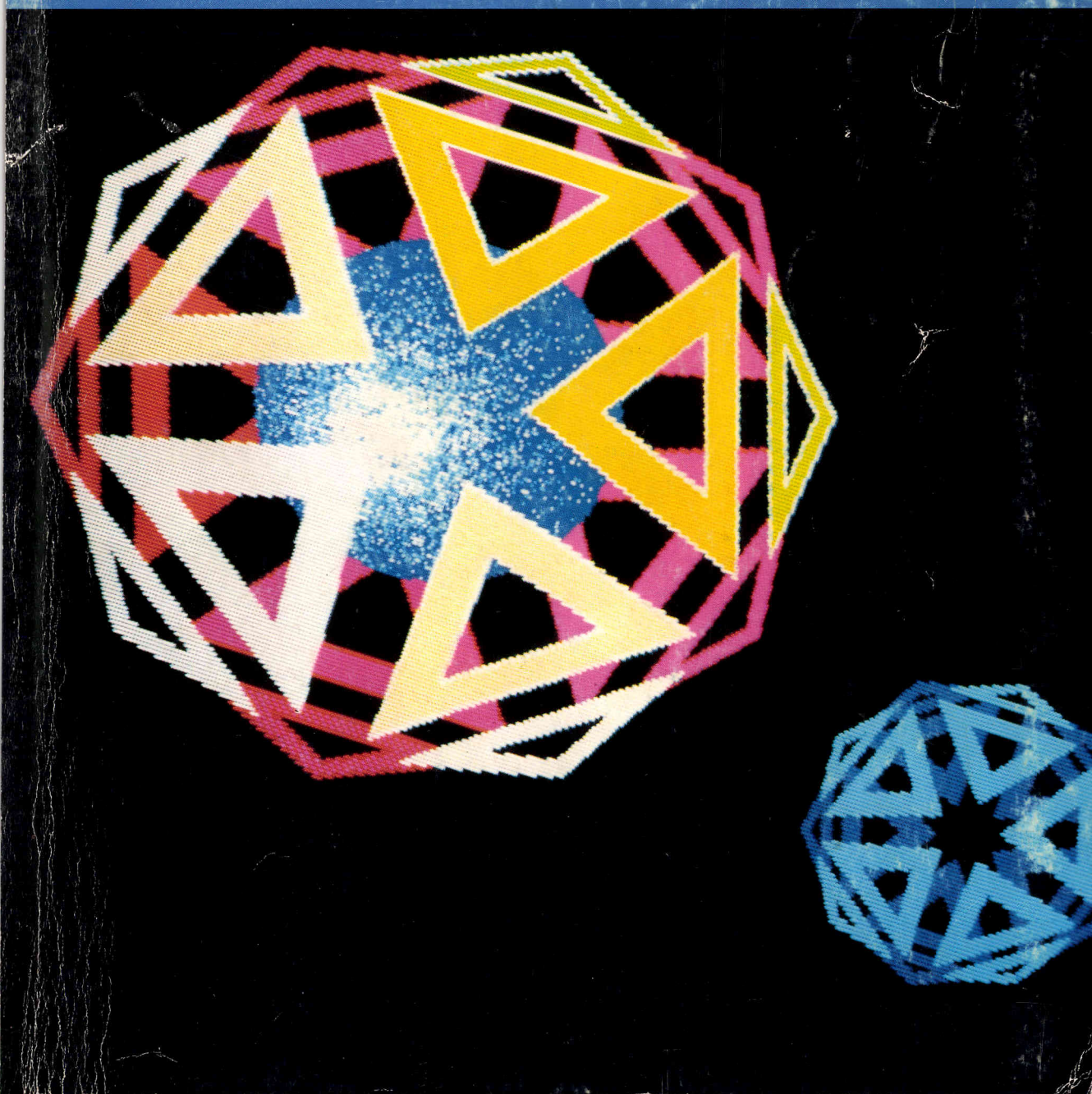
SHARP®

New Life Now

パソコンテレビ X1turbo
パーソナルコンピュータ

形名
CZ-850C
CZ-851C
CZ-852C

USER'S MANUAL



■ユーザーズマニュアル

はじめに

このマニュアルはＢＡＳＩＣの初歩から、本機の特長であるグラフィック機能、日本語処理機能、デジタルテロッパー機能などを使いこなしていただくため項目別に説明しています。

目的に応じて各章をお読みください。

なお、本機を正しくご使用いただくため、まず取扱説明書からお読みください。

ＤＩＳＫ　ＢＡＳＩＣが起動されると自動的にＳｔａｒｔ　ｕｐというプログラムが実行され、各種の初期設定が行なわれます。

初期設定には、ＤＩＳＫ　ＢＡＳＩＣが起動されるときにすでに行なわれているものと、Ｓｔａｒｔ　ｕｐプログラム中で行なわれているものがあります。

ＤＩＳＫ　ＢＡＳＩＣ起動時の初期設定は変更できませんが、Ｓｔａｒｔ　ｕｐプログラム中で行なわれるものは、このプログラムを自分の希望するように書き換えておけば、初期状態を自由に決めることができます。

（ｉ）ＤＩＳＫ　ＢＡＳＩＣ起動時の初期状態

ＩＮＩＴ
ＷＩＤＴＨ　８０，２５（高解像度ディスプレイモードのとき）
ＷＩＤＴＨ　８０，１２（標準ディスプレイモードのとき）
ＫＭＯＤＥ１
ＯＰＴＩＯＮ　ＳＣＲＥＥＮ１
ＣＬＥＡＲ　＆ＨＦ４００
ＣＬＩＣＫ　ＯＮ
ＲＥＰＥＡＴ　ＯＮ

（ⅱ）Ｓｔａｒｔ　ｕｐで設定するもの

ＣＬＥＡＲ　＆ＨＦ０００（音訓辞書使用時）
ＦＵＮＣＴＩＯＮ　ＫＥＹの設定
内蔵時計の「年」の項の設定
ＮＥＷＯＮの設定

＊Ｓｔａｒｔ　ｕｐではＮＥＷＯＮ４を実行するようにプログラムが組まれています。
しかし、HELPキーを押しながらＤＩＳＫ　ＢＡＳＩＣを起動すると、
ＮＥＷＯＮ□
と表示し、キー入力待ちの状態となります。ここで０～９までの数値かまたは数値省略(全てのコマンド使用可)を入力すれば希望のレベルのNEWONが設定できます。
詳細は「２章、５、NEWONとフリーエリア」参照

ユーザーズマニュアル

はじめに

1章
プログラミングの初歩

1 コンピュータへのメッセージ

さて、それではディスプレイ画面を見て下さい。画面に四角くて明滅するものがあります。このチカチカするものを**カーソル**といいます。カーソルは表示位置を示しつつコンピュータがあなたからのメッセージを待っています。

では、手はじめに、

```
C
```

とキーを打って↵キーを押してみてください。（↵は画面に表示されません）

↵はリターン（Ｒｅｔｕｒｎ）キーで画面に表示されている命令をコンピュータに伝えた後、カーソルが次の行の先頭に移ります。

```
Ｓｙｎｔａｘ　ｅｒｒｏｒ
Ｏｋ
□
```

というメッセージが画面に返って来ましたね。これは、コンピュータが「"Ｃ"と言われてもいったい何のことかわからない」と言っているのです。

次に、

```
ＣＬ
```

とキーを打って↵キーを押してみてください。またまた、

```
Ｓｙｎｔａｘ　ｅｒｒｏｒ
Ｏｋ
□
```

というメッセージが返って来ましたね。これは、コンピュータが「"ＣＬ"と言われても、やっぱり何のことかわからない」と言っているのです。

では今度は、

```
ＣＬＳ
```

と打って↵キーを押してみてください。今度はいままでと少し違うようです。やっと、あなたのメッセージがコンピュータに通じたようです。ディスプレイ画面が消えて、画面の左上隅に、

```
Ｏｋ
□
```

と表示されましたね。

ＣＬＳには「画面をきれいにしてください」という意味があります。"ＣＬＳ"のようにコンピュータがわかる言葉を**キーワード**といいます。キーワードは全部で２００以上あります。いっぺんには覚えられませんから、少しずつ覚えるようにしましょう。

それでは次に、ＰＲＩＮＴというキーワードについてみてみましょう。

```
ＰＲＩＮＴ"Ｘ１"
```

と打って↵キーを押して画面をよく見てみると、すぐ下の行に、

```
Ｘ１
Ｏｋ
□
```

と表示されているのがわかります。

```
ＰＲＩＮＴ"２３＋４４"
```

と打って↵キーを押すと、

```
23+44
Ok
□
```

と画面に表示されます。

これは、PRINTが「引用符（"）で囲まれた文字を画面に表示しなさい」という意味をもったメッセージだからです。

それでは、次に示す例を順に試してみてください。↵の記号は、「その位置で↵の記号を打ってください」という意味です。

（例1）PRINT　23+44↵

```
67
Ok
□
```

（例2）PRINT（23+44）＊2／5↵

```
26.8
Ok
□
```

（例3）PRINT　SUM（10）↵

```
55
Ok
□
```

（例4）？　23+44↵

```
67
Ok
□
```

上の例は、（例1）が23+44のたし算を、（例2）が（23+44）×2÷5の計算を、（例3）が1から10までの和を求める計算を、それぞれコンピュータにさせ、その結果を表示させたものです。

これは、PRINTに「引用符（"）で囲まれた文字を表示しなさい」という意味以外に、「計算した結果を表示しなさい」という意味もあるので、それを利用したものです。

なお、（例4）だけはちょっと別のキーワードの例のようですが、実は、PRINTは？（疑問符）で置き換えて使うことができます。キーボードからP、R、I、N、Tと5文字入れるところが、？の1文字で済むのでこれはたいへん便利ですね。

これまでみてきたように、キーワードは、その名の通りあなたがコンピュータにメッセージを伝えるための鍵となる単語で、キーボードから正しく打ち込んで↵キーを押すと、コンピュータはそのメッセージに応じた仕事をしてくれます。

2　まずは簡単なプログラム作りから

それでは、

```
MUSIC↵
```

と打ち込んで下さい。

```
Missing operand
Ok
□
```

というメッセージが返って来ました。

これは、コンピュータが「MUSICというキーワードの意味はわかるけれども、MUSICの後ろに必要なデータが書かれていないので何もできません」と途方に暮れているのです。その必要なデータをいっしょに打ち込んでみましょう。

MUSIC"BG"↵

さあ、どうなりましたか？　何にも起こらないという人は、ディスプレイテレビのボリュームを上げてもう1度打ち込んでみてください。「シー・ソー」という電子音が出るはずです。

MUSICは音を出すためのキーワードで、音を表わす文字を引用符（"）で囲んで指定すると、音を出すことができます。ここでは、B（シの音）とG（ソの音）を指定したので「シー・ソー」という音が出たのです。これは、さきほどのPRINTと使い方が似ています。

PRINT"CHIME"↵

打ち込むと、引用符で囲んだ文字CHIMEが表示されますね。では、次のように打ち込むとどうでしょう？

10 PRINT"CHIME"↵

何ごとも起こりませんね。「CHIME」と表示しないし、「Syntax error」などのメッセージが返って来るわけでもありません。

今度は、

20 MUSIC"BG"↵

と打ち込んでください。やはり何ごとも起きません。でも、次のように打ち込むとどうなりますか？

RUN↵

画面に「CHIME」と表示されて、「シー・ソー」という音が出たでしょう。

RUNと打ち込んで、↵キーを押してやれば、何度でも同じことを繰り返すことができます。いったいどういうわけでしょう。

LIST↵

と打ち込んでみてください。

```
10 PRINT"CHIME"
20 MUSIC"BG"
Ok
□
```

と画面に表示されるでしょう。

PRINTやMUSICの前に10とか20の番号をつけるかつけないかでコンピュータの反応が違っています。

番号をつけないで、キーワードを打ち込んで↵キーを押すと、コンピュータはすぐその場で実行して、画面に文字を表示したり、音を出したりします。

番号をつけると、コンピュータはすぐには実行しませんが、打ち込まれたメッセージをきちんと覚えているので、あとからRUN↵と打ち込めば何度でも実行することができます。ここで、

```
10 PRINT"CHIME"
20 MUSIC"BG"
```

の2行のことを**プログラム**といい、10や20の番号のことを**行番号**、行番号の後ろのPRINT"CHIME"や MUSIC"BG"のことを**ステートメント**といいます。

また、プログラムを実行するために打ち込んだRUNというキーワードのことを**コマンド**といいます。コマンドは、プログラムをどうするのかコンピュータに指示すときに使います。アンダーラインのついた言葉は大切ですのでよく覚えるようにしてください。

RUNは「プログラムを実行しなさい」
LISTは「プログラムを画面に表示しなさい」
CLSは、「画面に書かれた内容を消しなさい」
という意味のコマンドです。

LIST↵

と入れると、画面にプログラムが表示されます。

RUN↵

と入れると、プログラムを実行します。

CLS↵

と入れると、画面がきれいになりますが、プログラムもいっしょにきれいに消えてしまったかというと決してそうではありません。その証拠に、

```
LIST↵
```

と入れると、プログラムが表示されます。

3 プログラムの作成(その1)

3.1 プログラムの入力

プログラムをキーボードから打ち込んでコンピュータに覚え込ませることを、プログラムを**入力する**といいます。

さきほど入力したプログラムはたった2行でしたが、何十行、何百行となるのが普通です。そういった大きくて複雑なプログラムを入力する際には必らず途中で追加、変更、削除などの編集を行なわねばなりません。

ここでは、小さなプログラムを例にとり、プログラムの編集方法を具体的に説明しましょう。

まず、NEWというコマンドを使って、コンピュータ内に残っているプログラムをきれいに消しましょう。

```
NEW↵
```

試しに、

```
LIST↵
```

と打ち込んでみてください。

```
Ok
□
```

と表示されて、プログラムがきれいに消えていることがわかります。

次に画面をきれいにしましょう。

```
CLS↵
```

さあ、プログラムを入力する準備は整いました。スペルを間違えないように、プログラムを入力して行きましょう。

```
10 PRINT"X1"↵
```

プログラムを1行入力したら、まず正しく入力されたかどうか見なおします。間違いが見つかったら、面倒でももう一度入力しなおしてください。何も全部入力しなおさなくても、もっとよい方法があるのですが、それについてはあとで説明します。

正しく入力されたことを確かめて、

```
RUN↵
X1
Ok
□
```

と表示されたらオーケイ。もし、

```
Syntax error in 10
Ok
□
```

とメッセージが出たら、

```
10 PRINT"X1"↵
```

と再び入力しなおしましょう。

```
20 MUSIC"BGADDR"↵
LIST↵
10 PRINT"X1"
20 MUSIC"BGADDR"
```

```
Ok
□
```

これでプログラムが2行になりました。実行してみましょう。

```
RUN↵
X1
```

と表示されて音が出るでしょう。

```
30 MUSIC"DE#FGGR"↵
LIST↵
10 PRINT"X1"
20 MUSIC"BGADDR"
30 MUSIC"DE#FGGR"
Ok
□
```

これでプログラムが3行に増えました。RUN↵と入力して、このプログラムを走らせる（RUNの本来の意味は「走る」です）と、「X1」と表示して、さきほどより長いメロディーが流れます。

3.2 1行の挿入

```
15 PRINT"CHIME"↵
LIST↵
10 PRINT"X1"
15 PRINT"CHIME"
20 MUSIC"BGADDR"
30 MUSIC"DE#FGGR"
Ok
□
```

10と20の間に新しい15の行が**挿入**されて、プログラムが4行に増えました。

```
RUN↵
X1
CHIME
```

上のように表示されて、音が流れましたか？

プログラムは基本的には若い行番号から順に実行されます。ここでは、まず行番号10のPRINT、次いで15のPRINT、そして20と30のMUSICの順に実行されます。

3.3 1行の削除

それでは、X1と表示しないようにするにはどうすればよいのでしょうか。最も簡単な方法はプログラムから10番の行を取ってしまえばよいのです。そのためには、行番号のうしろに何も入力しないでリターンキーを押せばよいので、

```
10↵
```

と入力します。

```
LIST↵
15 PRINT"CHIME"
20 MUSIC"BGADDR"
30 MUSIC"DE#FGGR"
```

10番が**削除**されて3行のプログラムになっていることがわかります。走らせると、

```
RUN↵
CHIME
```

と、「X1」を表示しないでメロディーが流れます。

このように、プログラムから1行削除するには、その行番号だけを入力して↵キーを押せば済み

ます。

3.4 1行の変更

30のMUSICの音を変えてみましょう。30のステートメントを新たに入力すればよいのです。

```
30 MUSIC"DABGGR"↵
LIST↵
15 PRINT"CHIME"
20 MUSIC"BGADDR"
30 MUSIC"DABGGR"
Ok
□
```

前の30番の行がいま入力した行に変更されました。さっそくプログラムを走らせてみましょう。どんな音が出るのかな？

```
RUN↵
```

もし「Syntax error」が出たら、もう1度入力しなおすことを忘れずに。

```
5 CLS↵
LIST↵
5 CLS
15 PRINT"CHIME"
20 MUSIC"BGADDR"
30 MUSIC"DABGGR"
Ok
□
```

CLSは、「画面をきれいにする」という意味のコマンドですが、プログラムのステートメントとしても使うことができます。

```
RUN↵
```

画面をきれいに消してから「CHIME」と表示して音がでるようになりました。

3.5 行番号のつけなおし

```
RENUM↵
Ok
□
```

あれっ、何をしたのかな？

```
LIST↵
10 CLS
20 PRINT"CHIME"
30 MUSIC"BGADDR"
40 MUSIC"DABGGR"
```

よく見るとプログラムの行番号が変わったのがわかります。でも、プログラムの内容まで変わったわけではないので安心してください。

```
RUN↵
```

と入力して走らせると、コンピュータは、前と同じことをやってくれるでしょう。

RENUMは、「プログラムの行番号をつけなおしなさい」という意味をもつコマンドです。このコマンドは、プログラムに新しい行を挿入するために行間のスペースがなくなったときに使うと便利です。

さて、プログラムを増やしましょう。

```
50 MUSIC"GBADDR"↵
60 MUSIC"DABGGR"↵
```

```
70 PRINT"END"↵
```

プログラムが正しく入力されたかどうかを確かめてから、走らせてみましょう。

```
LIST↵
10 CLS
20 PRINT"CHIME"
30 MUSIC"BGADDR"
40 MUSIC"DABGGR"
50 MUSIC"GBADDR"
60 MUSIC"DABGGR"
70 PRINT"END"
Ok
RUN↵
```

どうですか。チャイムらしい音が出ましたか。

このプログラムはいま先頭の番号が１０になっていますが、ＲＥＮＵＭを使ってプログラムの先頭の行番号を変えることができます。たとえば、先頭を１００にしたいときは

```
RENUM100↵
Ok
□
```

と入力します。

```
LIST↵
100 CLS
110 PRINT"CHIME"
120 MUSIC"BGADDR"
130 MUSIC"DABGGR"
140 MUSIC"GBADDR"
150 MUSIC"DABGGR"
160 PRINT"END"
Ok
□
```

試しに走らせてみると、さきほどと同じプログラムであることがわかるでしょう。

```
RUN↵
```

3.6 複数行の削除

さて、今度はこのプログラムを消すことを考えてみましょう。残らず消してしまうにはＮＥＷというコマンドを使えばよいのですが、もし１００～１６０まであるプログラムのうち１１０～１５０の行を消したいときにはどうすればよいのでしょうか。

消したい行の行番号を１つ１つ入力して行くのもよいでしょうが、面倒だという人のためにもっとよい方法があります。それは、ＤＥＬＥＴＥというコマンドを使う方法です。このコマンドを使えば、プログラムのいらなくなった行をまとめて消してしまうことができます。

１１０～１５０の行を消したいときは次のように入力します。

```
DELETE110-150↵
```

これで、コンピュータは１１０～１５０のプログラムをすべて消してしまいました。確かめるため、プログラムを見てみましょう。

```
LIST↵
100 CLS
160 PRINT"END"
```

このプログラムを走らせると、画面が消えて「ＥＮＤ」と表示されるはずです。

```
RUN↵
```

```
END
Ok
```

ここで、これまで説明したプログラムの編集方法について１度まとめておきましょう。

―――――――――☆　プログラムの編集方法（その１）　☆―――――――――

１．１行挿入するには
　⇨挿入したい行の上と下の行の行番号の間の番号を行番号に選ぶ。
　たとえば、行番号２０と３０の間に挿入したいときは２１～２９の行番号で始まる１行を入力する。
２．１行削除するには
　⇨削除したい行の行番号のみを入力する。
３．何行かまとめて削除するには
　⇨ＤＥＬＥＴＥコマンドを使う。
　たとえば、行番号１１０～１８０を削除したいときは、
　　ＤＥＬＥＴＥ１１０－１８０↵
　と入力する。
４．１行変更するには
　⇨変更したい行の行番号で始まる１行を改めて入力する。

4　プログラムの作成（その2）

次に進む前に、プログラムと画面をきれいに消して起きましょう。

```
NEW↵
Ok
□
CLS↵
Ok
□
```

4.1 カーソルの移動と文字の修正

ここでは、新たに簡単な曲を演奏するプログラムを入力してみましょう。

```
１０ MUSOC"CRCRGRGRARARGGRR"↵
```

ここでＭＵＳＩＣと入力するところを誤ってＭＵＳＯＣと入力したことに気づいたとします。

こういう場合、どうやって誤りを訂正したらよいでしょうか。もう１度「１０……」と行番号１０から入力しなおすのも一つの手ですが、もっとよい方法があります。

それは、キーボードのテンキー部分の下方にある**カーソルコントロールキー**を使う方法です。

カーソルコントロールキーは矢印のついたキーで、↑↓←→の４つあり、画面上のカーソルを上下左右好きな方向に動かすことができます。

ＭＵＳＯＣの「Ｏ」を「Ｉ」になおすためには、まずカーソルコントロールキーを使ってカーソルをＭＵＳＯＣの「Ｏ」のところまでもって行きます。

```
１０ MUSOC"CRCRGRGRARARGGRR"
□
↑
カーソル
```

の状態で、↑を１回、→を６回打ってみてください。カーソルが、ちょうど「Ｏ」のところに来たらオーケイです。もし、「Ｏ」を通り過ぎてしまっても、慌てないで４つのカーソル・キーをうまく使って「Ｏ」のところに持って行ってください。

```
１０ MUSOC"CRCRGRGRARARGGRR"
```

（□はカーソルを表わす）

ここでＩを押すと、「Ｏ」が「Ｉ」に変わってＭＵＳＩＣとなります。カーソルは「Ｃ」のところに来ていますから、そのままそこで↵キーを押してください。

これで**修正**が完了しました。

試しにプログラムを見てみましょう。

```
LIST↵
10 MUSIC"CRCRGRGRARARGGRR"
Ok
```

それでは次に、１行入力中、↵を押す前に入力の誤りに気づいた場合についてみてみましょう。

```
20 MUSOC□
```

ここまで打って、ＭＵＳＩＣの「Ｉ」が「Ｏ」になっている誤りに気づいたとき、それをなおすには２通りの方法があります。

まず、第１の方法は左向き矢印のカーソルコントロールキー←を２回打ってカーソルを「Ｏ」のところに持って行き、Ｉと打つ方法がありますが、第２の方法として INS DEL キーを２回打って、

```
20 MUS□
```

「ＯＣ」の２文字を消してしまい、Ｉ、Ｃと打って修正する方法もあります。

さて、どちらかの方法で修正が終わったらプログラムの入力を続けることにしましょう。

4.2 効率のよい入力方法

```
20 MUSIC"FRFRERERDRDRCCRR"↵
30 MUSIC"GRGRFRFRERERDDRR"↵
```

次の４０番の行が上の３０番とまったく同じステートメントでよいときは、カーソルを３０の「３」のところに持って行って、４を打ち↵キーを押します。

```
20 MUSIC"FRFRERERDRDRCCRR"
40 MUSIC"GRGRFRFRERERDDRR"
□
```

これで３０番とまったく同じ行がもう１つ、４０番の行に作られました。

プログラムを見てみましょう。

```
LIST↵
10 MUSIC"CRCRGRGRARARGGRR"
20 MUSIC"FRFRERERDRDRCCRR"
30 MUSIC"GRGRFRFRERERDDRR"
40 MUSIC"GRGRFRFRERERDDRR"
Ok
□
```

どうです。４０番の行がきちんとできているでしょう。

次の５０番の行は１０番の行とまったく同じでよいのですが、どうすればよいでしょうか？　そうです。１０の「１」のところにカーソルを持って行って５と打って↵キーを押せばよいのです。

```
50 MUSIC"CRCRGRGRARARGGRR"
20 MUSIC"FRFRERERDRDRCCRR"
30 MUSIC"GRGRFRFRERERDDRR"
40 MUSIC"GRGRFRFRERERDDRR"
Ok
```

次の６０番の行は２０番の行とまったく同じです。いまちょうど、カーソルが２０の「２」のところに来ていますね。その位置で６と打って↵キーを押してください。

```
50 MUSIC"CRCRGRGRARARGGRR"
60 MUSIC"FRFRERERDRDRCCRR"
30 MUSIC"GRGRFRFRERERDDRR"
```

```
40 MUSIC"GRGRFRFRERERDDRR"
Ok
```

プログラムの上にカーソルがあり、プログラムを書き換えてしまう恐れのあるときは、CLR HOMEキーを使って画面を消してしまいましょう。

SHIFTキーを押しながら、CLR HOMEキーを押してみてください。画面がすっかりきれいになりましたね。

ここで、プログラムを見てみましょう。

```
LIST↵
10 MUSIC"CRCRGRGRARARGGRR"
20 MUSIC"FRFRERERDRDRCCRR"
30 MUSIC"GRGRFRFRERERDDRR"
40 MUSIC"GRGRFRFRERERDDRR"
50 MUSIC"CRCRGRGRARARGGRR"
60 MUSIC"FRFRERERDRDRCCRR"
Ok
```

これは「キラキラ星」でだれもが知っている有名な曲ですが、このままこのプログラムを実行すると、テンポが非常に遅いながらも曲が流れます。

```
RUN↵
```

音楽を途中で止めたいときは、SHIFTキーを押しながら、BREAKキーを押してください。一般的に、実行中のプログラムを止めるには、SHIFTキーを押しながら、BREAKキーを押します。

曲のテンポを速くするには、曲のテンポを指定するステートメントを入れます。

```
5 TEMPO300↵
```

ついでに、画面を消すステートメントと、タイトルを表示するステートメントも入れましょう。

```
3 CLS↵
4 PRINT"キラキラボシ"↵

LIST↵
 3 CLS
4 PRINT"キラキラボシ"
5 TEMPO300
10 MUSIC"CRCRGRGRARARGGRR"
20 MUSIC"FRFRERERDRDRCCRR"
30 MUSIC"GRGRFRFRERERDDRR"
40 MUSIC"GRGRFRFRERERDDRR"
50 MUSIC"CRCRGRGRARARGGRR"
60 MUSIC"FRFRERERDRDRCCRR"
Ok
□
```

行番号を揃えて、

```
RENUM↵
OK
LIST↵
10 CLS
20 PRINT"キラキラボシ"
30 TEMPO300
40 MUSIC"CRCRGRGRARARGGRR"
50 MUSIC"FRFRERERDRDRCCRR"
60 MUSIC"GRGRFRFRERERDDRR"
70 MUSIC"GRGRFRFRERERDDRR"
```

```
80 MUSIC"CRCRGRGRARARGGRR"
90 MUSIC"FRFRERERDRDRCCRR"
Ok
□
```

走らせてみると、「キラキラ星」が演奏されます。

4.3 文字の挿入

ここで、タイトルの「キラキラボシ」を画面の中央に表示したいときは、タイトルを書く位置をLOCATEというキーワードを使って指定します。

「15 LOCATE12,12↵」と、15番の1行を入力してもよいのですが、20の番号とPRINTの間に「LOCATE12,12:」を挿入しても同じプログラムになります。

まずカーソルをPRINTの「P」のところに持って行きましょう。そして、SHIFT+INS DEL キーを12回押す（SHIFTを押したままINS DELキーを12回打つ）と、カーソルのあるP以降が1文字ずつ計12文字分右にずれ、挿入するスペースがあきます。

```
20 PRINT"キラキラボシ"
     ↑
   カーソル
     ⇩
20 □                    PRINT"キラキラボシ"
     ↑
   カーソル
```

ここで「LOCATE12,12:」と入力して、↵キーを押すと挿入修正した20番の行が入力されます。

```
20 LOCATE12,12:PRINT"キラキラボシ"
30 ……
……
```

プログラムを確かめてみましょう。

```
LIST↵
10 CLS
20 LOCATE12,12:PRINT"キラキラボシ"
30 TEMPO300
40 MUSIC"CRCRGRGRARARGGRR"
50 MUSIC"FRFRERERDRDRCCRR"
60 MUSIC"GRGRFRFRERERDDRR"
70 MUSIC"GRGRFRFRERERDDRR"
80 MUSIC"CRCRGRGRARARGGRR"
90 MUSIC"FRFRERERDRDRCCRR"
Ok
```

さっそく実行してみましょう。画面に「キラキラボシ」と表示されて音楽が流れます。

4.4 文字の削除

```
LIST↵
10 CLS
20 LOCATE12,12:PRINT"キラキラボシ"
30 TEMPO300
40 MUSIC"CRCRGRGRARARGGRR"
50 MUSIC"FRFRERERDRDRCCRR"
60 MUSIC"GRGRFRFRERERDDRR"
```

```
70 MUSIC"GRGRFRFRERERDDRR"
80 MUSIC"CRCRGRGRARARGGRR"
90 MUSIC"FRFRERERDRDRCCRR"
OK
```

「LOCATE12，12」を消したいときは、まず、カーソルを消したい文字のすぐ後ろ、ここではPRINTの「P」のところに持って行きます。そして、[INS DEL]キーを12回打つと、カーソルの左隣りの文字が1文字ずつ計12文字消され、それに続く文字の「PRINT……」を引っ張ってきます。ここで[↵]キーを押すと、削除修正した行が入力されます。

```
20 LOCATE12，12：PRINT"キラキラボシ"
                  ↑
                カーソル
   ⇩

20 PRINT"キラキラボシ"
30 ……
↑
カーソル
```

[SHIFT]キーを押しながら[CLR HOME]キーを押して画面を消してから、プログラムを確かめてみましょう。

```
LIST↵
10 CLS
20 PRINT"キラキラボシ"
30 TEMPO300
40 MUSIC"CRCRGRGRARARGGRR"
50 MUSIC"FRFRERERDRDRCCRR"
60 MUSIC"GRGRFRFRERERDDRR"
70 MUSIC"GRGRFRFRERERDDRR"
80 MUSIC"CRCRGRGRARARGGRR"
90 MUSIC"FRFRERERDRDRCCRR"
Ok
□
```

これまで簡単な例を示しながら、プログラムの編集についてほんの触りの部分を説明しました。

これでプログラムを作るということが、どのようなことなのか、ある程度おわかりになられたと思います。

☆ プログラムの編集方法（その2） ☆

1．カーソルはカーソルキーを使って自由に移動できる。

- [↑]カーソルを上へ移動する。
- [↓]カーソルを下へ移動する。
- [←]カーソルを左へ移動する。
- [→]カーソルを右へ移動する。

2．文字を挿入するには

⇨カーソルを挿入したい部分に持って行って、[SHIFT]＋[INS DEL]キーを押し、カーソルの右に空白部分を作ってから行ないます。

3．文字を削除するには

⇨カーソルを削除したい部分のすぐ後に持って行って、[INS DEL]キーを必要なだけ押す。

4．画面をきれいにするには

⇨[SHIFT]＋[CLR HOME]キーをおす。（＝CLS↵）

（[SHIFT]＋[CLR HOME]は「[SHIFT]キーを押しながら[CLR HOME]キーを押す」の意味です）

◇ コマンドのまとめ ◇

いままで出てきた基本的なコマンドについてまとめてみましょう。

1．CLS↵	画面をきれいにする。
2．NEW↵	記録されたプログラムを消す。
3．LIST↵	記録されているプログラムを画面に表示する。
4．RUN↵	プログラムを実行する。
5．RENUM↵	プログラムの行番号をつけなおす。
6．DELETE　n－m↵	プログラムのｎ番からｍ番までを消す。

5 スクリーンエディタ

ＢＡＳＩＣプログラムを作成するときに、誤って隣りのキーを押してしまったり、途中の文字を抜かしてしまったりする場合があります。また、行やステートメントを新しくつけ加えたり、今まであったものをとってしまったりする編集作業をすることがあります。このような時のために、いろいろなスクリーンエディタ（画面編集）機能が用意されています。

5.1 基本的な編集機能

(1)カーソルの移動

文字を修正したい時には、その修正箇所までカーソルを移動させます。カーソルを移動させるには以下のカーソルコントロールキーを使用します。

[→] を押すとカーソルは右へ移動します。画面の右端にカーソルがあるときにこのキーを押すと１つ下の行の左端に移動します。また、画面の一番下の行の右端にカーソルがあるときにこのキーを押すと、画面は１行上へスクロール（巻きあがり）し、カーソルは一番下の行の左端に移動します。

[←] を押すとカーソルは左へ移動します。画面の左端にカーソルがあるときにこのキーを押すとカーソルは１つ上の行の右端に移動します。また、画面の一番上の行の左端にカーソルがあるときにこのキーを押しても、カーソルは画面の右端に移動します。

[↑] を押すとカーソルは上へ移動します。画面の一番上の行にカーソルがあるときにこのキーを押してもカーソルは移動しません。

[↓] を押すとカーソルは下へ移動します。画面の一番下の行にカーソルがあるときにこのキーを押すと画面は１行上へスクロール（巻きあがり）します。

通常キーボードのキーを押し続けると同じ文字あるいは同じ動作が継続して入力されます（リピート機能）。カーソルコントロールキーも押し続けると押している間カーソルが連続して移動します。ＲＥＰＥＡＴ　ＯＦＦ↵と入力することにより、このリピート機能はなくなり、１文字ずつしか受け付けなくなります。再度リピート機能をはたらかせるにはＲＥＰＥＡＴ　ＯＮ↵と入力します。

カーソルを瞬時にホーム位置（画面の左上隅）に移動させるには[CLR HOME]キーを押します。また[SHIFT]＋[CLR HOME]（[SHIFT]キーを押しながら[CLR HOME]キーを押す）を押すとテキスト画面がすべて消され、カーソルはホーム位置に移動します。ただし、記憶されたプログラムは消えていませんのでＬＩＳＴ↵とすれば画面に表示できます。

(2)文字の削除

文字の削除を行なう場合には、削除したい文字の次の文字にカーソルを移動させた後、[INS DEL]キーを押します。この場合、カーソルの左にある文字が削除されカーソルから右の文字列が左へ移動します。

(3)文字の挿入

文字の挿入を行う場合には、文字を挿入したい位置の次の文字にカーソルを移動させた後[SHIFT]＋[INS DEL]キーを押します。この場合、カーソルから右の文字列が右へ移動し、カーソル位置にはスペース（空白）ができます。このスペース位置に追加したい文字を書き込みます。

このようなスクリーンエディタ機能を用いてプログラムの訂正を行なった場合には、各行番号の文の訂正が終わる度に、必ずキャリッジリターンキー（↵）を押してください。これにより、画面に表示されているステートメントの文字列がプログラムとしてメインメモリーに登録されます。

(例)「10 PLIINT CH$(65)」を「10 PRINT CHR$(65)」に訂正する場合次のようにしてください。□はカーソルを示します。

10 PLIINT CH$(65)□

[←] キーを押してカーソルを「L」の位置に移動させ[R]キーを押します。

10 PR[I]INT CH$(65)

[→] キーを押してカーソルを次の「I」の位置に移動させ[INS DEL]キーを押して「I」を1文字削除します。

10 PR[I]NT CH$(65)

[→] キーを押してカーソルを「$」の位置に移動させ[SHIFT]+[INS DEL]キーを押して「H」と「$」の間に1文字分のスペースを作ります。

10 PRINT CH□$(65)

[R] キーを押して「H」と「$」の間に文字「R」を挿入します。

10 PRINT CHR[$] (65)

[↵]キーを押してプログラムをメインメモリへ登録します。

以上のスクリーンエディタ機能を利用してプログラムの修正は行なえますが、画面編集をより効率よく迅速に行なうために、さらに便利な機能として次のような機能が用意されています。

5.2 知っていると便利な編集機能

(1)カーソルの移動

○水平TAB

画面の初期状態では水平TAB位置は画面左端から8文字単位に設定されています。[H TAB]キーを押すとカーソルは現在位置から次の水平TAB位置（初期状態では8文字右）へ瞬時に移動します。この機能を利用すると、文字の始まり位置を統一することが容易に行なえます。また、水平TAB位置は表示画面内で任意に設定できます。まず、設定されているTAB位置を取り消す場合にはその取り消す水平TAB位置へカーソルを移動させ[CTRL]+[Y] キーを押します。逆に新しく水平TABを設定する場合には、設定したい水平位置にカーソルを移動させ[CTRL]+[T] キーを押します。

○1ワード単位でのカーソル移動

[CTRL]+[F] キーを押すとカーソルは現在位置より右（先）にあるワードの先頭に移動します。また、[CTRL]+[B] キーを押すとカーソルは現在位置より左（後）にあるワードの先頭に移動します。ここで、ワードとは空白、英記号を除く連続した文字列のことです。ただし、コロン（:）はワードとみなされますが、πはワードとはみなされません。

(2)文字の削除

○カーソル位置からその行の終わりまでの削除

[CTRL]+[E] キーを押すとカーソル位置からその行の終わりまでの文字が削除されます。この後で、キャリッジリターンキーを押すことによりプログラムが修正されます。

○カーソルのある位置以降の画面クリア

[CTRL]+[Z] キーを押すとカーソル位置以降の画面をすべてクリアします。

(3)文字の挿入

文の途中で何文字か挿入したい場合には、文字を挿入したい位置の次の文字にカーソルを移動させ[CTRL]+[A] キーを押します。これで、インサートモードに設定され、それ以後に入力された文字がカーソル位置に表示されると共にカーソルより右の文字列が右へ移動していきます。インサートモードを解除するには[CTRL]+[S] キー（BREAKと同じ）を押します。また、カーソルコントロールキーやキャリッジリターンキーを押してもインサートモードは解除されます。また、インサートモード時に[INS DEL]キーを押して文字を解除してもインサートモードは解除されません。

(4)行の分割

1行の内容を2行にわたるように書き直したい場合には、区切りたい箇所へカーソルを移動させ、[CTRL]+[J] キーを押すと、カーソルからその行の終わりまでの文が、自動的にメインメモリから削除され、次の行へ移動します。移動した行の先頭に行番号をつけてキャリッジリターンキーを押せば

これまでの1行の内容が2行にわたって書かれます。

（例）次に示す行番号50で定義されているプログラムをCIRCLE文とPAINT文に分離してみましょう。

```
50 CIRCLE(100,100),50,1:PAINT(100,100),
HEXCHR$("AA00AA00AA00"),1
```

カーソルをPAINT文の「P」の位置に移動させ CTRL +J キーを押します。するとPAINT文以降が次の行へ移動します。

```
50 CIRCLE(100,100),50,1:
PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

SHIFT + INS DEL キーを押してPAINT文の前に行番号を挿入するためのスペースを作り、行番号60を入力します。

```
50 CIRCLE(100,100),50,1:
60 PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

キャリッジリターンキー↵を押して行番号60の文をプログラムとしてメインメモリへ登録します。

(5)行の結合

2行にわたる文を1行に納めたい場合には、行の結合機能を利用します。CTRL +W キーを押すと、カーソルのある水平1行と次の1行とが結合されます。よって2行にわたる文が1行として扱われ、これまでの2行の文と文の間隔は INS DEL キーを押して文字を削除することによりつめることができます。結合する2つの行のうち前の1行の文が画面の数行にわたるときには、カーソルをその文1行内の最下行に移動させてCTRL +W キーを押してください。

（例）次に示す行番号50と60で定義されているプログラムを1つの行番号の文にまとめてみましょう。

```
50 CIRCLE(100,100),50,1:
60 PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

行番号50のライン上にカーソルを移動させCTRL +W キーを押します。これにより行番号50と60の文が結合されました。この時点ではまだ、メインメモリへは登録されていません。

```
50 CIRCLE(100,100),50,1:
60 PAINT(100,100),HEXCHR$("AA00AA00AA00"),1
```

カーソルをPAINT文の「P」の位置に移動させ INS DEL キーを押すとPAINT文が左へ移動し画面左端まで移動します。さらに押し続けると1行上の50行の画面右端よりPAINT文が現われます。CIRCLE文の右にPAINT文を並べた時点で↵キーを押します。

```
50 CIRCLE(100,100),50,1:PAINT(100,100),HEXCH
R$("AA00AA00AA00"),1□
```

これにより、文番号50にCIRCLE文とPAINT文がプログラムとしてメインメモリへ登録されました。しかし文番号60にはまだPAINT文が残った状態となっていますので文番号60の文を消す必要があります。

