動作速度についていけないためではないか、と推測されています。２５６ＫＲＡＭ内蔵版のほうは、高速ＲＡＭが使用されているようです。
アスキー社発行「ＭＳＸ２＋パワフル活用法」には、主にＭＳＸ－ＭＵＳＩＣのレジスタの説明などにかなりの誤りがあります。アスキーに問い合わせをすることをおすすめします。
ＤＯＳ１では、ディスクのメディアタイプ（メディアＩＤ）を決定するのに、ディスクのＦＡＴの第１バイトを読んでいるようですが、ＤＯＳ２ではブートセクターのメディアＩＤの値を読んでメディアタイプを決定するようです。
ＤＯＳ１フォーマットしたディスクにオプションなしでＦＩＸＤＩＳＫをかけると、ＦＳ－Ａ１ＳＴでＺ８０モードでＤＯＳ２が起動します。
漢字ＲＯＭ内蔵機にＨＡＬＮＯＴＥを接続すると、本体漢字ＲＯＭが優先となる現象が起きることがあるようです。
ソニー製ＭＳＸ２＋、ＨＢ－Ｆ１ＸＤＪには、内部の回路設計に致命的な欠陥があり、買ってから１年くらいすると内蔵ＦＭ音源がぼそぼそとした変な音になって、最終的には全く音が出なくなってしまいます。これは、すべてのＨＢ－Ｆ１ＸＤＪで起こり得ます。もしこのような状況になってしまったときは、ＦＭ音源チップＹＭ２４１３のそばにあるＣ９９という５０Ｖ　１μＦのコンデンサーを新しいものと取り替えてください。そうすれば再び音が鳴るようになります。面倒な人はメーカー修理に出せば無料で修理してくれます（保証期間が過ぎていても）。間違ってメーカーのサービスの人が修理代を請求してくることもありますが、その場合、「これは回路のミスだと設計のほうが認めてるんだ！」と騒ぐことにより無償修理となります。なおＨＢ－Ｆ１ＸＶでは回路が変更されているためこの現象は起きません。

第７部　　サンプルプログラム

◆

ここでは、付属サンプルプログラムディスクの内容について簡単に説明します。ディスクのほうにも説明をつけておきましたので、そちらのほうも参考にしてください。ファイルの中で、拡張子が「BAS」のものは実行可能なベーシックファイル、「MAC」のものは機械語部分のソースファイル、「DOC」のものはベーシックファイルで、走らせるとそのプログラムの詳しい説明が表示されます。

ＥＳＣＡＰＥ．ＢＡＳ　　（ＭＳＸ１　ＲＡＭ８Ｋ）
エスケープシーケンスを利用して、画面を上下にスクロールさせます。オールベーシックです。本文第１部第１章を参照してください。

ＲＡＭＣＨＫ．ＢＡＳ　　（ＭＳＸ１　ＲＡＭ３２Ｋ，一部機械語）
第２部第１章第２節で紹介したアルゴリズムを用いて、裏ＲＡＭを検索し、存在したスロットを表示します。拡張ＲＡＭカートリッジやＤＯＳ２カートリッジを差すと、そちらのほうのＲＡＭもちゃんと検索します。分かりやすくするために、かなりＢＡＳＩＣを使っていますが、実際に検索するときには、完全機械語で組んでください。

ＫＡＮＪＩＲＯＭ．ＢＡＳ　　（ＭＳＸ１　ＲＡＭ１６Ｋ）
第一、第二水準漢字ＲＯＭの存在をチェックし、さらに、入力されたＪＩＳコードに対応するフォントをスプライトで表示します。本文第２部第４章参照のこと。

ＳＴＯＰＢＡＴ．ＢＡＳ　　（ＭＳＸ１　６４Ｋ）
ＳＴＯＰＢＡＴ．ＣＯＭという名前のファイルをこしらえます。このＳＴＯＰＢＡＴ．ＣＯＭを実行すると、それ以後のバッチファイルの実行を停止します。長いバッチコマンドの途中で使ってみてください。例えば、「ＰＡＵＳＥ，ＰＡＵＳＥ，ＳＴＯＰＢＡＴ，ＰＡＵＳＥ，ＰＡＵＳＥ」というバッチファイルを作って実行させると、最後の二回のＰＡＵＳＥが実行されません。ＣＯＭＭＡＮＤ２．ＣＯＭ不可です。本文第３部第11章参照のこと。

ＭＯＵＳＥ．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
トリガーを押しながらマウス（またはトラックボール）を動かすと、線が引けます。本文第２部第６章参照のこと。

ＸＦＯＲＭＡＴ．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
ＳＣＲＥＥＮ２モードでメニューを表示してディスクをフォーマットします。ほんとにフォーマットしますから注意してください。プログラム中の「ＳＣＲＥＥＮ２」を５や６に書き換えると、それらの画面モードでもフォーマットできます。１ドライブの方は、ドライブＢを指定してみてください。２ドライブシミュレーションのメッセージを変えてあります。本文第３部第２章、第７章参照のこと。

ＥＸＴ．ＢＡＳ　　（ＭＳＸ１　６４Ｋ，一部機械語、裏ＲＡＭ使用）
裏ＲＡＭを利用して「ＣＡＬＬ　ＡＳＣＡＴ」という命令を拡張します。途中、ページ１のＲＡＭのスロット番号をきいてくるので、先ほどの「ＲＡＭＣＨＫ．ＢＡＳ」で調べたスロット番号をいれてください。「ＣＡＬＬ　ＡＳＣＡＴ」という命令が新設されて、使えるようになります。もっとも、ただメッセージを表示するだけですが。本文第１部第３章参照のこと。

ＭＯＮＤＩＳＫ．ＢＡＳ　　（ＭＳＸ１　１６Ｋ）
ディスクの内容を直接読み書きするプログラムです。下手に使うとディスクが使用不能になってしまうので、必ずコピーしたディスクで実行してください。使用方法は、まずドライブ番号をいれると、そのディスクのブートセクターを読み込んで表示して、コマンド待ちになります。ここで、セクター番号を入れてリターンキーを押すと、そのセクターを読み込みます。カーソルの上下で表示しきれなかった部分を表示します。何も入れずにリターンキーを押すと、データ書き換えモードになってアドレス（現在表示中のページの先頭アドレスを０とする）をきいてくるので入れます（０から１２７まで）。そしてデータ

を入れます（０から２５５まで)。アップデート（ディスクへの書き込み）はＳＥＬＥＣＴキーで行ないます。本文第１部第１章参照のこと。

ＥＸＴＢＩＯＳ．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
　現在接続されている拡張デバイスの一覧を表示します。ただし、漢字ドライバだけは調べません。どうしてこうなるのかというと、漢字ドライバだけはほかの拡張デバイスとは少し違った動作をするためです。本文第２部第７章参照のこと。機種によっては正常動作しないという報告もありますが、原因は未確認です。

ＲＳ２３２Ｃ．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
　ＲＳ－２３２Ｃの存在の有無を調べ、存在すれば、接続されているスロット番号と、ジャンプテーブルのスタートアドレスを表示します。モデムカートリッジの場合も表示します。さらに、オプション機能の有無についても表示します。２つ以上のＲＳ－２３２Ｃが存在した場合、すべてについて検索しますが、複数チャンネル対応でないＲＳ－２３２Ｃは、一度に２つ以上接続できません。本文第４部参照のこと。

ＭＳＸＪＥ．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
　ＭＳＸ－ＪＥが存在するかどうかを調べ、あれば、そのスロット番号と、ジャンプテーブルのスタートアドレス、オプション機能の有無を表示します。さらに、動作するのに必要なワークエリアの大きさも表示します。ＭＳＸ－ＪＥ内蔵機の場合は、スロット番号は大抵０－３になるでしょう。本文第５部参照のこと。

ＤＩＳＫＥＲＲ１．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
　ディスクを１セクターずつ読み、そのセクターに異常があれば、エラーの種類を表示します。試しにまだフォーマットしていないディスクなどを読ませると、エラーが山のように出ます。ディスクをいれずに走らせてもやはりエラーが起きます。市販ソフトの中には、特定のセクターに異常なデータを書き込んでいて、エラーが起きれば本物と判断するソフトがあります。そういうソフトをいれて走らせるとどこかでエラーが出るかも知れません。本文第３部第３章参照のこと。

ＤＩＳＫＥＲＲ２．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
　ＤＩＳＫＥＲＲ１．ＢＡＳとほとんど同じですが、今度はエラーが起きると、「RETRY, IGNORE」が選べるようになります。「ABORT」がないのは、それを選ぶとほんとにリセットがかかってしまうためです。ＤＯＳ２だと少し動作が変になることもあります。本文第３部第３章参照のこと。

ＰＨＹＤＲＩＶＥ．ＢＡＳ　　（ＭＳＸ１　３２Ｋ，一部機械語）
　第３部第８章のアルゴリズムを使って、現在接続されている物理ドライブを調べます。ＤＯＳ２のＲＡＭディスクが使用されているときは、ちゃんとＲＡＭディスクも物理ドライブとして表示します。

ＳＵＢＲＯＭ．ＣＯＭ　　（ＭＳＸ２　ＶＲＡＭ６４Ｋ，機械語）
　ＤＯＳ上からサブＲＯＭを呼びだし、ＢＥＥＰ音を鳴らしてＢＡＳＩＣのプロンプトを表示します。プログラム自体はかなりわけの分からないことをしています。このプログラムを読んで意味が分からなかった人は、サブＲＯＭをＤＯＳ上から使うのはやめたほうがいいかもしれません。本文第２部第２章参照のこと。

ＭＯＤＥＭ．ＣＯＭ　　（ＭＳＸ１　６４Ｋ，要ＭＳＸ－ＤＯＳ）
　第４部第５章で紹介したアルゴリズムを用いて、モデムが接続されているのか、ＲＳ－２３２Ｃが接続されているのかを調べます。

ＰＣＭＲＥＣ．ＣＯＭ　　（turboR　要ＭＳＸ－ＤＯＳ）
　第６部第３章第６節で紹介したアルゴリズムを使用して、まず約８kHzでサンプリングを行い、それを、８kHz，16kHz，４kHzで再生します。

ＳＲＡＭ．ＣＯＭ　　（ソニーＨＢ－Ｆ１ＸＤＪ，ＸＶ，ＨＢＩ－Ｊ１専用）
　ソニー製のＭＳＸ－ＪＥは、まれにＳ－ＲＡＭに異常なデータが書き込まれて、長音記号で始まる文節を変換しようとしたり、単語を登録、削除しようとしたときに暴走することがあります。これを回避するために、Ｓ－ＲＡＭの内容をクリアするプログラムです。ＭＳＸ－ＤＯＳが必要です。