```
50 CIRCLE(100,100),50,1:PAINT(100,100),HEXCH
R$("AA00AA00AA00"),1
60 ↵
```

文番号60のみを入力してキャリッジリターンキーを押すことにより文番号60のプログラムがメインメモリから削除されます。以上により2つの行番号で定義されていたプログラムが1つの文番号内に定義されました。

(6)文字のコピー

本機は画面に表示されている文字をそのまま別の場所にコピーして表示する機能を持っています。コピーをするにはまず、コピーする文字を表示したい場所にカーソルを移動しCOPY キーを押します。カーソルは点滅を停止し静止した状態になります。ここで、カーソルコントロールキーを押すと、静

止したカーソル位置から点滅したカーソルが現われます。このカーソルをコピーしたい文字の先頭まで移動させます。[↵]キーを押すと、点滅したカーソル位置にある文字が点滅を停止したカーソル位置へコピーされ、両カーソルは右へ移動します。コピーを終了するには再度[COPY]キーを押します。

このコピー機能によりコピー表示する行は水平1行にのみ有効です。次の行にわたる必要がある場合には以上のコピー操作を再度行なってください。

また、この[COPY]キーは次のようにプリンタへの画面コピー機能を持っています。

[SHIFT]＋[COPY]キーを押すとテキスト画面のみをプリンタにコピーします。

[GRAPH]＋[COPY]キーを押すとグラフィック画面のみをプリンタにコピーします。

[CTRL]＋[COPY]キーを押すとテキスト画面とグラフィック画面をプリンタにコピーします。

(7)行間への新しい行の追加

表示画面の行と行の間をひろげ新しく行を追加したい場合には、[ROLL UP]キーあるいは[ROLL DOWN]キーを使用します。[ROLL UP]キーを押すとカーソルのある行から上の行が1行上にスクロールします。また、[ROLL DOWN]キーを押すとカーソルのある行から下の行が1行下にスクロールします。これにより、行と行の間がひろげられ新しい行を追加できます。ただし、次の項(8)に示すように、エディットモードではこれらのキーの働きが異なります。

(8)ＥＤＩＴ機能

プログラムを実行させた場合、プログラムに何らかのエラーがあると、エラーの種類とそのエラーが生じた行番号を表示して実行が中断されます。このような場合ＥＤＩＴ文を使用することによりエラーの発生したプログラムの修正を能率よく行なえます。例えばＥＤＩＴ．と入力すると、エラーの発生した行を画面に表示して、カーソルはその行の先頭に表示され、エディットモードに入ります。エディットモードでは[ROLL UP]キーあるいは[ROLL DOWN]キーを使用することにより、巻き物を見るように、画面に表示されている文の前後を表示できます。この機能を利用して、訂正する文を画面に表示させ訂正します。エディットモードを解除するには[SHIFT]＋[CLR HOME]キーあるいは[SHIFT]＋[BREAK]キーを押してください。

エディットモードで[ROLL UP]または[ROLL DOWN]キーを使用して、プログラムを画面全体に表示してスクロールしている状態で、新しく行を追加するには[CTRL]＋[E]あるいは[CTRL]＋[Z]キーを使用して画面に空白部分を作成してから行番号付文をキー入力し、最後にキャリッジリターンキーを入力してください。

また、ＥＤＩＴ機能はエラーが発生した時のみでなく、プログラム作成後、プログラムの見直しを行なう場合に便利です。ＬＩＳＴ文の実行でもプログラムの見直しはできますが、画面上部へ消えていった行を表示するには再度ＬＩＳＴ文を実行する必要があります。

ＥＤＩＴ文は次の形式で表わします。

```
EDIT [{n}]
     [{.}]
```

n…行番号

.…ピリオド（.）にはエラーが発生してときの行番号が入ります。

(9)行番号の整理

プログラムを修正して行を取り除いたり、新しい行を追加すると行番号がばらばらになります。このようなとき、ＲＥＮＵＭ文を使用して行番号を整理することができます。ＲＥＮＵＭ文は次の形式で表わし、旧行番号mで指定した行以降の行番号を新行番号ℓで始まる行番号に増分nで付け換えます。この場合、プログラム内のＧＯＴＯ文などでジャンプする行番号も新しい行番号に書き換わります。

```
RENUM〔〔l〕,〔m〕,〔n〕〕
```

l…新行番号。省略すると１０。

m…旧行番号。省略するとプログラムの最初の行。（l≧m）

n…増分。省略すると１０。

5.3 全角文字の編集

本機は一般の英数カナ文字に加えて、日本語文字も簡単にテキスト画面に表示することができ、プログラム文内でも日本語文字を使用できます。ただし、標準解像度モード（縦 200ライン表示モニタ用）でテキスト画面が40×25、40×20、80×25、80×20行モード時には日本語文字は表示できません。一般の英数カナ文字が１バイトコードで表わされて半角文字と呼ばれるのに対して、日本語文字は２バイトコードで表わされ全角文字と呼ばれ、文字の大きさも英数カナ文字の水平方向が２倍の大きさで表示されます。

プログラムを編集する際に、全角文字は半角文字と取り扱い方やカーソルの動作が異なります。以下に、全角文字の編集方法について説明します。

⑴全角文字の表示

プログラムの作成中に全角文字を使用する場合には、まず CTRL ＋ XFER キーを押して日本語入力モードにします。画面の最下行に変換フィールドが表われますので、表示したい全角文字を指定してキャリッジリターンキーを押すとカーソル位置に指定した全角文字が２桁（１桁は半角文字１文字分を示します）の大きさで表示されます。再度 CTRL ＋ XFER キーを押すと日本語入力モードが解除されます。日本語入力モードに関しては「９章．日本語処理」の項を参照してください。

⑵全角文字上のカーソルの移動

カーソルコントロールキー ⇨ を操作してカーソルを全角文字上に移動するとカーソルは全角文字全体を覆う大きさに変わります。これは全角文字の１桁目にカーソル位置があることを示します。もう１回 ⇨ キーを押すとその全角文字の右半分のみの大きさに変わります。これは全角文字の２桁目にカーソル位置があることを示します。このように、全角文字列の中では ⇨ キーを２回押すことによりカーソルは１文字進みます。カーソルを ⇦ キーにより戻す場合にはカーソルはこの逆の動作を行います。

⑶全角文字の削除

全角文字の削除を行う場合には、削除したい全角文字の次の文字にカーソルを移動（次の文字が全角文字の場合にはカーソル位置はその文字の前半でも後半でもかまいません）させた後、INS DEL キーを押します。この場合、カーソルの左にある全角文字が削除されカーソルから右の文字列が左へ１文字分（２桁）移動します。

⑷全角文字の挿入

文字列の途中に全角文字を挿入する場合には、挿入したい位置の次の文字にカーソルを移動（次の文字が全角文字の場合には、カーソル位置はその文字の前半でも後半でもかまいません）させた後、SHIFT ＋ INS DEL キーを２回押します。この場合、カーソルから右の文字列が右へ２桁移動し、全角文字１文字分のスペースができます。このスペースに追加したい全角文字を書き込みます。

⑸全角文字の修正

入れまちがえた全角文字を正しい全角文字に置き換えたい場合には、まちがった全角文字の前半の位置にカーソルを移動させた後、正しい全角文字を入力します。もし、カーソルが全角文字の後半にあった場合に新しく全角文字を入力すると、修正したい全角文字の次にある文字を消してしまいます。

このような機能を利用してプログラムの訂正を行なった場合には各行番号の文の訂正が終わる度に、必ずキャリッジリターンキー（↵）を押してください。これにより、画面に表示されているステートメントの文字列がプログラムとしてメインメモリに登録されます。

（例）「１０　ＰＲＩＮＴ"本日語の文を使います。"」を
「１０　ＰＲＩＮＴ"日本語文字を使います。"」に訂正する場合次のようにしてください。
□はカーソル位置を示します。
１０　ＰＲＩＮＴ"本日語の文を使います。"□
⇦ キーでカーソルを「本」の位置まで移動します。
１０　ＰＲＩＮＴ"本日語の文を使います。"
COPY キーを押した後、点滅するカーソルを「日」の位置に移動させて↵キーを押して「日」

を「本」の位置にコピーした後、再度[COPY]キーを押してコピーモードを解除します。
10 PRINT"日[日]語の文を使います。"
[SHIFT]＋ [XFER] キーを押して日本語入力モードにした後、全角文字、ローマ字入力、音訓変換モードになっていることを確認して[H] 、[O] 、[X] 、 [XFER] と入力して「本」を指定して、[↵] キーを押します。
10 PRINT"日本[語]の文を使います。"
[⇨] キーを4回押してカーソルを「文」の位置に移動させます。
10 PRINT"日本語の[文]を使います。"
[INS DEL] キーを1回押して「の」を削除します。
10 PRINT"日本語[文]を使います。"
[⇨] キーを2回押してカーソルを「を」の位置に移動させます。
10 PRINT"日本語文[を]使います。"
[SHIFT]＋ [INS DEL] キーを2回押して「文」と「を」の間に全角文字1字分のスペースを作ります。
10 PRINT"日本語文[ ]使います。"
日本語入力モードの状態で[J] 、[I] 、 [XFER] と入力して「字」を指定して[↵] キーを押します。[SHIFT]＋ [XFER] キーを押して日本語入力モードを解除します。
10 PRINT"日本語文字[を]使います。"
[↵] キーを押して修正したプログラムをメインメモリへ登録します。

6 コントロールコード表

以下にコントロールコードの一覧表を示します。なお下の表で[CTRL]＋A は[CTRL]キーを押しながら[A]キーを押すことを意味します。

CTRL＋	コード	処　理　内　容
@	00	ダミー。
AまたはA	01	インサートモード1）にする。解除は[BREAK]または[↵] で行なえる。
BまたはB	02	現在のワード2）の先頭にカーソルを戻す。
CまたはC	03	実行を停止する。（＝[SHIFT]＋[BREAK]）
DまたはD	04	画面などを標準の状態に戻す。（＝INIT）
EまたはE	05	現在のカーソル以降1行を消す。
FまたはF	06	次のワードの先頭にカーソルを戻す。
GまたはG	07	ビープ音を鳴らす。（＝BEEP）
HまたはH	08	1文字分消して戻る。（＝[INS DEL]）
IまたはI	09	水平タビュレーションを行なう。(＝ [H TAB])
JまたはJ	0A	現在のカーソル以降を次の行に分ける。
KまたはK	0B	カーソルを画面のホーム位置に移す。（＝[CLR HOME]）
LまたはL	0C	テキスト画面を消去する。（＝[SHIFT]＋[CLR HOME]）
MまたはM	0D	キャリッジリターンをする。（＝[↵] ）
NまたはN	0E	現在のカーソルから上を上方向にスクロールする。（＝[ROLL UP]）
OまたはO	0F	現在のカーソルから下を下方向にスクロールする。（＝[ROLL DOWN]）
PまたはP	10	ダミー。
QまたはQ	11	実行の一時停止を解除する。
RまたはR	12	空白を挿入する。（＝[SHIFT]＋[INS DEL] ）
SまたはS	13	実行を一時停止する。3)（＝[BREAK]）
TまたはT	14	水平タビュレーション位置の新たな設定を行なう。

CTRL+	コード	処　理　内　容
UまたはU	15	ダミー。
VまたはV	16	ダミー。
WまたはW	17	現在のカーソルがある行と次の行をつないで1つにする。
XまたはX	18	ダミー。
YまたはY	19	現在のカーソル位置の水平タビュレーションの解除を行なう。
ZまたはZ	1A	現在のカーソルより下のテキスト画面をすべて消去する。
[	1B	ダミー。
¥	1C	カーソルを右へ移動する。（＝[⇨]）
]	1D	カーソルを左へ移動する。（＝[⇦]）
^	1E	カーソルを上へ移動する。（＝[⇧]）
_	1F	カーソルを下へ移動する。（＝[⇩]）
0		画面の背景色を黒（透明）にする。（＝COLOR，0）
1		画面の背景色を青にする。（＝COLOR，1）
2		画面の背景色を赤にする。（＝COLOR，2）
3		画面の背景色をマゼンタにする。（＝COLOR，3）
4		画面の背景色を緑にする。（＝COLOR，4）
5		画面の背景色をシアンにする。（＝COLOR，5）
6		画面の背景色を黄色にする。（＝COLOR，6）
7		画面の背景色を白にする。（＝COLOR，7）

テンキーの 0		文字グラフィックの色を黒（透明）にする。（＝COLOR0）
1		文字グラフィックの色を青にする。（＝COLOR1）
2		文字グラフィックの色を赤にする。（＝COLOR2）
3		文字グラフィックの色をマゼンタにする。（＝COLOR3）
4		文字グラフィックの色を緑にする。（＝COLOR4）
5		文字グラフィックの色をシアンにする。（＝COLOR5）
6		文字グラフィックの色を黄色にする。（＝COLOR6）
7		文字グラフィックの色を白にする。（＝COLOR7）
/		キャラクタゼネレータのROM／RAMを切り換える。（＝CGEN）
*		文字を点滅モードとノーマルモードとに切り換える。（＝CFLASH）
−		文字を反転モードとノーマルモードとに切り換える。（＝CREV）

1）インサートモード（insert　mode）　挿入したい文字キーを押すたびに、カーソルから右の部分が自動的に右に移動して、キーボードから入力した文字が挿入されるモード。

2）ワード（word）　隙間のない英数字の文字列をいいます。

3）プログラムの実行中は、キーを押している間だけ停止し、キーを離すと再び実行を開始します。

2章
フリーエリアについて

1 フリーエリア

本機では、CPUにZ80Aを採用しています。したがって、メインメモリは64Kバイトのメモリ空間を持っています。このうち、BASICインタープリタに使用している領域を除いた残りの部分がフリーエリアになります。

このフリーエリアの領域にプログラムやデータが格納されるわけですが、それらがフリーエリアより大きくなるとメモリが足りなくなり"Out of memory"エラーになります。

したがって、大きなプログラムを走らせたり、多量のデータを扱うためには、大きなフリーエリアを確保することが必要ですがメインメモリは64Kバイトと制限されていますので、この中でフリーエリアを増やそうとすると、BASICを小さくするしかありません。

しかし、BASICを小さくすると、その分機能が減り、かえって使いにくいものになってしまいます。

そこで、本機ではグラフィックVRAMを変数領域として使用できるようにし、その分メインメモリのプログラム領域を増やし、フリーエリアの増大を図っています。

1．フリーエリア

本機のメモリはBIOSROM、メインメモリ、グラフィックVRAMなどから構成されています。

BASICは、このうちのメインメモリにロードされ、残りの領域にプログラムテキストやプログラムに使用される変数が置かれ通常この部分をフリーエリアといいます。

しかし、フリーエリアは、プログラムの実行前と実行後やグラフィックVRAMを変数に使うか否かなどによって異なります。

そこで、もう少しフリーエリアについて詳しく見ていくことにします。

1.1 BASICの起動直後

BASICを起動した直後はメインメモリにはBASICシステムとそれに必要なワークなどがとられるだけです。また、グラフィックVRAMは変数として使用されるモードとなっていますので、図－1のようになります。

1.2 プログラムロード後（実行前）

プログラムをロードするとBASICシステムの後ろにプログラムテキストが置かれます。したがって、フリーエリアはメインメモリからBASICシステムワークエリア、プログラムテキストを除いた部分と、グラフィックVRAMの部分で図－2のようになります。

1.3 プログラム実行後

プログラムを実行するとプログラムテキストの後ろにプログラムで使用される変数がとられます。またVDIM命令を使っていれば、グラフィックVRAMにも変数がとられます。

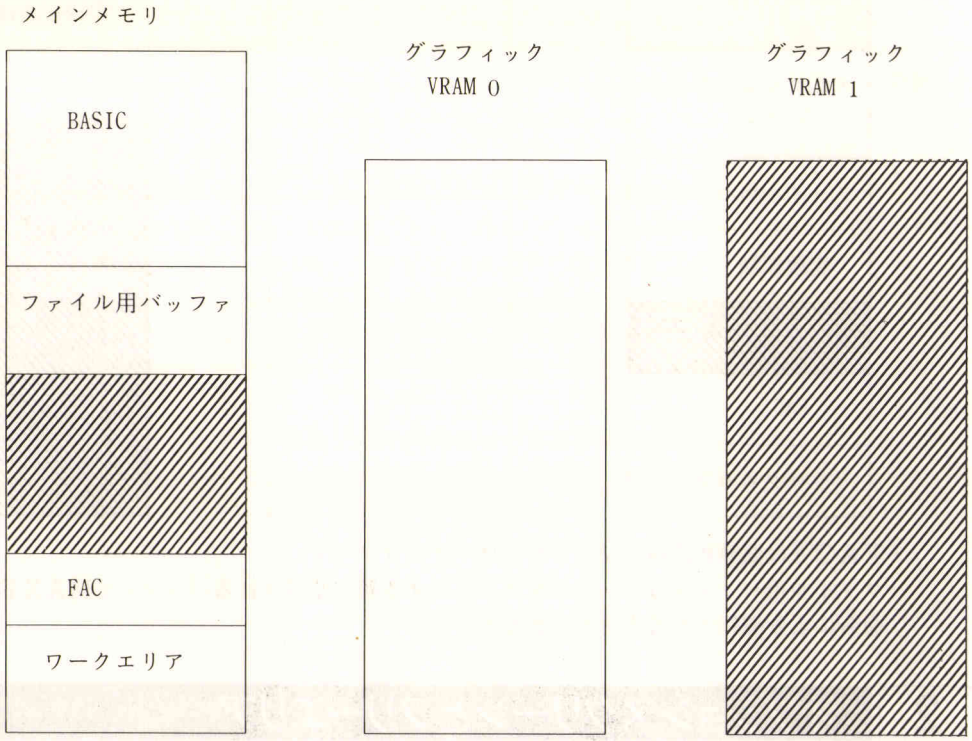
この場合のフリーエリアは図－3で示すようにプログラムロード後のフリーエリアからさらに変数エリアを除いた部分になります。

1.4 グラフィックVRAMを変数として使用しない場合

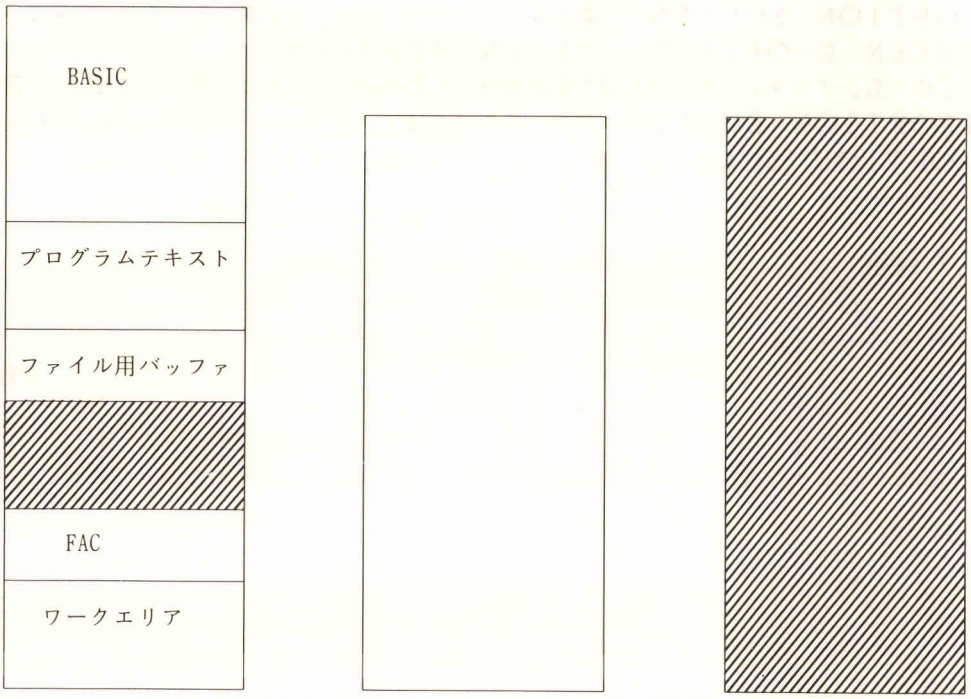
高解像度グラフィックを使用する場合、グラフィックＶＲＡＭのバンク１は、グラフィック描画用に使用され、変数を格納することはできません。また、外部記憶として使用する場合も同様に変数を格納することはできません。

これらの場合、メインメモリの部分のみがフリーエリアとなります。

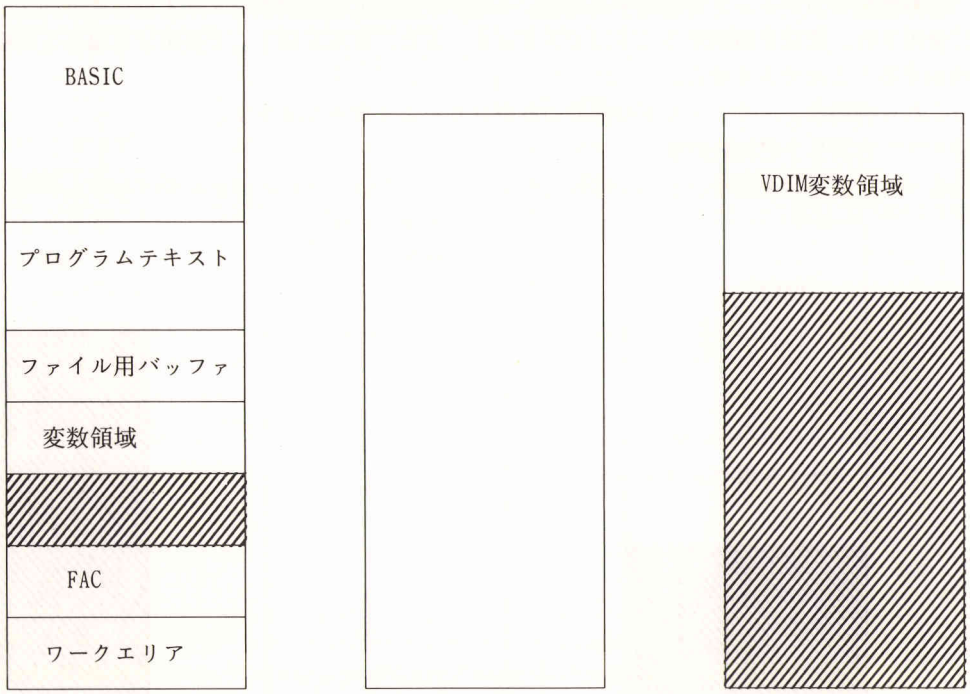
図－１　ＢＡＳＩＣ起動直後



図－２　プログラムロード後（実行前）



図－3　プログラム実行後



・FACとは演算処理時に使用されるワークエリアです。
・ファイル用バッファはファイルをアクセスする際に使用されるバッファでMAXFILES命令により確保される大きさが異なります。

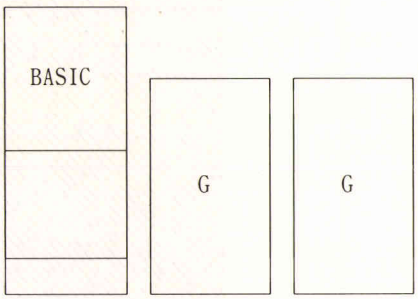
2　オプションスクリーンとフリーエリア

グラフィックVRAMをどのような用途で使用するかを決めるにはOPTION SCREENという命令を使います。

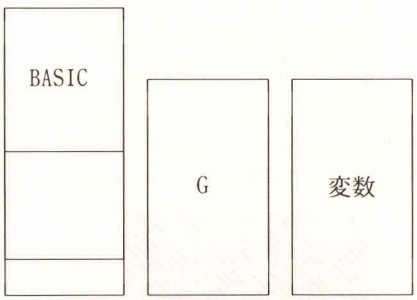
OPTION SCREENで定義されるスクリーンモードは全部で5つあり、OPTION SCREEN 0～OPTION SCREEN 4で設定します。

このうち、グラフィックVRAMを変数領域として使用できるのはOPTION SCREEN 1とOPTION SCREEN 2で、それ以外はすべてグラフィックまたは外部記憶とします。

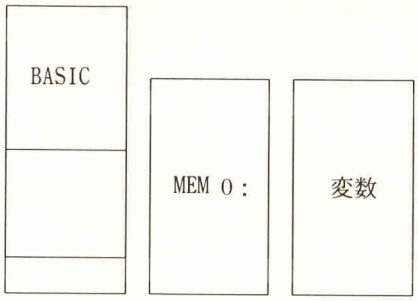
(1)OPTION SCREEN 0



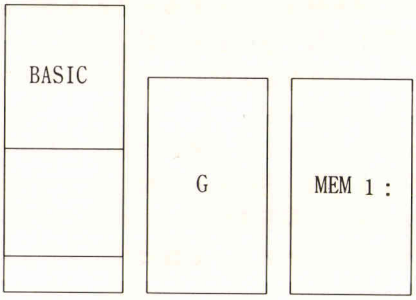
(2)OPTION SCREEN 1



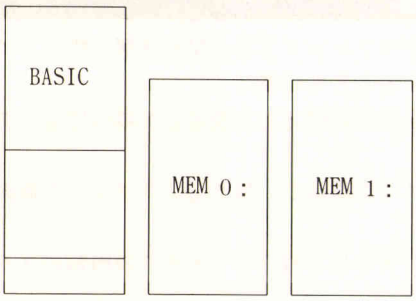
⑶OPTION SCREEN 2



⑷OPTION SCREEN 3



⑸OPTION SCREEN 4



Gはグラフィック描画用

MEM0:,MEM1:はデバイス名を示します。

⑴OPTION SCREEN 0

グラフィックVRAMのバンク0、バンク1ともグラフィックの描画に使用されます。

その結果、グラフィックとしては次の画面モードで使用できます。

640×400	1画面	320×400	2画面
640×384	1画面	320×384	2画面
640×200	2画面	320×200	4画面
640×192	2画面	320×192	4画面

このモードではVDIM、VDIM CLEAR、INIT"MEM0:"、INIT"MEM1:"の命令を実行しようとすると"Bad screen mode"エラーとなります。

また、変数、プログラム領域として使用できるのはメインメモリ上のみです。

⑵OPTION SCREEN 1

グラフィックVRAMのバンク0は、グラフィック描画用として使用し、バンク1を変数領域として使用します。

画面モードとしては、

640×200	1画面	320×200	2画面
640×192	1画面	320×192	2画面

となり、スクリーンの解像度は制限されますが、VDIM命令によりグラフィックエリアに変数を確保することができます。

このモードでのフリーエリアはメインメモリ上の領域とグラフィックVRAMのバンク1を合せた部分となります。

⑶OPTION SCREEN 2

グラフィックVRAMのバンク0はグラフィック描画用として使用し、バンク1を外部記憶として使用します。

したがって、フリーエリアはメインメモリ上の領域とグラフィックVRAMのバンク1を合せた部分となります。

このモードではVDIM、VDIM CLEAR、INIT"MEM0:"は使用できますが、INIT"MEM1:"とすると"Bad screen mode"のエラーとなります。

⑷OPTION SCREEN 3

グラフィックVRAMのバンク0はグラフィック描画用として使用し、バンク1を外部記憶として使用するモードです。

プログラムテキスト、変数は共にメインメモリ上にのみ確保されます。

⑸OPTION SCREEN 4

グラフィックVRAMのバンク0、バンク1共に外部記憶用として用います。

プログラムテキスト、変数は共にメインメモリ上にのみ確保されます。

3 グラフィック描画と外部記憶

OPTION SCREEN 2、3、4のモードではグラフィックVRAMを外部記憶として使用できますが、このメモリ上にグラフィックデータを描画することも可能です。

すなわち、次の点に注意すれば、グラフィックVRAMを外部記憶として設定した場合でも、グラフィックを描画することもできます。

また、48Kバイトのうち一部を外部記憶として、ファイルを格納し、残りをグラフィック描画にすることも可能となります。

⑴グラフィックVRAMを外部記憶として、設定した場合、WIDTH命令でメモリの内容がクリアされることはありません。

⑵グラフィックVRAMに対し、INIT命令を実行すると、グラフィックVRAMにはメモリ管理のためのデータが書き込まれます。

⑶外部記憶として、使用した場合でも、グラフィック命令によって、グラフィックを描画できますので、ファイルと共存して使用する場合は注意が必要です。

たとえばCLS命令などによってファイルをこわさないようにしてください。

ファイルはグラフィックVRAMの上位アドレス（青のメモリ）から順に書き込まれていきますので、上位16Kバイトにファイルを格納し、残り32Kバイト（赤と緑）をグラフィックに使うということも可能です。

4 日本語入力とフリーエリア

BASICを起動した時点では、日本語入力は音訓入力モードになっています。音訓入力モードでは音訓辞書をディスクから読み込んで変換を行ないますので、音訓辞書をアクセスするためのルーチンが必要です。

このルーチンはメインメモリ上のF000H番地からF3FFH番地を使用するため、

```
CLEAR &HF000
```

となっています。

したがって、音訓変換を使用しない場合には、

```
CLEAR &HF400
```

とすることができますので、フリーエリアが1Kバイト増加します。

5 NEWONとフリーエリア

本機のBASICはゲーム、ビジネス、教育用などあらゆる用途に対応できるため、ひじょうに多くの命令を含んでいます。

その結果、短いステップで豊富な処理を扱うことができます。

しかし、実際のプログラムではすべての命令を使用しているわけではありません。

たとえば、外部デバイスにディスクしか使わないプログラムではカセットの命令やRS－232C

の命令は不要です。それらの命令に使用しているメモリ領域をユーザー用のフリーエリアとすればより大きなプログラムやデータが扱えます。

そこで、プログラムで使用しない命令を削除してフリーエリアを確保しようとする場合に使われるのがNEWON命令です。

この命令によってBASICをある程度カスタマイズすることができます。

NEWONで削除される命令群は10のレベルがあり0～9の番号で指定し番号が小さくなるほどレベルは高くなります。

レベルの高い番号を指定するとそれより低いレベルの命令群はすべて削除されます。

たとえば、NEWON0とするとNEWON0～NEWON9のすべての命令が削除されます。

NEWON命令で一度削除された命令はBOOT命令もしくはIPLリセットで再度BASICを起動しない限り使用できません。

（例）

n	
省　略	すべてのコマンド、ステートメント、関数など使用可
9	MIRROR$,KANJI$,DTL,RANDOMIZE,WAIT, KEYLIST,KLIST,KBUF,CANVAS,LAYER,TVPW, CHANNEL,VOL
8	VERIFY,LOAD?,"CAS:",CMT,CMT関数,REW, FAST,EJECT,APSS
7	"COM:",ON COM GOSUB,COM ON/OFF/STOP, POSITION,PATTERN,CIRCLE@,SCROLL,STICK, STRIG,PUSH,POP,ATTR$,RUN"?.Sys"
6	CRT,ON KEY GOSUB,KEY ON/OFF/STOP ON TIME$ GOSUB,TIME$ ON/OFF/STOP
5	MKDIR,CHDIR,RMDIR,HDOFF,SET,NAME,FPOS
4	MOUSE,MOUSE関数,HCOPY
3	GET@,PUT@,CGPAT$,DEVI$,DEVO$,LPOUT CONSOLE#,COPY
2	LIST,LLIST,DELETE,RENUM,AUTO,EDIT, TRON,TROFF,SAVE,SEARCH,KILL,CONT, エラーメッセージ
1	PLAY,MUSIC,TEMPO,SOUND,CBLACK
0	WINDOW,LINE,PSET,PRESET,CIRCLE,POLY, PAINT,SYMBOL,POINT,PRW,CLICK,NEWON n

6 フリーエリアの大きさを確認するには

フリーエリアはＦＲＥ関数によって内容を確認することができます。

ＦＲＥ関数は

ＦＲＥ（ｎ）

という書式で書くことができ、ｎの値が１のときメインメモリのフリーエリア、ｎの値が２のときグラフィックＶＲＡＭのフリーエリア、ｎの値が０のとき、それらの合計がこの関数の値になります。

3章
ファイルについて

1 ファイル

「ファイル」とは、日常、「書類などの情報をバインダーなどにとじたもの」の意味で使われていますが、ＢＡＳＩＣの「ファイル」もこれとほとんど同じ意味をもっており、違うのは、情報を記録しておくものが「バインダー」ではなく、「電気や磁気による記憶装置（デバイス）」になっている点です。

ファイルは、一つの単位として取り扱われる関連したレコード（ｒｅｃｏｒｄ）から成り、そこに、プログラムやデータの集合が記録されます。

2 ファイルの種類

ファイルは、アクセス1）の仕方によって、シーケンシャルアクセスファイルとランダムアクセスファイルに分類されます。

1）アクセス（ａｃｃｅｓｓ）　メモリやＩ／Ｏを読み書きすることは、すべてアクセスといいますが、ここでは、「ファイルを読み書きする」の意味で使用しています。

2.1 シーケンシャルアクセスファイル

ファイル中のデータが書き込まれた順に並んでおり、データを読み書きする際には、先頭から順番にしかできないようなファイルのことを、シーケンシャルアクセスファイルといいます。

一般に、このファイルではデータが順序良く並んでいて、無駄がありませんが、特定のデータだけを読み込むことはできません。そのため、データの内容を変更する場合は、内容を変更しないデータも順に読み出し、処理を行なった後、別のファイルに書き込まなければなりません。

2.2 ランダムアクセスファイル

ファイル中の特定のデータを内容のインデックスによって、直接その記憶位置を指定して読み書きできるようになっているファイルを、ランダムアクセスファイルといいます。

この場合、データは１レコード（＝２５６バイト）単位で読み書きされます。

このファイルでは、任意のデータを指定できるので、シーケンシャルアクセスファイルではできなかった、特定のデータの変更、追加、削除をすることができます。

3 デバイス

バインダーにとじたファイルは書棚にしまって整理するように、コンピュータでも、ファイルを「デバイス」と呼ばれる書棚にしまって整理します。

デバイスには、次のようなものがあります。

- フロッピーディスク
- カセットテープ
- グラフィックメモリ
- 外部メモリ（EMM）
- ハードディスク
- ディスプレイテレビ
- キーボード
- プリンタ
- ＲＳ－２３２Ｃ

4 ファイルの管理

書棚にしまってあるファイルには見出しを付けておくように、デバイスにしまってあるファイルにも名前が付けられています。

さらにファイルを探しやすいように名前などの情報はディレクトリとよばれる場所にまとめて書かれています。

ファイルの管理はこのディレクトリによって行なわれていますが、本機ではそれを階層ディレクトリと呼ばれる方法によって行なっています。

以下階層ディレクトリについて説明します。

現在、パーソナルコンピュータの記憶媒体として、フロッピーディスクが多く使われており、比較的安価で持ち運びに便利なため、保存するファイルの数が増えると、何枚かのフロッピーディスクに分けて記録し、それぞれにインデックスシールを貼って区別するようにします。ところが、ハードディスクのようにフロッピーディスク３０枚分ものデータを扱えるデバイスでは、１台の中にすべてのファイルを保存することができます。このように、何枚ものフロッピーディスクに分けて行なっていたことを１つのハードディスクで行なおうとすると、ファイルの管理上問題が生じてきます。

たとえば、ディスクに入っているファイルのリストを見るためにＦＩＬＥＳコマンドを実行するとします。このとき、ファイルの数が１０や２０であれば、その中から目的のファイルを見つけ出すことはそれほど苦になりませんが、ファイルの数が１００、２００となった場合は、考えただけでも探す意欲がなくなってしまいます。

そこで考え出されたのが階層ディレクトリと呼ばれるファイルの管理方法です。

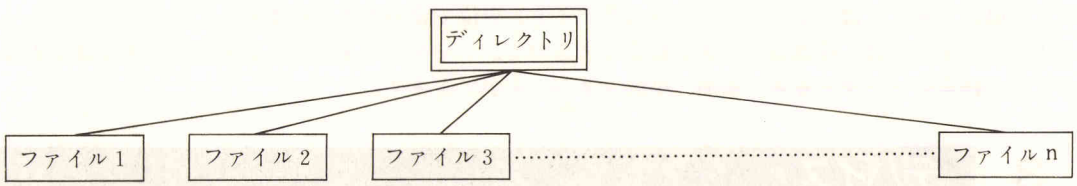
4.1 ディレクトリ

ディレクトリというのは、ディスクの中に記憶されているファイルの登録簿のことで、ちょうど本の目次に当たります。すなわち、ディスクにおいて、ファイル名やその種類、属性、ディスク上の位置、サイズ、作成年月日などファイルに関する情報が書かれている部分をいいます。

ＦＩＬＥＳコマンドを実行すると、このディレクトリを参照して、その中に含まれているファイル名の一覧表が表示されます。

4.2 単層ディレクトリ

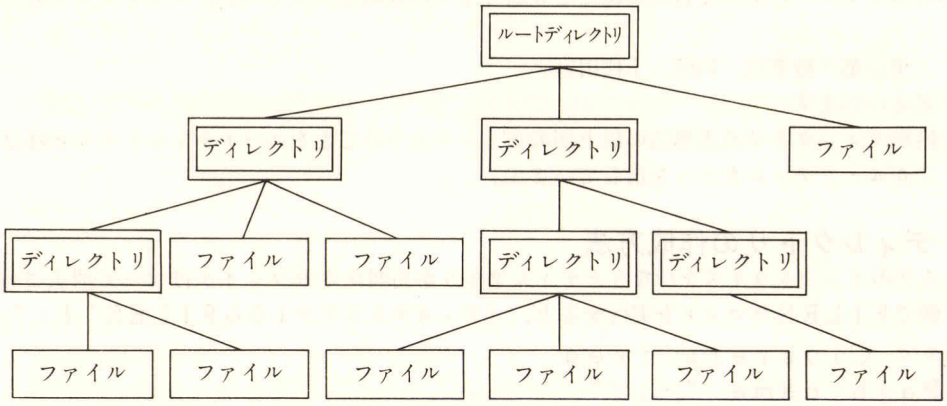
従来のファイルの管理は、ディスク１枚に１つのディレクトリ、すなわち、下図のような単層で行なわれていました。



このとき、ＦＩＬＥＳを実行するとファイル１、ファイル２、……、ファイルのすべてのファイル名が一度にリストされることになります。

4.3 階層ディレクトリ

容量の大きなデバイス上のファイルを効率的に管理するために考え出されたものが、下図のような木構造をもつ階層ディレクトリです。

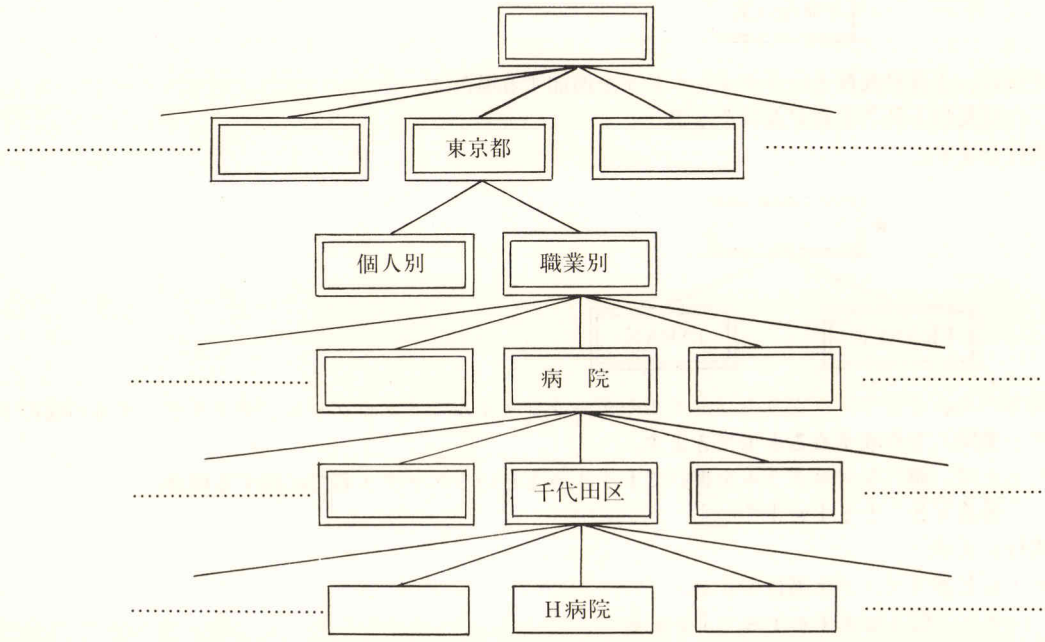


たとえば、国内のHという病院の電話番号を電話帳を使って調べる場合を考えます。H病院が東京都千代田区にあったとすると、まず東京都２３区の電話帳が必要となります。電話帳は個人別のものと職業別のものとに分かれていますが、病院を調べる場合は職業別のものを使います。次に電話帳の目次から病院の項のページを調べ、そのページを開きます。すると、病院名と住所、電話番号が列記してありますが、各区ごとに項目が分かれています。そこで今度は千代田区の項目を探し出し、さらにその項のところに列記してある病院の中からH病院を探し出し、電話番号を知ることができます。つまり、H病院の電話番号を見つけるまでの経路を示すと、

電話帳の山⟶東京都⟶職業別⟶病院⟶千代田区⟶H病院

となります。

この例を、階層ディレクトリに当てはめると、東京都、職業別、病院、千代田区がディレクトリに対応し、H病院というのがファイルに対応します。



本機の階層ディレクトリでは階層の経路を表わすのにスラッシュ（／）を使用します。
したがって前記の例は次のように表わされます。

／東京都／職業別／病院／千代田区／

このときファイルにあたるH病院にたどり着くまでの経路をパスといい、ディレクトリを／で区切った

／東京都／職業別／病院／千代田区／

をパス名といいます。

また階層ディレクトリの木構造の最上部のディレクトリのことをルートディレクトリと呼び、先頭の"／"がルートディレクトリを指しています。

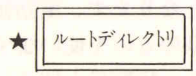
4.4 ディレクトリの作成方法

ディスクのイニシャライズを行なうとディレクトリが初期化されファイルはすべて消去されます。この状態でＦＩＬＥＳコマンドを実行すると、（ディスクドライブ１ならＦＩＬＥＳ"１："↵）

```
××  Clusters  free
Path  name  "×:／"
```

と表示されます。

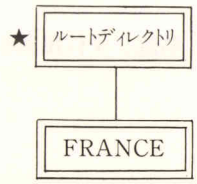
この表示はディスク上にルートディレクトリのみが存在していることを示しています。



ここに新しくディレクトリを作成する場合、ＭＫＤＩＲというコマンドを使用します。
たとえば、ルートディレクトリの下にＦＲＡＮＣＥというディレクトリを作成する場合、

```
MKDIR"FRANCE"↵
```

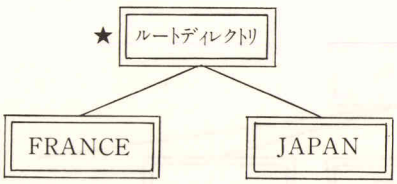
とします。



さらに、ＪＡＰＡＮというディレクトリを追加する場合は、

```
MKDIR"JAPAN"↵
```

を実行します。



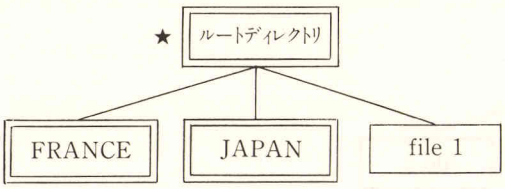
もちろん、ここでプログラムファイルやデータファイル（「プログラム、データファイルの保存と再生」参照）を作成することもできます。
たとえば、適当なプログラムを書いてｆｉｌｅｌというファイル名で記録する場合

```
SAVE"filel"↵
```

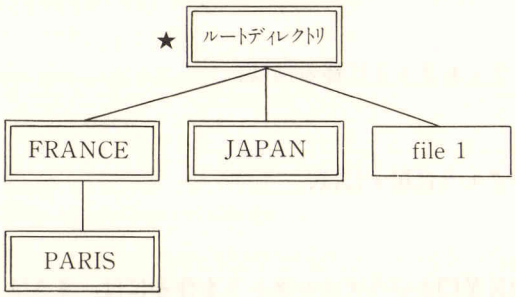
を実行します。
ＦＩＬＥＳコマンドを実行すると、

```
××  Clusters  free
Path  name  "×:／"
Dir＊        "×:FRANCE     .DIR"
```

```
Dir*          "×:JAPAN      .DIR"
Bas           "×:file1      .    "
```
と表示されます。



FRANCEというディレクトリの下にPARISというディレクトリを作るには、
```
MKDIR"FRANCE／PARIS"
```
を実行します。



さらに下の階層を作る場合も同様に"／"で区切って追加します。
```
MKDIR"……／……／……／……"
```

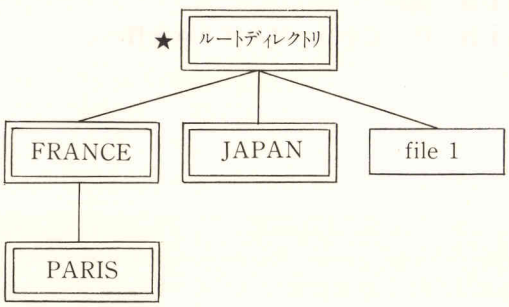
4.5 カレントディレクトリの変更

階層ディレクトリ構造では各ディレクトリはそれぞれ独立してファイルのアクセスを行ないます。したがって、同じファイル名のファイルを別々のディレクトリの中に作成することができます。

ファイルをアクセスする場合、そのファイルの存在するディレクトリに移動します。その後はそのディレクトリを起点としてファイルを指定することができます。このディレクトリのことをカレントディレクトリと呼びます。カレントディレクトリの移動にはCHDIRコマンドを使います。

たとえば、カレントディレクトリをルートディレクトリからFRANCEというディレクトリに移す場合、
```
CHDIR"FRANCE"
```
を実行します。

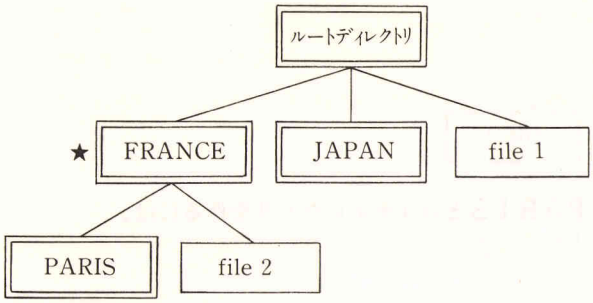
このときFILESコマンドを実行すると、
```
×× Clusters free
Path name  "×:／FRANCE／"
Dir*       "×:PARIS      .DIR"
```
と表示されます。



ここで、プログラムをｆｉｌｅ２というファイル名で記録する場合、

SAVE"ｆｉｌｅ２"↵

とします。



ここからさらにその下のＰＡＲＩＳというディレクトリに移るには、

CHDIR"PARIS"↵

を実行します。

またカレントディレクトリをルートディレクトリに戻すには、

CHDIR"／"↵

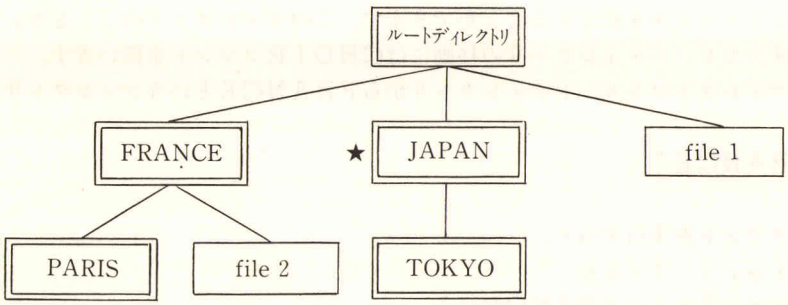
とします。

ＪＡＰＡＮというディレクトリの下にＴＯＫＹＯというディレクトリを作るには、さきほどの方法で、

MKDIR"JAPAN／TOKYO"

としてもよいのですが、いったんカレントディレクトリをＪＡＰＡＮに移してから作る方法もあります。

CHDIR"JAPAN"↵
MKDIR"TOKYO"↵



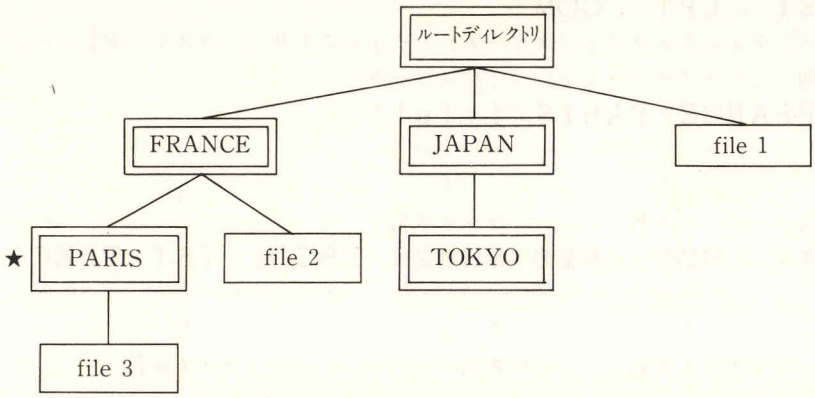
この状態でＰＡＲＩＳというディレクトリの下にファイルを作りたいときは、

CHDIR"／FRANCE／PARIS"↵

を実行して、カレントディレクトリをＰＡＲＩＳに移してから、プログラムを作り、

SAVE"ｆｉｌｅ３"↵

とします。



ここでＦＩＬＥＳを実行すると、

```
××  Clusters  free
Path  name  "×:／FRANCE／PARIS"
Bas         "×:file3        "
```

と表示されます。

4.6 ディレクトリの削除

不要なディレクトリを削除するにはＲＭＤＩＲというコマンドを実行します。

今、カレントディレクトリをＪＡＰＡＮとして、その下のＴＯＫＹＯというディレクトリを削除する場合、

```
CHDIR"JAPAN"
RMDIR"TOKYO"
```

とします。

このときディレクトリ"ＴＯＫＹＯ"にあるファイルおよびディレクトリはあらかじめ消去されている必要があります。

もしファイルが消去されていなければＫＩＬＬコマンドで消去してください。

5 ファイルの指定

ＢＡＳＩＣからファイルをアクセスするときは、シーケンシャルアクセスファイル、ランダムアクセスファイルのいずれの場合も、まず「書棚の所まで行って、使用したいファイルを取り出し、机の上に運んで開く」という操作をしなければなりません。このことを「ファイルをオープンする」といいます。

コンピュータには、デバイスと呼ばれる何種類かの書棚があり、いくつかのファイルを同時に使用できるように、机が何台か用意されています。

ファイルをオープンするには、書棚と使用したいファイルと机を決定する必要があります。書棚、使用したいファイル、および机を決定することを、ファイルの指定といいます。

ファイルの指定は、どの書棚かを示すデバイス名、分類を示すパス名、使用したいファイルを示すファイル名、および机を示すファイル番号によって行ないます。

ファイルの指定は、ファイルディスクリプタと呼ばれる指定子によって行ないます。ファイルディスクリプタは、次の書式で書かれます。

```
"〔デバイス名：〕〔パス名〕〔ファイル名〕"
```

デバイス名：……０：～３：、Ｆ０：～Ｆ３：、ＨＤ０：～ＨＤ３：、ＣＡＳ：
ＭＥＭ０：ＭＥＭ１：、ＥＭＭ０：～ＥＭＭ９：、ＣＲＴ：、ＳＣＲ：

KEY：、LPT：、COM：

パス名……〔／〕ディレクトリ〔；パスワード〕／ディレクトリ〔；パスワード〕／…／

ファイル名……名前〔，エクステンション〕〔；パスワード〕

(例) LOAD"0：／FRANCE／PARIS／file1"

↑ デバイス名　↑ パス名　↑ ファイル名

OPEN"R"，#1，"HD0：／営業部；SH023／PROG1．TST；TAKO"

↑ デバイス名　↑ パス名　↑ ファイル名

6 デバイス名

デバイス名は、ファイルの存在する書棚（デバイス）を指定するもので、半角文字で最大4文字の文字列とそれに続くコロン（：）で表わされます。

次に示すような種類があります。

デバイス名	指定されるデバイス	用途		
		シーケンシャル入力	シーケンシャル出力	ランダム入出力
0： ｜ 3：	ディスク 0（3or5 インチ版） ｜ ディスク 3（3or5 インチ版）	○ ｜ ○	○ ｜ ○	○ ｜ ○
F0 ： ｜ F3 ：	ディスク 0（8インチ版） ｜ ディスク 3（8インチ版）	○ ｜ ○	○ ｜ ○	○ ｜ ○
HD0 ： ｜ HD3 ：	ハードディスク 0 ｜ ハードディスク 3	○ ｜ ○	○ ｜ ○	○ ｜ ○
CAS ：	カセットテープ	○	○	
MEM0： MEM1：	グラフィックメモリ 0 グラフィックメモリ 1	○ ○	○ ○	○ ○
EMM0： ｜ EMM9：	外部メモリ 0 ｜ 外部メモリ 9	○ ｜ ○	○ ｜ ○	○ ｜ ○
CRT ： SCR ：	ディスプレイテレビ ディスプレイテレビ		○ ○	

KEY ：	キーボード	○		
LPT ：	プリンタ		○	

通信デバイス

COM ：	RS-232C ポート	○	○	

＊COM：は特にシーケンシャル入出力用として使用することができます。

7 パス名

パス名は、アクセスしようとするファイルが木構造をもつディレクトリのうちのどこにあるかを示します。

パス名は、次の規則に従ってつけられます。

a）パス名はスラッシュ（／）によっていくつかの階層部分に区切られています。

／ディレクトリ／ディレクトリ／……／

b）1つのディレクトリは、半角文字で1～13文字の文字列で表わされます。このとき、全角文字は、1文字で2文字分と数えます。

c）各ディレクトリの後ろにセミコロン（；）をつけて、続けてパスワードを指定することができます。パスワードは、ファイル内容の秘密を守るためのもので、いったん指定すると、以後パスワードを省略してファイルをアクセスすることができなくなります。

／ディレクトリ；パスワード／……／

d）デバイス名、ディレクトリ、セミコロン（；）、パスワードを合わせて、半角文字128文字以内でなければなりません。

デバイス名：／ディレクトリ；パスワード／……／

128文字以内

e）パス名の先頭のスラッシュ（／）はルートディレクトリを指しています。したがって、先頭のスラッシュを省略すると、現在のカレントディレクトリの下のディレクトリを指定することになります。

f）パス名の指定の仕方が違うときは、「Bad file descripter」のエラーが出ます。

（例）／営業／鈴木／

／営業／鈴木；AYAKO／

／技術；T007／田中；Yoshio／

8 ファイル名

ファイル名は、使用するファイルの名前でデバイス名で指定されたデバイス上のどのファイルかを示します。

ファイル名は、次の規則に従ってつけられます。

a）ファイル名は3つの部分に区切ることができます。

名　前〔.エクステンション〕〔;パスワード〕

b）名前は、半角文字で1～13文字の文字列、エクステンションは1～3文字で表わされます。ファイル名は、名前、ピリオド、エクステンション、セミコロン、パスワードをあわせて31文字以内でなければなりません。

このとき、全角文字は1文字で2文字分と数えます。

名前，エクステンション；パスワード

３１文字以内

c）名前が１３文字より長かったり、エクステンションが３文字より長い場合およびファイル名が３１文字を越えた場合は、「Ｂａｄ　ｆｉｌｅ　ｄｅｓｃｒｅｐｔｅｒ」のエラーが出ます。

d）パスワードはファイルの機密を保持するためにつけ、いったん指定すると以後パスワードを省略してファイルをアクセスすることができなくなります。

（例）ＴＥＳＴ．Ａｓｃ
　　在庫管理．Ｂｉｎ
　　Ｔｅｌｅｐｈｏｎｅ　ｌｉｓｔ．Ｂａｓ；ＫＡＭＩＳＡＮ

9 ファイル番号

ファイル番号は、ファイルのオープン時、用意される机の番号で、０～１５の整数で表わされます。
使用する机（ファイル）の数は、ＭＡＸＦＩＬＥＳステートメントで指定することができます。
（例）ＭＡＸＦＩＬＥＳ　２……１、２の２つの机（ファイル）を使用することができます。

4章
プログラム、データファイルの保存と再生

本機のメインメモリにキーボードから入力したプログラムは、いったん電源を切ると、メインメモリ上には保存されません。そこで、カセットテープやフロッピーディスクなど外部ファイルへの保存が必要になります。また、プログラム中で多くのデータを処理したい場合にも、外部ファイルにデータの保存・再生が必要になります。ＢＡＳＩＣでは、このプログラム・データの保存・再生をそれぞれプログラムファイル、データファイルという形で行ないます。ここではプログラム・データの保存および再生とそれに関するコマンドを説明します。

1 プログラムファイル

コンピュータのメインメモリのプログラムは、カセットテープやフロッピーディスクなどの外部ファイルにプログラムファイルとして保存・再生が可能です。その保存・再生は次のコマンドを使用して行ないます。

SAVE、LIST、SAVEM、VERIFY、LOAD?、INIT
OPTION SCREEN、KILL、DEVICE、FILES、
LFILES、RUN、MERGE、CHAIN

1.1 プログラムの保存

メインメモリ内のプログラムを外部ファイルに記録することを、プログラムの保存（セーブ）といい、ＳＡＶＥ，ＬＩＳＴ，ＳＡＶＥＭコマンドによって実行できます。

SAVE〔"〔デバイス名：〕ファイル名"〕〔,A〕

デバイス名……０：～３：，ＣＡＳ：，ＭＥＭ０：～ＭＥＭ１：，ＥＭＭ０：～ＥＭＭ９：
Ｆ０：～Ｆ３：，ＨＤ０：～ＨＤ３：，ＣＯＭ：

メインメモリ内のＢＡＳＩＣプログラムを指定されたデバイスへ指定されたファイル名を付けてセーブします。また、最後にＡを指定するとプログラムをアスキー形式でセーブします。アスキー形式とはプログラムをキャラクタコード（アスキーコード）に変更してセーブすることです。Ａを指定しないときが一般的な記録方式であり、プログラムは中間コードに変換されてセーブされます。アスキーセーブは一般的なセーブよりも多くのファイル領域が必要になりますが、後でのべるＭＥＲＧＥコマンドで使用することができます。

〔注意〕①ファイル名は、プログラムをセーブするときにファイルにつける名前です。名前は、英数字、カナ、記号、日本語を使用し半角文字で１３文字（２バイト文字は２文字として扱かいます）までです。なお、名前の中に．"：；，は使用できません。

②グラフィックメモリにプログラムをセーブする場合、あらかじめ次の２つのコマンドを実行しておく必要があります。

ＯＰＴＩＯＮ　ＳＣＲＥＥＮ　ｎ　（ｎ：０～４）

グラフィックメモリを外部記憶装置として使用することを定義します。その場合、ｎの値は２～４を指定します。詳しくは、「ＢＡＳＩＣリファレンスマニュアル」を参照してください。

ＩＮＩＴ"ＭＥＭ｛０／１｝："

グラフィックメモリをイニシャライズしセーブできる状態にします。ダイレクト命令で実行した場合、「Are you sure?（y or n）」と表示されますのでYキーを入力してください。

③ディスクにプログラムをセーブする場合、ディスクをイニシャライズしておく必要があります。ディスクのユーティリティプログラムを参照してください。

④デバイス名にCOM：を指定した場合には、RS－232Cポートへのプログラムの送り出しが行なわれます。この場合には、Aオプションをつけなくてもアスキー形式でプログラムを送出します。

では、実際にプログラムをセーブしてみましょう。次のプログラムを「X1－WORLD」という名前でセーブしてみることにします。

〔例〕
```
10 INPUT "N=";N
20 FOR P=1 TO N
30 PRINT "X1-WORLD"
40 NEXT P
50 END
```

＊プログラムをRUNすると、回数をきいてきますから、数字を入力してください。入力した回数だけ「X1－WORLD」と表示します。

上のプログラムを入力後、次のようにしてセーブを行ないます。

フロッピーディスク（ドライブナンバー1）にプログラムをセーブする場合。
```
SAVE"1:X1-WORLD"↵
Ok
```
そのセーブをアスキー形式で行なう場合
```
SAVE"1:X1-WORLD",A↵
Ok
```
グラフィックメモリにセーブする場合
```
OPTION SCREEN2↵
Ok
INIT"MEM0:"↵
Are you sure?(y or n) Y↵
Ok
SAVE"MEM0:X1-WORLD"↵
Ok
```

LIST〔"デバイス名：ファイル名"〔,〕〕〔〔開始行番号〕〔－〔終了行番号〕〕

デバイス名…SAVEコマンド指定できるもの以外にSCR：、CRT：（どちらも画面）
LPT（プリンタ）があります。

このコマンドを実行することによりメインメモリ内のプログラムを指定されたデバイスへ指定されたファイル名を付けてアスキー形式でセーブすることができます。「SAVE"ファイルディスクリプタ",A」と異なる点はセーブする行番号を指定できることです。すなわち、メインメモリにあるプログラムの開始行番号から終了行番号まで（開始行番号だけを指定した場合はその行番だけ）をセーブすることができます。このコマンドと後でのべるMARGEコマンドを合わせてもちいることによって、あるプログラムのサブルーチン（プログラムの一部でまとまった作業を行なう部分）を他のプログラムに追加することができます。

また、すべてのパラメータを省略することによってメインメモリ中のプログラムリストを画面に表示することができます。

〔例〕 LIST"0:TEST",10-40↵
Ok
この実行によりメインメモリにあるプログラムの行番号10から40までがアスキー形式でフロッピーディスクのドライブ0にセーブされます。

SAVEM〔"〔デバイス名:〕ファイル名"〕,開始アドレス,終了アドレス〔,実行開始アドレス〕

＊デバイス名……SAVEコマンドと同じものが指定できます。
メインメモリ内の機械語プログラムを外部ファイルにセーブします。
〔注意〕開始アドレス、終了アドレスは機械語プログラムをセーブするときにその範囲を指定するもので、メインメモリ上の番地を指定します。これはファイルにも記録され、再生のときに使われます。
実行開始アドレスは機械語プログラムを実行するときの開始場所を指定するもので、省略するとセーブの開始アドレスが実行開始アドレスになります。
〔例〕SAVEM"CAS:TEST",&H8000,&H80FF,&H800E
メインメモリ内の&H8000から&H80FFにある実行番地&H800Eのプログラムをカセットにセーブします。

1.2 プログラムの再生

カセットテープ、フロッピーディスクなどの外部ファイルにセーブされたプログラムをメインメモリに移すことをプログラムの再生（ロード）といいます。ロードはLOAD、RUN、LOADMコマンドによって実行できます。

LOAD〔"〔デバイス名:〕ファイル名"〕

デバイス名……SAVEコマンドと同じものが指定できます。
外部ファイルに記録されているBASICプログラムをメインメモリにロードします。
〔注意〕①プログラムをロードすると、それまでメインメモリにあったプログラムは消えます。またファイル名はセーブしたときと、全く同じでなければなりません。もし、指定したファイル名のファイルがない場合には「File not found」のエラーメッセージを表示します。
②デバイス名にCOM:を指定した場合にはRS-232Cポートよりプログラムを読み込みます。この場合、通信回線より送られてくるプログラムはアスキー形式でなければなりません。
〔例〕SAVEコマンドの所でセーブしたファイル名"X1-WORLD"のプログラムをロードします。
フロッピーディスク（ドライブナンバ1）からプログラムをロードする場合、
LOAD"1:X1-WORLD"↵
Ok
＊なおアスキー形式でセーブされたプログラムも同様にしてロードできます。
グラフィックメモリからプログラムをロードする場合、
LOAD"MEM0:X1—WORLD"↵
Ok

LOADM〔"〔デバイス名:〕ファイル名"〔,ロード開始アドレス〕
LOADM〔"〔デバイス名:〕ファイル名"〔,〔ロード開始アドレス〕,R〕

デバイス名……ＳＡＶＥコマンドと同じものが指定できます。

外部ファイルに記録されている機械語プログラムをメインメモリにロードします。最後のパラメータＲを指定した場合ロード終了後プログラムを直ちに実行します。

〔注意〕ロード開始アドレスを指定すると、そのアドレスから機械語プログラムのロードを開始し、この開始アドレスを省略した場合、ＳＡＶＥＭで指定したセーブ開始アドレスからロードが始まります。

〔例〕　ＬＯＡＤＭ"ＣＡＳ：ＴＥＳＴ"↵
　　Ｏｋ

カセットにセーブされた"ＴＥＳＴ"という機械語プログラムがロードされます。

　　ＬＯＡＤＭ"ＣＡＳ：ＴＥＳＴ"，，Ｒ↵
　　Ｏｋ

"ＴＥＳＴ"という機械語プログラムがロードされた後、直ちに実行されます。

RUN〔"〔デバイス名：〕ファイル名"〕

＊デバイス名……ＳＡＶＥコマンドと同じものが指定できます。

外部ファイルに記録されているＢＡＳＩＣプログラムをメインメモリにロードした後、直ちに実行します。また、すべてのパラメータを省略した場合、メインメモリ中のプログラムを実行します。

〔例〕　ＲＵＮ"１：Ｘ１－ＷＯＲＬＤ"↵

フロッピーディスク（ドライブナンバー１）の"Ｘ１－ＷＯＲＬＤ"プログラムをロードした後、直ちに実行します。

1.3 プログラムのベリファイ

カセットテープにセーブしたプログラムをメインメモリにあるプログラムとくらべて、正しくセーブされたかどうかを調べることをプログラムのベリファイといいます。このときに使うのが、ＶＥＲＩＦＹまたはＬＯＡＤ？です。この２つは全く同じ意味のコマンドです。

VERIFY〔"〔CAS：〕ファイル名"〕
LOAD?　〔"〔CAS：〕ファイル名"〕

このコマンドで扱えるデバイスはＣＡＳ：のみです。

カセットテープにセーブしたプログラムが正しく記録されているかどうかを調べます。なお、このコマンドの実行はカセットテープにプログラムをセーブ後すぐに行なってください。

〔例〕ここではＳＡＶＥコマンドの項で示したプログラム「Ｘ１－ＷＯＲＬＤ」をカセットテープにセーブした後、ベリファイします。

①プログラムを入力します。

②カセットテープに"Ｘ１－ＷＯＲＬＤ"をセーブします。

プログラムをセーブできるカセットテープをデータレコーダにセットします。テープカウンタを０にしておくか、カウンタ番号を紙に書いておけばカセットテープのどこにセーブしたかの目印になります。

ＳＡＶＥ"ＣＡＳ：Ｘ１－ＷＯＲＬＤ"↵

と入力してください。画面にＯｋが表示されカーソルが点滅したらセーブ終了です。

③テープをセーブした位置まで巻きもどし、次のように入力します。

ＶＥＲＩＦＹ"ＣＡＳ：Ｘ１－ＷＯＲＬＤ"↵

正しくセーブされていれば次のように表示されます。

Ｆｏｕｎｄ"Ｘ１－ＷＯＲＬＤ"
Ｏｋ

正しくセーブされていなかった場合は次のように表示されます。

Ｔａｐｅ　ｒｅａｄ　ＥＲＲＯＲ

この場合、もう一度セーブしてください。

1.4 ファイルの削除

メインメモリにあるプログラムを消すにはＮＥＷを使いますが、外部ファイルに記録したプログラムを消すには、ＫＩＬＬという命令を使います。

ＫＩＬＬ"〔デバイス名：〕ファイル名"

デバイス名……０：～３：、ＭＥＭ０：～ＭＥＭ１：、ＥＭＭ０：～ＥＭＭ９：、
Ｆ０：～Ｆ３：、ＨＤ０：～ＨＤ３：

指定したデバイス内の指定したファイルを削除します。このコマンドによって削除されたプログラムはもとにもどりません。十分注意して行なってください。

〔注意〕デバイス名に"ＣＡＳ："を指定して、カセットに記録されたプログラムを消すことはできません。

〔例〕　ＫＩＬＬ"１：Ｘ１－ＷＯＲＬＤ"↵

Ｏｋ

フロッピーディスク（ドライブナンバ１）にセーブされた"Ｘ１－ＷＯＲＬＤ"プログラムを削除します。

1.5 ファイル名の一覧表

外部ファイルにセーブされたファイル名の一覧表はＦＩＬＥＳ、ＬＦＩＬＥＳのコマンドを実行することによって与えられます。

ＦＩＬＥＳ〔"デバイス名："〕

ＬＦＩＬＥＳ〔"デバイス名："〕

デバイス名……０：～３：、ＣＡＳ：、ＭＥＭ０：～ＭＥＭ１：、ＥＭＭ０：～ＥＭＭ９：
Ｆ０：～Ｆ３：、ＨＤ０：～ＨＤ３：

デバイス名で指定したデバイス内にあるファイル名の一覧表がＦＩＬＥＳの場合は画面に表示され、ＬＦＩＬＥＳの場合はプリンタに出力されます。

〔注意〕ＦＩＬＥＳを行なった後、その中に含まれるファイルをロード、削除、実行したい場合は次のようにすると簡単に行なえます。

(1)そのファイル名の初めに表示されたＢａｓまたはＡｓｃの所へ、カーソルを移動する。

(2)ＢａｓまたはＡｓｃのかわりにＬＯＡＤ、ＫＩＬＬ、ＲＵＮを入力し↵キーを押す。

〔例〕ＦＩＬＥＳ"１："↵

フロッピーディスク（ドライブナンバー１）のファイルが出力されます。

＊ファイルリストの初めに表示されるＢａｓ、Ａｓｃ、Ｂｉｎ、Ｄｉｒには次の意味があります。

Ｂａｓ……ＢＡＳＩＣプログラムのファイル

Ａｓｃ……アスキー形式で書き込まれたファイル

Ｂｉｎ……機械語プログラムのファイル

Ｄｉｒ……ディレクトリのファイル

1.6 デバイス名の省略

ファイルに関するコマンドやステートメントを使うプログラムにおいて、何度も同じデバイス名を指定しなければならないことがあります。そのような時DEVICEコマンドを使用してデバイス名を省略した場合の指定が行なえます。

DEVICE"デバイス名"

デバイス名……すべてのデバイス名が指定できます。

CRT:、SCR:、KEY:、LPT:、CAS:、0:～3:、
F0:～F3:、HD0:～HD3:、EMM0:～EMM9:、
MEM0:MEM1:、COM:

FILES、SAVE、LOADなどのファイル入出力コマンド、およびLIST、RUNなどのコマンドにおいて、デバイス名を省略すると、このコマンドで指定されたデバイス名が指定されたことになります。

〔注意〕DEVICEコマンドでデバイス名が指定されていない時にデバイス名を省略した場合、BASIC起動時のデバイス名（カセットBASIC起動時はCAS:ディスクBASIC起動時は0:）が指定されます。

〔例〕
```
DEVICE"1:"↵
Ok
LOAD"X1-WORLD"↵
Ok
```

ロードされるプログラムは、フロッピーディスクのドライブ1にある"X1-WORLD"になります。

1.7 プログラムのマージ

メインメモリ上にあるプログラムと、外部ファイルに記録されたプログラムを合成して、1つのプログラムにすることをマージすると言います。これは、次のMERGEコマンドを実行することによって行なわれます。

MERGE"デバイス名:ファイル名"

デバイス名……SAVEコマンドと同じものが指定できます。

このコマンドが実行されると、指定したデバイスに入っているプログラムが読み込まれ、現在メモリに存在するプログラムと組み合わさった1つのプログラムが作成されます。

〔注意〕双方の行番号が重複した場合は、ファイルから読み込まれた方が優先します。また、セーブされているプログラムはアスキー形式でセーブされている必要があります。アスキー形式でセーブされていないプログラムをマージしようとすると「Bad file mode」のエラーが出ます。

〔例〕①次のプログラムを入力してください。

```
10 DATA 10,20,30,40,50,60,70,80,90,100
20 INPUT N
30 FOR I=1 TO N
40 READ DA(I)
50 NEXT
```

②入力したプログラムをフロッピーディスク（ドライブナンバ0）にアスキー形式でセーブします。

```
SAVE"0:TEST", A↵
Ok
```

＊ＬＩＳＴコマンドによっても行なうことができます。

③メインメモリのプログラムを消します。

```
NEW↵
Ok
```

④メインメモリにマージする次のプログラムを入力してください。

```
60 FOR J=1 TO N
70 KEI=KEI+DA(J)
80 NEXT
90 PRINT KEI
```

⑤マージを実行します。

```
MERGE"0:TEST"
```

⑥ここでメインメモリのプログラムリストをとります。次のようにマージされた1つのプログラムになっていることが確認できます。

```
LIST
10 DATA 10,20,30,40,50,60,70,80,90,100
20 INPUT N
30 FOR I=1 TO N
40 READ DA(I)
50 NEXT
60 FOR J=1 TO N
70 KEI=KEI+DA(J)
80 NEXT
90 PRINT KEI
Ok
```

＊このプログラムは行番号１０～５０でＤＡＴＡ文から指定数だけデータを読み込み行番号６０～９０でその合計を出し表示するというものです。

1.8 プログラムのチェイン

プログラムが長すぎて、メインメモリで処理できない場合があります。この場合プログラムをいくつかに分割し、それぞれ別のファイル名で外部ファイルに記録しておきます。そして、1つのプログラム実行後、自動的に次のプログラムをロードして実行するようにします。このことを「プログラムをチェインする」といい次のＣＨＡＩＮコマンドを使用して行ないます。

CHAIN"〔デバイス名：〕ファイル名"

デバイス名……ＳＡＶＥコマンドと同じものが使えます。

メインメモリ内にあるプログラムの変数を保護し、指定したデバイス内の指定したファイル名のプログラムをロードして実行します。

〔注意〕ロードする前にオープンされていたデータファイルは、このコマンドでは閉じません。

〔例〕サンプル(1)

```
10 DIM IN(20)
20 INPUT"DATA ﾉ ｺｽｳ",N
30 FOR L=1 TO N
40 PRINT L;"ﾊﾞﾝﾒﾉ DATA"
50 INPUT IN(L)
60 NEXT
70 CHAIN"0:SAMPLE2"
```

サンプル(2)

```
10 FOR M=1 TO N-1
20 B=ABS(IN(M)-IN(M+1))
30 PRINT
40 PRINT IN(M);TAB(10);"-";IN(M+1);TAB(22);"=|";B;"|"
50 NEXT
```

①サンプル(2)のプログラムを入力し、フロッピーディスク（ドライブナンバー0）に"SAMPLE2"というファイル名でセーブします。

```
SAVE"0:SAMPLE2"↵
Ok
```

②メインメモリをクリアします。

```
NEW↵
Ok
```

③サンプル(1)のプログラムを入力し、実行します。データの個数とデータを入力すると隣り合う2つの数の差をとって、その絶対値が表示されます。

```
RUN
DATA ノ コスウ3↵
1バンメノ DATA
? 153.3↵
2バンメノ DATA
? 12345.67↵
3バンメノ DATA
? 410↵
153.3-12345.67=|12192.37|
12345.67-410  =|11935.67|
Ok
```

サンプル(1)の行番号10～60はデータの入力を行なうだけですが、70行のCHAINコマンドが実行されることにより、フロッピーディスク（ドライブナンバ0）からサンプル(2)のプログラムがロードされ実行さます。また、サンプル(2)のプログラムはデータの処理と表示を行なうだけですが、CHAINコマンドによってサンプル(1)の変数が保護されているため、サンプル(1)のプログラムで入力したデーたが処理され表示されます。

1.9 ファイル名の付けかえ

1度セーブしたファイル名を次のNAMEコマンドを実行することによって変更することができます。

NAME"〔デバイス名：〕旧ファイル名"AS"〔デバイス名：〕新ファイル名"

デバイス名……0：～3：、MEM0：～MEM1：、EMM0：～EMM9：、F0：～F3：、HD0：～HD3：

旧ファイル名で指定したファイルのファイル名を新ファイル名に変更します。

〔注意〕2つのデバイス名は同じものでなければなりません。また、デバイス名にCAS：は使用できません。

〔例〕

```
NAME"0:TEST"AS"0:SAMPLE"↵
Ok
```

フロッピーディスク（ドライブナンバ0）の"TEST"というファイルを"SAMPLE"というファイル名に変更します。

1.10ファイルの属性指定

作成したファイルを誤って消去しないようにそのファイルにライトプロテクト（書き込み禁止）をかけたり、リードアフタライト（ファイルに書き込みを行う際にデータを書き込んだ後自動的にベリファイを行なう）やシークレット（ＦＩＬＥＳコマンドによりファイル名の一覧を表示させてもシークレット指定されたファイルは表示されない）というファイルの属性を指定する場合にはＳＥＴコマンドを実効します。

```
SET { #ファイル番号                  } , { "P" }
    { "〔デバイス名：〕ファイル名"   }   { "R" }
                                         { "S" }
                                         { "" }
```

ファイル番号……オープンで指定したファイル番号
デバイス名……０：～３：、ＭＥＭ０：、ＭＥＭ１：ＥＭＭ０：～ＥＭＭ９：、
Ｆ０：～Ｆ３：、ＨＤ０：～ＨＤ３：

属性を指定する文字として「Ｐ」と「Ｒ」と「Ｓ」を使用し、「Ｐ」を指定するとライトプロテクト（書き込み禁止）がかかり、「Ｒ」を指定するとリードアフタライトが実効され、「Ｓ」を指定するとシークレットが実行されます。属性を取り除くにはヌルストリング（""）を指定します。

〔例〕

①フロッピーディスク（ドライブ０）内の「ＳＡＭＰＬＥ」というＢＡＳＩＣプログラムファイルにライトプロテクトをかけてみましょう。

```
SET"0:SAMPLE","P"↵
Ok
```

これでライトプロテクトがかかりました。ＦＩＬＥＳコマンドによりファイル名のリストを表示させてみてください。「ＳＡＭＰＬＥ」というファイルのＢａｓの次に「＊」マークが付いています。このマークがライトプロテクトのかかったファイルであることを示します。

②リードアフタライトの指定は次のようにします。

リードアフタライトを指定するファイルをＯＰＥＮコマンドで開いた後（ファイル番号を１とすると）

```
SET#1,"R"
```

を実行すると、以降このファイルにデータを書き込むと自動的にベリファイ動作を行ないます。このファイルをＣＬＯＳＥコマンドにより閉じた時点でこのリードアフタライトの属性は解除されます。

③例①でライトプテクトをかけたファイルにさらにシークレット指定を行なってみましょう。

```
SET"0:SAMPLE","S"↵
Ok
```

これでシークレット指定されました。ＦＩＬＥＳ"０："↵と入力してファイル名の一覧を表示させてみてください。「ＳＡＭＰＬＥ」というファイル名はどこにも表示されていません。ただし、「ＳＡＭＰＬＥ」というファイルが削除されてしまったわけではなく、単にファイル名の表示を行なわないというだけでそのファイルはディスクに残っています。

1.11ファイルのパスワード

自分で作成したファイルを他人に使用されたくないという場合には、そのファイルにパスワード（合い言葉）を指定して、データをファイルに書き込みます。すると、このファイルを読み出す時には指定したパスワードを入力しないと読み出せません。このパスワードを指定してファイルのアクセスを行なうには、名前〔，エクステンション〕；パスワードという形式でファイル名を使用します。

〔注意〕

パスワードには．"：；／記号は使用できません。またファイル名全体の長さは３１文字以内でなければなりません。

〔例〕

例えば、メインメモリにあるＢＡＳＩＣプログラムをファイルネーム「ＴＥＳＴ」でパスワード「ＡＢＣ」を付けてディスク（ドライブ０）にセーブするには、

ＳＡＶＥ"０：ＴＥＳＴ；ＡＢＣ"↵

と入力します。ＦＩＬＥＳ"０："↵と入力してファイル名のリストを表示してみてください。指定したパスワードは表示されておらず、パスワードを指定せずにセーブしたファイルと表面上は何の変化もありません。しかし、ＬＯＡＤ"０：ＴＥＳＴ"↵と入力して読み出そうとしても「Ｎｏ ｐａｓｓｗｏｒｄ」とエラー表示され読み出すことができません。読み出すためにはＬＯＡＤ"０：ＴＥＳＴ；ＡＢＣ"↵と入力します。これにより、指定したパスワードを知らない人はそのファイルを扱うことができなくなります。

2 データファイル

プログラム中で使用するデータは、外部ファイルにデータファイルという形で再生・保存が可能です。また、データファイルにはシーケンシャルファイルとランダムアクセスファイルの２種類があります。

シーケンシャルファイルとは、連続的に書かれたデータファイルのことで、次のような特徴があります。

1）ファイルの中では、データは書き込まれた順序で並んでおり、データを保存・再生する際には先頭から順番にしか行なえません。
2）データの内容を変更する際には、内容を変更しないデータも順に読み出し、処理を行なった後、ファイルに書き出すことが必要です。
3）記録したファイルの最後にデータを追加することができます。
4）ファイルの中のデータは順序良く並んでおり、無駄がありません。
5）ファイルの作成が比較的簡単に行なえます。

シーケンシャルファイルを使用してデータの保存・再生を行なう場合、次のコマンドを使用します。

ＯＰＥＮ、ＰＲＩＮＴ＃、ＷＲＩＴＥ＃、ＩＮＰＵＴ＃、ＬＩＮＰＵＴ＃、ＣＬＯＳＥ、ＥＯＦ、ＩＮＰＵＴ＄、ＤＥＶＩＣＥ、ＯＰＴＩＯＮ ＳＣＲＥＥＮ、ＩＮＩＴ

これに対してランダムアクセスファイルとは、１レコード（＝２５６バイト）単位でその記録位置を指定できるデータファイルのことで、次のような特徴があります。

1）ファイル中のデータは、内部のインデックスによって直接その記憶位置を指定できるようになっているため、データの保存・再生はレコードごとにランダムに行なえます。
2）ファイル中の一部のデータだけを変更することができます。
3）記録したファイルの最後にデータを追加することができます。
4）データが少ない場合でもロード、セーブを１レコードごとに行なうため無駄な部分があります。
5）ファイルの作成のために多くの設定が必要なため、シーケンシャルファイルに比べてプログラムが複雑になります。

ランダムファイルを使用してデータの保存・再生を行なう場合、次のコマンドを使用します。

OPEN、CLOSE、FIELD、LSET、RSET、GET、PUT、DEVICE、OPTION SCREEN、INIT、MKI$、MKS$、MKD$、CVI、CVS、CVD

シーケンシャルファイルを使えば、ほとんどのデータ処理が行なえますが、ランダムファイルを自由に使いこなしてこそ、初めてコンピュータの真価が発揮できます。

〔注意〕データファイルもプログラムファイルと同様にKILL、FILES、LFILES、NAME　SET命令によりファイルの削除、ファイル名の一覧表示等が行なえます。

2.1 シーケンシャルファイルへのデータの保存・再生

シーケンシャルファイルへのデータの保存・再生を行なう場合、プログラムファイルのように1つのコマンドだけで行なうことはできません。シーケンシャルファイルの場合書き込み、読み出しのコマンド（PRINT#、INPUT#など）の前後でファイルの状態を設定する必要があります。ここでは、シーケンシャルファイルへのデータの保存・再生について、実際のプログラム例を用いて説明します。なお、コマンドの詳細に関しては「BASICリファレンスマニュアル」を参照してください。

次のプログラムを入力してください。