ＴＡＮＧＯＣＨＯ．ＢＡＳ　（ＭＳＸ１　３２Ｋ，要漢字ドライバ、ＭＳＸ－ＪＥ）
ＭＳＸ－ＪＥのユーザー辞書を登録・削除・管理するためのユーティリティーです。ＭＳＸ－ＪＥを使うプログラムを作るときの参考にでもなるかな､と思って載せてみました。ただし、機械語部分のソースファイルがないので、逆アセンブルして解析してください。

この他にもちょっとしたプログラムがまぎれて入っているかも知れません。印刷したらそれっきり変更の利かない本と違って、ディスクはそれこそ売る直前まで変更が利くので、今これを書いている著者にも、どんなおまけプログラムがついてくるのか見当がつかないからです。出荷時期によって微妙に内容が変わっている可能性もあります。まあ、オートスタートさせれば何か説明が出るでしょう。（かどうかはわからないが）

☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆　はみ出しコラム　☆☆☆☆☆☆☆☆☆☆☆☆☆☆☆
漢字ドライバを積んでいて、ＭＳＸ－ＪＥも積んでいるときに、ＢＡＳＩＣでＣＡＬＬ ＫＡＮＪＩで漢字ドライバを起動している状態から、ＤＥＦＵＳＲ＝０：Ａ＝ＵＳＲ（０）とやってリセットをかけると、あら不思議、次に漢字ドライバを起動したときにＭＳＸ－ＪＥが切り離されて単漢字変換になってしまいます。
漢字ドライバにはいくつかのバージョンがあることが知られています。例えば、ＤＯＳ２カートリッジに搭載されている漢字ドライバよりも、ＭＳＸ２＋に搭載されている漢字ドライバのほうが、バージョンが上なのです。ＭＳＸ２＋版は、ＳＣＲＥＥＮ１０～１２もサポートするようになっています。漢字ドライバが二つ以上あるときは，当然新しいバージョンのほうが起動しますが、ユーザーにはちょっと分かりにくいのが困りものです。
メーカー別のＭＳＸ－ＪＥの動作の違いについて。実をいうと、メーカーによってＭＳＸ－ＪＥの機能には差があります。例えば、仮想端末の機能キーの割り当てがアスキーとソニーでまるで違います。単語登録のとき、アスキーのものは読みデータに英字は使えないようですが、ソニーのものは英字を使っても普通に動作するし、ちゃんと変換もします。
ＭＳＸ－ＪＥで特に違うのが使用するワークエリアの大きさです。アスキーのVJE-80Aは、５１２バイトでも動作しますが、ソニーのものは最低２５６０バイト必要です。ＭＳＸ－ＪＥ用ソフトを作るときは、少なくとも２５６０バイトのワークエリアを確保しなければならないことになっています。
アスキー、松下のＭＳＸ－ＪＥは、単語登録のとき単語データをページ３に置かないと動作しません。ソニー、ＨＡＬ研究所のものはページ２でも動作します。これは、単にアスキーのバグのようです。また、仮想端末のファンクション０９Ｈで、指定されたウィンドーが開けないとき、アスキーのものはディフォルトのウィンドーを返しますが、ソニーのものは開ける最大の大きさのウィンドーを返すようです。このように、メーカーによって動作がかなり違いますので、ＭＳＸ－ＪＥ対応ソフトを作るときは、複数の機種で動作確認をしたほうがいいでしょう。著者の体験で言うと、ソニーで動いたので安心していたところ、松下で動作しなかったということがありました。
ＭＳＸ－ＪＥには、現在、アスキー、松下、三洋が採用しているＶＪＥ系のものと、ＨＡＬ研究所、ソニーが採用しているＥＧ系の大きく分けて２種類があります。前者はＶＪＥという１６ビット用のフロントエンドを８ビット用にコンバートしたもの、後者はエルゴソフトのＥＧブリッジというやはり１６ビット用のフロントエンドをコンバートしたものです。動作チェックはこれら２系統のそれぞれ最低１機種で行ったほうがいいでしょう。
デバイスイネーブルについて。Ｉ／ＯのＦ５Ｈに値を書き込むことによって、内蔵デバイスを切り離すことができます。試しに漢字ＲＯＭ内蔵機でＢＡＳＩＣから「ＯＵＴ＆ＨＦ５，＆ＨＦＣ」とやると、なんと漢字ＲＯＭが切り離されてしまいます。ここで漢字ＢＳＩＣなんかを立ちあげると、１６進コードでずらずらと漢字が表示されます。
ＤＯＳ２ではルートディレクトリーには１１２しかファイルが作れませんが、（ディレクトリーも含む）サブディレクトリーにはディスク容量の許す限り無数のファイルが作れます。著者は、１４０２ファイル作ったところで疲れてそれ以上作るのをやめました。
漢字ＲＯＭは内蔵のものより外付けのものが優先的に使用されます。
ＤＯＳ２のＲＡＭディスクは１セクター５１２バイトで１クラスターが１セクターです。容量が２Ｍバイトを越えて４Ｍバイトまでは１クラスター２セクターになります。
ソニー製ディスクドライブは、ＤＯＳ１上で２ＤＤ，８セクターのディスクを読み書きする際、正常動作をしないことがわかりました。松下の外付けドライブの中にも、ソニー製のもの（ＦＳ－ＦＤ１）が存在します。注意してください。
松下ＦＳ－Ａ１ＷＸ／ＦＸ／ＷＳＸでは、「OUT64,8:OUT65,0」とやるとＣＰＵが６ＭＨｚになります。このままソフトウェアリセットすると、市販ソフトも高速で動作します。

第８部　　アペンディックス

◆

ここでは、「ＭＳＸ２テクニカルハンドブック」に載っていないＢＩＯＳ、ワークエリア、フック、その他有用と思われるコード表などについて一覧表形式で紹介します。

第１章　ＢＩＯＳ

ここでは、知られていないＢＩＯＳについて説明します。エントリについては、次のように記述します。

ラベル名（アドレス）
機能　　機能解説および注意
入力　　呼びだし時に必要なパラメータ
出力　　リターンされるパラメータ
レジスタ　内容が破壊されるレジスタ

ＭＳＸ２，ＭＳＸ２＋になってから追加されたものは、「機能」のところに説明があります。そのほかはＭＳＸ１からあるものです。

●ＭＡＩＮ－ＲＯＭ

ＰＨＹＤＩＯ（０１４４Ｈ）
機能　ディスクをセクター単位で高速で読み書きする。ディスクがないときは何もせずに戻る。いわゆる隠し（未公開）ＢＩＯＳ。しかし、多くのアプリケーションが使用しているので、使ってもあまり問題はないと思われる
入力　Ｃｙフラグ，セットなら書き込み
　　リセットなら読み込み
　　Ａにドライブナンバー（０はＡ、１はＢなど）
　　Ｂにセクターの数
　　Ｃにメディア ＩＤ
　　ＤＥに先頭のセクター番号
　　ＨＬにＤＭＡアドレス
出力　Ｃｙフラグリセットなら成功
　　セットなら失敗，さらに
　　Ａにエラーコード
　　Ｂに読み残した（書き残した）セクター数
レジスタ　すべて

ＦＯＲＭＡＴ（０１４７Ｈ）
機能　ディスクをフォーマットする。ただし、ユーザーがフォーマットするときはここではなくマスタースロットの４０２５Ｈを利用する

ＲＤＲＥＳ（０１７ＡＨ）
機能　リセットの状態を調べる。ＭＳＸ２＋になってから追加されたもの
入力　なし
出力　Ａのビット７が０ならハードウェアリセット，
　　１ならソフトウェアリセット
レジスタ　なし

ＷＲＲＥＳ（０１７ＤＨ）
機能　リセットの状態を書き込む。ＭＳＸ２＋になってから新設されたもの。０番地ジャンプする直前に必ず利用する。０１７ＡＨでリセットの状態を読み、８０ＨでＯＲをとってからこのＢＩＯＳで書き込み、それから０番地ジャンプする（詳しくは第２部第２章第５節を参照のこと）
入力　Ａにリセットの状態
出力　なし
レジスタ　なし

ＣＨＧＣＰＵ（０１８０Ｈ／ＭＡＩＮ）
　機能　　ＣＰＵのモードを切り換える。turboRになってから新設されたもの。Ａレジスタのbit7が立っていれば、どちらのＣＰＵが動いているかを示すＬＥＤが変化する。なお、Ｒ８００ ＤＲＡＭモードとは、メインＲＯＭ、サブＲＯＭ、漢字ドライバをＲＡＭに転送して高速化するモード。そのかわり、ＲＡＭが６４Ｋバイト減少する
　入力　　Ａのbit7を立てればＬＥＤを変化させる。bit1,0がそれぞれ0,0ならＺ８０モード。0,1ならＲ８００モード。1,0ならＲ８００ ＤＲＡＭモード。bit6～bit2は０を指定する
　出力　　なし
　レジスタ　ＡＦ

ＧＥＴＣＰＵ（０１８３Ｈ／ＭＡＩＮ）
　機能　　動作中のＣＰＵのモードを調べる。turboRで新設されたもの
　入力　　なし
　出力　　Ａが０ならＺ８０モード
　　１ならＲ８００モード
　　２ならＲ８００ ＤＲＡＭモード

ＰＣＭＰＬＹ（０１８６Ｈ／ＭＡＩＮ）
　機能　　ＰＣＭの再生。turboRで新設されたもの。Ａレジスタのbit7でＰＣＭのデータがＲＡＭにあるのかＶＲＡＭにあるのかを指定する。サンプリング周波数のうち、15.75kHzはＣＰＵがＲ８００ ＤＲＡＭモードのときのみ有効。それ以外のときはエラーになる。なお、Ｄ，Ｅレジスタはデータがあるときのみ意味を持つ
　入力　　Ａのbit7が０ならデータはメインＲＡＭ、１ならＶＲＡＭ．bit6～bit2は０。bit1,0がそれぞれ0,0なら周波数15.75kHz、0,1なら7.875kHz、1,0なら5.25kHz1,1なら3.9375kHz
　ＥＨＬでＰＣＭデータの開始番地
　ＤＢＣでデータの長さ（バイト数）
　出力　　Ｃｙリセットなら正常終了
　セットなら異常終了でさらに、
　Ａに異常の原因。０なら周波数指定の誤り。１ならCTRL+STOPによる中断
　ＥＨＬに中断番地