```
10 OPEN"O",#1,"1:DATA"
20 INPUT"名前は";NA$
30 INPUT"何才ですか";AGE
40 PRINT#1,NA$
50 PRINT#1,AGE
60 CLOSE#1
100 OPEN"I",#1,"1:DATA"
110 INPUT#1,NA$
120 INPUT#1,AGE
130 PRINT"名前は";NA$;"です。"
140 PRINT"歳は";AGE;"才です。"
150 CLOSE#1
```

入力がおわりましたら、プログラムのリストをとり、間違いがないかどうかを調べてください。

それでは、プログラムの説明を行ないます。

```
10 OPEN"O",#1,"1:DATA"
```

シーケンシャルファイルへデータを書き込んだり、読み出したりするには、どのファイルにどのような操作を行なうのかを宣言する必要があります。その宣言を行なうのがこのOPENコマンドで、各パラメータによって次のことが設定されます。

"O"……ファイルへのデータの書き込み（OUTPUT）を設定します。
#1……OPENコマンドによって出力するファイル番号を設定します。
"1:DATA"……出力デバイスを設定します。

すなわち、行番号10の実行でフロッピーディスク（ドライブナンバー1）にデータの出力が可能になります。また、以後のファイル処理は、ファイル名ではなくファイル番号で行ないます。

＊この設定を行なうことをファイルを開く、あるいはファイルをオープンするといいます。

```
20 INPUT"名前は";NA$
30 INPUT"何才ですか";AGE
```

行番号20、30でシーケンシャルファイルに出力するデータを用意します。

```
40 PRINT#1,NA$
50 PRINT#1,AGE
```

行番号４０、５０では、ファイル番号１でオープンされたファイルにデータを出力します。シーケンシャルファイルにデータを出力するコマンドは、このＰＲＩＮＴ＃の他にＷＲＩＴＥ＃があります。この２つのコマンドの違いについては「ＢＡＳＩＣリファレンスマニュアル」を参照してください。

```
60 CLOSE#1
```

すべてのデータを書き込んだならば、その内容を保持するためにオープンしたファイルをとじるという処理が必要です。そのためのコマンドがＣＬＯＳＥで、行番号６０ではファイル番号１のファイルを閉じています。このようにして閉じられたファイルは、再びオープンされない限り読み出すことも書き込むこともできなくなります。以上でシーケンシャルファイルへのデータの出力は終りです。

```
100 OPEN"I",#1"1:DATA"
```

次に、書き込んだデータを読み出します。読み出す場合も書き込む場合と同様にＯＰＥＮ文でファイルをオープンする必要があります。ただ、書き込みを指定していたパラメータ"Ｏ"を"Ｉ"に変え、読み出し（ＩＮＰＵＴ）状態に設定する必要があります。行番号１００の実行によって、フロッピーディスク（ドライブナンバ１）のシーケンシャルファイル"ＤＡＴＡ"からデータの読み出しが可能になります。

```
110 INPUT#1,NA$
120 INPUT#1,AGE
```

行番号１１０、１２０ではファイル番号１でオープンされたファイルからデータを変数ＮＡ$、ＡＧＥに読み込みます。シーケンシャルファイルでは、データを読み出す場合、書き込んだ順番に行なう必要があり、一部のデータを読み出す場合でも、その前に書き込まれたデータをいったん読み出さなければなりません。

```
130 PRINT"名前は";NA$;"です。"
140 PRINT"歳は";AGE;"才です。"
```

行番号１３０、１４０では、シーケンシャルファイルから読み込んだデータを画面に表示します。

```
150 CLOSE#1
```

行番号１５０ではファイル番号１のファイルを閉じています。データ書き込みだけでなく読み込みの場合でも、その実行が終ったならばファイルを閉じる必要があります。

〔注意〕データの保存、再生を行なう場合、プログラムの保存、再生と同様にディスクならば、イニシャライズが、グラフィックメモリならばＯＰＴＩＯＮ　ＳＣＲＥＥＮ、ＩＮＩＴが実行されている必要があります。

〔例〕では、実際の実行例を示します。

```
RUN ↵
名前は？パソコンテレビ ↵
何才ですか？2 ↵
名前はパソコンテレビです。
歳は2才です。
Ok
```

＊プログラムの実行前にＫＭＯＤＥ１が実行されていない場合は漢字を表示しません。

2.2 シーケンシャルファイルへのデータの追加

シーケンシャルファイルでは、ファイルの任意の位置への書き込み、読み出しは行なえませんが、ファイルの一番最後にデータを追加することはできます。ここでは、２．１のプログラムで作られたシーケンシャルファイルにデータを追加する例を用いて説明します。

次のプログラムを入力してください。

＊なお、このプログラムは２．１のプログラムの一部を変更することで簡単に作れます。

```
10 OPEN"A",#1,"1:DATA"
20 INPUT"名前は";NA$
25 IF NA$="END"THEN 60
30 INPUT"何才ですか";AGE
```

```
40 PRINT#1,NA$
50 PRINT#1,AGE
55 GOTO 20
60 CLOSE#1
100 OPEN"I",#1,"1:DATA"
105 IF EOF(1) THEN 150
110 INPUT#1,NA$
120 INPUT#1,AGE
130 PRINT"名前は";NA$;"です。"
140 PRINT"歳は";AGE;"才です。"
145 GOTO 105
150 CLOSE#1
```

それでは、プログラムの説明を行ないます。

```
10 OPEN"A",#1,"1:DATA"
```

データの追加でファイルをオープンする場合、パラメータを"A"にしデータの追加（APPEND）状態にしなければなりません。行番号10ではフロッピーディスク（ドライブナンバー1）の"DATA"というシーケンシャルファイルにデータを追加することを宣言しています。

```
20 INPUT"名前は";NA$
25 IF NA$="END"THEN 60
30 INPUT"何才ですか";AGE
40 PRINT#1,NA$
50 PRINT#1,AGE
55 GOTO 20
60 CLOSE#1
```

行番号25のIF文と行番号55のGOTO文で何回もデータが入力できるようになっています。シーケンシャルファイルにデータを追加する場合OPENコマンドのパラメータ以外は、入力と同じ手続きで行ないます。追加されるデータは、それ以前のデータの後に入力順に追加されます。行番号60まででデータの追加は終わりです。

```
100 OPEN"I",#1,"1:DATA"
105 IF EOF(1) THEN 150
110 INPUT#1,NA$
120 INPUT#1,AGE
130 PRINT"名前は";NA$;"です。"
140 PRINT"歳は";AGE;"才です。"
145 GOTO 105
150 CLOSE#1
```

行番号100～150では、追加されたシーケンシャルファイルからデータの入力を行ない画面に表示します。行番号105のIF文はシーケンシャルファイルファイルの終りをEOF（END OF FILEの略）関数により検出し、終わりであれば行番号150へジャンプすることを意味します。EOF関数は、最後のデータならば真の値（－1）を与え、そうでないならば偽の値（0）を与えます（詳しくは、BASICリファレンスマニュアルを参照ください。）

〔例〕以下は、プログラムの実行例です。

```
RUN
名前は？　小沢
何才ですか？　28
名前は？　相原
何才ですか？　24
名前は？　END
```

名前は．パソコンテレビです。
歳は２才です。
名前は小沢です。
歳は２８才です。
名前は相原です。
歳は２４才です。
Ｏｋ
＊「名前は？」の所でＥＮＤをキー入力することによってデータの追加が終わります。

2.3 ランダムアクセスファイルへのデータの保存・再生

ランダムファイルもシーケンシャルファイルと同様に書き込み、読み出しのコマンドの前後でファイルの状態を設定する必要があります。また、ランダムファイルは、データの入出力をファイルバッファ（メインメモリ上でデータを一時的に蓄える部分）を介して行ない、そのファイルバッファへの入出力は文字型のデータでなければなりません。そのためランダムファイルよりも次のように多くの手続きを必要とします。

１．データの保存

①ファイルをオープンします。……〔ＯＰＥＮ〕
②ランダムファイルのデータを用意します。データが数値型ならば文字型に変更を行なう必要があります。……〔ＭＫＩ＄、ＭＫＳ＄、ＭＫＤ＄〕
③ファイルバッファに文字変数を割りつけます。……〔ＦＩＥＬＤ〕
④ファイルバッファにデータを書き込みます。……〔ＬＳＥＴ、ＲＳＥＴ〕
⑤ファイルバッファの内容をディスク、グラフィックメモリに書き込みます。……〔ＰＵＴ〕
⑥ファイルを閉じます。……〔ＣＬＯＳＥ〕

２．データの再生

①ファイルをオープンします。……〔ＯＰＥＮ〕
②ファイルバッファに文字変数を割りつけます。……〔ＦＩＥＬＤ〕
③ランダムファイル上のデータをファイルバッファに読み込みます。……〔ＧＥＴ〕
④ファイルバッファ上のデータを取り出して処理します。

文字型に変換していた数値をもとの数値型に変換しなおします。……〔ＣＶＩ、ＣＶＳ、ＣＶＤ〕
⑤ファイルを閉じます。……〔ＣＬＯＳＥ〕

それでは、実際のプログラム例を用いて説明していきます。なお、各コマンドの詳細に関しては「ＢＡＳＩＣリファレンスマニュアル」を参照してください。

次のプログラムを入力してください。

プログラム(1)……データ保存例

```
10 OPEN"R",#1,"1:RDATA"
20 INPUT"名前は";NA$
30 INPUT"何才ですか";AGE%
40 T$=MKI$(AGE%)
50 FIELD#1,10 AS A$,2 AS B$
60 LSET A$=NA$
70 LSET B$=T$
80 PUT#1,1
90 CLOSE#1
```

それでは、プログラムの説明を行ないます。

```
10 OPEN"R",#1,"1:RDATA"
```

ランダムファイルをオープンする場合"R"のモード（Random file mode）指定を行ないます。なお、ランダムファイルの場合入力、出力で別のモード指定を行なう必要はなく1つのOPEN文でデータの入出力をすることも可能です。行番号10では、フロッピーディスク（ドライブナンバー1）に"RDATA"という名のランダムファイルをオープンしています。

```
20 INPUT"名前は ";NA$
30 INPUT"何才ですか ";AGE%
40 T$=MKI$(AGE%)
```

ランダムファイルに出力するデータは文字型のものだけで、数値型のものは使用することができません。そのため、数式を文字列に変換した後、出力する必要があります。数式を文字列に変換するコマンドは、行番号40のMKI$の他にMKS$、MKD$があります。それぞれのコマンドに関しては「BASICリファレンスマニュアル」を参照してください。

```
50 FIELD #1,10 AS A$,2 AS B$
```

ランダムファイルにデータの入出力を行なう場合、1レコード（256バイト）のデータをファイルバッファに蓄えておき、1レコード分づつ入出力を行ないます。また、そのファイルバッファは、入出力の前にデータの割り付けがなされていなければなりません。それを行なうのがFIELD文で行番号50の場合、ファイル番号1用のファイルバッファ（256バイト）の最初の10バイトをA$、次の2バイトをB$に割り付けています。なおこの場合、残りの244バイトは未使用となり、たいへんぜいたくな使い方といえます。

＊FIELD文はファイルバッファに変数の割り付けを行なうだけで、データをファイルバッファおよびファイルに入出力しません。

```
60 LSET A$=NA$
70 LSET B$=T$
```

行番号60、70ではFIELD文で割り付けたファイルバッファにデータを出力しています。ファイルバッファにデータを出力する場合、このLSETの他にRSETというコマンドがあります。2つの違いについては「BASICリファレンマニュアル」を参照してください。

```
80 PUT#1,1
```

PUT文によってファイルバッファの内容（256バイト）がランダムファイルに出力されます。行番号80では、ファイル番号1のファイルバッファにあるデータをOPEN文で指定した"RDATA"というファイルのレコード番号1に書き込んでいます。ランダムファイルには、レコード番号が記録されるようになっており、このレコード番号の指定を変えることによりデータのランダムな入出力が可能となります。

```
90 CLOSE#1
```

行番号90では、オープンされていたランダムファイルを閉じています。ランダムファイルの場合、入力出力別にOPEN文を実行する必要がないため、ファイルを閉じなければ、データの入出力を1つのOPEN文、FIELD文によって行なうことができます。

〔例〕実行例を以下に示します。

```
RUN
名前は？ パソコンテレビ
何才ですか？ 2
Ok
```

次のランダムファイルからデータ入力を説明します。

次のプログラムを入力してください。

プログラム(2)……データ再生例

```
10 OPEN"R",#1,"1:RDATA"
20 FIELD#1,10 AS A$,2 AS B$
30 GET#1,1
```

```
40 AGE=CVI(B$)
50 PRINT"名前は";A$;"です。"
60 PRINT"歳は";AGE;"才です。"
70 CLOSE#1
```

それでは、プログラムの説明を行ないます。

```
10 OPEN"R",#1,"1:RDATA"
20 FIELE #1,10 AS A$,B$
```

行番号10でランダムファイルをデータ入力のためにオープンしています。ランダムファイルの場合、前に述べたようにデータの入出力でオープン方法は変りません。行番号20では、ファイルバッファに変数を割り付けています。この場合、出力時と変数名はかえることができますが、ファイルバッファの割り付けは同じにする必要があります。

```
30 GET#1,1
```

ファイルから1レコード（256バイト）分のデータをファイルバッファに入力するにはGET文を使います。行番号30の場合、ファイル番号1のファイルバッファにOPEN文で指定した"RDATA"というファイルのレコード番号1からデータを256バイト読み込んでいます。

```
40 AGE=CVI(B$)
50 PRINT"名前は";A$;"です。"
60 PRINT"歳は";AGE;"才です。"
70 CLOSE#1
```

ファイルバッファに読み込まれたデータは、FIELD文で割り付けた文字変数を使用することによって取り出せます。また、一旦文字式に変換した数値は、CVI、CVS、CVDコマンドを実行することによりもとの数値に変換することができます。3つのコマンドの違いについては「BASICリファレンスマニュアル」を参照してください。以上のことは行番号40～60で行なわれており、行番号70では入力のためにオープンしたファイルを閉じています。

〔例〕それでは実行例を示します。なおこのプログラムの実行前にプログラム(1)が実行されている必要があります。

```
RUN
名前はパソコンテです。
歳は2才です。
```

＊名前の入力をする際、パソコンテレビと入力しましたが、FIELD文でそのファイルバッファの領域を10バイトしか設けなかったため、このプログラムではハソコンテとしか表示されません。データをファイルバッファに割り付ける場合、その大きさに注意する必要があります。

2.4 ランダムアクセスファイルに関するデータの追加・変更

ランダムファイルはシーケンシャルファイルに比べて多くの手続きを必要としますが、データの追加・変更は自由に行なうことができます。すなわち、レコード番号を指定することによって、どの場所のデータもアクセス（呼び出し）可能です。ここでは、2．3で作成したランダムファイルにデータを追加・変換するプログラム例を説明します。

次のプログラムを入力してください。

プログラム(3)……データ追加・変換プログラム

```
10 OPEN"R",#1,"1:RDATA"
20 INPUT"名前は ";NA$
25 IF NA$="END"THEN 90
30 INPUT"何才ですか ";AGE%
40 T$=MKI$(AGE%)
50 FIELD#1,10 AS A$,2 AS B$
60 LSET A$=NA$
```

```
70 LSET B$=T$
75 INPUT"レコード番号 ";N
80 PUT#1,N
85 GOTO 20
90 CLOSE#1
```

ほとんどがプログラム(1)の場合と同じです。行番号７５で入力するレコード番号を変えることにより、どの場所のデータでも追加変更できます。なお、このプログラムでは名前に「ＥＮＤ」を入力するまで何回でも追加・変更を繰り返えすことができます。

〔例〕実行例を示します。

```
RUN
名前は ? 小林
何才ですか ? 29
レコード番号 ? 1
名前は ? 中村
何才ですか ? 25
レコード番号 ? 3
名前は ? 早川
何才ですか ? 70
レコード番号 ? 2
名前は ? END
Ok
```

それでは、プログラム(3)で作成したランダムファイルからデータを取り出す次のプログラムを入力してください。

プログラム(4)

```
10 OPEN"R",#1,"1:RDATA"
20 FIELD#1,10 AS A$,2 AS B$
25 INPUT"レコード番号";N
28 IF N=0 THEN 70
30 GET #1,N
40 AGE=CVI(B$)
50 PRINT"名前は ";A$;"です。"
60 PRINT"歳は";AGE;"才です。"
65 GOTO 25
70 CLOSE#1
```

この場合もまた、プログラム(2)とほとんど同じです。行番号３０でアクセスする行番号を入力し、行番号４０、９０によって繰り返しが行なえるようになっています。

〔例〕実行例を示します。

```
RUN
レコード番号　？　１
名前は　小林　です。
歳は２９才です。
レコード番号？　２
名前は　早川　です。
歳は７０才です。
レコード番号　？　３
名前は　中村　です。
歳は２５才です。
レコード番号　？　０
Ｏｋ
```

＊ファイル番号に０を入力することによってプログラムの実行が終ります。
また、「ＲＤＡＴＡ」というファイルのレコード番号１には、プログラム(1)の実行により「パソコンテ」というデータが入っていましたが、プログラム(3)の実行例で示すようにレコード番号１に新しく「小林」というデータを書き込んだため、前のデータは消去されています。

5章
ディスクの使い方

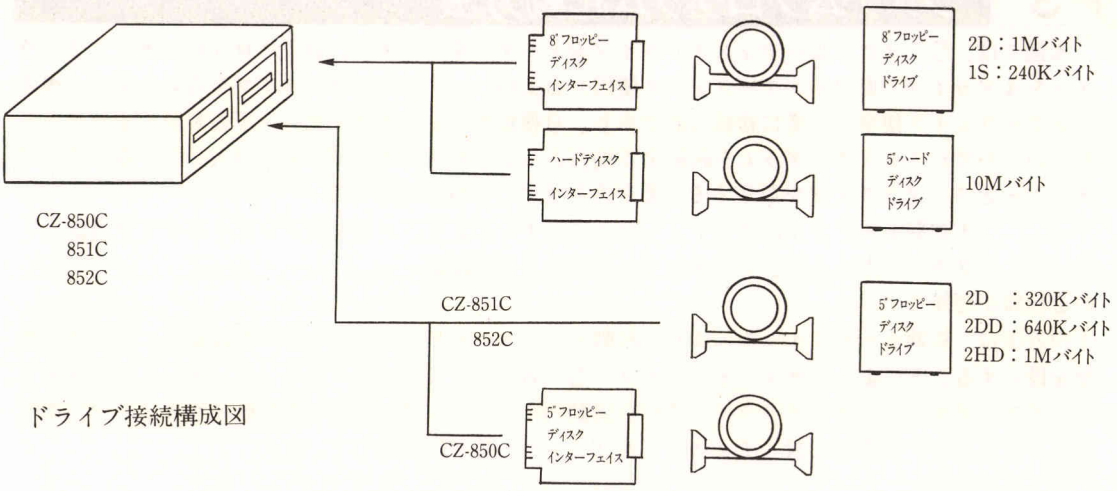
本機では、5インチミニフロッピーディスク（3インチコンパクトフロッピーディスク）のほかに8インチフロッピーディスク、5インチハードディスクをBASICでサポートしています。ここでは　これらのディスク装置を入出力装置として使う場合に、必要な事柄を説明しています。
なお、プログラムやデータを保存または再生するときは「4章プログラム、データファイルの保存・再生」をご覧ください。

1 ディスクの使用準備

新らしいフロッピーディスクを使用される場合、あらかじめフロッピーディスクのフォーマッティングを行なっておく必要があります。
フォーマッティングとは新らしいフロッピーディスクを使用できるようにすることで、DISK BASICのユーティリティプログラムにフォーマッティングのためのプログラムが用意されていますので、フロッピーディスクのフォーマッティングの手順に従って行なってください。（別冊の「アプリケーションソフトの説明」参照）

2 ドライブ接続構成

本機は下記構成図の様に各々3"or5"フロッピーディスクドライブ、（容量：320Kバイト、640Kバイト、1Mバイト）、8"フロッピーディスクドライブ（容量；1Mバイト、240Kバイト）及び5"ハードディスクドライブ容量（10Mバイト）を接続可能です。



ドライブ接続構成図

・CZ－850Cは3"or5"、8"フロッピーディスク、5"ハードディスクの専用インターフェイスボードを各々拡張Ⅰ／Oポートに挿入し各インターフェイスボードに各タイプのドライブを外部接続可能です。
・CZ－851C、CZ－852Cは5"フロッピーディスクドライブ及びインターフェイスを内蔵しているので3"or5"フロッピーディスクドライブは直接増設可能であり、　8"フロッピーディスクドライブ、ハードディスクドライブは専用

インターフェイスボードを使用して外部接続ができます。

ドライブ接続台数

接続ドライブ ＼ 機種		CZ-850C	CZ-851C	CZ-852C	備考
3" 5" フロッピー ディスク ドライブ	内蔵（台）	—	1	2	最大接続台数はトータルで4台
	内部増設（台）	—	1	—	
	外部増設（台）	4	2	2	
8"フロッピーディスクドライブ（台）		4	4	4	専用インターフェイス使用
5"ハードディスクドライブ（台）		4	4	4	専用インターフェイス使用

（サポートドライブを最大接続した時の容量は４８Mバイトです。）

なお、ＣＺ－８５１Ｃ、ＣＺ－８５２Ｃには外部５"フロッピーディスク　インターフェイスは接続しないでください。

3 システムプログラム起動方法

本機では、５"フロッピーディスクドライブをトータル４ドライブ、８"フロッピーディスクドライブを４ドライブ（専用インターフェイス使用）およびハードディスクドライブを４ドライブ（専用インターフェイス使用）一度に接続可能であり、８種類のディスクタイプ　（下表参照）をサポートしているのでシステムプログラムを起動する場合あらかじめシステムディスクの入っているドライブナンバーとディスクタイプを設定する必要があります。その方法として

(1)初期モードスイッチ（ディップスイッチ）により、システムディスクのタイプ設定

(2)キー入力により、システムディスクのタイプ設定

の２種類が可能です。

(1)の方法は　初期モードスイッチにより　起動するシステムディスクナンバーを設定し、システム電源を投入することによってサポートドライブ（３"ｏｒ５"および８"フロッピーディスクドライブ、ハードディスクドライブ）のドライブ０から自動的にシステムプログラムが起動できます。＊出荷時ディップスイッチは全てＯＮ（ディスクタイプナンバー０）に設定しています。

ディスクタイプナンバーとディスクの種類

ディップスイッチ			ディスクタイプNo.	内　　容	記録方式	記録容量	ディスク名
SW1	SW2	SW3					
ON	ON	ON	0	2D x1 フォーマット	両　面 倍密度記録	320K	3" 5"FD
OFF	ON	ON	1	2DD x1 フォーマット	両面倍トラック 倍密度記録	640K	
ON	OFF	ON	2	2HD x1 フォーマット	両面高密度 倍密度記録	1 M	
OFF	OFF	ON	3	＊2HD 8"標準 フォーマット	両面高密度 倍密度記録	1 M	
ON	ON	OFF	4	2D x1 フォーマット	両　面 倍密度記録	1 M	8"FD
OFF	ON	OFF	5	＊2D 8"標準 フォーマット	両　面 倍密度記録	1 M	
ON	OFF	OFF	6	1S 8"標準 フォーマット	片　面 単密度記録	240K	
OFF	OFF	OFF	7	x1 フォーマット	4ヘッド 倍密度記録	10 M	ハードディスク

＊但しヘッド0、シリンダ0のみ1セクタ＝128 バイトの単密度記録

またシステム　プログラムを起動したサポートドライブのドライブ0に対する

ディスクタイプも同時に　システムの　ディスクタイプ設定レジスタに書き込まれ、以後のプログラム上で、ドライブのデバイス名を省略してアクセスしようとした場合は、このドライイブナンバーが指定されます。

一般にこの様にシステムプログラムを起動したドライブを、カレントドライブと呼んでいます。

一方、システムプログラムが起動できなかった時は

```
     Make your device ready
Press selected Key to start driving:
          F:FLoppy
          R:ROM
          C:CMT
          T:Timer
```

という表示となるので初期モードスイッチで設定したディスクタイプのドライブ状態の再確認後、再び電源を入れてシステムプログラムを起動するか、キーボード入力での操作で、システムプログラム

を起動してください。

(2)キー入力で　システムプログラムを起動する場合

　初期モードスイッチの設定に関係なくキー入力によって　ドライブの選択を行ない任意のディスクタイプ（ナンバー0から7）のディスクから　システムプログラムが起動できます。またそのドライブナンバーがカレントドライブに設定され　デバイス名　を省略してアクセスした時はカレントドライブ（システムプログラムを起動したドライブ）が指定されます。

操作方法

　本機の電源をONすると

```
IPL is set for device ..... (1)
```

という表示が表示画面に出　キーボードから [F] キーを入力します。

```
Drive No.?  (0-3) ..... (2)
```

という表示となりますので、次にシステムプログラムを起動しようとする　ドライブナンバー（0～3）を指定するために、数字キーの [0] ～ [3] のうち指定のナンバーをキー入力すると、

```
Type of DISK ?  (0-7)

5"FD  320K bytes     2D      ..... 0
  or
3"FD  640K bytes     2DD     ..... 1
      1M   bytes     2HD     ..... 2
      1M   bytes    *2HD     ..... 3
                                ..... (3)
8"FD  1M   bytes     2D-256  ..... 4
      1M   bytes    *2D-256  ..... 5
      240K bytes    *1S-128  ..... 6

5"HD  10M  bytes              ..... 7
```

とサポートしている全てのディスクタイプが表示されますので、起動したいディスクタイプに対応しているナンバーを指定するために、数字キーの0～7キーを使って入力すると任意のドライブナンバーから任意のディスクタイプでシステムプログラムが起動されます。表示画面のとしては

```
IPL is looking for a program from FD [NO] ... (4)
                                （NOは0～3のうちのどれか1つ）
```

となり、設定したディスクタイプで読み出し可能な時は

```
IPL is loading BASIC CZ-8FB02 ..... (5)
```

表示となり、BASICの読み出しを開始します。

一方設定したドライブナンバーから、設定したディスクタイプで読み出し不可能だった時は

```
      Make your device ready
Press selected key to start driving:
          F:Floppy
          R:ROM
          C:CMT
          T:Timer
```

という表示となるので、入出力装置の状態を確認の上、再操作してください。

キーボードから設定して、システムプログラムを起動した場合、起動したドライブに対するディスクタイプのナンバーは自動的にシステムのディスクタイプ設定レジスタに書き込まれます。他のサポートドライブに対しては初期設定（３"ｏｒ５"ドライブの場合　ディスクタイプ０、８"ドライブの場合ディスクタイプ４）されているので、各ドライブに対してディスクタイプを設定する必要があります。

4 接続ドライブのディスクタイプ設定

本機では、計１２ドライブと接続が可能です。

		デバイス名
・３"ｏｒ５"（2D、2DD、2HD）	---------- ４ドライブ ；	０：～３：
・８"（2D、1S）	---------- ４ドライブ ；	Ｆ０：～Ｆ３：
・ハードディスク	---------- ４ドライブ ；	ＨＤ０：～ＨＤ３：

３"ｏｒ５"および８"のフロッピーディスクドライブにどのタイプのディスクが挿入されているか、本機で認識する必要があります。本機で認識しているディスクタイプと違うディスクを挿入すると、書き込み／読み出しができない場合が生じます。すなわちディスクタイプによってヘッド数、シリンダ数、セクタ数が異なり、また、ＦＭ記録／ＭＦＭ記録、３"ｏｒ５"（２Ｄ、２ＤＤ）／５"（２ＨＤ）、８"（２Ｄ、１Ｓ）の切換えを各フロッピーインターフェイスに指示する必要があるためです。

ディスクタイプを正確に認識することによって　自動的にプログラムの書き込み／読み出しの管理ができます。

ディスクタイプの設定方法として

(1)　ユーティリティプログラムの実行
(2)　ＤＥＶＩＣＥ命令
(3)　ワークレジスタの書き換え

の３方法があります。

(1)は、ユーティリティプログラムを走らせ

各デバイス名、に対応するディスク・タイプＮｏ．をキーボードから入力すると自動的に設定できます。

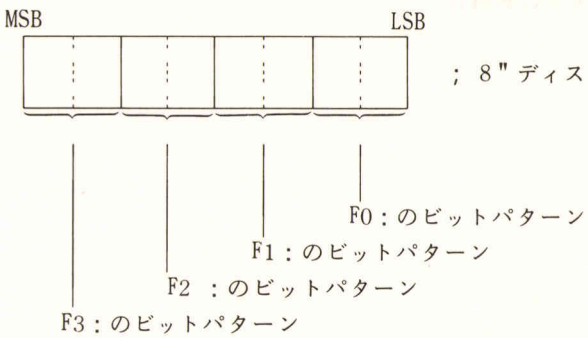
(2)は、ＢＡＳＩＣの　ＤＥＶＩＣＥ命令で

DEVICE "デバイス名：ディスクタイプNo．"
によって設定できます。（下表参照）

デバイス名	ディスクタイプNo.
0： ～ 3：	0---5"or3"FD 2D 1--- 2DD 2--- 2HD 3--- 2HD
F0： ～ F3：	0--- 8" FD 2D 1--- 2D 2--- 1S

(3)はモニタコマンドまたはBASICで、ワークレジスタを変更する方法です。
ディスクタイプのワークレジスタは　各デバイス名について
5"ディスクは1バイト、　8"ディスクは2ビット毎のビットパターン1バイトから成っています。

	デバイス名	ワークレジスタアドレス	データ
5" FD	0：	FAB4	0：2D　x1フォーマット
	1：	FAB5	1：2DD　x1フォーマット
	2：	FAB6	2：2HD　x1フォーマット
	3：	FAB7	3：2HD　8"標準フォーマット
8" FD	F0：	FAB8	00：2D　x1フォーマット
	F1：		01：2D　8"標準フォーマット
	F2：		10：1S　8"標準フォーマット
	F3：		（ビットパターン）



例えば　接続ディスクが

3"or5"	0：	2D	8"	F0：	2DX1フォーマット
FD	1：	2D	FD	F1：	2DX1フォーマット
	2：	2DD		F2：	2D8"標準フォーマット
	3：	2HDX1フォーマット		F3：	1S8"標準フォーマット

の場合

ディスクタイプのワークレジスタは

FAB4 ； 00
FAB5 ； 00
FAB6 ； 01
FAB7 ； 02
FAB8 ； 90　　となります。

ディスクタイプのデフォルト（初期モードスイッチの設定の場合）

(1)3"or5"ディスクでBOOTした時

デバイス名	デフォルトタイプ
0：	BOOTしたディスクタイプ
1：～3：	2D
F0：～F3：	2DX1フォーマット

(2) 8"ディスクでBOOTした時

デバイス名	デフォルトタイプ
0：～3：	2D
F0：	BOOTしたディスクタイプ
F1：～F3：	2DX1フォーマット

(3) ハードディスクでBOOTした時

デバイス名	デフォルトタイプ
0：～3：	2D
F0：～F3：	2DX1フォーマット

また、キー入力で、BOOTした時は、BOOTしたドライブのデバイス名が、そのディスクタイプに設定され、他の3"or5"、8"フロッピーディスクドライブは

3"or5"FD…　2D

8"FD…　2DX1フォーマット

に設定されます。

5　ディスクのフォーマット構成

ディスクにデータの書き込み読み出し動作を行なう場合　ディスク上のアドレス情報（ヘッドNo.、シリンダーNo.、セクターNo等）を基にして、データの格納場所を検索し実行します。　このため、ディスク上にこのアドレスを割り付ける必要があります。　この動作をイニシャライズ（1次フォーマット）といいます。

3"or5"FD2D、2DDの市販ディスクはイニシャライズされていないディスクなので、必ずイニシャライズする必要があります。一方、5"FD2HDおよび8"FD2D、1Sは8"FD標準フォーマットでイニシャライズされているので、必ずしもイニシャライズする必要はありませんが全レコード256バイト／セクター，倍密度記録のX1フォーマット注1)に、イニシャライズして使用してもかまいません。

サポートしているディスクタイプおよびフォーマットを下記に示します。

ディスク＼項目		ヘッドNo	シリンダNo	セクターNo	セクターデータ（バイト）	内　容
3" or 5" FD	2D	0,1	0～39	1～16	256	X1 フォーマット
	2DD	0,1	0～79	1～16	256	X1 フォーマット
	2HD	0,1	0～76	1～26	256	X1 フォーマット
		0,1	0～76	1～26	256　注 2) ヘッド0 シリンダー0 128	8"FD標準 フォーマット
8" FD	2D	0,1	0～76	1～26	256	X1 フォーマット
		0,1	0～76	1～26	256　注 2) ヘッド0 シリンダー0 128	8"FD標準 フォーマット
	1S	0	0～76	1～26	注 3) 128	8"FD標準 フォーマット
ハードディスク		0～3	0～305	1～33	256	X1 フォーマット

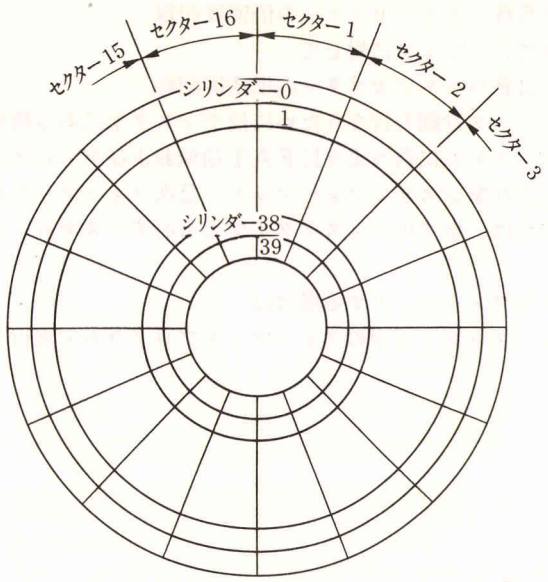
注1）X1フォーマートは、全てのシリンダーに対して
256バイト／セクターの倍密度記録

注2）8"FD　標準フォーマットは
ヘッドNo.0、シリンダーNo.0のセクターNo.1～26は、
128バイト／セクターの単密度記録

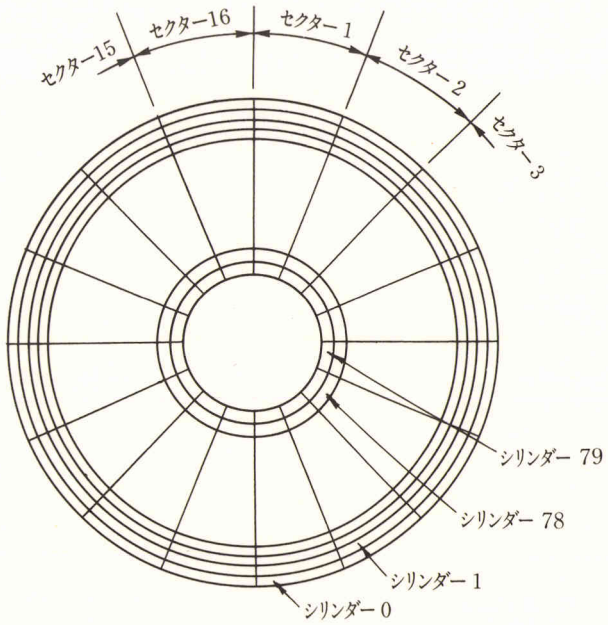
その他の領域は
２５６バイト／セクターの倍密度記録
注３）全てのシリンダに対して
１２８バイト／セクターの単密度記録

また、システムで　ディスク管理を行なうためにはディスク上にある特別な領域、ＦＡＴ領域とディレクトリ領域を設け、システムに合うようにＦＡＴ領域およびディレクトリ領域に初期データを書き込む必要があります。これをシステムフォーマット（２次フォーマット）と呼んでいます。　本機ではディスク管理する場合は、全てのディスクタイプとも必ずシステムフォーマットする必要があります。

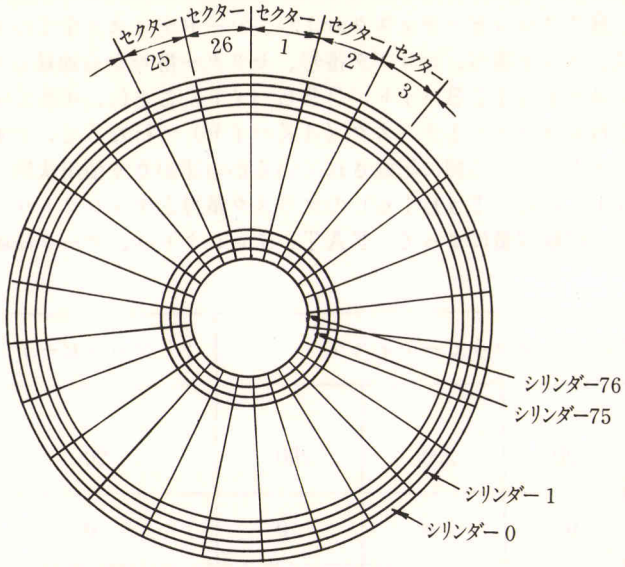
実際にディスクをフォーマットする場合は
ユーティリティ　プログラム中のフォーマットプログラムで実行してください。



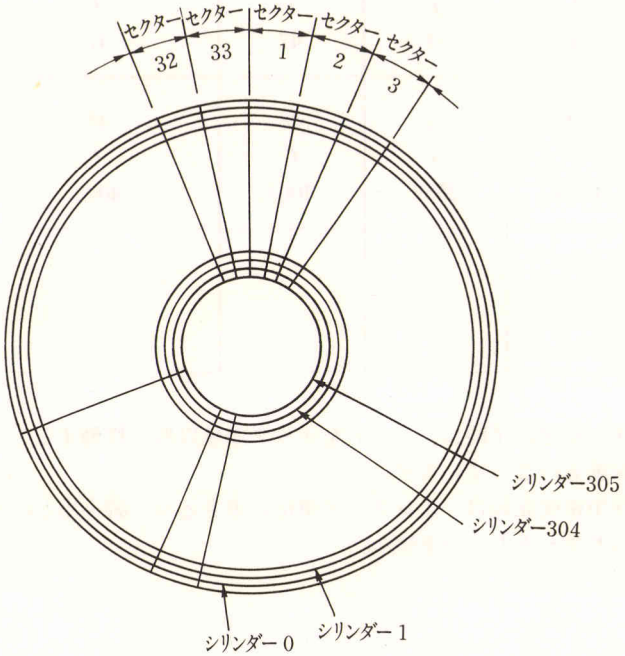
5″ FD 2D タイプのディスク構造



5″ FD 2DD タイプのディスク構造



5″ FD 2HD
8″ FD 2D 1S タイプのディスク構造



5″ ハードディスクのディスク構造

6 ディスクファイル管理方法

3"or5"、8"フロッピーディスクおよび、ハードディスク全てのタイプのディスクへの書き込み／読み出しは、ヘッド番号、シリンダ番号、セクター番号から連続した論理アドレスに変換し、その最小単位のレコード（128バイト／256バイト）ごとに、可能ですが実際のファイル管理での最小単位は、16レコード＝1クラスタ（4Kバイト）で行なっています。

ファイルがディスク上にどの様に記録されているかの連がりや使用状態の管理は、FAT（File Allocation Table）のクラスタ番号とディレクトリデータで行なっています。各ディスクタイプの記録容量によって FAT、ディレクトリ、データ領域は下記の構成となっています。

（単位＝レコード）

	3"or 5"フロッピーディスク			8"フロッピーディスク	ハード
ディスク / 項目	2D	2DD	2HD	2D	ディスク
システム領域	0	0	0	0	0
FAT領域	14	14 ～ 15	28 ～ 29	28 ～ 29	8 ～ 27
ディレクトリ領域	16 ～ 31	16 ～ 31	32 ～ 47	32 ～ 47	48 ～ 63
データ領域	32 ～ 1279	32 ～ 2559	48 ～ 4003	48 ～ 4003	64 ～ 40127
代替領域	—	—	—	—	40128 ～ 40391

ハードディスクの使用終了時に、ヘッドをデータ領域以外に移動することにより、ディスクデータの破カイからの保護を行なっています。

8"FDの片面単密度記録は 1レコード単位の書き込み／読み出しは可能であるがBASICでのファイル管理はサポートしていません。

(1)ディレクトリ

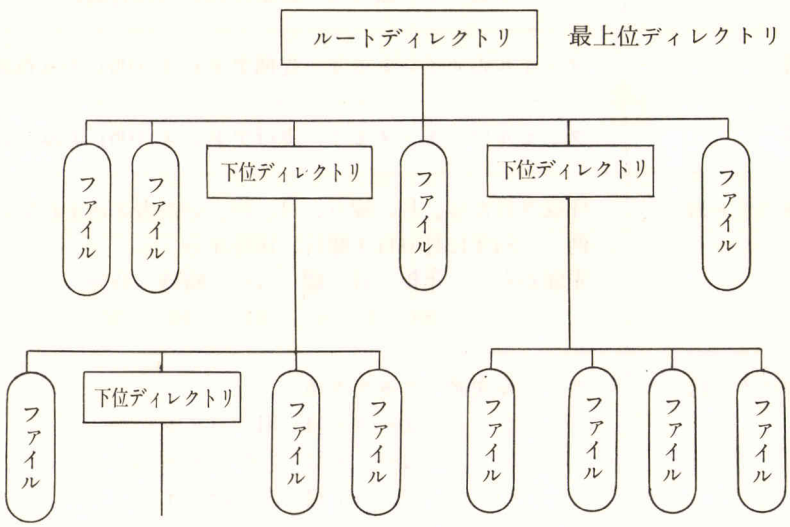
ディレクトリはファイル管理のためのファイル名、格納先頭クラスタ番号、データ数、日付等を表わし、１ファイル３２バイトで表現しています。

各タイプのディスクとも階層ディレクトリ形式なので、データ領域内に下位ディレクトリが設定され、各下位ディレクトリはディスクタイプに関係なく、１クラスタ分（４Ｋバイト：１２８ファイル数）の領域を確保できます。最上位ディレクトリ（ルートディレクトリ）で管理できるファイル数はディスクタイプで異なり下表に示すとおりです。

ディスクタイプで
管理可能な最大ファイル数

ディスク ＼ 項目		ルートディレクトリ領域　レコード数	ファイル数
3" or 5" FD	2D	16	78
	2DD	16	158
	2HD	16	247
8" FD	2D	16	247
ハードディスク		16	2504

ディレクトリの構成は下図の様なツリー構造になります。



ディレクトリのファイルに対する構成を以下に示します。（1ファイル＝32バイト）

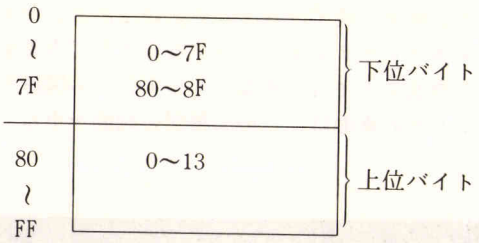
1　ファイル名に対するディレクトリの構成

	内　　容
1バイト目	種類を表わす。　00はKILLされたファイルまたは未使用領域。FFは使用ディレクトリテーブルの終り。 bit　0が1 ……Bin　ファイル（機械語で書かれたファイル） bit　1が1 ……Bas　ファイル（BASIC テキストで書かれたファイル） bit　2が1 ……Asc　ファイル（ASCII セーブされたファイル） bit　4が1 ……FILES で表示しない： 0…表示する bit　5が1 …リードアフターライトON： 0… OFF bit　6が1 …書き込み禁止ファイル： 0…書き込みOK bit　7が1 …下位ディレクトリ bit　3は予備
2バイト目～14バイト目	ファイル名（13文字）
15バイト目～17バイト目	ユーザー指定　EXTENTION　エリア　（3文字）
18バイト目	パスワードのバック（無指定なら20（16）の値）
19・20バイト目	ファイルのバイト数（Bas およびObj のみ有効）
21・22バイト目	ファイルのメインメモリー先頭アドレス（Obj のみ有効）
23・24バイト目	ファイルのメインメモリー実行アドレス（Obj のみ有効）
25バイト目～29バイト目	作成された年、月、曜日、日、時、分が書き込まれています。例：'84年12月01日土曜日、16時36分 先頭から、　年年　月　曜　日日　時時　分分 84　C　6　01　16　36
30バイト目～32バイト目	ファイル先頭　クラスタ値 30バイト目　HIGHバイト 31バイト目　LOW バイト 32バイト目　MIDDLEバイト

(2) FAT

ＦＡＴはディスクで管理されるファイルのチェーン状態を表わし、２バイトのクラスタ番号　で表現しています。　ＦＡＴの必要容量（バイト数）は（管理可能な最大ファイル数）×２バイトです。

ＦＡＴ構成としては



図の様にＦＡＴ領域の１レコードを１２８バイトに分割し、各々をクラスタの下位、上位バイトに指定しています。

各ディスクタイプによってディスク管理用の予約バイトとして　ＦＡＴ領域の先頭から

3" or 5" FD2D,2DD	2バイト
5" FD2HD 8" FD2D	3バイト
ハードディスク	4バイト

上表のように必要です。ＦＡＴデータとしては、次のようなものがあります。

ディスク / 項目	3" or 5" FD			8" FD		ハードディスク
	2D	2DD	2HD	2D		
ファイルがチェーンしているクラスタ	02～4F	02～7F 100 ～11F	03～7F 100 ～179	03～7F 100 ～179		04～7F 100 ～17F 1300～1353
ファイル終了クラスタ	80～8F	80～8F	80～8F	80～8F		80～8F
未使用クラスタ	0	0	0	0		0

6章
ディスプレイモード

BASICは、図形文字（アルファベット、数字、漢字、カタカナ、ひらがな、セミグラフィク文字など）、点、線、その他複雑な図形をディスプレイテレビに表示することができます。

アルファベットなどの図形文字を表示する画面をテキスト画面、ドットによる図形を表示する画面をグラフィック画面と呼び、ディスプレイテレビの画面は、この2種類の画面を重ねて表示していると考えることができます。

1 テキスト画面

英数字などの文字列から構成されるBASICのプログラムはテキストと呼ばれ、このテキストをはじめとする図形文字を表示する画面のことをテキスト画面といいます。

テキスト画面の表示文字数（横×たて）は、WIDTHステートメントを使って、次の8通りの設定をすることができます。ただし、横の文字数は、表示する文字が1バイトコード文字（半角文字）のときの数です。

ステートメント	表示文字数	備　　考
WIDTH40,10,g,d	40×10	標準ディスプレイモードのときのみ設定開始。 アンダーラインが引ける。 グラフィック画面表示不可。
WIDTH80,10,g,d,	80×10	
WIDTH40,12,g,d,	40×12	
WIDTH80,12,g,d	80×12	
WIDTH40,20,g,d,	40×20	アンダーラインが引ける。 グラフィック画面表示不可。
WIDTH80,20,g,d,	80×20	
WIDTH40,25,g,d,	40×25	
WIDTH80,25,g,d,	80×25	

※ g：グラフィック画面の解像度の設定　0＝200／192ドット　1＝400／384ドット

d：ディスプレイモードの設定

0＝本体の標準／高解像度切換スイッチの状態に従う。

▄▄：標準ディスプレイモード（STANDARD）

▄▄：高解像度ディスプレイモード（HIGH）

1＝標準ディスプレイモード

2＝高解像度ディスプレイモード

（注）・次のＷＩＤＴＨステートメントは使用できません。
・ＷＩＤＴＨ　ｘ，ｙ，１，１
・本体の標準／高解像度切換スイッチが標準ディスプレイモード（▬）にセットされている時、ＷＩＤＴＨ　ｘ，ｙ，１，０
ただし、ｘ＝４０　ｏｒ　８０、ｙ＝２５　ｏｒ　１２
・標準ディスプレイでは、標準ディスプレイモードのみ、高解像度ディスプレイでは、高解像度ディスプレイモードのみ使用可能です。
・専用ディスプレイＴＶでは、標準ディスプレイモード及び高解像度ディスプレイモードの双方使用可能です。
その切換えは、ＷＩＤＴＨステートメントの第４パラメータによって行ないます。

2 グラフィック画面

グラフィック画面は、２つのグラフィックメモリ（４８ＫＢが２つ）から構成されており、次のような解像度（横×たて）をもつ画面を設定することができます。

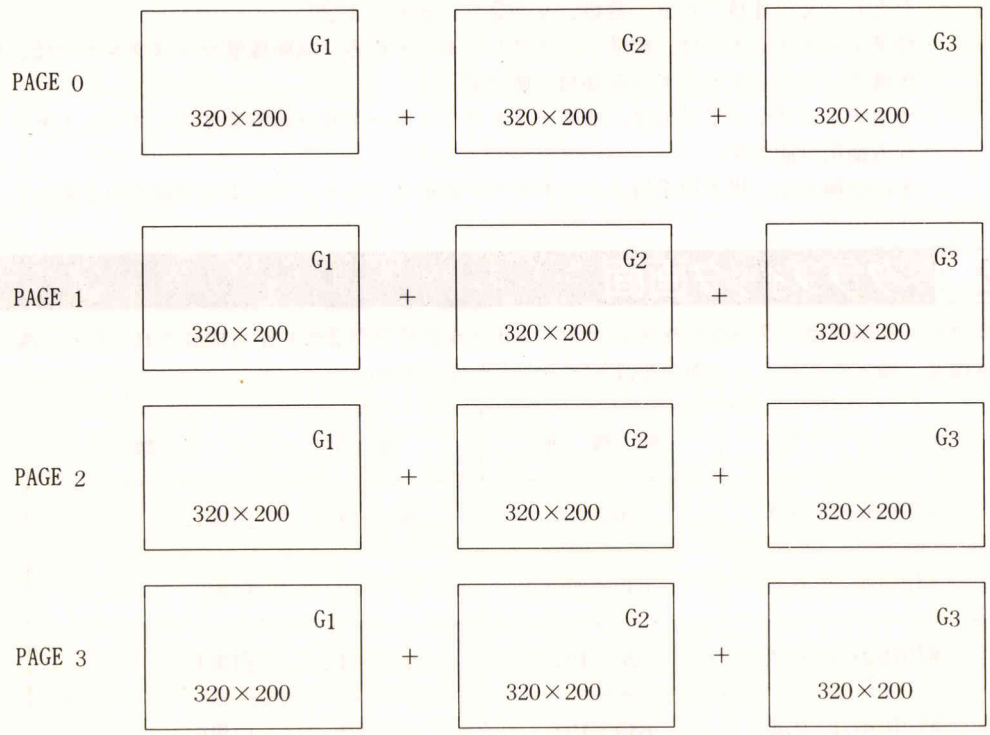
ステートメント	解像度	使用ページ数
WIDTH40,25,0,d	320×200	カラー4　／　白黒12
WIDTH80,25,0,d	640×200	カラー2　／　白黒6
WIDTH40,12,0,d	320×192	カラー4　／　白黒12
WIDTH80,12,0,d	640×192	カラー2　／　白黒6
WIDTH40,25,0,D	320×200	カラー4　／　白黒12
WIDTH80,25,0,D	640×200	カラー2　／　白黒6
WIDTH40,12,0,D	320×192	カラー4　／　白黒12
WIDTH80,12,0,D	640×192	カラー2　／　白黒6
WIDTH40,25,1,D	320×400	カラー2　／　白黒6
WIDTH80,25,1,D	640×400	カラー1　／　白黒3
WIDTH40,12,1,D	320×384	カラー2　／　白黒6
WIDTH80,12,1,D	640×384	カラー1　／　白黒3

ｄ：０ｏｒ１（標準ディスプレイモードに設定。ｄ＝０のとき、本体の標準／高解像度切換スイッチは標準ディスプレイモード（▬）にセットされていること
Ｄ：０ｏｒ２（高解像度ディスプレイモードに設定。ｄ＝０のとき、本体の標準／高度解像度切換えスイッチは高解像ディスプレイモード（■）にセットされていること

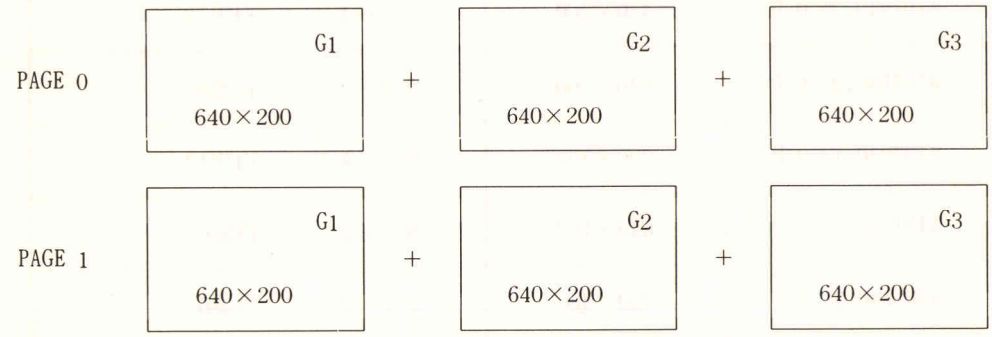
グラフィック画面の基本的な構成図を次に示します。

この説明書では、1枚目のグラフィック画面をG1、2枚目の画面をG2、3枚目の両面をG3と呼んで区別しています。

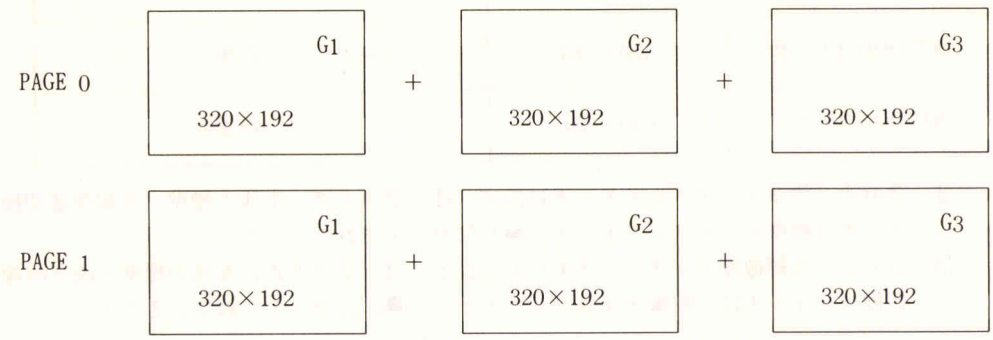
＜高解像度／標準ディスプレイモードで、WIDTH40,25,0,Dを設定（D =0 or1 or2 ）＞



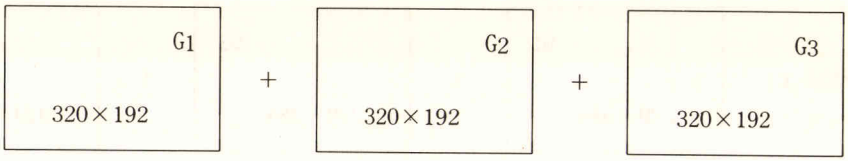
＜高解像度／標準ディスプレイモードで、WIDTH80,25,0,Dを設定（ D= 0or1or2）＞



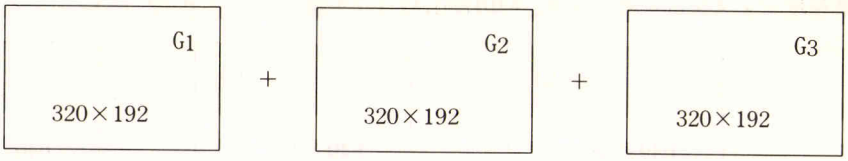
＜高解像度／標準ディスプレイモードで、WIDTH40,12,0,Dを設定（ D= 0or1or2）＞



PAGE 2

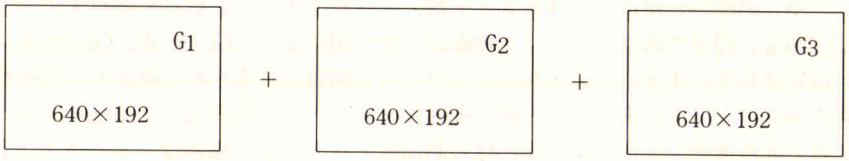


PAGE 3

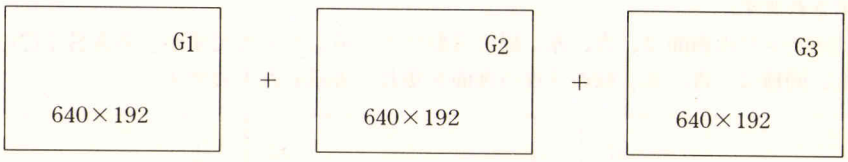


<高解像度／標準ディスプレイモードで、WIDTH80,12,0,Dを設定 (D= 0or1or2) >

PAGE 0

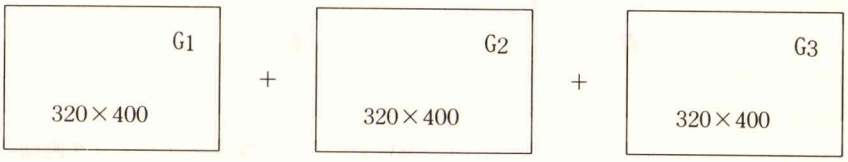


PAGE 1

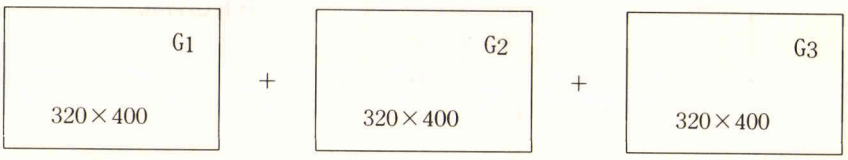


<高解像度ディスプレイモードで、WIDTH40,25,1,Dを設定 (D= 0or2) >

PAGE 0

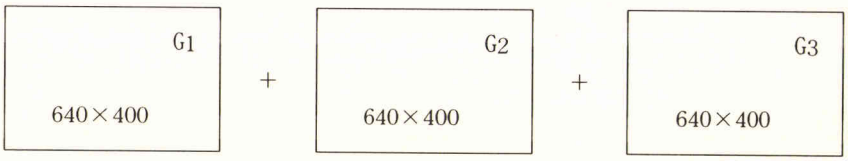


PAGE 1



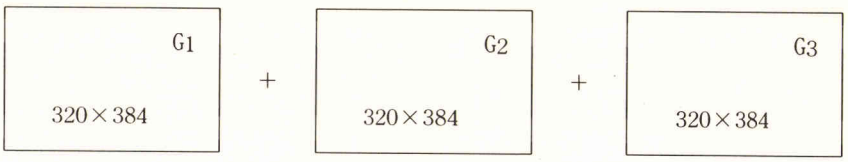
<高解像度ディスプレイモードで、WIDTH80,25,1,Dを設定 (D= 0or2) >

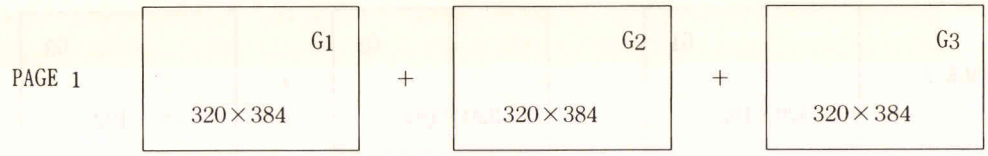
PAGE 0



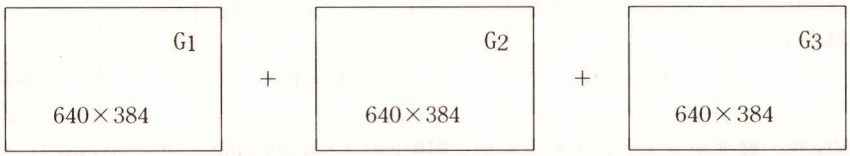
<高解像度ディスプレイモードで、WIDTH40,12,1,Dを設定 (D= 0or2) >

PAGE 0





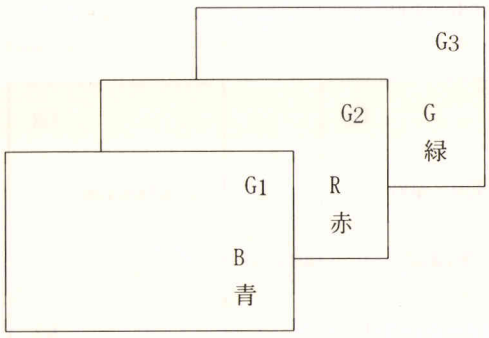
〈高解像度ディスプレイモードで、WIDTH80,12,1,Dを設定（ D= 0or2 ）〉



※　画面内の積は、（横のドット数）×（たてのドット数）を表わす。

G1、G2、G3の各グラフィック画面は、次の図のようにG1が青、G2が赤、G3が緑というように設定されています。この3枚のグラフィック画面は合成されて画面上に表示されます。たとえば、青の画面にドットがあり、赤と緑の画面にドットがない場合はそのドットは青で表示されます。また、青と赤の画面にドットがあり、緑の画面にドットがない場合は、そのドットはマゼンタ（紫）で表示されます。

（カラーテレビの画面は、青、赤、緑の3原色を合成したものですが、BASICのカラーグラフィックも、同様に、青、赤、緑の3枚の画面を重ねて表示したものです。）



3枚のグラフィック画面の合成
（BRG合成）

3 カラーコード

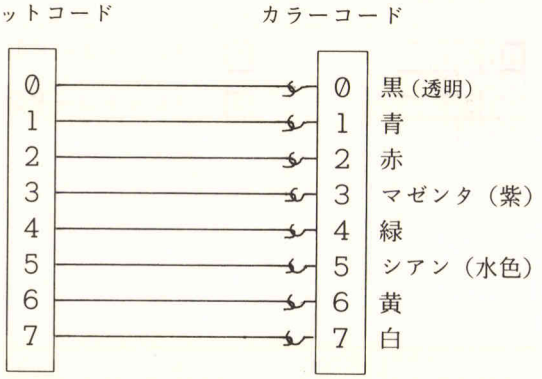
カラーコードは0～7の整数で表わされ、その値によって画面の色が設定されます。
次の図はG1、G2、G3の画面とカラーコードの関係を示しています。
このようにカラーコードとは、青の画面をビット0、赤の画面をビット1、緑の画面をビット2とした3ビットの2進数を10進数表現したものとなっています。

色	G3 G2 G1	2進数ビット表現	カラーコード
黒（透明）	○ ○ ○	000	0
青	○ ○ ●	001	1
赤	○ ● ○	010	2
マゼンタ	○ ● ●	011	3
緑	● ○ ○	100	4
シアン	● ○ ●	101	5
黄	● ● ○	110	6
白	● ● ●	111	7

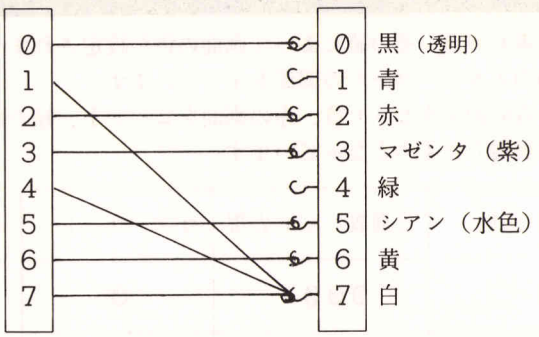
※ ●はドットがある、○はドットがないことを示しています。

4 パレットコード

カラーコードは色そのものを指定するコードで、
0＝黒(透明)、1＝青、2＝赤、3＝マゼンタ、4＝緑、5＝シアン、6＝黄、7＝白
とういうように、数字と色が絶対的に対応しています。
これに対してパレットコードも色を指定するコードで0～7の数字で表わされますが、カラーコードのように数字に対する色がきまっていず、自由に色を設定することができます。
パレットコードの色の設定はPALETステートメントを使って行ないます。
次の図はパレットコードの設定の概念を示すものです。



BASICが起動した最初の状態は上のようにパレットコード＝カラーコードとなっていますが、次のようにフックをかけなおすと、パレットコードの1と4が白として使えるようになります。



5 中間色コード

グラフィックステートメントのうち

LINEのBF（ボックスフル）
PAINT
SYMBOL

の３つに対して、パレットコードを指定する代わりに「中間色コード」を指定することができます。

中間色コードは、

&Hmn（またはm＊16＋n）　m＝0～7のパレットコード
n＝0～Fのコード

の16進数２けたで表わされ、mとnの２色を混合した中間色を出すことができます。

ただし、mとnが同じパレットコードの場合は、パレットコードmかnを単独で指定したときの色になります。また、黒と他の色との混合色

&H00、&H01、……、&H07

はカラーコードの

0、1、……、7

そのもので中間色を出すことができないので、黒と他の色との混合色は、

&H08、&H09、……、&H0F

で代用しています。

〈&Hmnと&Hnmの相違点〉

&H12と&H21はパレットコード１（初期状態が青）とパレットコード２（初期状態が赤）を混合した同じ中間色（初期状態で紫）を表わしますが、グラフィック画面上のドット構成は次のように全く反転しています。

&H12

(0,0)

1	2	1	2
2	1	2	1
1	2	1	2
2	1	2	1

&H21

(0,0)

2	1	2	1
1	2	1	2
2	1	2	1
1	2	1	2

[1]；パレットコード１でセット

[2]；パレットコード２でセット

〈中間色一覧表〉

中間色コードと表示される色との関係を次の表にまとめます。実際の色を確認する場合は表の下のプログラムを実行してください。

中間色コード　＆Hmn または m＊16＋n

m＼n	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	黒 (0)	青 (1)	赤 (2)	ﾏｾﾞﾝﾀ (3)	緑 (4)	シアン (5)	黄 (6)	白 (7)	黒 (8)	ダークブルー (9)	茶 (10)	紫 (11)	深緑 (12)	青緑 (13)	黄土色 (14)	灰色 (15)
1	ダークブルー (16)	青 (17)	紫 (18)	青紫 (19)	青緑 (20)	空色 (21)	灰色 (22)	薄紫 (23)								
2	茶 (32)	紫 (33)	赤 (34)	ピンク (35)	黄土色 (36)	灰色 (37)	褐色 (38)	はだ色 (39)								
3	紫 (48)	青紫 (49)	ピンク (50)	ﾏｾﾞﾝﾀ (51)	灰色 (52)	薄紫 (53)	はだ色 (54)	薄ふじ色 (55)								
4	深緑 (64)	青緑 (65)	黄土色 (66)	灰色 (67)	緑 (68)	緑青 (69)	黄緑 (70)	ホワイトグリーン (71)								
5	青緑 (80)	空色 (81)	灰色 (82)	薄紫 (83)	緑青 (84)	シアン (85)	ホワイトグリーン (86)	水色 (87)								
6	黄土色 (96)	灰色 (97)	褐色 (98)	はだ色 (99)	黄緑 (100)	ホワイトグリーン (101)	黄 (102)	クリーム (103)								
7	灰色 (112)	薄紫 (113)	はだ色 (114)	薄ふじ色 (115)	ホワイトグリーン (116)	水色 (117)	クリーム (118)	白 (119)								

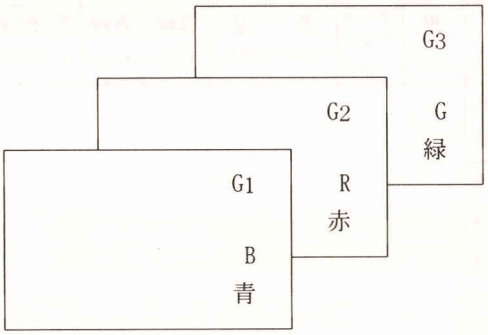
・色は、初期状態のパレットコードによって出されるもの。
・（　）内は10進数表現。

```
100 ' 中間色コード
110 INIT:WIDTH 80,12,0:CLS4
120 CSIZE2:PRINT#0, "   中間色コード（
&Hmn or m*16+n）":CSIZE0:PRINT
130 PRINT"␣␣␣␣m";CHR$(&H815F);"n␣␣␣0␣␣␣1
␣␣␣2␣␣␣3␣␣␣4␣␣␣5␣␣␣6␣␣␣7␣␣␣8␣␣␣9␣␣␣A␣␣␣B
␣␣␣C␣␣␣D␣␣␣E␣␣␣F";
140 FOR I=0 TO 7
150 LOCATE 3,I+3 :PRINT I;
160 NEXT
170 FOR J=0 TO 7
180 FOR I=0 TO 15
190 LINE(67+I*32,48+J*16)-(96+I*32,62+J
*16),PSET,J*16+I,BF
200 NEXT
210 NEXT
220 LOCATE 0,0:END
```

6 カラーモード

カラーモードとは、「SCREEN文の第3パラメータ（グラフィックモードの設定）が0となっていて、全グラフィック画面がアクセスできるモード」のことをいいます。

このモードのとき、グラフィック画面の各ドットに対して8色のうち任意の1色を指定することができます。



3枚のグラフィック画面を合成した画面で、8色のカラーを指定できる画面をカラー画面といいます。

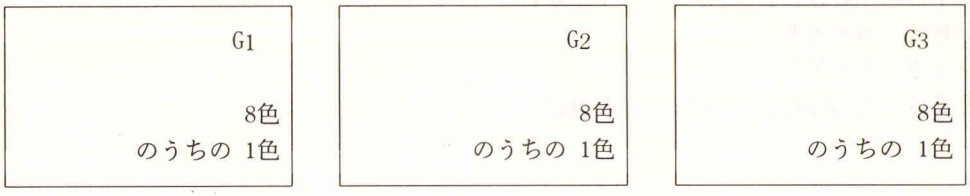
画面の解像度の設定により、何ページかの独立したカラー画面を設定することができます。

解　像　度	カラー画面のページ数
320×200 、 320×192	4（PAGE0 ～PAGE3 ）
640×200 、 640×192 320×400 、 320×384	2 （PAGE0 , PAGE1 ）
640×400 、 640×384	1

カラー画面が複数ページであるときは、SCREENステートメントの第1パラメータと第2パラメータで表示するページと書き込むページを指定することができます。

7 マルチページモード

マルチページモードとは、「SCREEN文の第3パラメータ（グラフィックモード）の指定が1、2、3のいずれかになっていて、G1、G2、G3の3枚の画面を各々単色画面としてアクセスできるモード」のことをいいます。このモードのとき、CANVASステートメントによって、各画面の色を任意の1色に設定することができます。



画面の解像度により、使用できる画面のページ数を設定することができます。

ステートメント	解像度	マルチ画面のページ数
WIDTH40,25,0,d	320×200	3×4
WIDTH40,12,0,d	320×192	
WIDTH80,25,0,d	640×200	3×2
WIDTH80,12,0,d	640×192	
WIDTH40,25,0,D	320×400	
WIDTH40,12,0,D	320×384	
WIDTH80,25,0,D	640×400	3×1
WIDTH80,12,0,D	640×384	

※d：0、1、2（標準ディスプレイモード又は高解像度ディスプレイモードに設定。）
D：0、2（高解像度ディスプレイモードに設定。D＝0の時本体の標準／高解像度切換スイッチは、HIGH側にセットされていること。）

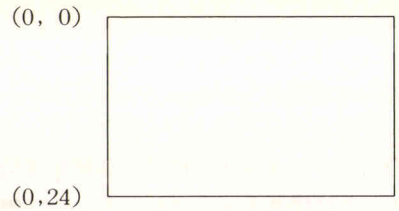
8 ディスプレイ画面上の座標系

テキスト画面、グラフィック画面とも画面の左上が原点（0，0）であり、横方向がx軸、たて方向がy軸となっています。

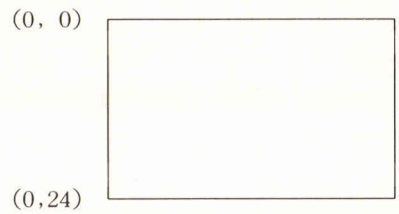
8.1 テキスト座標系

テキスト座標系は、テキスト画面において設定されている座標系で、表示文字数によって次の図のようになっています。

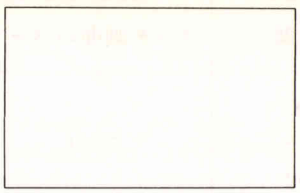
<WIDTH40,25,G,D（G=0or1、D=0or1or2）のとき>



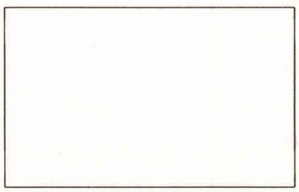
<WIDTH80,25,G,D（G=0or1、D=0or1or2）のとき>



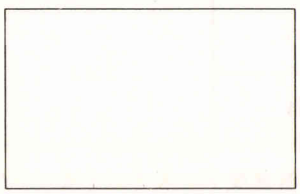
<WIDTH40,12,G,D (G=0or1、 D=0or1or2) のとき>



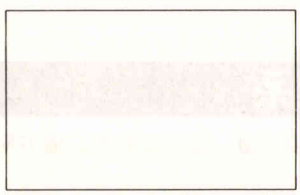
<WIDTH80,12,G,D (G=0or1、 D=0or1or2) のとき>



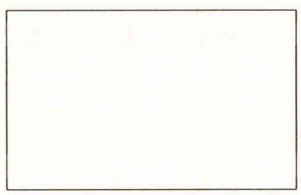
<WIDTH40,20,G,D (G=0or1、 D=0or1or2) のとき>



<WIDTH80,20,G,D (G=0or1、 D=0or1or2) のとき>

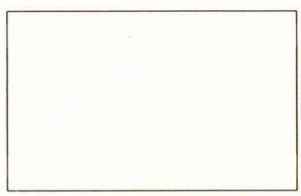


<WIDTH40,10,G,D (G=0or1、 D=0or1) のとき>



※D＝0のとき本体の標準／高解像度切換スイッチは標準ディスプレイモード（▬）にセットされていること。

<WIDTH80,10,G,D (G=0or1、 D=0or1) のとき>



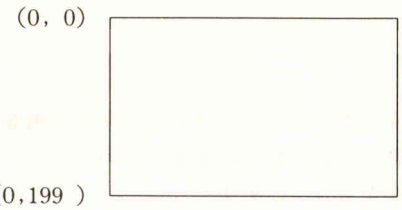
※D＝0のとき本体の標準／高解像度切換スイッチは標準ディスプレイモード（▬）にセットされていること。

CONSOLEおよびLOCATEステートメントは、上に示した座標の範囲で設定することができます。

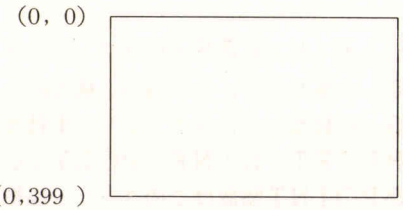
8.2 グラフィック座標系

グラフィック座標系は、グラフィック画面において設定されている座標系です。
グラフィックの解像度によって、次のような座標の範囲が設定されます。

<WIDTH40,25,0,D (D=0or1or2) のとき>

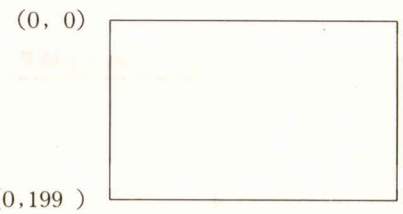


<WIDTH40,25,1,D (D=0or2) のとき>

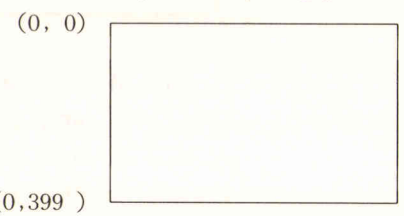


※D＝0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード（■）にセットされていること。

<WIDTH80,25,0,D (D=0or1or2) のとき>

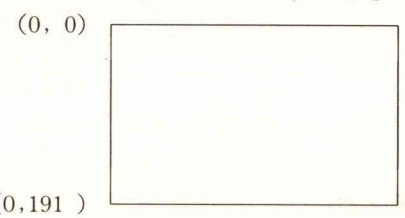


<WIDTH80,25,1,D (D=0or2) のとき>

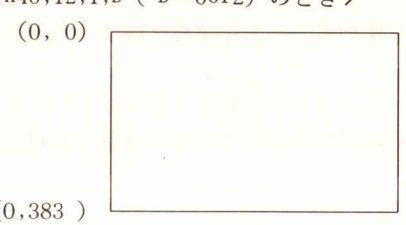


※D＝0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード（■）にセットされていること。

<WIDTH40,12,0,D (D=0or1or2) のとき>



<WIDTH40,12,1,D (D=0or2) のとき>



※D＝0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード（■）にセットされていること。

※WIDTH40,20,G,D (G=0or1、 D=0or1or2) 、
WIDTH80,20,G,D (G=0or1、 D=0or1or2) 、
WIDTH40,10,G,D (G=0or1、 D=0or1) 、
WIDTH80,10,G,D (G=0or1、 D=0or1) のときはグラフィック画面が使用できません。したがって、グラフィック座標系は設定されません。

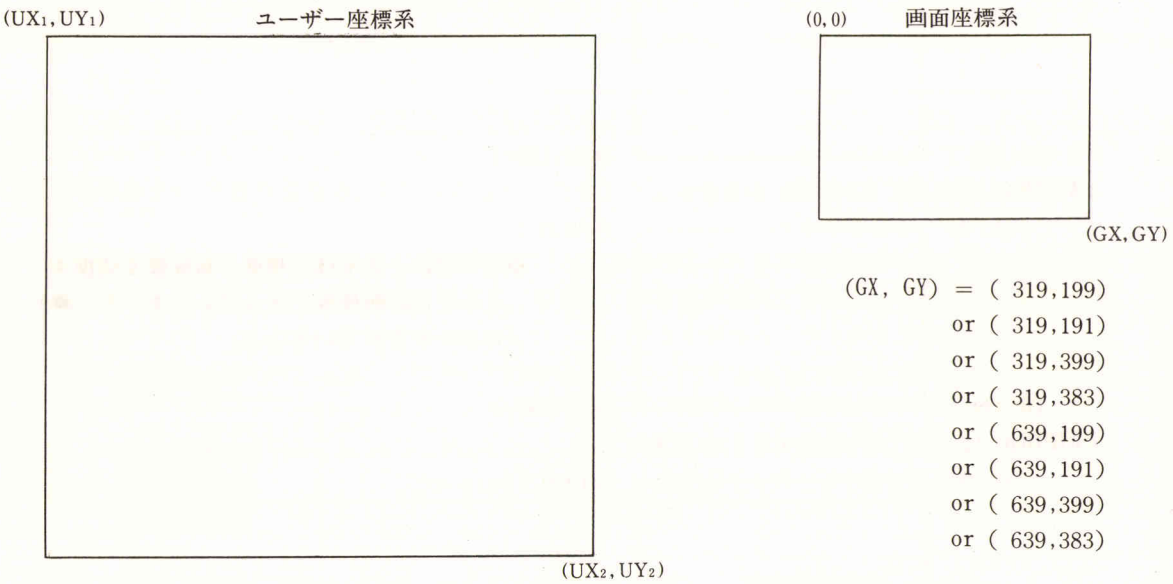
8.3 ユーザ座標系と画面座標系

ディスプレイ画面は、テキスト画面とグラフィック画面が重なって表示され、テキスト画面にはテキスト座標系、グラフィック画面にはグラフィック座標系が設定されています。このうちグラフィック座標系は画面座標系とユーザ座標系という２つの座標系をもっています。

画面座標系とは、グラフィック座標系で説明した座標に一致するもので、画面上の１ドットが座標単位の１に対応します。

ユーザー座標系とは、ＢＡＳＩＣのＷＩＮＤＯＷステートメント（ＢＡＳＩＣリファレンスマニュアルの2.8.2 ＷＩＮＤＯＷ参照）によってユーザーが指定した座標系で、たて方向、横方向とも－１．７０１４１１８Ｅ+３８～１．７０１４１１８Ｅ+３８（指数部－３８～+３８）の単精度の範囲で指定できます。後ろの文法編で詳述するＰＳＥＴ，ＰＲＥＳＥＴ，ＬＩＮＥ，ＰＯＬＹ，ＣＩＲＣＬＥ，ＰＡＩＮＴのグラフィックステートメント、およびＰＯＩＮＴ関数はこのユーザー座標で使用されます。

下の図は、ユーザー座標と画面座標系の概念図です。



(UX1,UY1) = (-1.7014118E +38, -1.7014118E +38)
(UX2,UY2) = (1.7014118E +38, 1.7014118E +38)

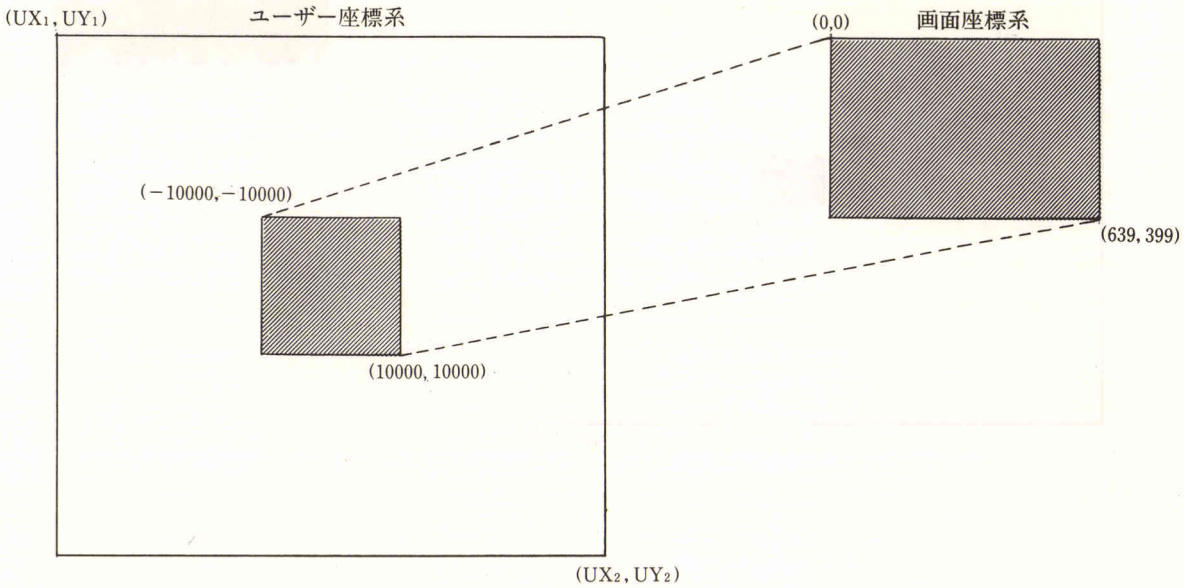
画面座標系は実際に目で見ることができますが、ユーザー座標系の広大な座標平面は目で見ることができません。

したがってユーザーはＢＡＳＩＣのＷＩＮＤＯＷステートメント（ＢＡＳＩＣリファレンスマニュアルの2.8.2 ＷＩＮＤＯＷ参照）を使って、ユーザ座標系の座標平面中の任意の領域を画面座標系の平面上に割り当てなければなりません。

たとえば、WIDTH80、25、1、2が設定されているとき、

WINDOW（0，0）－（639，399），（－10000，－10000）－（10000，10000）

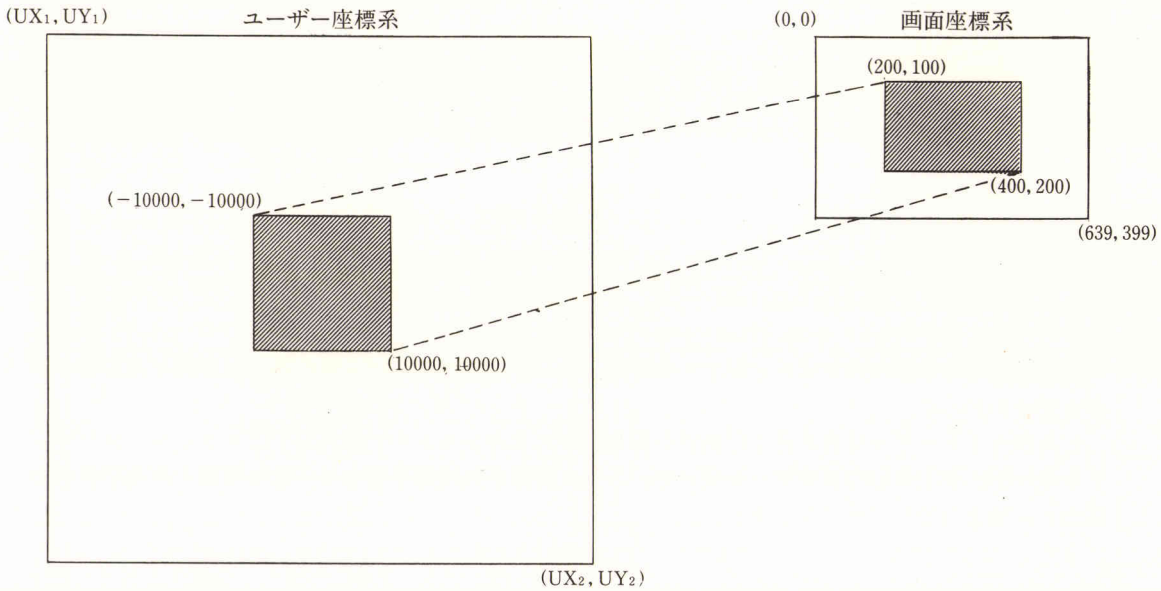
を実行すると、次のようにユーザー座標系の（－10000，－10000）から（10000，10000）を対角線とする長方形の領域を画面座標系全域（グラフィック画面全体）に表示することができます。



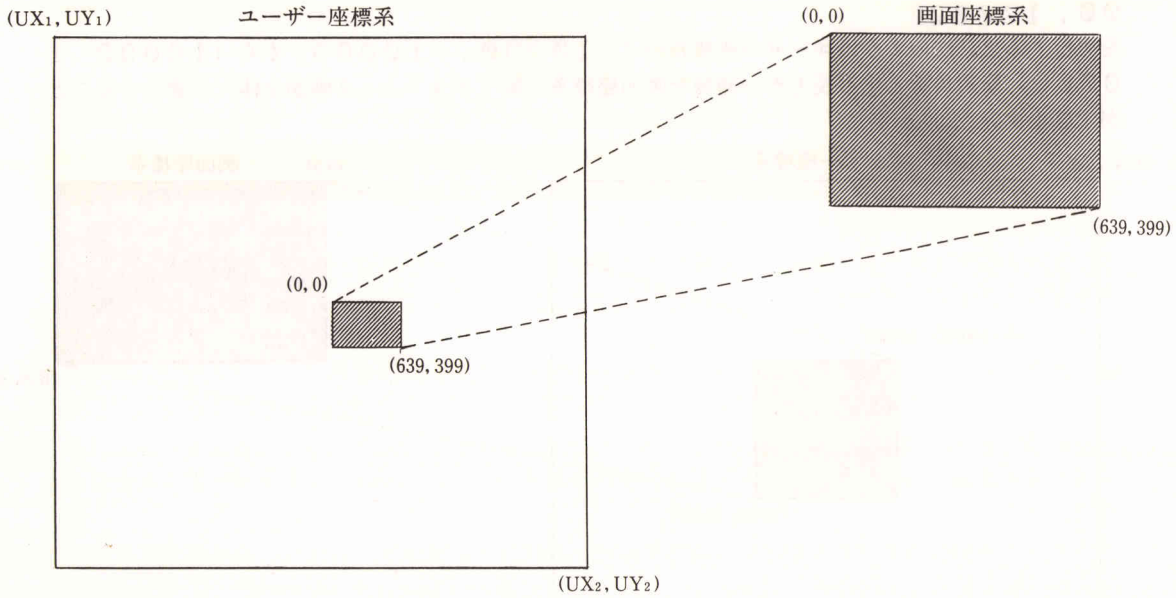
また、

WINDOW（200，100）－（400，200），（－10000，－10000）－（10000，10000）

を実行すると、ユーザー座標系の（－10000，－10000）から（10000，10000）を対角線とする長方形の領域を画面座標系の（200，100）から（400，200）を対角線とする長方形の領域に表示することができます。



後ろのパラメータを省略して、ＷＩＮＤＯＷのみを実行すると、画面座標系の範囲と同じ領域が自動的に指定されます。



7章
テキスト

1 テキスト画面とグラフィック画面

キーボードから入力された文字やプログラムのリストはディスプレイ画面に表示されます。このとき　もしディスプレイ画面に円や線などのグラフィックが描かれていれば　その上に重なって文字が表示されます。ここで画面消去の命令　CLS↵を入力すると文字のみが消去され　グラフィックはそのまま残って表示されています。このように　グラフィックを描く画面と文字を表示する画面は独立しており、それぞれグラフィック画面とテキスト画面といいます。

1.1 テキスト画面

テキスト画面にはアルファベット、数字、漢字、カタカナ、ひらがな、セミグラフィック文字などが書かれます。

テキスト画面に表示できる文字数は使用するディスプレイが標準ディスプレイか高解像度ディスプレイかにより異なり、次の通りです。

標準ディスプレイ（文字数／行）×（行数／画面）

４０文字×２５行	,	８０文字×２５行
４０文字×１２行	,	８０文字×１２行
４０文字×２０行	,	８０文字×２０行
４０文字×１０行	,	８０文字×１０行

注）文字数は半角文字１文字を１文字とする。

高解像度ディスプレイ

４０文字×２５行	,	８０文字×２５行
４０文字×１２行	,	８０文字×１２行
４０文字×２０行	,	８０文字×２０行

電源を投入し　SHARP　HuBASIC　CZ－８FB０２　を起動した直後は、標準ディスプレイ使用時には、８０文字×１２行に設定され、高解像度ディスプレイを使用時には、８０文字×２５行に設定されています。

この画面モードを変更するには　WIDTH　命令を使用します。

WIDTH　〔1行当たりの文字数〕，〔画面に表示する行数〕

たとえば、４０文字×２５行の表示画面に変更するには