ＰＣＭＲＥＣ（０１８９Ｈ／ＭＡＩＮ）
　機能　　ＰＣＭの録音。turboRで新設されたもの。トリガーレベルは録音を始めるきっかけとなる音の大きさをいう。これが０ならただちに録音が開始される。bit2を１にすると、録音データを圧縮して録音する
　入力　　Ａのbit7が０ならデータはメインＲＡＭ、１ならＶＲＡＭ。bit6～bit3はトリガーレベル。bit2が０ならデータ圧縮なし、１ならあり。bit1,0は周波数指定で、指定の仕方はＰＣＭＰＬＹと同じ
　出力　　Ｃｙリセットなら正常終了
　セットなら失敗でさらに
　Ａに異常の原因。０なら周波数指定の誤り。１ならCTRL+STOPによる中断

第２章　システムワークエリア

ここでは、あまり知られていない公開システムワークエリアについて説明します。なお、ディスクのワークエリアおよびフックについては、第５章以下で紹介します。表記は次のとおりです。長さはバイト数です。

ラベル名（アドレス，長さ）
　初期値、内容、使用目的

ＲＯＭＩＤ　（００２ＤＨ，１）メインＲＯＭ
　内容　　ＢＡＳＩＣのバージョンナンバー。ここの値が０ならＭＳＸ１、１ならＭ

ＳＸ２、２ならＭＳＸ２＋、３ならturboR
ＲＧ０ＳＡＶ（Ｆ３ＤＦＨ，１）
：　　：　　：
ＲＧ７ＳＡＶ（Ｆ３Ｅ６Ｈ，１）
内容　　ＶＤＰレジスタ０～７を保存しておく。ユーザーが参照しても構わないが、プログラム中でＶＤＰに直接アクセスするときは、ここに値を書き込むようにしておかないと、読み込むときに意味のある値は帰ってこない。ＢＩＯＳを使って書き込んだときは、ここにＢＡＳＩＣ側が値を書き込むので、ユーザーが書き込む必要はない
ＢＵＦ　（Ｆ５５ＥＨ，２５８）
内容　　ＢＡＳＩＣインタープリタ用のバッファだが、ＢＡＳＩＣはここの値が保存されることは期待していないので、機械語プログラムが一時的に使っても構わない
ＭＯＤＥ　（ＦＡＦＣ，１）
内容　　bit7　０：ひらがな、１：カタカナ　（ローマ字変換用）
bit6　０：第２水準漢字ＲＯＭなし、１：あり　（ＭＳＸ２＋で追加）
bit5　０：SCREEN10、１：SCREEN11　（ＭＳＸ２＋でのＲＧＢ処理用）
bit4　０：クリッピングしない、１：する　（ＭＳＸ２＋で追加）
BIT3　０：マスクしない、１：する　（SCREEN0～3のＶＲＡＭアドレス）
bit2,1　ＶＲＡＭ容量。bit2,1がそれぞれ0,0なら１６Ｋ、0,1なら６４Ｋ、1,0なら１２８Ｋ
bit0　０：ローマ字変換しない、１：する
ＮＯＲＵＳＥ（ＦＡＦＤ，１）
内容　　漢字ドライバが使用。bit7:１にするとグラフィック、文字混在モード、bit6:１にするとＳＨＩＦＴ＋カーソルで画面が上下スクロールする、bit5,4:内部で使用、bit3-0:ＶＤＰのロジカルオペレーションが入っている
ＢＡＳＲＯＭ（ＦＢＢ１Ｈ，１）
内容　　本来は、プログラムがＲＡＭにあるのかＲＯＭ上にあるのかが書いてあるが、ここに０以外を書き込むと、ＢＡＳＩＣ上でＣＴＲＬ＋ＳＴＯＰを利かなくすることができる
？？？？？？（ＦＣＢ１Ｈ，１）
内容　　Ｉ／ＯポートＡ７Ｈ番地の値の保存場所。turboRで追加
ＥＸＰＴＢＬ（ＦＣＣ１Ｈ，４）
内容　　ＦＣＣ１ＨはメインＲＯＭのスロット番号
ＦＣＣ２Ｈから３バイトはそれぞれスロット１，２，３が拡張されているかどうか。拡張されていれば８０Ｈ、そうでなければ００Ｈが入っている
ＸＸＸＸＸＸ（ＦＤ０９Ｈ，１）
内容　　ここのビット５をセットするとＢＡＳＩＣのＲＡＭディスク機能が使えなくなる。ここのビット６が立っているとＢＡＳＩＣのＲＡＭディスクが使われている
Ｈ．ＰＨＹＤ（ＦＦＡ７Ｈ，５）
内容　　ここの先頭番地がＣ９Ｈならば、ディスクは接続されていない。Ｃ９Ｈ以外なら、ディスクが接続されている。ＢＩＯＳのＰＨＹＤＩＯから直接呼ばれる。そこで、高速化を必要とするソフトでは、直接ここをコールしている場合もある
ＲＧ８ＳＡＶ（ＦＦＥ７Ｈ，１）
：　　：　　：
ＲＧ２３ＳＡ（ＦＦＦ６Ｈ，１）
内容　　ＶＤＰレジスタ８～２３の保存場所。そのほかはＲＧ０ＳＡＶなどと同じ。ＭＳＸ２になってから追加されたもの
ＭＩＮＲＯＭ（ＦＦＦ７Ｈ，１）
内容　　メインＲＯＭのスロットアドレスが入っているが、ユーザーはここではなくＦＣＣ１Ｈ番地のほうを使うこと。ＭＳＸ２で追加されたもの
ＲＧ２５ＳＡ（ＦＦＦＡＨ，１）
：　　：　　：
ＲＧ２７ＳＡ（ＦＦＦＣＨ，１）
内容　　新設ＶＤＰレジスタ２５～２７の保存場所。そのほかはＲＧ０ＳＡＶなどと同じ。ＭＳＸ２＋で追加されたもの

第３章　ディスクインターフェイスＲＯＭ内ルーチン

ディスク操作をするときにマスタースロットの特定の番地をコールするといくつかのディスク操作が行なえます。マスタースロットのスロット番号はＲＡＭのＦ３４８Ｈ番地に書いてあります。これらの機能のうち、ユーザー向けに公開されたものは４０２５ＨのＦＯＲＥＮＴと４０２９ＨのＭＴＯＦＦだけですが、恐らくハードウェアメーカー向けには公開されているものと思われます。市販ソフトの中にもこれらのルーチンを直接使っているものがあります。ただし、ＤＯＳ２上でこれらの機能を使おうとすると、大抵の場合何もせずにエラーが出て返ってきます。turboRではＤＯＳ２上でもこれらの機能は使えます。
これらの機能及びラベル名は著者が独自に調査したもので、実際のものとは異なることがあります。ご了承ください。

ＤＳＫＩＯ　（４０１０Ｈ）
機能　　ディスクをセクタ単位で直接読み書きする。レジスタの設定などはＰＨＹＤＩＯと同じ。未公開ルーチン。ＤＯＳ２ではエラーになる。ドライブ番号は、そのディスクインターフェイスが制御しているドライブ、つまり、０または１のみが指定可能

ＤＳＫＣＨＧ（４０１３Ｈ）
機能　　ディフォルトドライブを変更する。あらかじめＧＥＴＤＰＢでＤＰＢをセットしておく必要がある。このファンクションでは、存在しないドライブを指定すると暴走する。また、システム側ではこのファンクションでドライブが変えられても、ディフォルトドライブは元のままとして動作をする。つまり、ディフォルトがＡドライブなのにＡドライブにディスクを入れるようにというメッセージが出たりする。未公開。ＤＯＳ２では不可
入力　　Ａにドライブ番号（０～１）
　　Ｂに０
　　Ｃにメディア ＩＤ
　　ＨＬにＤＰＢの先頭アドレス
出力　　不明

ＧＥＴＤＰＢ（４０１６Ｈ）
機能　　ＨＬで指定されたアドレスを先頭番地としてそこから１９バイトにＤＰＢをセットする。このファンクションでは、ＤＰＢの最後の２バイトのＦＣＢの先頭アドレスは返らない。そのため、ＤＰＢの長さが通常のものより２バイト短い。未公開。ＤＯＳ２では不可
入力　　Ａにドライブ番号（０～１）
　　ＢにＦＡＴの先頭の値（Ｃレジスタと同じ）
　　Ｃにメディア ＩＤ
　　ＨＬにＤＰＢの先頭アドレス
出力　　ＨＬで指定されたアドレスにＤＰＢが入る

ＣＨＯＩＣＥ（４０１９Ｈ）
機能　　ＨＬレジスタにフォーマット時のメニューの先頭アドレスを返す。未公開。フォーマットが選べない機種（一種類のフォーマットしかできない機種）やＤＯＳ２では、このアドレスに００Ｈが入っている
入力　　なし
出力　　ＨＬにメニューの先頭アドレス。メニュー文字列の最後は００Ｈで終了する
レジスタ　すべて

ＤＳＫＦＭＴ（４０１ＣＨ）
機能　　指定されたディスクを、直接フォーマットする。Ａレジスタでフォーマットの種類を指定するのだが、機種によってメニューの順番が違うことがあるので、ユーザーがここを直接コールしてはならない。例えば、Ａレジスタに２を入れた場合、１ＤＤ．９セクターフォーマットするディスクドライブもありうるし、２ＤＤ．９セクターフォーマットするドライブもありうるし、２ＨＤフォーマットするドライブもありうる。未公開。ＤＯＳ２不可。なお、このファンクションではＤＯＳ１フォーマットを行なう
入力　　Ａにメニューで選んだフォーマットの種類。（１～９まで）１種類のフォー

マットしかできない機種では０を入れる。
Ｄにドライブナンバー（０～１）
ＨＬにワークエリアの開始アドレス
ＢＣにワークエリアの長さ。８ＫＢｙｔｅ取る
出力　Ｃｙ、リセットなら正常終了
セットなら異常終了、さらに、
Ａにエラー番号
レジスタ　すべて

？？？？？？（４０１ＦＨ）
機能　ドライブを初期化（停止）する。未公開。ＤＯＳ２では何もしない。
ＭＴＯＦＦはすべてのドライブを停止させるが、このファンクションではそのディスクインターフェイスが制御しているドライブだけを停める
入力　なし
出力　なし