```
WIDTH　４０，２５ ↵
```

と入力します。すると、画面をクリアした後、４０文字×２５行画面に変わります。

WIDTH命令で画面に表示する文字数が設定できました。さらに、CONSOLE命令を使用すると、表示画面をテキストの命令が有効な領域と無効な領域に分けることができます。

CONSOLE　たて方向の表示開始行，たて方向の表示行数，横方向の表示開始行，横方向の表示桁数

たとえば、横方向の最初の桁から２０桁行とたて方向の最初の行から１０行の領域をテキストの命令が有効な領域に設定するには、

```
CONSOLE　０，１０，０，２０ ↵
```

と入力します。すると、ＬＩＳＴやＦＩＬＥＳ命令を実行した場合この設定された領域（２０文字×１０行）内でリストやファイル名が表示され、それ以外の領域には表示されません。このようにして、テキスト表示領域内で、表示文字の変化可能領域と変化不可能領域とに分けることができます。この画面分割を解除するには、再度ＣＯＮＳＯＬＥ命令を実行して表示画面全体を指定するかまたはCTRL＋Ｄ キーを押してください。

また、テキスト画面の任意の位置に文字を表示したりする場合にはＬＯＣＡＴＥ命令を使用します。

LOCATE　横方向の位置，たて方向の位置

たとえば、横方向１０桁目、たて方向１０行目の位置からＡＢＣという文字を表示させるには

```
LOCATE 10,10:PRINT"ABC"
```

と入力します。

1.2 ファンクションキーの内容表示

テキスト画面の最下行は　ファンクションキーの内容表示に使用することができます。

ただし、表示画面の最下行は、ＣＯＮＳＯＬＥステートメントによりテキストへの命令が無効領域に設定されている必要があります。

ファンクションキーの表示は　ＫＥＹ　ＬＩＳＴ　という命令を使って行ないます。

```
KEY LIST 0
```

と入力するとファンクションキーの内容は表示されず、

```
KEY LIST 1
```

と入力すると表示されます。

1.3 漢字とセミグラフィックパターン

本機は　漢字をテキスト画面に表示します。漢字を表示する場合画面モードを漢字表示のモードに設定しておく必要があります。この設定を行なうには、ＫＭＯＤＥという命令を使います。

KMODE { 0 / 1 }

```
KMODE 1
```

と入力すると漢字表示モードになります。

ただし、標準ディスプレイモードでたて方向の行数が２５行または２０行に設定されている場合には、漢字は表示できません。

このモードで「漢字」を表示するには

```
PRINT"漢字"
```

あるいは

```
PRINT CHR$(&J3441,&J3B7A)
```

と入力します。また、

```
PRINT CHR$(&H8ABF,&H8E9A)
```

と入力しても

漢字

と表示されます。

漢字コードは内部では２バイトで表現されており、最初の１バイトが&Ｈ８０～&Ｈ９Ｆ　もしくは、&ＨＥ０～&ＨＦＦではじまるコードとなっています。&Ｈ８０～&Ｈ９Ｆ，&ＨＥ０～&ＨＦＦのコードはセミグラフィックパターンのコードと重複しています。したがって漢字とセミグラフィックパターンとは同時に表示できません。そこで漢字の代りにセミグラフィックパターンを表示するには

```
KMODE 0
```

と入力します。ここで

PRINT CHR$(&H8ABF) [↵]

と入力してみてください。今度は「漢」という文字ではなく▌ソというセミグラフィックパターンが表示されます。

2 テキストの属性

テキストは最大80文字×25行の2000文字を表示することができますが、1文字単位で、色を指定することができます。また、反転や点滅も同様に1文字単位で行なえます。
この色や反転、点滅のことを文字の属性といいます。
そのほか文字の属性には倍文字、アンダーライン、ROM／RAMキャラクタジェネレータがあります。
文字の属性は英数字、カナ、漢字すべてに指定できます。

2.1 色の指定

文字の色はCOLORという命令で行なえます。

COLOR　色の番号

0：黒、1：青、2：赤、3：マゼンタ、4：緑、5：シアン、6：黄、7：白

とすると、この命令を実行されてから以後に入力される文字は指定された色になります。
また、[CTRL]キーを押しながらテンキーの[0]～[7]を押すことによっても文字の色指定ができます。

2.2 反転文字

文字は標準状態では指定された色で表示されていますが、反転モードにして入力すると、文字の色が補色になり白い四角で囲まれた状態になります。
補色は

青←→黄
赤←→シアン
緑←→マゼンタ
白←→黒

となります。

CREV { 0または省略 / 1 }

反転モードにするには

CREV 1 [↵]

と入力します。
また

CREV 0 [↵]　もしくは　CREV [↵]

とすると標準モードにもどります。
また、[CTRL]キーを押しながらテンキーの[-]キーを押すことによっても反転モード、標準モードの切り換えができます。

2.3 点滅文字

点滅モードにすると、入力された文字は反転の状態と標準の状態を繰り返し表示します。

```
CFLASH { 0または省略 }
       {    1       }
```

点滅モードにするには

```
CFLASH 1↵
```

とします。

標準モードにするには

```
CFLASH 0↵   または   CFLASH↵
```

とします。また、CTRLキーを押しながらテンキーの×を押すことによっても点滅モード、標準モードの切り換えができます

2.4 倍文字

テキスト画面に表示される文字は、たて方向と横方向をそれぞれ２倍にして表示することができます。

たて方向、横方向とそれぞれ独立して２倍にできますので、合計４通りの大きさの文字を表示することができます。この大きさを指定する命令がCSIZEで

```
CSIZE 番号
```

という書式で書きます。

番号は、

0---------標準のサイズの文字
1---------たて方向２倍の文字
2---------横方向２倍の文字
3---------たて、横　共に２倍の文字

を示します。

さらに、CSIZEで指した大きさの文字を表示するには　PRINT　#0を使います。

たとえば　横に2倍の大きさの文字"ABC"を表示する場合

```
CSIZE 2↵
PRINT #0,"ABC"↵
```

とします。

倍文字のうち、たて倍文字と標準サイズの文字を同一行中に含むことができないなど、使用する条件がありますので、詳しくはBASICリファレンスマニュアルを参照してください。

2.5 ROMCGとRAMCG

テキスト画面に表示される文字はキャラクタジェネレータと呼ばれる部分から呼び出されたパターンが表示されますが、キャラクタジェネレータ（略してCG）は、あらかじめ　固定されたパターンが格納されているROMCGと、ユーザーが自由に定義できるRAMCGがあります。

ROMCGの中には、英数字、カナ、セミグラフィックなどのパターンが入っており、標準状態でキーを入力するとROMCGのパターンが表示されます。RAMCGの内容は電源を切ると消えてしまいます。したがって、RAMCGを使用する場合にはあらかじめRAMCGにパターンを定義しておかなければなりません。この定義に使用するステートメントがDEFCHR$です。また、RAMCGとROMCGとの表示を切り換えるステートメントがCGENです。

そこで例をあげて説明します。

```
CGEN 0↵
PRINT  "A"↵
```

とします。
すると画面にはAという文字が表示されます。
次に

```
DEFCHR$(65)=STRING$(8,&HFF)+STRING$(8,&HF0)
           +STRING$(8,&H81)↵
```

CGEN1↵ PRINT"A"↵ （CGEN1↵以後はどのキーを押しても白ベタに表示されます。）
とすると画面には縞模様のパターンが表示されます。
このパターンが　RAMCGにDEFCHR$で定義されたパターンです。
DEFCHR$（65）の65はASC11コードを示しています。ASC11コード65は
ROMCGの"A"にあたります。
RAMCGは8×8ドットのパターンで256個まで定義できます。
またRAMCGとROMCGとの表示を切り換えるにはCTRLキーを押しながらテンキーの/を押すことによっても可能です。

2.6 アンダーライン

テキスト画面で、行と行の間にアンダーラインを表示することができます。
アンダーラインは画面に表示する行が20行もしくは10行のときにのみ設定することができます。
つまり、アンダーラインを使用する場合、あらかじめ

```
WIDTH,20↵　　もしくは　WIDTH,10↵
```

としておきます。
これらのモードにすると、行と行の間にアンダーライン用のドットがあけて表示されます。
アンダーラインを表示する場合、KSENステートメントを使用します。

```
KSEN { 0 または 省略
     { 1
```

```
KSEN 1↵
```

としますと、このステートメントを実行されたあとに入力された文字の下にアンダーラインが表示されます。
さらにアンダーラインの色は　KSEN命令の第2パラメータで指定できます。

```
KSEN 1,色↵
```

アンダーラインを表示しないモードにもどすには

```
KSEN 0↵　または　KSEN↵
```

とします。

3 グラフィックと文字表示

LINE、SYMBOL、GET@、PUT@はグラフィックだけでなく文字表示にも使用できます。
たとえば

```
LINE(0,0)—(39,9),"A",B↵
```

とすると（0，0）と（39，9）を対角線とするBOXをテキスト画面上に描きます。
また、

```
SYMBOL(0,0),"ABCD",1,1,7,0,"#"↵
```

とすると、"ABCD"という文字を"#"で描きます。
これらのステートメントの応用として、属性のみの変更に使用できます。
たとえば、80文字のモードで、画面の上から3行目の文字にすべてアンダーラインを付ける場合

```
KSEN 1,7↵
```

LINE（0，2）—（79，2），""↵

とします。（このとき、WIDTHステートメントでアンダーライン表示可能な画面モードにしておきます）つまり、文字列のかわりに""（ヌルコード）を指定すると、その時点の属性で、LINEやSYMBOLを描くことができます。

次に、GET@はテキスト画面のデータを配列変数に取り込むことができます。

たとえば、画面（0，0）—（5，5）の範囲に表示されている文字をそのまま、（25，15）—（30，20）の位置に書く場合

DIM A（25）↵

GET@（0，0）—（5，5），A↵

とし、配列変数Aに文字を読み込みます。

そして、

PUT@（25，15）—（30，20），A↵

とします。

8章
グラフィックス

本機の特長の1つに、豊富なグラフィック処理機能があげられます。特に、CZ－851C／CZ－852Cでは、96Kバイトものグラフィック用メモリを標準装備していますので、複雑、多様な表現が可能です。本機のBASICでは、多彩なグラフィック処理コマンドによって、これらの機能をサポートしています。本章では、多くのプログラム例をまじえながら、これらコマンド群の使い方を説明します。本章で解説するコマンド・ステートメントを次に示します。

INIT、CLS、PSET、PRESET、LINE、CIRCLE、POLY、PAINT、PALET、POSITION、PATTERN、SYMBOL、GET@、PUT@、PRW、COLOR

（注）本章は、「6章、ディスプレイモード」の内容を土台にして書かれています。本文中で何かわからない点がありましたら、6章の記述を参照してください。
なお、40×20、80×20、40×10、80×10行の各モードではグラフィックを表示できません。

1 画面の初期化

まず最初に、画面を初期化するステートメントについて説明します。コマンド・ステートメントの中には、画面状態の設定も変えてしまうものもありますので、そのままでは思い通りのグラフィックを描けない場合があります。そのような時、次のステートメントを実行することによって、画面状態はもとの設定にもどります。

```
INIT
```

また、CTRL ＋ D とダイレクトに入力することによっても、INITステートメントと同様の結果が得られます。

これらの命令は、例えば次の症状が起こって困った時などに実行してください。

1	文字がすべて点滅文字になってもとにもどらないとき。
2	文字がすべて反転文字になってもとにもどらないとき。
3	文字の内容が、キーの表示内容と合わなかったり、わけのわからないパターンが出てきた時
4	文字が標準の大きさでなく、倍文字となるとき。
5	文字の色を白色にもどしたいとき。
6	SHIFT ＋ CLR HOME キーを押しても画面の文字が消えずに残っているとき。
7	40文字モードで、キー入力しても画面に何もあらわれないとき。

8	コンピュータの音が鳴りっぱなしで止まらないとき。
9	画面のグラフィックがCLS0の命令でも消えないとき。
10	キー入力した文字がグラフィックのうしろにかくれてしまうとき。
11	グラフィックで描いた図形が指定した大きさ、色と合わないとき。
12	ＬＩＳＴ，ＦＩＬＥＳコマンド等によって、グラフィック画面が表示されなくなったとき。

上記ＩＮＩＴステートメントは、画面の設定状態を初期化する命令ですので、画面に描かれている内容をクリアすることはできません。画面をクリアする場合は、次のステートメントを実行します。

ＣＬＳ〔ｎ〕注）　（ｎ＝０～４）

注）〔 〕で囲まれたパラメータは省略することができます。
ＣＬＳステートメントは、ｎの値によって処理が異なってきます。
ＣＬＳ０……グラフィック画面Ｇ１、Ｇ２、Ｇ３注）を同時にクリアします。
ＣＬＳ１……グラフィック画面Ｇ１のみをクリアします。
ＣＬＳ２……グラフィック画面Ｇ２のみをクリアします。
ＣＬＳ３……グラフィック画面Ｇ３のみをクリアします。
ＣＬＳ４……グラフィック画面Ｇ１、Ｇ２、Ｇ３およびテキスト画面を同時にクリアします。
ＣＬＳ……テキスト画面のみをクリアします。（SHIFT＋CLR HOMEと同じ処理をします）。
注）グラフィック画面Ｇ１、Ｇ２、Ｇ３については、「６章　２．グラフィック画面」の項を参照してください。

上記ＩＮＩＴ、およびＣＬＳステートメントによって、画面状態の初期化とクリアが行なわれます。したがって、これからいろいろなコマンド、ステートメントを実行していく中で、画面状態を元に戻したい場合は、上記ステートメントを実行してください。

2　グラフィックステートメントの座標指定

グラフィック画面の座標系には、「６章ディスプレイモード」の「８．３ユーザー座標系と画面座標系」で説明したように画面座標系とユーザー座標系の２種類あります。画面座標系は、画面上の１ドットが１座標に１対１対応した絶対的な座標系ですので、不変ですが、ユーザー座標系はＷＩＮＤＯＷステートメントの設定値によって変化します。これからこの章で述べていく各種グラフィックステートメントのグラフィック座標は、以上２つの座標系のどちらかに分類されますが、その分類の様子は次のようになります。

座標系	画面座標系	ユーザ座標系
グラフィック ステートメント	ＰＯＳＩＴＩＯＮ，ＳＹＭＢＯＬ， ＧＥＴ@，ＰＵＴ@	ＰＳＥＴ，ＰＲＥＳＥＴ，ＬＩＮＥ，ＣＩＲＣＬＥ，ＰＯＬＹ，ＰＡＩＮＴ

このようにグラフィックステートメントによって座標系が異なりますので、ＷＩＮＤＯＷステートメントを使用した場合は、まず、そのステートメントがどちらの座標系に所属しているかを調べてから座標指定を行なうようにしてください。ただし、ＢＡＳＩＣ起動時には、ユーザー座標系は、画面座標系とまったく同じものですので、ＷＩＮＤＯＷステートメントでユーザー座標系を変えない限り、

両者の区別はありません。座標系の初期化、すなわち、ユーザー座標系を画面座標系と同一にするには、ＩＮＩＴステートメントまたはCTRL＋Dを実行します。

（注）ＷＩＮＤＯＷステートメントの使用方法については「6章　8.3 ユーザー座標系と画面座標系」の項を参照してください。

3 グラフィックステートメントの色指定

カラーモードにおいてグラフィック画面は、１ドット単位に８色の色指定が可能です。これらの色指定には、カラーコードとパレットコードの２種類が用いられます。カラーコードは、色とコードが１対１に対応した絶対的なコードなので不変ですが、パレットコードは、色とコードの対応関係が、ＰＡＬＥＴステートメントによって変化します。したがって、これからこの章で述べていく各種グラフィックステートメントの色指定がどちらのコードで行なわれているのか、あらかじめ知っておく必要があります。以下に、その区別の様子を示します。

色指定	カラーコード	パレットコード
グラフィック ステートメント	COLOR，CANVAS，KSEN	PSET，PRESET，LINE，CIRCLE，POLY，PAINT，SYMBOL，PRW

ただし、ＢＡＳＩＣ起動時には、パレットコードはカラーコードとまったく同じ色になっていますので、ＰＡＬＥＴステートメントでパレットコードの色指定を変えない限り、両者の区別はありません。パレットコードを初期化するためには、ＩＮＩＴステートメントまたはＰＡＬＥＴステートメントまたはCTRL＋Dを実行します。

また、グラフィックステートメントにおいて、色指定（パレットコード）を省略した場合は、自動的に、ＣＯＬＯＲステートメントの第１パラメータで指定した値（カラーコード）を、そのグラフィックステートメントのパレットコードとみなして実行します。

ＣＯＬＯＲステートメントの設定は、次の通りです。

```
COLOR 〔c1〕〔,c2〕
```

ｃ１：テキスト文字の表示色（カラーコード）

ｃ２：画面の背景色（カラーコード）

これより、グラフィックステートメント中の色指定を省略すると、パレットコードがカラーコードと等しい場合テキスト画面の文字の色と同じ色でグラフィックが描かれることになります。

例えば、ＣＯＬＯＲステートメントの第１パラメータをカラーコード１に設定した場合、パレットコードが初期状態ならば、パレットコードの指定を省略したグラフィックステートメントは青色として実行されます。また、ＰＡＬＥＴステートメントによって、パレットコードの１がカラーコードの４に設定されていれば、パレットコードの指定を省略した、グラフィックステートメントは緑色として実行されます。なお、ＣＯＬＯＲステートメントは、ＢＡＳＩＣ起動時には、次のように設定されています。

```
COLOR 7,0
```

また、ＣＯＬＯＲステートメントのかわりに、以下のようなダイレクト実行によっても同様な結果が得られます。

CTRL＋テンキー0　（＝COLOR 0）
CTRL＋テンキー1　（＝COLOR 1）
：　：
CTRL＋テンキー7　（＝COLOR 7）

（注）ＰＡＬＥＴステートメントの設定方法については、後述「９．色を瞬時に変える」の項を参照してください。
（注）本章では、特に断わらない限り、カラーモードにおけるグラフィック表示について延べます。（カラーモードについては、「6章　６．カラーモード」の項を参照してください。）

4 点を描く

ＰＳＥＴ，ＰＲＥＳＥＴステートメント

では、いよいよ具体的にグラフィック画面に対してアクセスしていきます。まず最初は、最も単純な点の描画です。グラフィック画面上の任意のドットをセットするには、ＰＳＥＴステートメントを使います。

ＰＳＥＴ（ｘ，ｙ〔，ｃ〕）

ｘ：横の座標（ユーザー座標系）
ｙ：縦の座標（　〃　）
ｃ：表示する色（パレットコード）

例えば、次のように入力すると、グラフィック画面上の座標（50、100 ）に緑色の点が表示されます。

```
PSET (50,100, 4)
```

点の色（パレットコード）の指定を省略すると、テキスト画面の文字の色に対応したパレットコードで表示されます。

```
PSET (50,100)
```

〔サンプルプログラム〕

座標（200,100 ）、（300,140 ）を対角とする白色の長方形を、ＰＳＥＴステートメントを使って描きます。

```
10 'PSET
20 WIDTH,25,0:INIT
30 FOR Y=100 TO 140
40 FOR X=200 TO 320
50 PSET(X,Y,7)
60 NEXT X
70 NEXT Y
```

ＰＳＥＴステートメントが任意の座標のドットをセットして点を表示するのに対して、ＰＲＥＳＥＴステートメントはドットをリセットして点を消す働きをします。

ＰＲＥＳＥＴ（ｘ，ｙ〔，ｃ〕）

ｘ：横の座標（ユーザー座標系）
ｙ：縦の座標（　〃　）
ｃ：消去する色（パレットコード）

正確に言うならば、ＰＲＥＳＥＴステートメントは点を消すのではなくて、指定した色を消す命令です。例えば、白色の点に対して、ＰＲＥＳＥＴステートメントの第３パラメータで緑色を消すようにすると、その点の色はマゼンタに変わります。これは、青、赤、緑の３色から成る白色から緑色を消した場合、残るのは青、赤から成るマゼンタだからです。

〔サンプルプログラム〕

サンプルプログラム１で作った白色の長方形の中で、座標（２３０、１１０）、（２７０、１３

0）を対角とする長方形の部分の緑色を消します。サンプルプログラム1の続きとして実行してください。

```
80 'PRESET
90 FOR Y=110 TO 130
100 FOR X=230 TO 270
110 PRESET (X,Y,4)
120 NEXT X
130 NEXT Y
```

〔サンプルプログラム〕

夜空に次々と花火が打ち上げられます。

```
10 '花火
20 WIDTH 40,25,0:CLS4:INIT
30 DEFFNX(CX)=COS(RAD(CX))*CR
40 DEFFNY(CY)=SIN(RAD(CY))*CR
50  FOR KAISU=1 TO 6
60   CCX=INT(RND*300)+10
70   CCY=INT(RND*150)+10
80   FOR UP=200 TO CCY STEP -1
90    C=INT(RND*7)+1
100   PSET(CCX,UP,C):PSET(CCX+1,UP,C)
110   PRESET(CCX,UP,C):PRESET(CCX+1,UP,C)
120  NEXT UP
130  H=INT(RND*10)+6
140  SOUND 7,&HF7:SOUND 6,43:SOUND 8,16:S
OUND 11,0:SOUND 12,150:SOUND 13,0
150  FOR CR=1 TO H*7 STEP H
160   FOR DEG=0 TO 360 STEP H
170    C=INT(RND*7)+1
180    PSET(FNX(DEG)+CCX,FNY(DEG)+CCY,C)
190   NEXT DEG
200  NEXT CR
210 NEXT KAISU
220 END
```

5 線を描く

LINEステートメント

線を描くには、LINEステートメントを使います。

LINEステートメントは、ただ単に実線を引くだけではなく、パラメータの設定によって次の動作を行なうことができます。

①ラインスタイルを設定することによって、点線、破線、一点鎖線等自由な形のラインを引くことができます。

②任意のラインスタイルで、長方形を描くことができます。

③長方形を描いて、その中を任意のパレットコード、中間色コードまたはタイリングパターンで塗りつぶすことができます。

④座標ばかりを連続入力することによって、任意の折れ線や多角形を描くことができます。

5.1 線を引く

線を引く場合、ＬＩＮＥステートメントのパラメータは次のようになります。

```
LINE〔(x1,y1)〕-(x2,y2)〔,mode,c,ls〕
```

ｘ１，ｙ１：始点座標（ユーザー座標系）
ｘ２，ｙ２：終点座標（ユーザー座標系）
ｍｏｄｅ：ＰＳＥＴ or ＰＲＥＳＥＴ or ＸＯＲ
ｃ：パレットコード　　ｌｓ：ラインスタイル

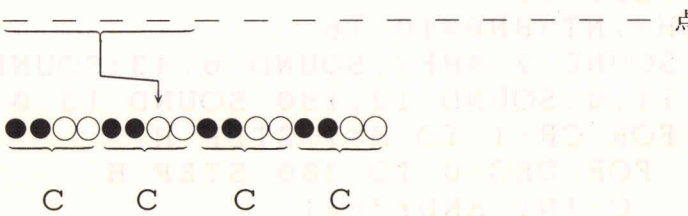
ただ単に実線を引くだけならば、例えば次の命令を実行するだけでＯＫです。

```
LINE（０，０）－（３１９，１９９）
```

これで、2点（０，０）、（３１９，１９９）を結ぶ実線が引かれます。線の色は、パレットコードを省略しているため、テキスト画面の文字の色に対応したパレットコードが与えられます。（初期状態では白色です）また、始点座標が省略されると、直前に実行したＬＩＮＥステートメントの終点座標が新たに始点座標としてみなされます。

第3パラメータ（ｍｏｄｅ）には、ＰＳＥＴ，ＰＲＥＳＥＴ，ＸＯＲの３通りの指定が可能です。ＰＳＥＴ，ＰＲＥＳＥＴについては「４．点を描く」で説明した通りです。ＸＯＲを指定すると、他のグラフィック図形と重なった直線部分の色が反転します。例えば、すでに白（青＋赤＋緑）でペイントされている所へ、XOR を使って青のラインを引いてやると、重なった部分のラインの色は黄（赤＋緑）になります。このパラメータが省略されると、直前に設定したモードの影響があります。

第5パラメータ（ｌｓ）には、ラインスタイルを指定します。ラインスタイルとは、点線、破線などを１６ビットのビットパターンで表わしたものです。例えば、次のような点線のラインスタイルは＆ＨＣＣＣＣで表わされます。



このように、ラインスタイルは、１６ドットの範囲内で任意のくり返しパターンを設定することができます。このパラメータを省略すると、実線（ラインスタイル＝＆ＨＦＦＦＦ）で引かれます。以下に主なラインスタイルとその破線のパターンを示します。

ラインスタイル	破線のパターン
$HFFFF	―――――――― 実線
$HFF00	― ― ― ― ―
$HCCCC	－－－－－－－－－－ 点線
$HAAAA	･･･････････････
$HFFCC	――－――－――－－ 一点鎖線
$HE4E4	―－－―－－―－－―
$HFCCC	――－－――－－―― 二点鎖線
$HFF24	――･･――･･――･･―

では、実際に直線を引いてみましょう。まず、画面を初期化して、グラフィック画面をクリアします。

```
INIT
CLS0
```

次に、2点（0，0）、（190，190）を結ぶ1点鎖線を黄色で引きます。

```
LINE（0，0）－（190，190），PSET，6，&HFFCC
```

どうですか？ちゃんと引かれましたか？もし、思いどおりの直線がひかれなければ CTRL ＋ D を押してみてください。さて、正しく表示されたら、今度は同じ座標間をXORモードを使って黄色の実線で結びます。

```
LINE（0，0）－（190，190），XOR，6
```

どうなりましたか？今まで1点鎖線が表示されていたところが透明になり、今まで透明だった部分に黄色が表示されましたね。このように、XORを指定すると、重なった部分のXOR論理がとられ、色が反転してしまいます。

〔サンプルプログラム〕

各種ラインスタイルを実際に表示させてみます。

```
10 'LINE STYLE
20 WIDTH 40,25,0:CLS0:INIT
25 PRINT "ラインスタイル";SPACE$(10);"パターン"
30 RESTORE 100
40 FOR I=2 TO 9
50 LOCATE 0,I*2
60 READ A$
70 PRINT A$
80 LINE(60,I*16+4)-(319,I*16+4),PSET,7,VA
L(A$)
90 NEXT
100 DATA &HFFFF,&HFF00,&HCCCC,&HAAAA,&HFF
CC,&HE4E4,&HFCCC,&HFF24
```

5.2 長方形を描く

長方形を描く場合、LINEステートメントのパラメータは次のようになります。

LINE〔(x1,y1)〕-(x2,y2),mode〔,c〕,B〔,ls〕

B：長方形を描く指定

指定した2点間の対角線を引くか、長方形を描くかは、記号"B"（BOXの略）があるかないかによって決ります。他のパラメータは、線を引く場合とまったく同じです。

では、実際に長方形を描いてみましょう。まず、画面の初期化とクリアを行ないます。

```
INIT
CLS0
```

次に、2点（0，0）（190，190）を対角とする長方形をシアンで描きます。

```
LINE（0，0）－（190，190），PSET，5，B
```

さらに、その上から、赤の点線でなぞってみます。

```
LINE（0，0）－（190，190），PSET，2，B，&HCCCC
```

これで、赤とシアンの点線で長方形が描かれました。

5.3 長方形を塗りつぶす

長方形を塗りつぶす場合、LINEステートメントのパラメータは次のようになります。

```
LINE〔(x1,y1)〕-(x2,y2),mode〔,c〕,BF
```

c：パレットコードまたは中間色コード

BF：長方形を塗つぶす指定

長方形を描いてその中を塗つぶすには、記号"BF"（Box Fillの略）を指定します。ただし、この場合、指定できる色はパレットコードだけでなく、中間色コードも指定できます。

例えば、2点（0，0）、（190，190）を対角とする長方形をシアンと赤の中間色コードで塗りつぶす場合は、次のステートメントを実行します。

```
LINE(0,0)-(190,190),PSET,&H25,BF
```

ただし、BFを指定する場合は、ラインスタイルを指定することができません。そのかわり、タイリングパターンを設定することができます。タイリングパターンというのは、より複雑な中間色や色模様を出すための文字列データで、最大8×8ドットの色パターンを定義することができます。最小のタイリングパターンは横8ドットの色パターンで、これを指定するには青、赤、緑各1バイトずつの計3バイトの文字列データが必要です。したがって、8×8ドットのタイリングパターンを定義する場合は、24バイトの文字列データが必要です。このように、中間色が2色のドットを交互に配置するだけなのに対して、タイリングパターンは、横8ドットの色パターンを自由に設定することができます。タイリングパターンを設定すると、その色パターンを使って長方形を描き、かつ塗りつぶします。タイリングパターンを使用する時、LINEステートメントは次のようになります。

```
LINE〔(x1,y1)〕-(x2,y2),mode,BF,t$
```

t$：タイリングパターン（文字列）

タイリングパターンの文字列は、CHR$やHEXCHR$関数によって用意するのが一般的です。

例えば、横8ドットのパターンを、1ドット1色ずつにして8色をすべて使って定義する場合は次のようにします。

横8ドットのパターン	黒	青	赤	マゼンタ	緑	シアン	黄	白	
	↓	↓	↓	↓	↓	↓	↓	↓	
青データ	0	1	0	1	0	1	0	1	=&H55
赤データ	0	0	1	1	0	0	1	1	=&H33
緑データ	0	0	0	0	1	1	1	1	=&H0F

∴t$=HEXCHR$("55330F")

そのタイリングパターンで、2点（0，0）、（190，190）を対角とする長方形を塗りつぶすには、次のステートメントを実行します。

```
LINE(0,0)-(190,190),PSET,BF,HEXCHR$("55330F")
```

5.4 折れ線を描く

LINEステートメントは、グラフ等に必要な折れ線を簡単に描くことができます。その場合のパラメータは次のようになります。

```
LINE(x1,y1)-(x2,y2)-(x3,y3)……-(xn,yn)
```

x1，y1：折れ線の始点座標

xi，yi：折れ線の頂点座標（i＝2，3……n－1）

xn，yn：折れ線の終点座標

このように、座標を連続して入力すると、各座標間を結んだ折れ線を描きます。これを利用すれば、任意の形の多角形を簡単に描くこともできます。例えば、4点（200，0）、（100，100）、

（０，１００），（１００，０）を頂点とする平行四辺形を描くには、次のように入力します。

```
LINE（２００，０）－（１００，１００）－（０，１００）－（１００，０）－（２００，０）
```

6 正多角形を描く

POLYステートメント

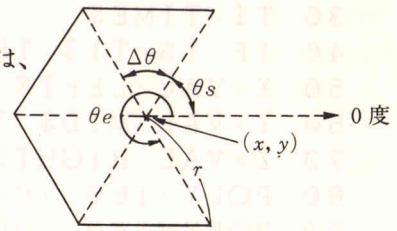
正多角形に限り、POLYステートメントを使って簡単に描くことができます。

POLY（x，y），r〔，c，Δθ，θs，θe〕

x，y：正多角形の中心座標（ユーザー座標系）
r：中心から頂点までの距離
c：正多角形の辺の色（パレットコード）
Δθ：ステップ角（０～３６０度）
θs：初期角（０～３６０度）
θe：終了角（０～３６０度）

多角形は、前述のＬＩＮＥステートメントでも描くことができますが、ＬＩＮＥの場合は各頂点の座標を計算して入力してやらなければなりません。しかし、POLYステートメントでは、円を描くのと同じ要領で入力することができますので、正多角形の中心（x，y）と半径r等の設定だけで簡単に描くことができます。（円については「７．円を描く」で述べます）。各パラメータについて以下に示します。

ステップ角（Δθ）は、正多角形の頂点間の角度で、０度に近いほど真円に近づきます（ただし、０度を指定するとただの線分になってしまいます）。初期角（θs）と終了角（θe）は、x座標の増分方向（画面の右方向）を０度とした場合の、多角形を描き始める角度と、描き終える角度を示します。



したがって、一般的に、正n角形を描くには、次のように指定します。

Δθ＝３６０／n
θe＝θs＋３６０

では、実際に、正多角形を描いてみます。まず、画面を初期化してクリアしてください。

```
INIT
CLS０
```

次に、正六角形を描いてみます。正六角形のステップ角は６０度（＝３６０÷６）です。

```
POLY（２００，１００），８０，２，６０
```

これで、中心座標（２００，１００）、頂点距離８０の赤い正六角形が描かれました。

今度は、右側半分だけの半円を描いてみます。右側だけの半円ですから、初期角は９０度、終了角は２７０度で指定します。

```
POLY（２００，１００），８０，６，１，９０，２７０
```

これで黄色の半円が描かれました。

（注）POLYおよびＣＩＲＣＬＥステーメメントにおける頂点距離または半径は、画面モードによって、実際の座標と対応していない場合があります。詳しくは、ＢＡＳＩＣリファレンスマニュアルのＣＩＲＣＬＥステートメントの項を参照してください。

〔サンプルプログラム〕

POLYで多角形を重ねます。

```
10 INIT:CLS4:WIDTH 40,25,0,0
20 RESTORE 160
30 FOR KAI=1 TO 5
40 READ STE1,STA1,OWA1,STE2,STA2,OWA2
50 C=INT(RND*6)+1
60 FOR X=1 TO 14
70 FOR Y=1 TO 9
80 POLY(X*20,Y*20),20,C,STE1,STA1,OWA1
90 POLY(X*20+2,Y*20),20,C+1,STE2,STA2,OWA2
100 NEXT Y
110 NEXT X
120 PAUSE 20
130 CLS4
140 NEXT KAI
150 GOTO 20
160 DATA 120,0,360,120,90,450
170 DATA 90,0,360,90,45,405
180 DATA 72,0,360,72,90,450
190 DATA 60,0,360,60,30,390
200 DATA 144,0,720,144,90,810
```

〔サンプルプログラム〕

時計を描きます。

```
10 INIT:CLS4:WIDTH 40,25,0,0
20 POLY(160,100),80,7,30
30 T$=TIME$
40 IF T$=T1$ THEN 30
50 X=VAL(LEFT$(T$,2))
60 Y=VAL(MID$(T$,4,2))
70 Z=VAL(RIGHT$(T$,2))
80 POLY(160,100),40,0,0,90-X1*30-INT(Y1/2)
90 POLY(160,100),50,0,0,90-Y1*6
100 POLY(160,100),60,0,0,90-Z1*6
110 POLY(160,100),40,5,0,90-X*30-INT(Y/2)
120 POLY(160,100),50,4,0,90-Y*6
130 POLY(160,100),60,6,0,90-Z*6
140 LOCATE 16,16:PRINT T$
150 X1=X:Y1=Y:Z1=Z:T1$=T$
160 GOTO 30
```

7 円を描く

CIRCLEステートメント

円、楕円および円弧を描くステートメントは、CIRCLEステートメントです。

CIRCLE〔@〕(x,y),r〔,c,f,θs,θe〕

f：偏平率（＝たて径／横径）

このように、CIRCLEステートメントのパラメータは、POLYステートメントと似ています。

違いは、POLYステートメントのステップ角が、CIRCLEステートメントでは偏平率に変わっている点だけです。また、CIRCLEステートメントでは、@記号がつくかつかないかで、半径及び偏平率の扱いが異なってきます。

(1) @記号がつかない場合

画面に表示される円の大きさは、320×200ドットモードの時を規準にしています。したがって、半径が同じならば、どの画面モードで描いても、画面に表示される円の大きさは同じになります。このことは、画面上の座標と半径の大きさとが、必ずしも一致しないことを示しています。偏平率を指定した場合も同様です。偏平率を省略すると、1が設定されます。

(2) @記号がつく場合

画面に表示される円の大きさは、各画面モードの座標に従います。したがって、同じ半径でも画面モードによって円の大きさは異なってきます。偏平率を指定した場合も同様です。したがって、偏平率を1に設定したからといって、画面モードによっては真円にならない場合があります。偏平率を省略すると、画面上に真円が描かれるように、自動的に偏平率を設定します。

なお、詳しくは、BASICリファレンスマニュアルのCIRCLEステートメント及びCIRCLE@ステートメントの項を参照してください。

では、実際に円を描いてみます。中心座標（200，100）、半径80の半円を描くためには、次のステートメントを実行します。

```
CIRCLE(200,100),80↵
```

〔サンプルプログラム〕

専用ディスプレイテレビにおいて、CIRCLEステートメントとCIRCLE@ステートメントにおける半径の取扱いの違いを示すプログラムです。

```
10 'CIRCLE
20 INIT:CLS4:OPTION SCREEN0
30 G=0:RES=1:GOSUB 70
40 G=0:RES=2:GOSUB 70
50 G=1:RES=2:GOSUB 70
60 END
70 'SUB
80 FOR X=40 TO 80 STEP 40
90 WIDTH X,25,G,RES
100 PRINT "WIDTH";X;",25,";G;",";RES
110 PRINT "   RED.....CIRCLE (200,100),100
,2"
120 PRINT "   GREEN...CIRCLE@(200,100),100
,4"
130 CIRCLE(200,100),100,2
140 CIRCLE@(200,100),100,4
150 A$=INKEY$:IF A$=""THEN 150
160 NEXT
170 RETURN
```

8 色を塗る

PAINTステートメント

グラフィック曲線で囲まれた任意の形の図形をペイントするには、PAINTステートメントを使います。

```
PAINT〔@〕（x，y），{ c / t$ }，b〔，b1，b2，……，b6〕
```

c　：パレットコードまたは中間色コード
t$：タイリングパターン
b，b1，b2……b6：境界色（パレットコード）

ペイントする色としては、パレットコード、中間色コード、またはタイリングパターンを指定することができます。（タイリングパターンについては「5．線を描く」の項を参照してください。境界色は、カンマで区切って7個まで指定できます。また、塗る色自身は常に境界色とみなされます。PAINTステートメントは、塗る範囲が1ドットでも開いていると、そこからはみ出して全体を塗ってしまうので、注意してください。

では、実際に色を塗ってみます。まず、境界をつくるために、次の命令を実行します。

```
CIRCLE（200，100），80，2
```

次に、境界色を赤に設定して、円内を青色で塗りつぶします。

```
PAINT（200，100），1，2
```

これによって、座標（200，100）から青色で塗り始められ、赤色で囲まれた部分をすべて塗りつぶします。

9 色を瞬時に変える

PALETステートメント

PALETステートメントは、グラフィック画面上の色を瞬時に他の色に変換します。

```
PALET p，c，b
```

p：パレットコード（0～7）
c：カラーコード（0～8）
b：ボーダーカラー（0or1）

PALETステートメントは、色の変換を行ないますが、実際に塗りつぶして変換するわけではありません。したがって、瞬時の色変換が可能です。変換は、パレットコードをどのカラーコードに対応させるか、で行ないます。パレットコードおよびカラーコードについては「6章　ディスプレイモード」の項を参照してください。例えば、パレットコード1は、初期状態ではカラーコード1すなわち青色に対応していますが、赤色（カラーコード2）に変換するには、次の命令を実行します。

```
PALET1，2
```

また、PALETステートメントによりグラフィックの透明と青の2色を、黒色に変換することができます。黒色は、カラーコード8に割り当てられており、PALET0，8orPALET1，8のどちらかを指定します。前者が透明を黒に、後者が青を黒に変換します。ただし、黒色といっても、コンピュータモードでは透明とまったく変わりません。透明との違いがでるのはスーパーインポーズ時のみです。透明はバックのテレビ画面を透き通しますが、黒色は透き通しません。

また、第3パラメータが1の時、画面のボーダー部分を黒色にします。0の時は透明のままです。これによって、スーパーインポーズ時にテレビ画面を黒ぬきにすることができます。

また、PALETステートメントは次のように設定することもできます。

```
PALET@c1，c2，c3，c4，c5，c6，c7，c8
```

c1，c2，……c8：カラーコード

この指定方法は、複数のパレットコードを一度に変換する場合に用いられます。パレットコードは、パラメータの位置に対応しています。すなわち、第1パラメータがパレットコード0に、第2パラメータがパレットコードの1に、第8パラメータがパレットコード7にそれぞれ対応しています。例えば、パレットコード1を赤に、パレットコード7を緑に変換するためには、次の命令を実行します。

```
PALET@0,2,2,3,4,5,6,4
```

〔サンプルプログラム〕

花火の点滅する感じをお楽しみください。

```
10 '花火
20 WIDTH 40,25,0:CLS4:INIT
30 SCREEN1,1:KLIST0:PRINT "Wait a moment!"
40 DEFFNX(CX)=COS(RAD(CX))*CR
50 DEFFNY(CY)=SIN(RAD(CY))*CR
60 CCX=INT(RND*300)+10
70 CCY=INT(RND*150)+10
80 H=INT(RND*10)+6
90 SCREEN1,0:KLIST0
100 FOR CR=1 TO H*7 STEP H
110 FOR DEG=0 TO 360 STEP H
120 C=INT(RND*7)+1
130 PSET(FNX(DEG)+CCX,FNY(DEG)+CCY,C)
140 NEXT DEG
150 NEXT CR
160 SCREEN1,1:CLS4
170 SOUND7,&HFE:PLAY"—E1"
180 FOR UP=200 TO CCY STEP -1
190 C=INT(RND*7)+1
200 PSET(CCX,UP,C):PSET(CCX+1,UP,C)
210 PRESET(CCX,UP,C):PRESET(CCX+1,UP,C)
220 NEXT UP
230 SCREEN0,0
240 SOUND 7,&HF7:SOUND 6,43:SOUND 8,16:SOU
ND 11,0:SOUND 12,150:SOUND 13,0
250 FOR I=1 TO 15
260 FOR J=1 TO 7
270 PALET J,((J+1)MOD 7)+1
280 NEXTJ
290 NEXTI
300 FOR I=1 TO 7
310 PALET I,0
320 PAUSE 1
330 NEXT I
```

10 テキスト文字とグラフィック画面の優先順位を決める

PRWステートメント

PRWステートメントは、テキスト画面上の文字とグラフィック画面上の図形が重なった時、どちらを優先して表示させるかを決めるステートメントです。

PRW〔n〕

n＝0～255

グラフィック画面の各色とテキスト文字との優先順位は、n の値で決定されます。n の値はコンピュータ内部では8ビットで表現されており、各ビットは次のように各パレットコードに対応しています。

	7	6	5	4	3	2	1	0	
n:	白	黄	シアン	緑	マゼンタ	赤	青	透明	(初期状態のパレットコード)

最下位ビットが透明に対応し、最上位ビットが白に対応しています。この時、あるビットの内容が1に設定されると、その色をもつグラフィックが優先され、0に設定されるとテキスト文字が優先されて表示されます。例えば、グラフィックの青と緑をテキストより優先させて表示させるには、次のように設定します。

PRW &B00010010↵ (＝PRW 18)

PRWステートメントは、パレットコードに対する命令ですので、例えば、PALET1，2を実行してパレットコード1を赤に変換すると、その部分の赤がテキストより優先されます。ただし、パレットコード2の赤はテキストに隠れたままです。このように、，PRWステートメントを利用すると、見かけ上同じ色でもテキストとの優先順位を変えることができます。

優先順位を初期状態にもどすには、PRW↵を実行します。

11 文字パターンの描画

SYMBOLステートメント

SYMBOLステートメントは、グラフィック画面上に文字列を指定の角度とサイズで描きます。

SYMBOL (x，y)，x$，h，v，{c' / t$}，θ，mode

x，y： 文字列の表示を開始する左上のドットの座標（画面座標系）
x$ ： 表示する文字列
h ： 横方向の倍率（1、2、3……）
v ： 縦方向の倍率（1、2、3……）
c' ： 文字の色（パレットコードor中間色コード）
θ ： 描く方向の指定（0～3）
mode ： PSET or PRESET or XOR or文字式or" "
t$ ： タイリングパターン

h，vは、それぞれ横、縦方向の倍率を指定します。設定値はすべて整数とみなされます。小数は四捨五入されます。この値によっては文字が画面から、はみ出してしまうので注意してください。

t$はタイリングパターンで、HEXCHR$などで指定します。タイリングパターンについては「5.3 長方形を塗りつぶす」の項を参照してください。

θは、表示する文字の方向を示しています。

θ＝0では、通常表示、θ＝1では、90度左に回転したもの
θ＝2では、180度左へ回転したもの
θ＝3では、270度左へ回転したもの
となります。

modeは、文字列の表示モードを示し、PSET，PRESET，XORを指定するとグラフィック画面に表示し、文字式または" "＝ヌルコードを指定するとテキスト画面に表示します。次に例を示します。

SYMBOL（100，100），"A"，1，1，1，0，PSET↵

これはグラフィック画面に、青色のAという文字をたて横1倍で表示します。

SYMBOL（100，100），"B"，2，2，4，1，PSET↵

これは、グラフィック画面に緑色のBという文字をたて横2倍の大きさで表示します。その時Bは90度左に回転されて表示します。

SYMBOL（0，0），"A"，1，1，1，0，"B"

これは、テキスト画面にキャラクタBでAという文字を表示します。

12 GET@とPUT@

GET@はグラフィック画面の情報を配列変数に読み込むステーメントです。

また、PUT@というステートメントはGET@の逆で、配列変数の情報を画面に描くステートメントです。

通常、これらの2つのステートメントを組み合わせて用いることによって、グラフィック画面のデータを別の位置に移動させることができます。

これらのステートメントは、画面データを一度配列変数に入れるので、データが入るだけの配列変数領域をあらかじめ定義しておく必要があります。

たとえば、次の例では、画面左上に描いた円を赤でぬり、それと同じ円を別の場所に描いています。

```
10 INIT :CLS4
20 DIM A%(100)
30 CIRCLE(10,10),5:PAINT (10,10),2,7
40 GET@(0,0)-(20,20),A%,7
50 FOR I=0 TO 100 STEP 20
60 PUT@(I,I)-(I+20,I+20),A%,PSET,7
70 NEXT
```

GET@およびPUT@の書式は

```
GET@（x1，y1）-（x2，y2），配列名，パレットコード
PUT@（x1，y1）-（x2，y2），配列名｛，PSET，パレットコード
                              ，PRESET
                              ，XOR
                              ，OR
                              ，AND
                              ，NOT｝
```

x1、y1：読みこむ領域の左上隅の座標（画面座標系）

x2、y2：読みこむ領域の右下隅の座標（画面座標系）

PUT@ステートメントでは表示のときにPSET，PRESET，XOR，OR，AND，NOTとモードがあり、画面にすでにあるドットと配列中のドットの演算結果を画面に表示することもできます。また、パレットコードを省略するとテキスト画面に対するステートメントになります。

13 グラフィックパターンの描画

POSITION・PATTARNステートメント

まず、グラフィック画面にドットパターンを描く際には描画開始位置を指定します。
それにはPOSITIONステートメントを用意します。

```
POSITIONx,y
```

x，y：ドット・パターン表示の左上隅の座標（画面座標系）

POSITIONステートメントの実行により描画開始位置を指定した後、PATTERNステートメントでドットパターンを描画します。

```
PATTERN n,x$〔,y$……〕
```

n：ドットパターンのたて方向の段数を指定し、n＜0のとき下の方向に重なりn＞0のとき上の方向に重なります。

x$，y$：ドットパターンを表わす文字式

例えば

```
POSITION 10,10
PATTERN-1,CHR$(&H55)
```

と入力すると、x座標10、y 座標10を描画開始位置として、ドットパターン

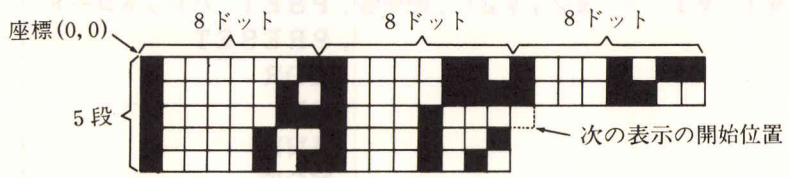
（■がセット。&H55＝&B01010101）が表示されます。

また、ドットパターンの積み重ねは段数を指定するnの値により、積み重ねの方向と段数が決められます。nが負のとき上から下へ、nが正のとき下から上へ積み重ねられます。
表示色は、テキスト画面の文字の色に対応したパレットコードで描れます。

例えば

```
100 A$=HEXCHR$("8182838485868788898A8B8C")
110 POSITION 0,0
120 PATTERN-5,A$
```

を実行すると下の様に表示されます。



9章
日本語処理

本機には、強力な日本語処理機能を備えた漢字ＢＡＳＩＣが標準装備されています。この漢字ＢＡＳＩＣは、漢字、ひらがな等の文字をＢＡＳＩＣプログラム上で容易に扱えるように、カナ—漢字変換機能や外字登録機能を持っています。この章では、これら全角文字と呼ばれる文字の扱いについて、説明します。

1 全角文字とは

全角文字とは、漢字やひらがなのように、そのフォントが16×16ドットで構成された文字のことを指します。それに対して、そのフォントが８×８or16×８ドットで構成された文字を、半角文字と呼んでいます。

従来のコンピュータでは、通常、英数字・カタカナ・記号・セミグラフィック等の半角文字のみ扱っており、漢字・ひらがな等の全角文字はプログラム中ではコード（ＪＩＳ漢字コード、区点コード等）として扱うことしかできませんでした。（例えば、"亜"という漢字はＫＡＮＪＩ＄（１６０１）で表わします）しかし、本機の漢字ＢＡＳＩＣでは、これらの全角文字をプログラム中で半角文字と同様に文字の形で扱うことができます。それは、次のような場合において可能です。

(1) ＰＲＩＮＴ文等のダブルクォテーション（"）で囲まれた部分
(2) ファイル名及びディレクトリ名
(3) ＲＥＭ文
(4) ＤＡＴＡ文　(5) 文字変数

このように、本機の漢字ＢＡＳＩＣでは、プログラム中に漢字やひらがなを混入させることができますが、ＢＡＳＩＣ内部ではこれらの全角文字は２バイトのコードとして処理されています。この意味から、従来の文字（半角文字）が１バイトコード文字と呼ばれているのに対して、全角文字は２バイトコード文字とも呼ばれています。

したがって全角文字の１文字分は半角文字の２文字分とみなされますので、例えば、ファイル名の長さ等、文字数に注意する必要があります。また、本機の漢字ＢＡＳＩＣでは、全角文字を扱う方式として、従来の半角文字のセミグラフィック部分の１バイトコードを利用するＳＨＩＦＴ　ＪＩＳコード体系をとっています。したがって、全角文字を扱う場合はセミグラフィック文字を混在させることができません。（ただし、英数字、カタカナ、記号等は使用できます）全角文字を表示させるか否かは、次のＢＡＳＩＣコマンドによって決められます。

ＫＭＯＤＥ　０……半角文字のみ扱うモード（セミグラフィック文字使用可）
ＫＭＯＤＥ　１……全角文字も扱うことができるモード（セミグラフィック文字使用不可）

また、全角文字の場合その文字のフォントがたて16ドットで構成されていますので、表示画面が標準ディスプレイモードの時には、全角文字は次の画面モードの時のみ表示することができます。

```
ＷＩＤＴＨ　８０，１２，０，｛１
                              ｛０
ＷＩＤＴＨ　４０，１２，０　｛１
                              ｛０
ＷＩＤＴＨ　８０，１０，０，｛１
                              ｛０
ＷＩＤＴＨ　４０，１０，０　｛１
                              ｛０
```

2 全角文字の入力方式

ひらがな、カタカナ、英数字等の全角文字の入力方式には、次の３通りの方式があります。（また、これらの方式では半角文字の入力も可能です）

(1)アルファベット、数字等の入力を簡単に行なえる英数字直接入力方式

(2)ローマ字表記にて入力し、それをひらがなまたはカタカナに変換するローマ字—カナ変換方式

(3)カナキーをロックしてカナ表記にて入力し、それをひらがなまたはカタカナに変換するカナ入力方式また、漢字への変換機能として、次の２通りの方式があります。

(1)漢字のコード（ＪＩＳ漢字コード又は区点コード）を入力して、直接、漢字、ひらがな等の全角文字に変換するコード入力方式

(2)上記、ローマ字—カナ変換方式およびカナ入力方式において入力したひらがなまたはカタカナを、漢字に変換させるカナ—漢字変換方式

さらに、カナ—漢字変換方式には、次の２つの方式があります。

(1)ひらがなまたはカタカナの文字列の先頭の１文字のみを判断して、その読みをもつ漢字をＪＩＳ漢字コードの配列通りに羅列して変換する一字変換方式

(2)漢字の音読み、訓読みを入力し、変換する音訓変換方式

なお、一字変換方式はコンピュータの内部プログラムとして存在しますので、ディスク等は一切不要ですが、音訓変換方式は附属のシステムディスクが必要です。

これらの全角文字入力機能は、日本語入力モードにおいて利用することができます。日本語入力モードに入るためには、まず次の条件が満たされていなければなりません。

〔条件１〕

標準ディスプレイモードの場合、画面モードを次のいずれかに設定すること。（高ディスプレイモードの場合は、制限はありません）

○WIDTH 80,12,0,{1 / 0}
○WIDTH 40,12,0 {1 / 0}
○WIDTH 80,10,0,{1 / 0}
○WIDTH 40,10,0 {1 / 0}

〔条件２〕

CONSOLEステートメントによって、表示画面の最下行が変換フィールドエリアとして確保れていること。

〔例〕

○25行モードならば、 CONSOLE 0,24
○12行モードならば、 CONSOLE 0,11
○20行モードならば、 CONSOLE 0,19
○10行モードならば、 CONSOLE 0,9

ただし、CONSOLEステートメントだけでなく、ＩＮＩＴ命令やCTRL＋Dキー入力によっても、変換フィールドエリアは確保されます。

〔条件３〕

"KMODE 1"を実行してあること。

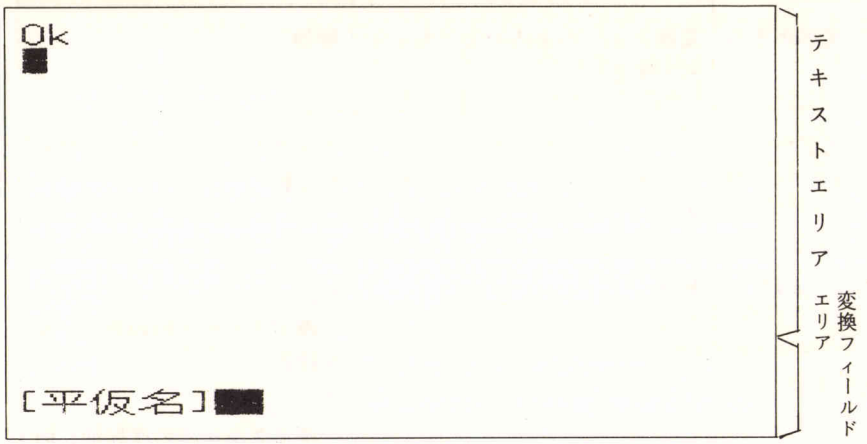
以上の条件が満たされていることを確認した後、日本語入力モードに入るには次の２つの方法がありますが、どちらを実行してもかまいません。

(a) CTRL キーを押しながら XFER キーを押す。

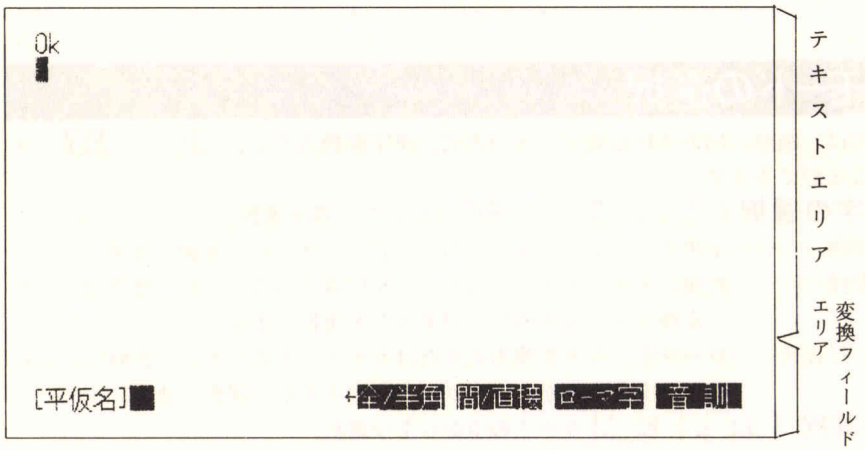
(b) SHIFT キーを押しながら XFER キーを押す。

日本語入力モードに入ると、表示画面の最下行に変換フィールドが現われます。（図９－１）

全角文字への変換は、すべてこの変換フィールド内で行なわれます。また、日本語入力モードから通常の状態に戻るには、再び前記(a)または(b)を実行します。



(a)40桁モードの場合の入力画面



(b)80桁モードの場合の入力画面

図９－１．日本語入力モードにおける表示画面

〔注〕日本語入力モードにおける表示画面は、40桁モードと80桁モードとでは若干異なりますが、動作に変わりはありません。

※日本語入力モード時の INS DEL キーとカーソルコントロールキーの動作

CTRL ＋ XFER キー又は SHIFT ＋ XFER キーによって日本語入力モードに入ると、INS DEL キーとカーソルコントロールキーの働きが次にように若干変わってきます。

(1)変換フィールド内に文字がある場合は、INS DEL キーは変換フィールド内の文字に対して働きます。（ただし、インサートはできません）また、カーソルコントロールキーは無視されます。

(2)変換フィールド内に文字がない場合、INS DEL キーは変換フィールド内のカーソルが左端にきた時点でテキストエリア内の文字に対して働きます。カーソルコントロールキーについても同様です。したがって、例えば変換フィールド内の文字をデリートする場合、フィールド内のカーソルが左端にきた時点でテキストエリア上のデリート動作に入りますので、特に注意する必要があります。

状　　態	INS DEL キー	カーソルコントロールキー
変換フィールド内に文字がある	変換フィールド内の文字をデリートする	無効
変換フィールド内に文字がなくて、フィールド内のカーソルが左端にある。	テキストエリア上の文字に対して、インサート／デリートを行なう	テキストエリア上のカーソルを移動
漢字グループを表示している	無効	← → 漢字グループ内のカーソルを移動 ↑ ↓ 漢字グループの次候補、前候補を選択

3 入力モードの選択

日本語入力モードでは、画面に出力される文字、入力方式、漢字変換方式を、 F1 ～ F10 キーによって選択することができます。

3.1 画面出力文字の選択（ F1 、 F2 、 F3 、について、各々選択）

F1 ：全角／半角……全角文字か半角文字のどちらを表示させるかを選択します。

F2 ：直接／間接……直接、テキストエリアのカーソル位置に出力するか、またはいったん変換フィールド内に出力するかを選択します。

F3 ：平仮名／片仮名……ローマ字－カナ変換方式またはカナ入力方式で入力した時に、ひらがなを出力するか、カタカナを出力するかを選択します。

3.2 入力方式の選択（ F4 、 F5 、 カナ の中から１つ選択）

F4 ：英数字直接入力方式……キーボード上のアルファベット、数字、記号の通りに入力します。この入力方式を選択すると、自動的に画面出力文字も英数字、記号となります。

F5 ：ローマ字―カナ変換方式……アルファベットを入力すると、自動的にローマ字とみなされ、ひらがなまたはカタカナに変換されます。

※ カナ キーロックによるカナ入力方式

入力方式が F4 の英数字又は F5 のローマ字―カナ変換のモードにある時、 カナ キーをロック（押し下げた状態）すれば、直接カナによる入力ができます。

3.3 漢字変換方式の選択（ F6 、 F7 、 F8 、 F9 、 F10 、の中から１つ選択）

F6 （ SHIFT ＋ F1 ）：コード入力方式……全角文字のコードを直接入力します。コードには、ＪＩＳ漢字コード（"16進"と表示）と区点コード（"区点"と表示）とがあり、このキーを押す毎に切換わります。

F7 （ SHIFT ＋ F2 ）：一字変換方式……漢字の読みを１文字だけ識別（他の文字は無視）して漢字に変換します。

F8 （ SHIFT ＋ F3 ）：音訓変換方式……漢字の音読み、訓読みを入力して、漢字に変換します。

F9 （ SHIFT ＋ F4 ）：システム辞書変換方式（ＳＹＳＴＥＭ辞書オプション）……ＳＹＳＴＥＭ辞書ディスクを使って、熟語やかなおくりなどの変換をします。

F10 （ SHIFT ＋ F5 ）：ユーザ辞書変換方式（ＵＳＥＲ辞書オプション）……ＵＳＥＲ辞書ディスクを使ってＳＹＳＴＥＭ辞書にない熟語を登録したり、オリジナル文字や記号

などを登録することができます。

※一字変換機能は、コンピュータの内部プログラムとして常駐しています。また、音訓変換方式、システム辞書変換方式、ユーザ辞書変換方式を利用するためには、それぞれ音訓辞書ディスク（システムディスク）、ＳＹＳＴＥＭ辞書ディスク、ＵＳＥＲ辞書ディスクが必要です。同時に、変換エリアをメモリー内部に確保するために、ＬＩＭＩＴコマンドまたはＣＬＥＡＲコマンドでエリアの確保を行なう必要があります。各方式におけるエリア領域は次の通りです。

○音訓辞書を利用する場合：ＬＩＭＩＴ&ＨＦ０００（またはＣＬＥＡＲ&ＨＦ０００）

○ＳＹＳＴＥＭ辞書を利用する場合：ＬＩＭＩＴ&ＨＦ０００（またはＣＬＥＡＲ&ＨＦ０００）

○ＵＳＥＲ辞書を利用する場合：ＬＩＭＩＴ&ＨＦ０００（またはＣＬＥＡＲ&ＨＦ０００）

以上の入力モードの選択は、 F1 ～ F10 キーのかわりにGRAPH キーを押しながら数字キー 1 ～ 0 を入力することによって行なうことができます。（ただし、数字キーとしてテンキーを用いることはできません）

〔例〕○全角／半角の切換え方法…… F1 又は GRAPH ＋ 1 を入力する。

○音訓変換方式への切換え方法…… F8 （ SHIFT ＋ F3 ）又は GRAPH ＋ 8 を入力する。

漢字ＢＡＳＩＣ起動時には、入力モードはそれぞれ以下のように初期設定されています。

(1) 画面出力文字の選択

○全角文字出力 Ｆ１

○間接出力 Ｆ２

○平仮名出力 Ｆ３

(2) 入力方式の選択

○ローマ字—カナ変換方式 Ｆ５

(3) 漢字変換方式の選択

○音訓変換方式 Ｆ８

また、初期状態では音訓辞書が使用できるように、ＬＩＭＩＴ&ＨＦ０００（またはＣＬＥＡＲ&ＨＦ０００）が実行されています。

※ HELP キーによる入力モードの確認

入力モードの確認は HELP キーによって行ないます。 HELP キーを押すと、 F1 ～ F10 キーの選択内容が最下行の変換フィールド部分にリストされます。この時、現在のモードは赤字で表示されます。なお、80桁モードの場合は全リストが表示されますが、40桁モードの場合は F1 ～ F5 の内容がリストされます。 F6 ～ F10 の内容を知りたい場合は、 SHIFT キーを押してください。また、これらのリストを表示したままの状態で F1 ～F10 キーを押せば、入力モードの変更ができます。日本語入力モードに戻るためには、再び HELP キーを押します。

4 ローマ字—カナ変換

カナとローマ字の対応関係を表９－１に示します。その他、以下のような変換も可能です。

(1) 小文字は、 SHIFT キーを押しながらアルファベットを入力することによって変換できます。

「ぁ」…… SHIFT ＋ A

「ぃ」…… SHIFT ＋ I

「ぅ」…… SHIFT ＋ U

「ぇ」…… SHIFT ＋ E

「ぉ」…… SHIFT ＋ O

「ゃ」…… SHIFT ＋ Y A

「ゅ」…… SHIFT ＋ Y U

「ょ」…… SHIFT ＋ Y O

「っ」…… SHIFT ＋ Z または SHIFT ＋ T U または SHIFT ＋ T S U または SHIFT ＋ ﾂｬﾔ

小文字の「っ」は、上記入力の他に、同じアルファベットを2度連続して入力することによって表示させることもできます。

〔例〕ローマ字入力「[I], [R], [A], [S], [S], [H], [A], [I]」→カナ変換出力「いらっしゃい」

(2) 「ん」は、次の様にも入力できます。

「ん」……[SHIFT]+[N] または [N], [SHIFT]+[ﾜﾔﾔ]

(3) 「きゃ」、「ちぇ」、「ぴゃ」などの文字は、大文字、小文字を別々に入力することもできます。

〔例〕「きゃ」……[K], [Y], [A] または [K], [I], [SHIFT]+[Y] [A]

大文字「き」入力　小文字「ゃ」入力

〔注〕上記の説明の中で、入力キーをカンマ（,）で区切った場合は、それらのキーを連続して入力することを示します。逆にプラス（+）で接続した場合は、それらのキーを同時に入力することを示します。

例 [N], [SHIFT]+[ﾜﾔﾔ]……[N] キーを入力した後連続して、[SHIFT] キーを押しながら [ﾜﾔﾔ] キーを入力することを示します。

表9－1　ローマ字－カナ変換一覧表

あ	い	う	え	お						
A	I	U	E	O						
か	き	く	け	こ	きゃ	きゅ		きょ		
KA	KI	KU	KE	KO	KYA	KYU		KYO		
さ	し	す	せ	そ	しゃ	しゅ	しぇ	しょ		
SA	SI	SU	SE	SO	SYA	SYU		SYO		
	SHI				SHA	SHU	SHE	SHO		
た	ち	つ	て	と	ちゃ	ちゅ	ちぇ	ちょ		
TA	TI	TU	TE	TO	TYA	TYU		TYO		
	CHI	TSU			CHA	CHU	CHE	CHO		
な	に	ぬ	ね	の	にゃ	にゅ		にょ		
NA	NI	NU	NE	NO	NYA	NYU		NYO		
は	ひ	ふ	へ	ほ	ひゃ	ひゅ		ひょ		
HA	HI	HU	HE	HO	HYA	HYU		HYO		
		FU								
ま	み	む	め	も	みゃ	みゅ		みょ		
MA	MI	MU	ME	MO	MYA	MYU		MYO		
や		ゆ		よ						
YA		YU		YO						
ら	り	る	れ	ろ	りゃ	りゅ		りょ		
RA	RI	RU	RE	RO	RYA	RYU		RYO		
わ		を		ん						
WA		WO		X						

が	ぎ	ぐ	げ	ご	ぎゃ	ぎゅ		ぎょ	
GA	GI	GU	GE	GO	GYA	GYU		GYO	
ざ	じ	ず	ぜ	ぞ	じゃ	じゅ	じぇ	じょ	
ZA	ZI	ZU	ZE	ZO	ZYA	ZYU		ZYO	
	JI				JA	JU	JE	JO	
だ	ぢ	づ	で	ど	ぢゃ	ぢゅ		ぢょ	
DA	DI	DU	DE	DO	DYA	DYU		DYO	
ば	び	ぶ	べ	ぼ	びゃ	びゅ		びょ	
BA	BI	BU	BE	BO	BYA	BYU		BYO	
ぱ	ぴ	ぷ	ぺ	ぽ	ぴゃ	ぴゅ		ぴょ	
PA	PI	PU	PE	PO	PYA	PYU		PYO	
ふぁ	ふぃ		ふぇ	ふぉ					
FA	FI		FE	FO					

※特殊文字の入力

入力方式がローマ字－カナ変換方式 [F5] または [カナ] キーロックによるカナ入力方式の場合、日本語文書作成の上で欠かすことのできない特殊文字を入力することができます。

(1) 濁点「〝」及び半濁点「゜」の入力には、次のキーを押します。

「〝」……[゛]

「゜」……[゜]

また間接出力モード [F2] の場合、変換フィールド内でひらがなやカタカナの直後に濁点又は半濁点を入力すると、自動的に濁点、半濁点のついた１文字のカナに変換されます。

〔例〕「し」の後に「〝」を入力すると、自動的に「じ」に変換されます。

逆に直接出力モード [F2] を選択した場合、カナと濁点は各１文字分のスペースをとりますので、２文字として扱われます

(2) 読点「、」及び句点「。」の入力には、次のキーを押します。

「、」……[SHIFT]＋[、]

「。」……[SHIFT]＋[。]

(3) 始めかぎ括弧「「」及び終わりかぎ括弧「」」の入力には、次のキーを押します。

「「」……[SHIFT]＋[「]

「」」……[SHIFT]＋[」]

5 カナ―漢字変換

間接出力モード [F2] 時に、ローマ字－カナ変換または [カナ] キーロックによるカナ入力で変換フィールド内に出力されたひらがなまたはカタカナは、[XFER] キーを押すことによってその読みに対応した漢字に変換されます。漢字は、画面右下に最高９文字ずつ羅列されます。目的の漢字が見つからない場合は、スペースキー、[XFER] キー、またはカーソルコントロールキー [↓] を押しますと、次の漢字のグループが表示されます。また、前のグループに戻すには、 カーソルコントロールキー [↑] を押します。目的の漢字が見つかったならば、その漢字を取り出すわけですが、その方法には次の２通りがあります。

(1) [1] ～ [9] の数字キーを使って指定する。

漢字グループの左端の漢字が[1]、右端の漢字が[9]のキーに対応しており、目的の漢字に対応した数字キーを押します。

(2) [←]、[→]のカーソルコントロールキーを使って指定する。

カーソルコントロールキーを使って目的の漢字の上にカーソルを合わせ、リターンキー([↵])を押します。

以上の操作によって、指定された漢字がテキストエリア内のカーソル位置に表示されます。

漢字グループを表示した状態で、入力誤りに気づいたり、目的の漢字がなくて変換を中止したい場合は、[ESC]キーを押します。すると、漢字グループの表示が消え変換フィールド内にカーソルが戻りますので、ひらがなまたはカタカナの訂正ができます。

〔変換例〕音訓変換方式によって、漢字"日本"を表示させます。

①入力モードを、全角文字出力[F1]、間接出力[F2]、音訓変換方式[F8]に設定します。

②変換フィールドに、カナ"にち"を入力します。

〔平仮名〕にち■　　←全／半角　間／直接　ローマ字　音　訓

③[XFER]キーを押して漢字変換します。この時、"にち"という読みをもつ漢字は、音訓辞書中には"日"しかありませんので、漢字グループを表示せずに直ちにテキストエリア上へ漢字"日"を表示します。

日

〔平仮名〕■　　←全／半角　間／直接　ローマ字　音　訓

④変換フィールドにカナ"ほん"を入力します。

日

〔平仮名〕ほん■　　←全／半角　間／直接　ローマ字　音　訓

⑤[XFER]キーを押して漢字変換します。すると、"ほん"の読みを持つ漢字グループがあらわれます。

日

〔平仮名〕ほん　　本翻叛　奔反品

⑥漢字グループの中の該当文字"本"を選びます（カーソルを移動させてリターンキー（[↵]）を押すか、数字キー[1]を入力します。

日本

〔平仮名〕■　　←全／半角　間／直接　ローマ字　音　訓

以上で、漢字"日本"が入力されました。

以上がカナ－漢字変換の概略ですが、一字変換方式及び音訓変換方式ではこのように1度に1文字の漢字変換しかできないため、変換フィールド内のカナも漢字1文字分の読みだけが有効となります。しかし、[H TAB]キーを用いると、1度に漢字数文字分の読みを変換フィールドに入力でき、次々に漢字変換していくことができます。この時、[H TAB]キーは、漢字の読みと読みの間に入力します。後は、[XFER]キーを押して、漢字変換をくり返すだけです。

〔変換例〕[H TAB]キーを使って、漢字"日本"を表示させます。

①入力モードを、全角文字出力[F1]、間接出力[F2]、音訓変換方式[F8]に設定します。

②変換フィールドに、カナ"にち"を入力します。

〔平仮名〕にち■　　←全／半角　間／直接　ローマ字　音　訓

③[H TAB]キーを押します。すると、記号"Î"が表示されます。

■

〔平仮名〕にちÎ■　　←全／半角　間／直接　ローマ字　音　訓

④続けて、カナ"ほん"を入力します。これで、読みの入力は終了です。

■

〔平仮名〕にちÎほん■　　←全／半角　間／直接　ローマ字　音　訓

⑤[XFER]キーを押します。すると、まず最初のカナ"にち"が漢字"日"に変換されます。

日

〔平仮名〕ほん■　　←全／半角　間／直接　ローマ字　音　訓

⑥続けて、 XFER キーを押します。すると、次のカナ＂ほん＂の読みをもつ漢字グループがあらわれます。

日

〔平仮名〕■ほん　　本翻叛　奔反品

⑦漢字グループの中の該当文字＂本＂を選びます。

日本■

〔平仮名〕■　　←全／半角　間／直接　ローマ字　音　訓

以上で、漢字＂日本＂が表示されました。

変換フィールドに入力できるかな数は、全角文字で20字、半角文字で40字です。 H TAB 記号＂Î＂は半角文字として入力されますので、1度に変換できる漢字数はこれらより求めることができます。

6 漢字変換方式について

6.1 コード入力方式

5種類ある漢字変換方式の中で、カナ－漢字変換を用いないのは、コード入力方式だけです。

このコード入力方式を選択するには、 F6 （ SHIFT ＋ F1 キー）を押すか、 GRAPH ＋ 1 キーを入力します。（コード入力方式を選択すると、画面出力文字の選択及び入力方式の選択は無視されます。）コードには、ＪＩＳ漢字コードと区点コードの2種類あり、どちらで入力するかは同じく F6 （ SHIFT ＋ F1 キー）か GRAPH ＋ 6 キーを入力することによって切換えることができます。（ＪＩＳ漢字コードが選択されると＂16進＂、区点コードが選択されると＂区点＂の文字が表示されます）

ＪＩＳ漢字コードと区点コードの間には、次式で示すような関係があります。ただし、ＪＩＳ漢字コードは16進数、区点コードは10進数で表わします。

区点コード（上位バイト）＝（ＪＩＳ漢字コード（上位バイト）－２０H)10
区点コード（下位バイト）＝（ＪＩＳ漢字コード（下位バイト）－２０H)10
ＪＩＳ漢字コード（上位バイト）＝（区点コード（上位バイト）＋３２)16
ＪＩＳ漢字コード（下位バイト）＝（区点コード（下位バイト）＋３２)16

コード入力方式が選択されると、変換フィールド内には数字の０～９及びアルファベット大文字のＡ～Ｆの入力以外受けつけられません。コードの入力は4桁で行ない（上位2桁：上位バイト、下位2桁：下位バイト）ＪＩＳ漢字コード選択時には16進数、区点コード選択時には10進数として自動的にみなされます。また、コードと漢字等の全角文字とは1対1に対応しており、その対応関係の詳細は別表に示される通りです。以下にその概略を示します。

JIS 漢字コード（16進数）	区点コード（10進数）	対応文字（全角文字）
0021～207E	0001～0094	入力無効（無表示）
2121～2F7E	0101～1594	第１水準　非漢字
3021～4F7E	1601～4794	第１水準　漢字
5021～757E	4801～8594	第２水準　漢字（オプション）
7621～7660	8601～8664	外字（PCG 登録文字）

（注）以下のコードを入力しても無効となります。（＊……任意の数）

ＪＩＳ漢字コード：＊＊００～＊＊２０、＊＊７Ｆ～＊＊ＦＦ

区点コード　　：＊＊００、＊＊９５～＊＊９９

（注）コード入力方式では、 XFER キーを押す必要はありません。4桁のコードを入力した時点で、全角文字に変換されてしまいます。

〔例1〕変換フィールド内でＪＩＳ漢字コード３０２１を入力すると、テキストエリア内に漢字＂亜＂が表示されます。

亜■

〔１６進〕3021■　　←全／半角　間／直接　ローマ字　コードＩＮ

〔例２〕変換フィールド内で区点コード１６０１を入力すると、テキストエリア内に漢字＂亜＂が表示されます。

亜■

〔区点〕1601■　　←全／半角　間／直接　ローマ字　コードＩＮ

ところで、全角文字には、漢字ＲＯＭに格納されている第１水準及び第２水準の漢字、非漢字文字のほかに、ＰＣＧを利用したユーザ登録文字（外字）があります。外字には、ＰＣＧ４キャラクタ分に定義する外字全角文字と、　ＰＣＧ２キャラクタ分に定義する外字半角文字の２種類がありますが、ここでは外字全角文字の表示方法について述べます。（外字の定義の仕方については、同梱のＤＥＦＣＨＲ　ＴＯＯＬの取扱い説明書を参照してください）コード入力方式においては、外字全角文字を表示させるために、外字の各文字にコードを割り当てています。外字用のコードは、第１水準及び第２水準の全角文字と衝突しないように、次のように割り当てられています。

ＪＩＳ漢字コード：７６２１～７６６０（64文字分）

区点コード　　：８６０１～８６６４（　〃　）

したがって、外字全角文字を表示させるためには、上記コードのいずれかをキー入力するだけでＯＫです。

上記コードのどれがＰＣＧのどのキャラクタに対応しているかについては、ＤＥＦＣＨＲ　ＴＯＯＬの取扱い説明書を参照してください。

（注）外字半角文字は、コード入力方式では表示できません。外字半角文字を表示するためには、次のようにします。

ＣＧＥＮ２ ↵

ＰＲＩＮＴ＃０，ＣＨＲ＄（Ｘ）↵

ただし、Ｘは、０からＦＥＨまでの偶数のＰＣＧキャラクタコードです。

詳しくは、ＤＥＦＣＨＲ　ＴＯＯＬの取扱い説明書を参照してください。

６.２ 一字変換方式

カナ－漢字変換方式の１つで、変換フィールド内に入力したカナの最初の１文字を判断して、先頭にその読みをもつ漢字を、ＪＩＳ漢字コードの順番通りに羅列します。羅列の仕方は、９文字を１つの漢字グループとみなして、１度に最大９文字ずつ、該当する漢字がなくなるまで交代に表示します。漢字の選び方については、「５．カナ－漢字変換」の項を参照してください。

〔変換例〕一字変換方式によって、漢字＂日本＂を表示させます。

①入力モードを、全角文字出力 [Ｆ１]、間接出力 [Ｆ２]、一字変換方式 [Ｆ７] に設定します。

②変換フィールドにカナ＂にち＂ [H TAB] キー、カナ＂ほん＂を入力します。

■

〔平仮名〕にち↑ほん■　　←全／半角　間／直接　ローマ字　１字変換

③ [XFER] キーを押します。すると、カナ＂にち＂の先頭文字＂に＂の読みを頭に持つ漢字が羅列されます。さらに、目的の漢字が見つかるまで、 [XFER] キーかカーソルコントロールキー [↓] かスペースキーを押し続けます。

■

〔平仮名〕にち　　二尼弐　邇匂賑　肉虹廿

[XFER] キー入力

〔平仮名〕にち　　日乳入　如尿韮　任妊忍

④目的の漢字＂日＂を取り出します（↵キーを押すか、数字キー [1] を入力します）。

日■

〔平仮名〕ほん■　　←全／半角　間／直接　ローマ字　１字変換

⑤同様に、 [XFER] キーを押してカナ＂ほん＂の先頭文字＂ほ＂の読みを頭にもつ漢字を羅列します。

日■
〔平仮名〕ほん

保舗鋪　圃捕歩　甫補輔
XFER キー入力
穂募墓　慕戊暮　母簿菩
XFER キー入力
倣俸包　呆報奉　宝峰峯
XFER キー入力
崩庖抱　捧放方　朋法泡
XFER キー入力
烹砲縫　胞芳萌　蓬蜂褒
XFER キー入力
訪豊邦　鋒飽鳳　鵬乏亡
XFER キー入力
傍剖坊　妨帽忘　忙房暴
XFER キー入力
望某棒　冒紡肪　膨謀貌
XFER キー入力
貿鉾防　吠頬北　僕卜墨
XFER キー入力
撲朴牧　睦穆釦　勃没殆
XFER キー入力

日■
〔平仮名〕ほん　　堀幌奔　本翻凡　盆

⑥目的の漢字＂本＂を取り出します。

日本■
〔平仮名〕■　　←全／半角　間／直接　ローマ字　１字変換

以上で、漢字＂日本＂が入力されます。

ところで、一字変換方式にはもう１つ別の変換の仕方があります。それは、変換フィールドに入力したキャラクタの先頭の１文字を判断して、そのキャラクタコードをＪＩＳ漢字コードの上位バイトに対応させた場合の該当する全角文字を羅列する方式です。例えば、キャラクタ＂！＂（キャラクタコード：&Ｈ２１）を入力した場合、変換フィールドエリアに羅列させる全角文字は、ＪＩＳ漢字コード２１２１～２１７Ｅまでの94個の文字群になります。この方式ですと、入力文字のキャラクタコードがそのままＪＩＳ漢字コードの上位バイトに対応しますので、変換フィールド内に入力するキャラクタは＂！＂（キャラクタコード：&Ｈ２１）から＂ｕ＂（キャラクタコード：&Ｈ７５）までの85文字ということになります。以下に、入力キャラクタと、対応するＪＩＳ漢字コードとの関係を簡単にまとめます。

入力キャラクタ	キャラクタコード	ＪＩＳ漢字コード	補　　足
！ ～ ／	&H21 ～&H2F	2121～2F7E	第１水準　非漢字
０ ～ ０	&H30 ～&H4F	3021～4F7E	第１水準　漢字
Ｐ ～ ｕ	&H50 ～&H75	5021～757E	第２水準　漢字（オプション）

羅列された漢字グループから目的の漢字を取り出すやり方は、通常の方式とまったく同じです。詳しくは、「５．カナ－漢字変換」の項を参照してください。

（注）入力方式としてローマ字－カナ変換方式またはカナ入力方式を選択している場合は、アルファベットや記号の一部を入力することができません。その場合は、英数字直接入力方式に切換えてください。

6.3 音訓変換方式

カナ－漢字変換方式の１つで、変換フィールド内に漢字の音読みまたは訓読みを入力して、それを音訓辞書を利用することによって全角文字に変換します。この方式の利用方法については、「５．カナ－漢字変換」の項に変換例を含めて述べますので、そちらを参照してください。

ところで、音訓変換方式には、漢字の音読み、訓読みだけでなく、各種記号、ギリシア文字、ロシア文字、外字全角文字を表示させるために、次の４種類のカナ文字を入力することができます。

(1) "キゴウ"……各種記号を羅列します。
(2) "ギリシア"……各種ギリシア文字を羅列します。
(3) "ロシア"……各種ロシア文字を羅列します。
(4) "ガイジ"……外字全角文字を羅列します。

変換方法は簡単です。すなわち、変換フィールド内に上記カナ文字列を入力した後 XFER キーを押します。すると、変換フィールドエリアに各種全角文字が羅列されますので、「５．カナ－漢字変換」の項で述べたように、カーソルコントロールキーや数字キーによって目的の文字列を取り出すことができます。

※ 音訓辞書の学習機能について

カナ－漢字変換の音訓変換方式では、使用した漢字を同一グループ内の先頭に配置変換する機能をもっています。したがって、直前に使用した漢字は、真先に羅列されます。

この機能は、上記数字キーによる漢字の選択を行なった場合にのみ行なわれ、カーソルコントロールキーによる選択時には行なわれません。また、音訓辞書ディスク（システムディスク）が書き込み禁止状態にある時には動きません。

※ 辞書ディスクを利用する前に ・・・

音訓辞書やシステム辞書、ユーザ辞書を利用して、漢字変換を行なうためには、それぞれ音訓辞書ディスク（標準装備）、ＳＹＳＴＥＭ辞書ディスク（オプション）、ＵＳＥＲ辞書ディスク（オプション）が必要です。これらの辞書ディスクを利用する場合は、辞書ディスクを挿入したドライブナンバーをコンピュータ側に教えてやらなければなりません。そのためには、次の操作を行ないます。

① ＤＥＶＩＣＥコマンドを使って、辞書ディスクを挿入したドライブナンバーをデフォルトデバイスとして指定します。
② 日本語入力モードにして、漢字変換方式を選択します。選択の仕方は次の通りです。
　・音訓辞書ディスクを利用する場合　　：SHIFT＋F3キー入力
　・ＳＹＳＴＥＭ辞書ディスクを利用する場合：SHIFT＋F4キー入力
　・ＵＳＥＲ辞書ディスクを利用する場合　：SHIFT＋F5キー入力
　このキーを入力することによって、コンピュータ側はそれぞれの辞書を利用する際のアクセス先を知ります。

以上の操作によって、コンピュータはそれ以後の辞書ディスクアクセスの際には、指定したドライブナンバーに対してアクセスを行ないます。一度この操作を行なうと、再度同様の操作によって指定を変更しない限り、ドライブナンバーは変わりません。各辞書ディスクを並行して使用する場合は、ディスクを別々のドライブに挿入した後、各辞書ディスクに対して上記①②の操作をくり返してください。

〔例〕
音訓辞書ディスクをドライブ１に挿入して利用する場合の操作方法

① ＤＥＶＩＣＥ"１：" ↵ …… デフォルト・デバイスの指定
② SHIFT＋XFER …… 日本語入力モードに切換え。詳しくは、本章「２．全角文字の入力方式」を参照してください。

③ SHIFT + F3 …… 音訓変換方式の選択

※ デバイスを変更しても、音訓辞書の読み出されるデバイスは、変化しません。

以上の操作を応用すれば、音訓辞書の高速利用が可能になります。これは、音訓辞書をそっくりそのままグラフィック用V－RAMに転送することによって行ないます。アクセス速度は、フロッピーディスクよりもグラフィック用V－RAMの方が高速ですので、グラフィック用V－RAM上に音訓辞書を移せば、音訓変換を一瞬のうちに行なうことができます。そのためには、次の操作を行ないます。

① グラフィック用V－RAMの使用目的を、外部記憶用に設定します。