？？？？？？（４０２２Ｈ）
機能　ディスクワークエリアを初期化する。ディスクのブートセクターで直接呼ばれる。ユーザーが使っても意味がない。未公開
入力　なし
出力　なし

ＦＯＲＥＮＴ（４０２５Ｈ）
機能　ディスクをフォーマットする。ワークエリアは開始アドレスが８０００Ｈより大きく、ＴＰＡにすべてが収まっている必要がある
入力　ＨＬにワークエリアの開始アドレス
ＢＣにワークエリアの長さ。８Ｋバイト取る
出力　Ｃｙリセットなら成功、セットなら失敗でＡにエラー番号
レジスタ　すべて

ＭＴＯＦＦ　（４０２９Ｈ）
機能　回転したままのドライブを止める。ただし、古い型のドライブでは何もせずに戻ってくることがある
入力　なし
出力　なし
レジスタ　すべて

ＧＥＴＳＬＴ（４０２ＤＨ）
機能　Ａレジスタに自分自身のいるスロットのスロット番号を返す。何の意味があってこのようなルーチンがあるのかは不明。ＤＯＳ２では、ＤＯＳ２のスロット番号が入る
入力　なし
出力　Ａに自分自身のいるスロットのスロット番号
レジスタ　すべて

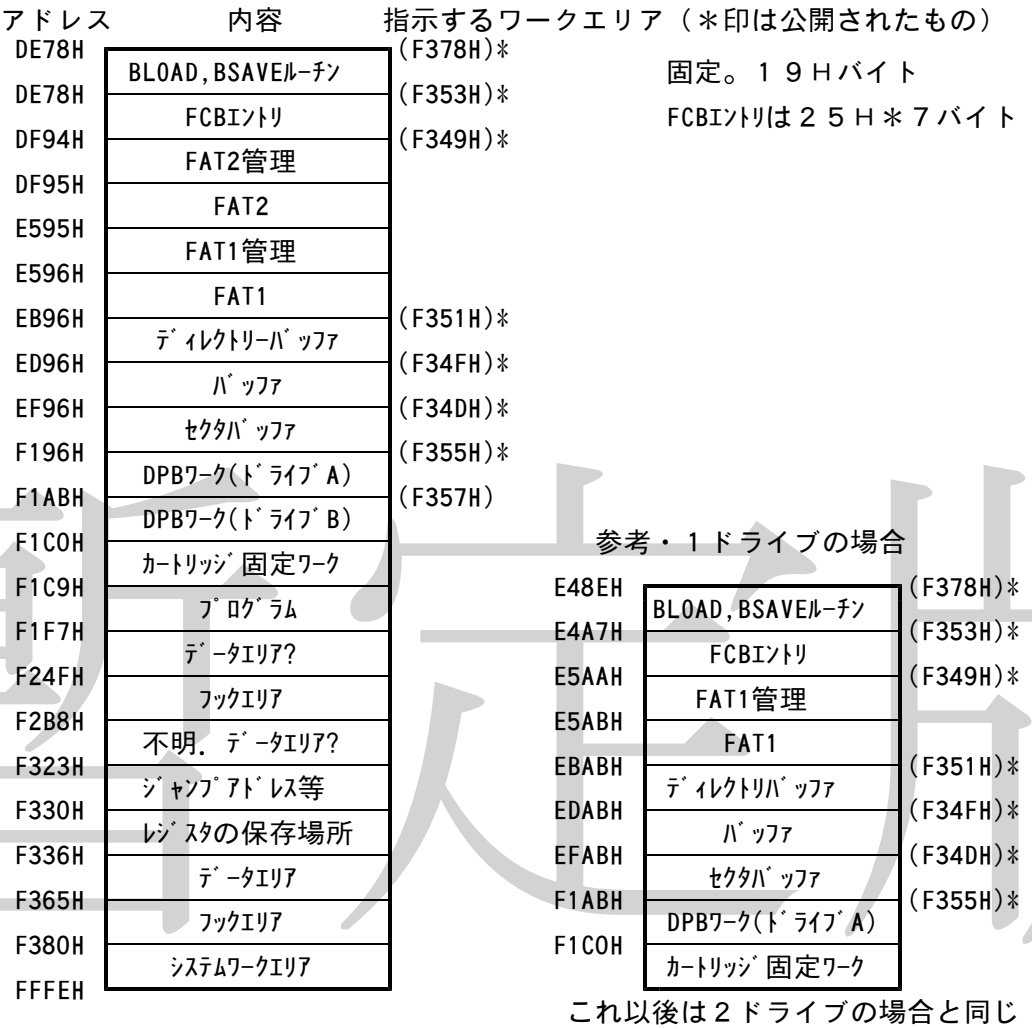
？？？？？？（４０３０Ｈ）
機能　ＨＬレジスタにＦ３４ＢＨ番地の内容を返す。Ｆ３４ＢＨはＤＯＳのシステム本体の開始開始番地。初期化のときに使われる。ＢＡＳＩＣでは無意味
入力　なし
出力　ＨＬにＦ３４ＢＨ番地の内容
レジスタ　なし

以上です。４０１０Ｈ～４０１ＦＨまでのエントリーでは、ドライブ番号はそのディスクインターフェイスがサポートしているドライブ、すなわち０または１が指定可能です。（２ドライブシミュレーションが行なわれていないときは、０しか指定できない）
　これらの機能の中には、ＣＨＧＤＲＶであらかじめドライブを変更しておかないと動作が変になる機能もあります。

第４章　ディスクワークエリアの内部構造

ディスクワークエリアのメモリーマップを掲載します。なお、それぞれの番地はソニーＨＢ－Ｆ１ＸＤのものです。

これらはあくまでも解析した結果を掲載しています。型番によってはこれとは違う結果が出ることもありえます。



BLOAD,BSAVEﾙｰﾁﾝ、FCBｴﾝﾄﾘは、ＤＯＳでは存在しません。また、ＤＯＳ２では、ワークエリアの配置はこれとは全く異なります。

ＤＯＳ１の場合、ディスクワークエリアはドライブが１つ増えるごとに１５５８バイト増加します。（内訳は、ＦＡＴに５１２＊３バイト、ＦＡＴ管理に１バイト、ＤＰＢに２１バイト）ディスクインターフェイスが増えたときはさらに９バイト（カートリッジ固定ワークの９バイト）増加します。

ＤＯＳ２の場合は、ドライブが１つ増えるたびにディスクワークエリアは２１バイト増加します。（ＤＰＢ用の２１バイト）ディスクインターフェイスが増えたときはさらに９バイト（カートリッジ固定ワークの９バイト）増加します。ＤＯＳ２はＦＡＴをメモリー上には全部置かないので、ワークエリアは劇的に減少します。

カートリッジ固定ワークの大きさは、メーカー、機種によって異なるので注意してください。例えば、松下の機種では、２６バイトです。つまり、松下の機種は、ソニーのものより、ユーザーの使えるエリアが小さくなります。

ＦＡＴ管理の１バイトは、ディスク上のＦＡＴがまだメモリ上に転送されていないとき２５５、メモリ上のＦＡＴだけを変更してまたディスク上に書き込んでいないとき１、ディスクとメモリ上の内容が同じのとき０になります。（公開資料）

第５章　未公開ディスクワークエリア

　未公開ディスクワークエリアについて説明します。なお、ＤＯＳ２ではここに挙げたもののうちいくつかは全く違う用途に使用されています。ご注意ください。

Ｆ１Ｅ２Ｈ（ルーチン）
　NOT READY処理用、ルーチン
Ｆ１Ｅ８Ｈ（ルーチン）
　ＨＬレジスタに書いてある番地に書いてある番地にジャンプ
Ｆ２４１Ｈ，１
　カレントドライブナンバー
Ｆ２４６Ｈ，１
　カレントドライブナンバー
Ｆ２４７Ｈ，１
　ディフォルトドライブナンバー
Ｆ２５５Ｈ，３
　デバイス名チェック用フック。新たなデバイスを接続するときのため
Ｆ２５ＢＨ，３
　ファイル名チェック用フック？
Ｆ２６１Ｈ，３
　ファイル名セット用フック？
Ｆ２６４Ｈ，３
　ファイルオープン用フック
Ｆ２６７Ｈ，３
　ＦＡＴリード用フック
Ｆ２６ＡＨ，３
　ＤＰＢセット用フック
Ｆ２６ＤＨ，３
　ファイルクローズ用フック
Ｆ２７０Ｈ，３
　論理セクターリード用フック
Ｆ２７３Ｈ，３
　ディスクエラー処理用フック
Ｆ２７６Ｈ，３
　ディスクのディレクトリーエリア書き換えフック
Ｆ２７９Ｈ，３
　論理セクターライト用フック
Ｆ２Ｅ１Ｈ，１
　カレントドライブナンバー
Ｆ３０９Ｈ，２
　カレントドライブのＤＰＢ先頭アドレス
Ｆ３３ＦＨ，１
　ＣＴＲＬキーが押されたかどうかのチェック。ＤＯＳが内部で使用する。押されれば２が入る
Ｆ３４０Ｈ，１
　ＡＵＴＯＥＸＥＣ．ＢＡＳ，ＡＵＴＯＥＸＥＣ．ＢＡＴ実行用フラグ。通常はＦ３Ｈが入っている
Ｆ３４５Ｈ，１
　不明
Ｆ３４７Ｈ，１
　接続されている論理ドライブの数。最高８
Ｆ３４ＢＨ，２
　ＤＯＳシステム本体の開始アドレス。ここから上の番地にＤＯＳが、ここから下の番地にＡドライブ用のワークエリアが置かれる。ＢＡＳＩＣでは意味がない
Ｆ３５７Ｈ～Ｆ３６４Ｈ，各２
　ドライブＢ～ＨのＤＰＢのアドレス
Ｆ３６５Ｈ，３
　フック。基本スロット選択レジスタの状態を読む
Ｆ３６８Ｈ，３
　ＤＯＳが使用するフック。ページ１をマススロットに切り換える

Ｆ３６ＢＨ，３
　ＤＯＳが使用するフック。ページ１をメインＲＡＭに切り換える
Ｆ３６ＥＨ，３
　ＤＯＳが利用するフック。ページ１をメインＲＡＭに切り換え、ＬＤＩＲ
Ｆ３７１Ｈ，３
　ＡＵＸデバイス（入力）拡張用フック。通常はＡレジスタに１ＡＨをいれて帰る
Ｆ３７４Ｈ，３
　ＡＵＸデバイス（出力）拡張用フック
Ｆ３７７Ｈ，３
　ＢＡＳＩＣが使うフック。ＢＬＯＡＤルーチンが使用する。公開ワーク
Ｆ３７ＡＨ，３
　ＢＡＳＩＣが使うフック。ＢＳＡＶＥルーチンが使用する
Ｆ３７ＤＨ，３
　ＤＯＳのファンクションコール。公開ルーチン
ＦＢ２９Ｈ，３
　第１ディスクインターフェイスＲＯＭ用の、割り込み処理ルーチンのアドレス。３バイトに、スロット番号、アドレス下位、アドレス上位の順で入っている。割り込み処理を新たに拡張しないインターフェイスの場合は３バイトとも０で埋められている
ＦＢ２ＣＨ，３～ＦＢ３２Ｈ，３
　それぞれ第２～第４ディスクインターフェイスＲＯＭ用割り込みルーチンのアドレス

第６章　公開ディスクワークエリア

　ここでは、公開ディスクワークエリアについて説明します。これらは安心してソフト内で参照することができます。もちろんディスクがつながっている場合のみ有効です。

Ｈ．ＰＲＯＭ（Ｆ２４ＦＨ，３）
　内容　　２ドライブシミュレーションメッセージ表示フック
ＤＩＳＫＶＥ（Ｆ３２３Ｈ，２）
　内容　　ディスクエラーが起きたときジャンプするアドレスの入っているアドレス
ＢＲＥＡＫＶ（Ｆ３２５Ｈ，２）
　内容　　ＣＴＲＬ＋Ｃが押されたときコールされるアドレスが入っているアドレス
ＲＡＭＡＤ０（Ｆ３４１Ｈ，１）
　内容　　ページ０のＲＡＭのスロット番号
ＲＡＭＡＤ１（Ｆ３４２Ｈ，１）
　内容　　ページ１のＲＡＭのスロット番号
ＲＡＭＡＤ２（Ｆ３４３Ｈ，１）
　内容　　ページ２のＲＡＭのスロット番号
ＲＡＭＡＤ３（Ｆ３４４Ｈ，１）
　内容　　ページ３のＲＡＭのスロット番号
？？？？？？（Ｆ３４６Ｈ，１）
　内容　　０以外の値をいれると、ＢＡＳＩＣ上からＣＡＬＬ　ＳＹＳＴＥＭでＤＯＳに行けるようになる
ＭＡＳＴＥＲ（Ｆ３４８Ｈ，１）
　内容　　マスターディスク（ドライブＡ）を制御しているディスクインターフェイスのスロット番号。ＤＯＳ２を差しているときはＤＯＳ２のディスク制御機能のある部分のスロット番号
ＨＩＭＳＡＶ（Ｆ３４９Ｈ，２）
　内容　　ディスクインターフェイスワークエリアの先頭番地
ＳＥＣＢＵＦ（Ｆ３４ＤＨ，２）
　内容　　ディスクドライバ用セクターバッファの先頭アドレス
ＢＵＦＦＥＲ（Ｆ３４ＦＨ，２）
　内容　　ＤＯＳ汎用セクターバッファの先頭アドレス
ＤＩＲＢＵＦ（Ｆ３５１Ｈ，２）
　内容　　ＤＯＳディレクトリー用セクターバッファの先頭番地
ＦＣＢＢＡＳＥ（Ｆ３５３Ｈ，２）
　内容　　ＦＣＢエントリの先頭アドレス
ＤＰＢＬＩＳＴ（Ｆ３５５Ｈ，２）

内容　　ＤＰＢテーブルの先頭アドレス
ＢＬＤＣＨＫ＋１（Ｆ３７８Ｈ，２）
内容　　ＢＬＯＡＤルーチンの先頭アドレス
ＤＲＶＴＢＬ（ＦＢ２１Ｈ，８）
内容　　第１ディスクインターフェイスＲＯＭに接続されている論理ドライブの数、第１ディスクインターフェイスＲＯＭのスロット番号、第２ディスクインターフェイスＲＯＭが制御している論理ドライブ数、……の順に第４ディスクインターフェイスＲＯＭのスロット番号までの値がそれぞれ１バイトずつで入っている

第７章　エスケープシーケンス表

＜ＥＳＣ＞Ａ　カーソルを上に移動
＜ＥＳＣ＞Ｂ　カーソルを下に移動
＜ＥＳＣ＞Ｃ　カーソルを右に移動
＜ＥＳＣ＞Ｄ　カーソルを左に移動
＜ＥＳＣ＞Ｅ　画面クリア
＜ＥＳＣ＞Ｈ　カーソルをホームポジションに移動
＜ＥＳＣ＞ｊ　画面クリア
＜ＥＳＣ＞Ｊ　カーソル位置から画面の終わりまでをクリア
＜ＥＳＣ＞Ｋ　カーソル位置から行の終わりまでをクリア
＜ＥＳＣ＞Ｉ　カーソルのある行をクリア（小文字のＬ）
＜ＥＳＣ＞Ｌ　カーソルのある行に１行挿入
＜ＥＳＣ＞Ｍ　カーソルのある行をクリア
＜ＥＳＣ＞ｘ４　カーソルを大きな形にする
＜ＥＳＣ＞ｘ５　カーソル消去
＜ＥＳＣ＞ｙ４　カーソルを小さな形にする
＜ＥＳＣ＞ｙ５　カーソル表示
＜ＥＳＣ＞Ｙ＜Ｙ座標＋２０Ｈ＞＜Ｘ座標＋２０Ｈ＞
　　カーソルを（Ｘ，Ｙ）位置に移動

第８章　ＭＳＸ－ＤＯＳ２コマンドインデックス

ＡＳＳＩＧＮ　[d:[d:]]
論理ドライブを物理ドライブに割り当てる。
例：「ＡＳＳＩＧＮ　Ａ：　Ｂ：」ドライブＡを論理ドライブＢに割り当てる

ＡＴＤＩＲ　＋Ｈ｜－Ｈ　[／Ｈ][／Ｐ]　複合ファイルスペック
ディレクトリの属性を変更する。／Ｈを指定すると不可視属性ディレクトリも対象となる。／Ｐはページごとの表示停止。＋Ｈを指定するとファイルは不可視になり、－Ｈで解除となる。

ＡＴＴＲＢ　＋Ｈ｜－Ｈ｜＋Ｒ｜－Ｒ[／Ｈ][／Ｐ]　複合ファイルスペック
ファイルの属性を変更する。＋Ｒで読み込み専用になり、－Ｒで解除。それ以外はＡＴＲＤＩＲと同じ。

ＢＡＳＩＣ　[プログラム名]
ディスクＢＡＳＩＣを起動する。

ＢＵＦＦＥＲＳ　[数値]
システムが使用するディスクバッファの量を指定する。

ＣＤ　（ＣＨＤＩＲの略）
ＣＨＤＩＲ　[d:][パス]
カレントディレクトリを表示・変更する。

ＣＨＫＤＳＫ　[d:][／Ｆ]
ディスクの使用状況を検査する。／Ｆを指定すると、内部に破壊された領域があるときにそこを修復するかどうか聞いてくる。

ＣＬＳ
画面を消去する。

ＣＯＭＭＡＮＤ２　[コマンド]
ＣＯＭＭＡＮＤ２を起動し、指定されたコマンドを実行する。ＥＸＩＴコマンドで元のＣＯＭＭＡＮＤ２に戻る。

ＣＯＮＣＡＴ　[／Ｈ][／Ｐ][／Ｂ][／Ｖ][／Ａ]　複合ファイルスペック　複合ファイルスペック
１番目のファイルスペックに該当するファイルがすべて連結され、２番目の複合ファイルスペックと同じ名前のファイルが作成される。／Ｈで不可視属性ファイルも対象とする。／Ｐは画面表示一時停止。／Ｂはバイナリファイルとして扱う。／Ａはアスキーファイルとして扱う（ディフォルト）。／Ｖは書き込みベリファイを行なう。
例：「ＣＯＮＣＡＴ　＊．ＤＯＣ　ＰＲＩＮＴ．ＴＸＴ」
　　「ＣＯＮＣＡＴ　ＦＩＬＥ１．ＴＸＴ＋ＦＩＬＥ３．ＴＸＴ　ＮＥＷ．ＴＸＴ」

ＣＯＰＹ　[／Ａ][／Ｈ][／Ｔ][／Ｖ][／Ｐ][／Ｂ]　複合ファイルスペック　複合ファイルスペック
１番目のファイルスペックに該当するファイルが２番目のファイルスペックの形で複写される。／Ｔで出力ファイルに複写時の日付、時刻が設定される。それ以外のオプションはＣＯＮＣＡＴと同じ。

ＤＡＴＥ　[日付]
現在の日付を表示・設定する。（ＴＩＭＥも参照）

ＤＥＬ　[／Ｈ][／Ｐ]　複合ファイルスペック
ファイルを削除する。／Ｈで不可視属性ファイルも対象とする。／Ｐは画面出力一時停止。

ＤＩＲ　[／Ｈ][／Ｗ][／Ｐ][複合ファイルスペック]
ディスクのファイル名を表示する。／Ｈで不可視属性も対象とする。／Ｗでワイド形式で表示。／Ｐは画面表示一時停止。

ＤＩＳＫＣＯＰＹ ［d:［d:］］［／Ｘ］［／Ｓ］
ディスクをまるごと複写する。／Ｘでメッセージ出力停止。／Ｓでブートセクターもコピーする（／ＳオプションはVer.2.30で追加）。

ＥＣＨＯ ［テキスト］
指定されたテキストを表示する。

ＥＲＡ （ＤＥＬと同じ）
ＥＲＡＳＥ （ＤＥＬと同じ）

ＥＸＩＴ ［数字］
ＣＯＭＭＡＮＤ２を終了して元のプログラムに戻る。数値はエラー番号。

ＦＩＸＤＩＳＫ ［d:］［／Ｓ］
ディスクをＤＯＳ２用にする。／Ｓで完全なＤＯＳ２にする。／Ｓを指定しないとディスク情報に誤りのあるＤＯＳ１でフォーマットされたディスクのみ修正される。

ＦＯＲＭＡＴ ［d:］
ディスクをフォーマットする。

ＨＥＬＰ ［項目］
項目で指定される項目の説明が表示される。

ＩＦ ［ＮＯＴ］ 条件 コマンド
条件が真でコマンドが実行される。条件は以下の２種類。
例：「ＩＦ ＥＸＩＳＴ ＡＢＣ．ＢＡＴ ＥＣＨＯ ＡＢＣ」ＡＢＣ．ＢＡＴというファイルがあるとＡＢＣと表示する。
例：「ＩＦ ％ＤＥＦ％＝＝Ｑ ＥＣＨＯ Ｑ－ＳＥＴ」環境変数ＤＥＦの内容が「Ｑ」ならＱ－ＳＥＴと表示する。
Ver.2.31で追加

ＫＭＯＤＥ 数値 | ＯＦＦ
ＫＭＯＤＥ ［数値 | ＯＦＦ］／Ｓ［d:］
漢字モードの設定を行なう。数値は０～３が有効。ＯＦＦを指定するとＡＮＫモードとなる。／Ｓオプションを付けるとディスクに指定したモードの情報が書き込まれ、以後、そのディスクを起動するとその漢字モードで起動する。／Ｓはハードディスクに対しては無効なので別の方法で漢字モード起動する（ハードディスクのマニュアル参照）

ＭＤ （ＭＫＤＩＲと同じ）
ＭＫＤＩＲ［d:］パス
パスで指定される名前のディレクトリーを作成する。

ＭＯＤＥ 数値
画面の１行の文字数を数値で設定する。

ＭＯＶＥ ［／Ｈ］［／Ｐ］複合ファイルスペック［パス］
複合ファイルスペックで指定されるファイルをパスで指定されたディレクトリーに移動する。パスが省略されるとカレンドディレクトリーが設定されたものとみなされる。／Ｈで不可視属性ファイルも移動する。／Ｐで画面一時停止を有効とする。

ＭＶＤＩＲ ［／Ｈ］［／Ｐ］複合ファイルスペック［パス］
複合ファイルスペックで指定されるディレクトリーをパスで指定されたディレクトリーに移動する。パスが省略された場合とオプションの意味はＭＯＶＥ命令と同じ。

ＰＡＴＨ ［［＋|－］［d:］パス［［d:］パス［［d:］パス....］］］
パラメーターが指定されないと現在の検索パス設定が「；」付きで表示される。＋を付けるとパス追加、－を付けると指定パス削除。両方ともつけないと現在のパス設定をすべて解除し指定パス名のみが有効となる。

ＰＡＵＳＥ ［コメント］

バッチコマンド中でプロンプトを表示し実行を一時停止する。コメントが指定されているときはそれを表示する。

ＲＡＭＤＩＳＫ［数値[Ｋ]][／Ｄ]
数値で指定される大きさのＲＡＭＤＩＳＫを作成する。／Ｄが指定されるか数値が０の場合ＲＡＭＤＩＳＫは削除される。既にＲＡＭＤＩＳＫが存在する場合は確認メッセージが出る。

ＲＤ　（ＲＭＤＩＲと同じ）

ＲＥＭ［コメント］
バッチコマンド中でコメントを無視し次のコマンドを実行する。

ＲＥＮ　（ＲＥＮＡＭＥと同じ）
ＲＥＮＡＭＥ［／Ｈ][／Ｐ］複合ファイルスペック　ファイル名
複合ファイルスペックで指定されるファイルをの名前をファイル名に変更する。／Ｈで不可視属性ファイルも対象とする。／Ｐで画面一時停止有効。

ＲＭＤＩＲ［／Ｈ][／Ｐ］複合ファイルスペック
複合ファイルスペックで指定されるディレクトリーを削除する。オプションの意味はＲＥＮＡＭＥと同じ。

ＲＮＤＩＲ［／Ｈ][／Ｐ］複合ファイルスペック　ファイル名
複合ファイルスペックで指定されるディレクトリーの名前をファイル名に変更する。オプションの意味はＲＥＮＡＭＥと同じ。

ＳＥＴ［名前][セパレーター][値]
環境変数を設定する。ＳＥＴのみだと全部の環境変数を表示。名前が付くとその環境変数を表示。それにセパレーターが付くとその環境変数を削除、それに値が付くと環境変数にその値を設定する。例：「ＳＥＴ　ＡＢＣＤ＝」ＡＢＣＤという環境変数を削除。「ＳＥＴ　ＡＢＣＤ＝ＲＲ」環境変数ＡＢＣＤにＲＲを設定。

ＴＩＭＥ［時刻]
時刻で指定された時刻に内蔵時計を設定する。時刻が指定されないと現在時刻を表示し時刻の入力を求めてくる。

ＴＹＰＥ［／Ｈ][／Ｐ][／Ａ][／Ｂ］複合ファイルスペック｜デバイス
複合ファイルスペックで指定されるファイルを表示する。／Ｈで不可視属性ファイルも対象とする。／Ｐで画面表示一時停止有効。／Ａでアスキーファイルとして扱う（ディフォルト)。／Ｂでバイナリファイルとして扱う。

ＵＮＤＥＬ［ファイルスペック]
ファイルスペックで指定されるファイルを復活する。ファイルスペックが指定されなかった場合は＊．＊を指定したことになる。

ＶＥＲ
システムのバージョン番号を表示する。

ＶＥＲＩＦＹ［ＯＮ｜ＯＦＦ]
書き込みベリファイ機能を[有効 ｜ 無効]にする。

ＶＯＬ［d:][ボリューム名]
パラメーターを指定しないかドライブ名のみ指定の場合はボリューム名を表示する。ボリューム名を指定するとボリューム名を設定する。

ＸＣＯＰＹ［ファイルスペック［ファイルスペック]][／Ｈ][／Ｔ][／Ａ][／Ｍ][／Ｓ][／Ｅ][／Ｐ][／Ｗ][／Ｖ]
ディレクトリーとファイルを一括して複写できる拡張ＣＯＰＹ命令。／Ｈで不可視属性ファイルも複写する。／Ｔで複写されたファイルのタイムスタンプが複写された時刻になる。／Ａでアーカイブ属性のファイルのみ複写。／Ｍは／Ａと同じだが複写するとファイルの

アーカイブ属性が解除される。／Ｓはファイルをディレクトリーごと複写する。／Ｅは／Ｓが指定されているとき、複写する該当ファイルがなくても、複写元と同じサブディレクトリーを作成する。／Ｐで複写前に複写するかどうか聞いてくる。／Ｗで複写開始前に一時停止する。／Ｖを指定するとベリファイが有効となる。

ＸＤＩＲ［ファイルスペック][／Ｈ]
ファイルと指定ディレクトリー中のサブディレクトリー内部のファイルのリストを表示する。／Ｈで不可視属性ファイルも表示する。

ＭＳＸ－ＤＯＳ２におけるリダイレクションとパイプ

ＭＳＸ－ＤＯＳ２では、コマンドラインに「＞」に続いてファイル名を入れることにより、画面に表示されるべきメッセージをファイルに出力することができます。また、「＜」を使うことによって、キーボードから入力されるべき入力をファイルから入力することもできます。例：「ＥＣＨＯ　ａｂｃｄｅ　＞ＦＩＬＥ．ＤＡＴ」というコマンドを実行すると、ＦＩＬＥ．ＤＡＴというファイルが新たに作成されてその中身は「ａｂｃｄｅ」となります。同様に、「ＤＩＲ　＞ＰＲＮ」でファイルの一覧をプリンターに出力することもできます。逆に、キー入力を促すＦＯＲＭＡＴやＦＩＸＤＩＳＫコマンドに対して、例えば「ＦＯＲＭＡＴ　Ａ：　＜ＦＩＬＥＮＡＭＥ」などとやることによって、入れるはずのキー入力をファイルで代用することができます。また、「＞＞」は同一名称ファイルが存在した場合、新たにファイルを作成せず、既にあるファイルの最後にデータを追加します。
　一方、パイプはコマンドを「｜」で区切って１行にまとめたもので、左側のコマンドの出力結果が一旦テンポラリーファイルと呼ばれる作業用ファイルに出力され、右側のコマンドの入力となります。テンポラリーファイルはシステムによって既に存在するファイル名と衝突しないような名前が自動的に決定されます。またパイプ処理が終わると自動的に消去されます。

ＭＳＸ－ＤＯＳ２の環境変数

　環境変数は表示の方式や参照するドライブなどを設定するための特殊な変数で、基本的に他の値を設定するか、リセットするまで有効です。システムが使用するもののほか、ユーザーが自分で設定することもできます。システムで使用している環境変数のリストを以下に挙げます。書式は「ＳＥＴ　環境変数名＝文字列」で、例えば「ＳＥＴ　ＥＣＨＯ＝ＯＮ」といった形になります。

ＥＣＨＯ
ＯＮ（またはｏｎ）を指定するとバッチコマンド中にコマンド行を表示します。これ以外の値を指定するか、何も指定しないとＯＦＦと同じになります。

ＰＲＯＭＰＴ
ＯＮ（またはｏｎ）を指定すると、ＤＯＳのプロンプトがドライブ名、カレントディレクトリー名になります。

ＰＡＴＨ
ＰＡＴＨコマンドで指定されたＰＡＴＨの内容が入っています。

ＳＨＥＬＬ
ＣＯＭＭＡＮＤ２．ＣＯＭを再ロードするときにどこからロードするかを指定します。例えば「ＳＥＴ　ＳＨＥＬＬ＝Ｈ：ＣＯＭＭＡＮＤ２．ＣＯＭ」とすることにより、ＲＡＭＤＩＳＫ上のＣＯＭＭＡＮＤ２．ＣＯＭを再ロードするようにしておけば、ロード時間が若干短くて済みます。指定が間違っていたら、（ＣＯＭＭＡＮＤ２．ＣＯＭが見つからなかったら）以前ロードしたところから探されます。

ＴＩＭＥ
２４を指定すると時計が２４時間表示になります。

ＤＡＴＥ
日付のフォーマットを設定します。（例：「ＳＥＴ　ＤＡＴＥ＝ＤＤ／ＹＹ／ＭＭ」）

ＨＥＬＰ
ＫＨＥＬＰ
ＨＥＬＰコマンドがヘルプファイルを探すディレクトリーを指定します。ＨＥＬＰはＡＮＫ版、ＫＨＥＬＰは漢字版です。Ver.2.30で追加。

ＡＰＰＥＮＤ
ＤＯＳ１用ソフトがファイルを探すディレクトリーを指定します。ＤＯＳ１用ソフトはサブディレクトリーが使用できないので該当ファイルがサブディレクトリーにあるときに使用します。使い方によっては若干面倒なことが起きる環境変数です。

ＰＲＯＧＲＡＭ
実行中の外部コマンドの名前が入っています。参照はできますが変更してはなりません。

ＰＡＲＡＭＥＴＥＲＳ
実行中の外部コマンドが存在するディレクトリー名が絶対パス指定で入っています。参照は可能ですが変更してはなりません。

ＴＥＭＰ
パイプ処理を使用するときに作られるテンポラリーファイルを作成するディレクトリーを指定しますＲＡＭＤＩＳＫを指定して高速化しようとしたりするとき使用します。

ＵＰＰＥＲ
ＯＮ（またはｏｎ）を指定するとコマンド行が大文字に変換されます。主にＤＯＳ１用ソフトを使用するときなどに使用します。

ＲＥＤＩＲ
ＯＦＦ（またはｏｆｆ）を指定するとリダイレクション・パイプ処理を行いません。

ＥＸＰＥＲＴ
ＤＯＳ１用ディスクに入れたコマンドを使用しようとすると、ＤＯＳ２ではエラーになりますが、この値をＯＮ（またはｏｎ）にするとエラーにならず使用できます。Ver.2.30で追加。

コマンド行

バッチコマンドなどに「％」のついたパラメーターを指定すると、そこにはコマンドラインから指定した文字がそのまま変換されて入ります。例えば、以下のようなバッチファイルを作成したとします。

```
ＥＣＨＯ　ｎａｍｅ：　％０
ＥＣＨＯ　ｐａｒ１：　％１
ＥＣＨＯ　ｐａｒ２：　％２
```

これにＥＸＡＭＰＬＥ．ＢＡＴと名前を付け、「ＥＸＡＭＰＬＥ　Ａ　１」として起動すると、表示結果は次のようになります。

```
ｎａｍｅ：　ＥＸＡＭＰＬＥ
ｐａｒ１：　Ａ
ｐａｒ２：　１
```

　％０にはコマンドの名前が、％１～％９まではコマンドパラメーター１から９までの内容が展開されます。ただし、ＡＵＴＯＥＸＥＣ．ＢＡＴには、起動時は％１に起動ディスクが展開されます。
　ＢＡＳＩＣから「ＣＡＬＬ　ＳＹＳＴＥＭ」でＤＯＳに戻ったときには、ＲＥＢＯＯＴ．ＢＡＴという名前のバッチファイルがあればそれが自動的に実行されます。このＲＯＢＯＯＴ．ＢＡＴにも、同様に％１に起動ディスクが展開されます．
　Ver.2.31以降は、コマンド行に環境変数が取り込めます。例：「ＥＣＨＯ　％ＰＲＯＭＰＴ％」で、環境変数ＰＲＯＭＰＴの内容を表示します。

第９章　２４ドット漢字プリンターの制御コード

この章の表はＭＳＸマガジン１９８８年１月号および１９８９年９月号を参考にさせていただきました。なお、先頭に＊印のついているものは、日本電気ＰＣ－８０２３についているコードで、ＭＳＸ用のプリンターなら、８、１６ドットのものも含めて、大抵のものに共通のものです。

大抵の２４ドット漢字プリンターについているコード

	記号	１６進コード	動作
＊	ＬＦ	０Ａ	改行
＊	ＦＦ	０Ｃ	改ページ
＊	ＣＲ	０Ｄ	キャリッジリターン
＊	ＳＯ	０Ｅ	拡大印字
＊	ＳＩ	０Ｆ	拡大印字解除
＊	ＥＳＣ ！	１Ｂ　２１	強調印字
＊	ＥＳＣ ”	１Ｂ　２２	強調印字解除
＊	ＥＳＣ Ｘ	１Ｂ　５８	アンダーライン
＊	ＥＳＣ Ｙ	１Ｂ　５９	アンダーライン解除
	ＥＳＣ Ｈ	１Ｂ　４８	高密度パイカ（英字、かな）印字
＊	ＥＳＣ Ｎ	１Ｂ　４Ｅ	高速度パイカ（英字、かな）印字
	ＥＳＣ Ｋ	１Ｂ　４Ｂ	漢字横印字
	ＥＳＣ ｔ	１Ｂ　７４	漢字縦印字
	ＥＳＣ ＋	１Ｂ　２Ｂ	外字登録
	ＥＯＴ	０４	外字登録終了
＊	ＥＳＣ Ｓ	１Ｂ　５３	８ドットビットイメージ印字
	ＥＳＣ Ｉ	１Ｂ　４９	１６ドットビットイメージ印字
	ＥＳＣ Ｊ	１Ｂ　４Ａ	２４ドットビットイメージ印字
	ＥＳＣ Ｕ	１Ｂ　５５	２４ドットビットイメージリピート印字
＊	ＥＳＣ ０１	１Ｂ　０１	１ドットスペース
＊	：	：	：
＊	ＥＳＣ ０８	１Ｂ　０８	８ドットスペース
	ＥＳＣ Ｆ	１Ｂ　４６	ドットアドレッシング
＊	ＥＳＣ Ａ	１Ｂ　４１	通常改行（行間あり）設定
＊	ＥＳＣ Ｂ	１Ｂ　４２	グラフィック用改行（行間なし）設定
＊	ＥＳＣ Ｔ	１Ｂ　５４	改行間隔１ドット単位指定
	ＥＳＣ Ｄ	１Ｂ　４４	コピーモード

注意・プリンターによって多少機能に差がある場合があります。例えば、外字の登録で、登録できるＪＩＳコードの番号が違ったりします。高密度、高速度パイカはアスキー文字のときのみ有効です。

なお、上記のコードに加えて、下記の４つコードを持っているプリンターは「ＴＹＰＥＢ」漢字プリンターと称されます。

ＥＳＣ ＣＳ	１Ｂ　４３　５３	スーパースクリプト
ＥＳＣ Ｃｓ	１Ｂ　４３　７３	サブスクリプト
ＥＳＣ ＞	１Ｂ　３Ｅ	片方向印字モート
ＥＳＣ ］	１Ｂ　５Ｄ	片方向印字モード解除

ただし、ＭＳＸ用漢字プリンターは両方向印字モードをサポートしていないものが多くあります。これらのプリンターでは片方向印字モード、片方向印字モード解除の二つのコードを指定しても何もしません。つまり、このコードが送られてきたとき異常動作さえしなければ、別に両方向印字ができなくても構わないのです。

最新の３機種（ブラザーＭ－１２２４Ｐ／Ｘ，松下ＦＳ－ＰＡ１，ソニーＨＢＰ－Ｆ１Ｃ）に共通のコード。なお、前の表と重複するところは省いてあります。つまり、この３機種は前の表のコードをすべて持っています。外字は、ＪＩＳコード７６２０Ｈから７６３３Ｈまでがこの３機種に共通です。

	名前	１６進コード	機能
＊	ＥＳＣ Ｌ	１Ｂ ４Ｃ	左マージン指定
☆	ＥＳＣ ／	１Ｂ ２Ｆ	右マージン指定
＊	ＥＳＣ ｆ	１Ｂ ６６	順方向改行
＊	ＥＳＣ ｒ	１Ｂ ７２	逆方向改行
	ＥＳＣ Ｒ	１Ｂ ５２	文字繰り返し
	ＣＡＮ	１８	印字データ取り消し
＊	ＥＳＣ （	１Ｂ ２８	水平ＴＡＢ設定
＊	ＥＳＣ ）	１Ｂ ２９	水平ＴＡＢ部分解除
＊	ＥＳＣ ２	１Ｂ ３２	水平ＴＡＢ全部解除
＊	ＨＴ	０９	水平ＴＡＢ実行
＊	ＧＳ	１Ｄ	ＶＦＵ設定開始
＊	ＲＳ	１Ｅ	ＶＦＵ設定終了
＊	ＵＳ	１Ｆ	ＶＦＵ実行（ＶＦＵ設定が必要）
＊	ＶＴ	０Ｂ	垂直ＴＡＢ実行（ＶＦＵ設定が必要）
☆	ＥＳＣ ｃ１	１Ｂ ６３ ３１	ソフトウェアリセット
＊	ＥＳＣ Ｅ	１Ｂ ４５	エリート
	ＳＵＢ Ｕ	１Ａ ５５	スーパースクリプト
	ＳＵＢ Ｌ	１Ａ ４Ｃ	サブスクリプト
☆	ＥＳＣ ｈ１	１Ｂ ６８ ３１	縦半角
☆	ＥＳＣ ｈ０	１Ｂ ６８ ３０	縦半角解除
☆	ＥＳＣ ｑ	１Ｂ ７１	縦半角組文字
	ＳＵＢ Ｖ	１Ａ ５６	縦拡大
	ＳＵＢ Ｗ	１Ａ ５７	縦拡大解除
	ＳＵＢ Ｆ	１Ａ ４６	ｎ／１２０インチ改行設定
	ＳＵＢ Ｇ	１Ａ ４７	ｎ／１８０インチ改行設定
	ＥＳＣ Ｍ	１Ｂ ４Ｄ	ネイティブモード
☆	ＳＵＢ Ｑ	１Ａ ５１	漢字幅２４ドットピッチ指定
☆	ＳＵＢ Ｎ	１Ａ ４Ｅ	漢字幅２７ドットピッチ指定
☆	ＳＵＢ Ｅ	１Ａ ４５	漢字幅３０ドットピッチ指定
☆	ＳＵＢ Ｐ	１Ａ ５０	漢字幅３６ドットピッチ指定
	ＳＵＢ ２	１Ａ ３２	外字全部消去
	ＥＳＣ Ｖ	１Ｂ ５６	８ドットビットイメージリピート
	ＥＳＣ Ｗ	１Ｂ ５７	１６ドットビットイメージリピート

左ページの表、上記のコードに加え、さらに下記の２つのコードをサポートしている漢字プリンターは、「ＴＹＰＥ　Ａ」漢字プリンターと称されます。ただし、「ＴＹＰＥ Ａ」漢字プリンターでは、上記の表中の「☆」マークのついたコードはサポートしていなくてもいいことになっています。これらのコードを使用するときは注意したほうがいいかもしれません。

ＥＳＣ Ｑ	１Ｂ ５１	コンデンスモード
ＥＳＣ Ｐ	１Ｂ ５０	プロポーショナルモード

これらのコードも、左ページの表と同様、サポートさえしていればこれらの機能をプリンターが持っていなくてもよいようです。

第10章　ＭＳＸ２テクニカルハンドブック正誤表

「ＭＳＸ２テクニカルハンドブック」（アスキー社刊）の第１版第２刷から第１版第15刷までの正誤表を、分かった範囲内で紹介します。なお、初版本にはこのほかにも間違いが５０箇所以上あるので、紹介しきれません。初版本を持っている方は、アスキー社へ問い合わせて正誤表をもらってください。すでにもらった人ももう一度もらってください。時期によって正誤表の内容が違っていることが判明したためです。

１７ページ、第２５行
誤　　６４Ｋ実装されているＭＳＸのうち
正　　６４Ｋ実装されているＭＳＸ２のうち

１３９ページ、第１７行
誤　　０ＦＦＨを、押されていなかった場　　００Ｈを
正　　００Ｈを、押されていなかった場合には ０ＦＦＨを

２０７ページ、図４．６１
誤　　R#5　A14 A13 A12 A11 A10　1　1　1
正　　R#5　A14 A13 A12 A11 A10　A9　A8　A7

２６１ページ、第１０行
誤　　ビット６を”１”に　　正　　ビット６を”０”に

２７０ページ、第６行
誤　　＝（２５６＊ＥＰ）／１．７８７７２５　［ＭＨｚ］
正　　＝（２５６＊ＥＰ）／１．７８９７７２５　［ＭＨｚ］

２７６ページ、第１行
誤　　レジスタ#15　　正　　レジスタ#14

２７６ページ、第１行
誤　　リスト５．３　　正　　リスト５．２

３２２ページ、図５．４１
誤　　イ）ＭＸＳ－ＤＯＳを使用している状態
正　　イ）ＭＳＸ－ＤＯＳを使用している状態

３７３ページ、右第３８行
誤　　ＤＲＯＷマクロ　　正　　ＤＲＡＷマクロ

３７５ページ、左３～２６行

誤	正
ＯＬＤＩＮＴ（ＦＢ１２Ｈ，５）	ＯＬＤＩＮＴ（ＦＢ１１Ｈ，５）
ＤＥＶＮＵＭ（ＦＢ１７Ｈ，１）	ＤＥＶＮＵＭ（ＦＢ１６Ｈ，１）
ＤＡＴＣＮＴ（ＦＢ１８Ｈ，３）	ＤＡＴＣＮＴ（ＦＢ１７Ｈ，３）
ＥＲＲＯＲＳ（ＦＢ１ＢＨ，１）	ＥＲＲＯＲＳ（ＦＢ１ＡＨ，１）
ＥＳＴＢＬＳ（ＦＢ１ＤＨ，１）	ＥＳＴＢＬＳ（ＦＢ１ＣＨ，１）
ＣＯＭＭＳＫ（ＦＢ１ＥＨ，１）	ＣＯＭＭＳＫ（ＦＢ１ＤＨ，１）
ＬＳＴＣＯＭ（ＦＢ１ＦＨ，１）	ＬＳＴＣＯＭ（ＦＢ１ＥＨ，１）
ＬＳＴＭＯＤ（ＦＢ２０Ｈ，１）	ＬＳＴＭＯＤ（ＦＢ１ＦＨ，１）
ＯＬＤＩＮＴ（ＦＢ１１Ｈ，５）	ＨＯＫＶＬＤ（ＦＢ２０Ｈ，５）
ＤＥＶＮＵＭ（ＦＢ１６Ｈ，１）	内容　拡張ＢＩＯＳ存在の有無

３９０ページ

誤	正
ＢＯＨ～Ｂ３Ｈ	ＢＯＨ～Ｂ３Ｈ
ＡＨ	ＢＯＨ
ＡＨ	Ｂ１Ｈ
ＡＡＨ	Ｂ２Ｈ
ＡＢＨ	Ｂ３Ｈ

このほか特にＶＤＰ関係のサンプルプログラムは、動作しなかったり、自ら規格違反のプログラムを書いていたりして、参考にできないということが判明しました。サンプルプログラムを使用するときは十分注意してください。

第11章　Ｉ／Ｏマップ

ここでは、ＭＳＸのＩ／Ｏマップを掲載します。ここに紹介されていないポートは将来拡張用なので、使用することはできません。＊印はturboRで削除されたものです。

ポート番号

00H～3FH　ユーザー拡張用。商用アプリケーションの使用は禁止
40H～4FH　メーカー別の拡張。松下規格12ドット漢字フォントＲＯＭなど
70H～73H　ＭＩＤＩ（ビッツー規格）
　　7CH　ＭＳＸ－ＭＵＳＩＣ（ＦＭ音源）
　　7DH　ＭＳＸ－ＭＵＳＩＣ。外付けも内蔵もポートは同じ
80H～87H　ＲＳ－２３２Ｃ
88H～8BH＊ ＭＳＸ２バージョンアップアダプターのＶＤＰ（Ｖ９９３８）
8CH～8DH　モデム
90H～91H　プリンター
98H～9BH　ＶＤＰ（ＴＭＳ９９１８，Ｖ９９３８，Ｖ９９５８）
A0H～A3H　サウンドジェネレーター（ＡＹ－３－８９１０）
　　A4H　Ｄ／Ａコンバーター。ＰＣＭ用。turboRで追加
　　A5H　Ｄ／Ａコンバーター。ＰＣＭ用。turboRで追加
　　A7H　ポーズランプ、ＣＰＵモード表示用ランプの点灯に使用。turboRで追加
A8H～ABH　８２５５
ACH～AFH　ＭＳＸエンジン（カスタムチップ）
B0H～B3H＊ ソニーＨＢ－５５付属４ＫバイトＳ－ＲＡＭカートリッジ
B4H～B5H　クロックＩＣ（ＲＰ－５Ｃ０１）。外付けも内蔵もポートは同じ
B8H～BBH＊ 三洋規格ライトペン。外付けも内蔵もポートは同じ
BCH～BFH＊ 日本ビクター規格ＶＨＤコントローラー
　　C0H＊ ＭＳＸ－ＡＵＤＩＯ（ＦＭ音源）第１チャンネル
　　（コントロール・ステータスポート）
　　C1H＊ ＭＳＸ－ＡＵＤＩＯ（ＦＭ音源）第１チャンネル
　　（データポート）
　　C2H＊ ＭＳＸ－ＡＵＤＩＯ　第２チャンネル（コントロール・ステータス）
　　C3H＊ ＭＳＸ－ＡＵＤＩＯ　第２チャンネル（データポート）
C8H～CFH＊ ＭＳＸインターフェイス
D0H～D7H＊ フロッピーディスクコントローラー。使用しない
　　D8H　第一水準漢字ＲＯＭ　b5-b0　下位アドレス（ライトのみ）
　　D9H　第一水準漢字ＲＯＭ　b5-b0　上位アドレス（ライト）
　　b7-b0　データ（リード）
　　DAH　第二水準漢字ＲＯＭ　b5-b0　下位アドレス（ライトのみ）
　　DBH　第二水準漢字ＲＯＭ　b5-b0　上位アドレス（ライト）
　　b7-b0　データ（リード）
　　DCH　漢字用予約（現在未使用）
E0H～E2H　ＭＳＸ－ＭＩＤＩ。turboR専用
　　E4H　ＣＰＵモード変更の際に使用する。turboRで追加
　　E5H　ＣＰＵモード変更の際に使用する。turboRで追加
　　E6H　システムタイマー（下位８ビット）。turboRで追加
　　E7H　システムタイマー（上位８ビット）。turboRで追加
E8H～EFH　ＭＳＸ－ＭＩＤＩ。turboR専用
　　F3H　ＶＤＰの画面モード（ＭＳＸ２＋で追加）
　　b7　ＹＡＥ　　b6　ＹＵＶ　　b5　ＴＰ
　　b4　Ｍ１　　b3　Ｍ２　　b1　Ｍ４　　b0　Ｍ３
　　F4H　リセットステータス（初期化の制御）。ＭＳＸ２＋で追加
　　F5H　デバイスイネーブル　b0　第一水準漢字ＲＯＭ
　　（書き込み専用）　b1　第二水準漢字ＲＯＭ（ＭＳＸ２＋で追加）
　　b2　ＭＳＸ－ＡＵＤＩＯ
　　b3　ＡＶ制御（スーパーインポーズ）
　　b4　ＭＳＸインターフェイス
　　b5　ＲＳ－２３２Ｃ（モデムは関係なし）
　　b6　ライトペン
　　b7　ＣＬＯＣＫ－ＩＣ
　　F6H　カラーバスＩ／Ｏ。オプションのＡ／Ｖ機能で使用。ＭＳＸ２で追加
　　F7H　Ａ／Ｖコントロール。オプションのＡ／Ｖ機能で使用。ＭＳＸ２で追加
　　FCH　メモリマッパ・ページ０切り換え用。書き込み専用
　　FDH　メモリマッパ・ページ１切り換え用。書き込み専用
　　FEH　メモリマッパ・ページ２切り換え用。書き込み専用
　　FFH　メモリマッパ・ページ３切り換え用。書き込み専用

索引

編集後記

この PDF 版は、第四版４刷を元に作成いたしました。しかし、当時とは開発環境が異なっているため、版組等に若干の差があります。ご了承ください。
また、第四版第４刷は、それまでのどの第四版とも版組が全く異なっています。もちろんページ割も全く異なっています。従いまして、本書の第３刷以前をお持ちの方と、第４刷をお持ちの方が、本書について会話をするときは充分気をつけてください。例えば、第３刷までには、第３部第15章という章は存在しません。
なお、一部の表現に現代の状況とそぐわない箇所や、社会通念上好ましくないとされる表現が含まれる可能性がありますが、本書の歴史的な背景及び歴史的資料性を尊重しそのままといたしました。
今回の PDF 版は今のところ暫定版です。版組等に問題があった場合、第四版第４刷にあった内容が消えていたり、ゴミデータが出力されているような場合には修正をかけてから正式に公表いたします。この版に問題点があることを発見された場合は、奥付のアドレスまでご連絡ください。

ＭＳＸテクニカルガイドブック第四版
1989年12月24日　初版発行
1989年12月29日　第１版第２刷発行
1990年７月24日　増補改訂第２版発行
1991年２月13日　新訂第三版発行
1992年７月24日　第四版発行
1992年９月７日　第四版第２刷発行
1993年２月13日　第四版第３刷発行
1994年２月13日　第四版第４刷発行
2010年１月８日　PDF版第四版暫定第５刷発行

検印
省略

発行所　アスキャット

http://ascat.jp
ascat@fetish-jp.org
（※メールアドレス及びドメインは、2010年現在のものです。
1992年時点でアスキャットが使用していたメールアドレス等とは異なります）