```
OPTION SCREEN 2 ↵
```

"OPTION SCREEN 3"または"OPTION SCREEN 4"でもかまいません。（詳しくは、BASICリファレンス・マニュアル「OPTION SCREEN」の項参照。）

② グラフィック用V－RAMを初期化します。

```
INIT"MEM0:" ↵
```

とすると、次のメッセージが表示されますのでY キーを押します。

```
Are you sure ?(y or n)
```

（"OPTION SCREEN 3"に設定した時は、"MEM0"のかわりに"MEM1"を実行してください（以後、同じようにします。）。）

③ 音訓辞書（ドライブ0）をグラフィック用V－RAMに転送します。

```
COPY"0:音訓　　変換.DIC"AS"MEM0:" ↵
```

④ デフォルト・デバイスをグラフィック用V－RAMに指定します。

```
DEVICE"MEM0:" ↵
```

⑤ 日本語入力モードに切換えます。

SHIFT + XFER

⑥ 音訓変換方式を選択します。

SHIFT + F3

⑦ デバイスの操作を④以前にもどします。

以上で、以後の音訓変換はグラフィック用V－RAMに対してアクセスされますので、非常に高速な漢字変換が行なえます。

注 システム辞書及びユーザー辞書については、グラフィック用V－RAMを利用して高速変換を行なうことができません。

7 文字のコピー機能 ： COPY キー

7.1 画面上のコピー

本機の漢字BASICには、画面上に表示されている文字群（全角文字、半角文字を問わず）を任意の場所にそっくりコピーする機能があります。コピーの方法を以下に示します。まず、コピー先にカーソルを移動し COPY キーを押します。すると、カーソルの点滅が止まります。（この位置からコピー文字の表示が開始されます）次に、カーソルコントロールキーでコピーしたい文章の先頭へカーソルを移動します。この時、移動したカーソルは点滅しています。（この位置がコピー元です）あとは、リターンキー（↵）を押す毎に1文字ずつ、カーソルが点滅している箇所の文字が、点滅していないカーソルの箇所に、コピーされます。途中でカーソルコントロールを押せば、点滅している方のカーソルが移動しますので、任意の箇所からコピーを再開することができます。また、コピー先

（点滅していない方のカーソル）が行割れをおこすと、自動的にコピー機能が停止します。コピー動作を中止する場合は、再び COPY キーを押すか、 ESC キーを押してください。

7・2 ハードコピー

COPY キーは、前述した画面上のコピー機能だけでなく、プリンタへのハードコピー機能も持っています。ハードコピーできる画面は、テキスト画面のみ、グラフィック画面のみ、テキスト画面とグラフィック画面を合成した画面の３通りです。ハードコピーを行なうためには、次のキーを入力します。

(1) SHIFT ＋ COPY ……テキスト画面のみハードコピー
(2) GRAPH ＋ COPY ……グラフィック画面のみハードコピー
(3) CTRL ＋ COPY ……テキスト画面とグラフィック画面を合成した画面のハードコピー

＊次の画面モードではハードコピーは行ないません。

４０文字×２０行　，　８０文字×２０行
４０文字×１０行　，　８０文字×１０行

10章
ミュージック

本機には、プログラマブルサウンドゼネレータ（ＰＳＧ）という、擬似音発生用ＬＳＩが内蔵されています。このＰＳＧを使うことにより音楽演奏やゲームの効果音を作ることができます。ここでは、このＰＳＧをコントロールするサウンド制御ステートメントの使い方を説明します。

なお、サウンド制御ステートメントには次のようなものがあります。

BEEP , MUSIC , PLAY , TEMPO , SOUND , SOUND@

1 ブザー音を出す

ブザー音を出すためにはＢＥＥＰステートメントを次のように使用します。

BEEP
- **0**……ブザー音を止める。
- **1**……ブザー音を出し続ける。
- **省略**……短い「ポッ」という音を出す。

〔例〕
```
BEEP 1 ↵
Ok
BEEP 0 ↵
Ok
```

2 楽譜を演奏する

ＭＵＳＩＣ、ＰＬＡＹ、ＴＥＭＰＯのステートメントを使用して楽譜の演奏（８オクターブ、３和音）が行なえます。

MUSIC 文字式

文字式で指定された楽譜に従って演奏します。文字式では、音程、長さ、休符、オクターブ、音量和音の指定を以下のようにして行なうことができます。

①音程……音程は下記のようにＣ～Ｂの文字を使用して指定します。

音程	ド	ド＃（レ♭）	レ	レ＃（ミ♭）	ミ	ファ	ファ＃（ソ♭）	ソ	ソ＃（ラ♭）	ラ	ラ＃（シ♭）	シ
指定文字	C	#C	D	#D	E	F	#F	G	#G	A	#A	B

②長さ……音程の後ろに0～9の整数をつけて、音の長さを指定します。

音の長さ	対応する整数
1/8（32分音符）	0
1/4（16分音符）	1
3/8（付点16分音符）	2
1/2（8分音符）	3
3/4（付点8分音符）	4
1（4分音符）	5
1 1/2（付点4分音符）	6
2（2分音符）	7
3（付点2分音符）	8
4（全音符）	9

＊音の長さは4分音符（整数5）を1としたときの相対的な値です。
また、整数の指定がない場合は前の音と同じ長さを意味します。
③休符……R0～R9で、休みの長さを指定します。
＊数字の値は上記の長さに対応しています。
④オクターブ……O1～O8のようにアルファベットのOの後ろに1～8の整数をつけてオクターブの指定をします。また、音符の前に＋をつけることによって上の音程を、－をつけることによって下の音程を指定できます。初期状態はO4です。
⑤音量……V0～V15で音量の指定をします。なお、この指定の次から音量が変化します。
⑥和音……2重音や3和音を演奏するには、上声部と下声部の間にチャンネルセパレータ：（コロン）をはさみます。
"チャンネルA：チャンネルB：チャンネルC"

TEMPO 数値（30～7500）

MUSIC文によって演奏される曲の速さ（テンポ）を指定します。
＊指定された数値は1分間あたりの4分音符の拍数を示します。
初期状態では120に設定されています。

PLAY { 数値 / 文字式 }

MUSIC文とTEMPO文の両方の機能を持ち、PLAYの後ろが数式の場合TEMPO文と同様の、文字式の場合MUSIC文と同様の機能を持ちます。
〔例〕オクターブを1から8まで変化させてからテンポを速くし、これを繰返します。

```
10 REM Music or Play
20 WIDTH 40:INIT
30 CLS
40 PLAY"V15R0:V15R0:V15R0"……音量最大 休符は最小。
50 FOR T=120 TO 2000 STEP 100…テンポを変化させる。
60 TEMPO T
70 FOR O=1 TO 8……オクターブを1から8まで変化させる。
80 PLAY"O"+CHR$(48+O)+"CDEFGABAGFEDC:DEFGA
BAGFEDCB:EFGABAGFEDCBA"
90 NEXT O
100 NEXT T
```

3 PSGのコントロール

本機で音を出すには、先に述べたＰＬＡＹ文等によって楽譜を文字列にして演奏する方法とＳＯＵＮＤ、ＳＯＵＮＤ@を使用して直接ＰＳＧにデータを送り音を出す方法があります。ここでは、ＳＯＵＮＤ、ＳＯＵＮＤ@の使い方を説明します。

SOUND　PSGのレジスタ番号，データ〔，データ……〕

SOUND@　PSGのレジスタ番号，データ〔，データ……〕

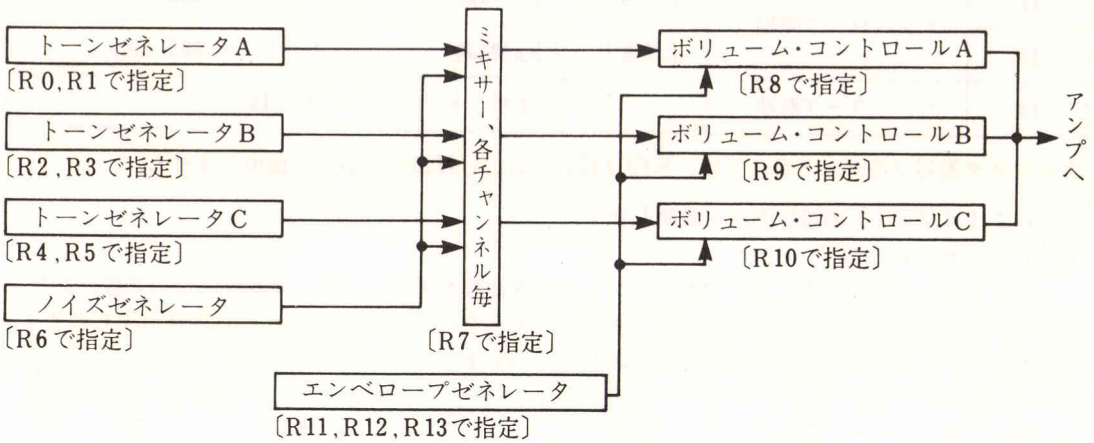
ＳＯＵＮＤ、ＳＯＵＮＤ@命令は、ＰＳＧの14個のレジスタへデータを書き込んで、ゲームの効果音など様々な音を作り出す事が出来ます。

なお、ＳＯＵＮＤ命令は、１つのデータで１つのレジスタ（８ビット）を定義しますが、ＳＯＵＮＤ@命令は、１つのデータで連続したレジスタ（16ビット）を定義できます。後で説明する周波数の設定のように連続したレジスタを定義する場合にＳＯＵＮＤ@を使用します。

また、データをカンマ（，）で区切って並べて書くと、連続した複数のレジスタにそのデータを書き込むことができます。

本機に内蔵しているＰＳＧは、３つのトーンゼネレータ、ノイズゼネレータ、エンベロープゼネレータ、ミキサー、３つのボリュームコントロールから構成されています。

そのブロック図を下に示します。



（注）図中のＲ０～Ｒ13はレジスタ番号です

上のブロックのレジスタにデータを送ることによりさまざまな音が発生します。

その各レジスタの機能表を以下に示します。

レジスタ番号	レジスタ機能	ビット構成 7 6 5 4 3 2 1 0	設定データ値（15進）	
0	チャンネルA周波数	下位8ビットT_L（微調）	0（高）～255（低）	
1		上位4ビット T_H（粗調）	0（高）～15（低）	
2	チャンネルB周波数	下位8ビットT_L（微調）	0（高）～255（低）	
3		上位4ビット T_H（粗調）	0（高）～15（低）	
4	チャンネルC周波数	下位8ビットT_L（微調）	0（高）～255（低）	
5		上位4ビット T_H（粗調）	0（高）～15（低）	
6	ノイズ周波数	5ビットN	0（高）～31（低）	
7	チャンネル選択	IN/OUT IOB IOA ノイズ C B A トーン C B A	0～63（各ビット共、0を設定すると選択される）	
8	チャンネルA音量	M 4ビット	0（小）～16（大）	各チャンネルとも M＝0のとき 4ビット（0～15）の値で音量設定 M＝1のとき（データ値16）音量はエンベロープに依存し4ビットデータは無効になる。
9	チャンネルB音量	M 4ビット	0（小）～16（大）	
10	チャンネルC音量	M 4ビット	0（小）～16（大）	
11	エンベロープ周期	下位8ビットE_L（微調）	0（小）～255（大）	
12		上位8ビットE_H（粗調）	0（小）～255（大）	
13	エンベロープ形状	4ビット	0～15	

＊レジスタ番号7のビット6、7は、SOUND、SOUND@命令では使用しません。

それでは各ブロックの設定方法を説明します。

(1) トーンゼネレータ

レジスタ0～5では、A、B、C各チャンネルの周波数を決定します。周波数は音の音程にあたり、1オクターブ音が上ると周波数は倍になります。また、その間を12に分けると半音階が得られます。下の表は、オクターブ4の各音の周波数を表わしています。

音階	O4C	#C	D	#D	E
周波数（Hz）	261.63	277.18	293.66	311.13	329.63

音階	F	#F	G	#G	A
周波数（Hz）	349.23	369.99	392.00	415.30	44.00

音階	#A	B
周波数（Hz）	466.16	493.88

＊音階はMUSIC文の文字列で表わしています。

ＰＳＧでは入力周波数である２ＭＨｚを内部で16分の１に分周（周波数を分割する操作）したものを、各チャンネルの12ビットデータでさらに分周して発生させます。

求めたい周波数をｆ〔Ｈｚ〕、分周値をＴ〔Ｈｚ〕とすると次の関係があります。

$$f = \frac{2 \times 10^6}{16 \times T} \cdots\cdots\cdots\cdots ①$$

例えば、１ＫＨｚの音を発生するには、①を変形した次の式に周波数を代入することにより分周値を求めることができます。

$$T = \frac{2 \times 10^6}{16 \times f} \cdots\cdots\cdots\cdots ②$$

代入すると

$$T = \frac{2 \times 10^6}{16 \times 10^3} = 125$$

この分周値を各チャンネルのレジスタ（Ａチャンネルの場合、レジスタ１の上位４ビットとレジスタ中の下位８ビット）に指定することにより１ｋＨｚの音が設定できます。

Ａチャンネルに１ＫＨｚを設定する場合次のようにします。

```
SOUND@０，１２５
```

＊この場合レジスタ０には１２５が、レジスタ１には０が設定されます。

また、４４０Ｈｚ（Ｏ４Ａ）を指定したい場合は次のようにします。

$$T = \frac{2 \times 10^6}{16 \times 440} \fallingdotseq 284$$

チャンネルＢに４４０Ｈｚ（Ｏ４Ａ）を指定する場合、次のようにします。

```
SOUND@２，２８４
```

＊この場合レジスタ２には２５６が、レジスタ３には２８が送られます。

(2) ノイズゼネレータ

レジスタ６によって雑音の平均周波数を指定します。雑音とは周波数が不規則に変化するものを指し、レジタス６の５ビットの値によって分周値を設定します。

例えば、５ＫＨｚのノイズを発生するには②の式に周波数を代入し分周値を求めます。

$$T = \frac{2 \times 10^6}{16 \times 5000} = 25$$

この値をレジスタに設定する場合１つのレジスタのみを指定すればよいためＳＯＵＮＤ命令を使います。

```
SOUND　６，２５
```

(3) ミキサー（チャンネル選択）

レジスタ７では、Ａ，Ｂ，Ｃ各チャンネル毎にトーン／ノイズ／（トーン＋ノイズ）出力の有無を選択します。例えば、

```
SOUND７，４４またはSOUND７，&Ｂ１０１１００
```

と設定すると、Ａチャンネルからはトーンだけ、Ｂチャンネルからはトーン＋ノイズ、Ｃチャンネルからはどちらも出力されません。

(4) ボリュームコントロール

レジスタ番号８～１０は、各チャンネルの音量を決定します。ただし、Ｍビット（５ビット目）の値により、４ビットデータによる０～15までの一定した音量が設定できるモード（Ｍ＝０）と、自動的に音量が０～１５まで変化するエンベロープモード（Ｍ＝１）とを選択できます。

例えば、

```
SOUND　９，１２
```

と設定するとＢチャンネルの音量を12に設定できます。

(5) エンベロープゼネレータ

レジスタ１１、１２では、エンベロープの周波数を指定し、レジスタ13ではエンベロープの形状を指定します。

まず、周波数の設定方法を説明します。ＰＳＧでは、エンベロープの周波数は入力周波数２ＭＨｚを２５６分の１に分周し、それをレジスタ１１、１２の１６ビットの値でさらに分周しています。そのため、エンベロープ周波数をｆＥ〔Ｈｚ〕、分周値をＴとすると次の関係があります。

$$f_E = \frac{2 \times 10^6}{256 \times T} \cdots\cdots ③$$

例えば、ｆＥ＝０．５〔Ｈｚ〕（エンベロープ周期１／ｆＥ＝２〔秒〕）を設定するには、③式を変形した次の式に周波数を代入することにより分周値を求めることができます。

$$T = \frac{2 \times 10^6}{256 \times f_E} \cdots\cdots ④$$

代入すると

$$T = \frac{2 \times 10^6}{256 \times 0.5} = 15625$$

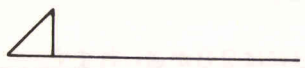
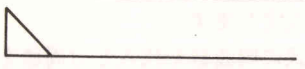
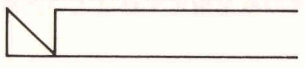
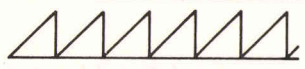
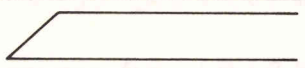
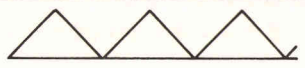
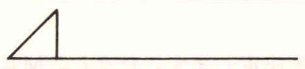
この分周値を次のように指定することによって０．５〔Ｈｚ〕が設定できます。

SOUND@１１，１５６２５

＊この場合、レジスタ１１に９が、レジスタ１２に６１が設定されます。

次に、エンベロープの形状を決定します。

それには、下の表に示すパターンに対応したデータをレジスタ１３に指定することが必要です。

R13の設定データ	エンベロープパターン
0～3	
4～7	
8	
9	
10	
11	
12	
13	
14	
15	

$1/f_E$＝エンベロープ周期

〔例〕では、実際にSOUND，SOUND@命令で音を出してみます。下のプログラムを実行してください。

```
１０　SOUND@０，２８４　　……Ａチャンネルの周波数設定。
２０　SOUND６，２５　　……ノイズ周波数設定。
３０　SOUND@１１，１５６２５……エンベロープ周期の設定。
４０　SOUND１３，８　　……エンベロープパターンの設定。
５０　SOUND８，１５　　……Ａチャンネルの音量設定
６０　SOUND７，&Ｂ１１１１１０……Ａチャンネルの選択。
```

Aチャンネルから４４０Ｈｚ（Ｏ４Ａ）の音が出ます。音をとめるには[CTRL]+[D]を押してください。次に６０行目を次のように再入力してください。

```
６０　ＳＯＵＮＤ７、＆Ｂ１１０１１０……Aチャンネルにノイズ＋トーンを設定。
```

Aチャンネルからは、音とノイズの両方が出力される。次に５０行目を次のように再入力してください。

```
５０　ＳＯＵＮＤ８、１６　　……エンベロープモードを設定。
```

Aチャンネルの音量にエンベロープがかかり、おもしろい音になります。

また、ＳＯＵＮＤ命令でエンベロープを指定することにより、ＰＬＡＹ（ＭＵＳＩＣ）文による同一音程の演奏を音符毎に区切ることができます。

```
ＰＬＡＹ１２０：ＰＬＡＹ"Ｖ１５Ｏ４Ｃ５ＣＤＤＥＥＤ"
```

上のＰＬＡＹ文を、次のようにかえてみてください。ただし、文字式での音量指定（Ｖ）はＶ１６にします。

```
１０　ＳＯＵＮＤ＠１１，５２４５：ＳＯＵＮＤ１３，０：ＳＯＵＮＤ８，１６
２０　ＰＬＡＹ１２０：ＰＬＡＹ"Ｖ１６Ｏ４Ｃ５ＣＤＤＥＥＤ"
```

11章

機械語サブルーチンとモニタ

ここでは、ＢＡＳＩＣプログラムと機械語サブルーチンをいっしょに使用する方法を説明します。

1 機械語サブルーチンの入力

1.1 機械語サブルーチンの記憶領域の設定

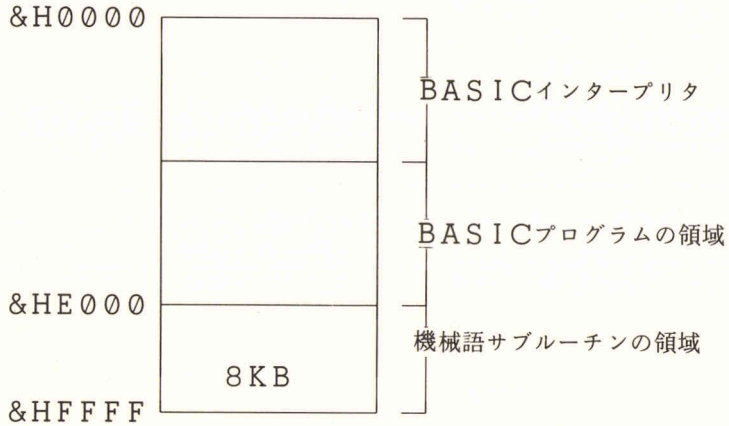
機械語サブルーチンを入力するためには、まずＣＬＥＡＲコマンドを用いて、その記憶領域を確保する必要があります。

たとえば、ＢＡＳＩＣの６４ＫＢのメインメモリ中、上部の８ＫＢ（最上部の３ＫＢはワークエリアとして使用されるので実際は５ＫＢ）を確保したければ、

```
CLEAR &HE000
```

というステートメントをプログラムの先頭に書くか、そのままダイレクトに実行します。

メモリの概略図



これで、ＢＡＳＩＣプログラムが機械語サブルーチンの領域に侵入することがなくなります。

ただし、最上部の&HF400～&HFFFFの３Ｋバイトは、ＢＡＳＩＣ、ＩＰＬ（Ｉｎｉｔｉａｌ　Ｐｒｏｇｒａｍ　Ｌｏａｄｅｒ）およびＢＩＯＳのワークエリアとなっているため使用することができません。また、&HF000～&HF3FFは辞書のエリアとして使用されるので、辞書を使うときはこのエリアを使用することができません。

1.2 機械語サブルーチンの入力方法

機械語サブルーチンをメモリに記憶させる方法には、次の２通りあります。

１．ＰＯＫＥステートメントを使う。

２．ＭＯＮコマンドで機械語モニタを起動する。

〔1〕　ＰＯＫＥステートメントによる入力

ＰＯＫＥステートメントを使って、機械語サブルーチンを入力する手順を次に示します。

１．機械語（16進数表現のコード）によってプログラムを作ります。

２．機械語を、１バイトごとにカンマ（，）で区切って、ＤＡＴＡ文の中にデータとして用意します。

３．機械語データをＲＥＡＤステートメントで逐次読み込んで、ＰＯＫＥステートメントでメモリに書き込みます。

この方法は、比較的小さなプログラムを記憶するときに使います。

次に簡単な例を示します。

```
100 'POKEステートメントによる入力
110 CLEAR &HD000
120 I=0
130 READ D$
140 D=VAL("&H"+D$)
150 POKE &HD000+I,D
160 IF D=&HC9 THEN 190
170 I=I+1
180 GOTO 130
190 FOR J=&HD000 TO &HD000+I
200 RD=PEEK(J)
210 PRINT HEX$(RD),
220 NEXT
230 END
240 '機械語データ
250 DATA 3E,07      :'LD   A,07H
260 DATA 01,00,20   :'LD   BC,2000H
270 DATA ED,79      :'OUT  (C),A
280 DATA 3E,41      :'LD   A,41H
290 DATA 01,00,30   :'LD   BC,3000H
300 DATA ED,79      :'OUT  (C),A
310 DATA C9         :'RET
```

このプログラムは、&HD000番地から機械語データを入力しています。

110：機械語サブルーチンの記憶領域を&HD000 から確保しています。

130～180：250～310のDATA文で指定された機械語データを読み
POKEステートメントでメモリに書き込んでいます。

190～220：書き込んだデータをPEEK関数で読み出して表示しています。

1.3 機械語モニタによる入力

機械語モニタ（後述「機械語モニタ」参照）を用いて、機械語サブルーチンを入力する方法を次に示します。

1．MON↵と打ち込んで、機械語モニタを起動します。

2．モニタのDコマンドかMコマンドを実行し、メモリに機械語（16進数表現のコード）を書き込みます。

3．必要に応じてSコマンドを使い、作った機械語サブルーチンをカセットテープに記録します。

4．Rコマンドで機械語モニタからBASICに戻ります。

2 BASICプログラムから機械語サブルーチンを呼び出す方法

BASICから機械語サブルーチンを呼び出すには、CALLステートメントを使う方法と、USR関数を使う方法の2通りがあります。

2.1 CALLステートメント

機械語サブルーチンは、BASICのCALLステートメントを使って呼び出すことができます。

CALLステートメントの形式は次の通りです。

文法：CALL a

a：機械語サブルーチンの開始アドレス。

機械語サブルーチンの最後には、もとのプログラムに戻るためにC9（16進数、ニーモニックではRET）が置かれます。

CALLステートメントを実行すると、プログラム・カウンタの内容をスタックに退避し、CALLステートメント中に指示されたアドレスa、すなわちサブルーチンの入口へジャンプします。

機械語サブルーチンの処理を終えてC9に来ると、スタックに退避しておいたプログラム・カウンタをもとに戻し、もとのBASICプログラムの実行を再開します。

CALLステートメントではBASIC上のデータを機械語サブルーチンへ引き渡す機能がありません。もし、データの引き渡しが必要な場合には次に述べるUSR関数を使用する方法が便利です。

2.2 USR関数

USR関数は、引数の値を受け渡すことができ、また、エラー処理のサブルーチンを呼び出すことができる、という利点を備えています。

USR関数を使うためには、まず、呼び出す機械語サブルーチンの実行開始アドレスを定義する必要があります。それにはDEFUSRステートメントを使って、次のように書きます。

文法：DEFUSRn＝a

n：USR関数識別番号。0～9の整数。

a：実行開始アドレス。

（例）100 DEFUSR0＝&HE000

110 DEFUSR1＝&HEC00

nは、関数識別番号で、0～9の最大10個の機械語サブルーチンを定義することができます。nを省略すると0が指定されます。

実行開始アドレスaは、機械語サブルーチンの実行が開始されるアドレスであり、メイン・メモリ64KB中のどこにでも指定できます。

DEFUSRでいったんアドレスを定義すると、再度定義しなおすまで、そのアドレスが保持されます。

DEFUSRで、nとaを定義したら次のようにしてUSR関数を使うことができます。

文法：〔1〕 y＝USRn（x）

y：数値変数。

n：USR関数識別番号。0～9の整数。

x：数式（引数）。

〔2〕 y$＝USRn（x$）

y$：文字変数。

n：USR関数識別番号。0～9の整数。

x$：文字式（引数）。

（例）200 A＝USR0（65）

210 A$＝USR1（C$）

USRの後ろにつける番号nは、DEFUSRで設定した番号に対応しています。

USR関数は、引数xやx$の値をもって、n番の機械語サブルーチンを呼び出し、リターン時にその値をyやy$に代入します。機械語サブルーチンの中で、引数の値を書き換えると、その結果がyやy$の値となります。

USR関数の引数x、x$は省略することができません。

xは、数式で、単独の定数、数値変数でもかまわず、精度も整数型、単精度型、倍精度型のいずれでもかまいません。

x$は、文字列を値にもつ文字式です。

USR関数で機械語サブルーチンを呼び出すとき、CPUの書くレジスタには次のような値が入っています。

・Aレジスタ（アキュムレータ）

Aレジスタ（アキュムレータ）には、引数の型を示す値が入っています。

引数の型	Aレジスタの値
整数型	2
単精度型	5
倍精度型	8
文字型	3

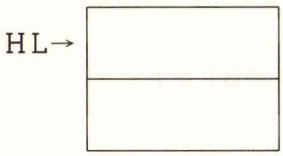
これらの値は、文字変数を除き、メモリ内において何バイトで表現されているかを表わしています。

・HLレジスタ

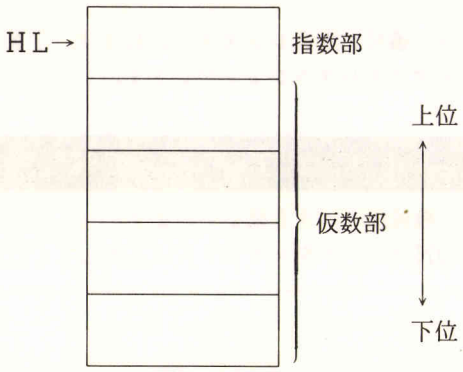
HLレジスタの値は、数値変数と文字変数の場合で異なっています。数値変数の場合は引数の入っているメイン・メモリのアドレスを示しており、引数が文字変数の場合は、直接文字列の入っているアドレスを示さず、ストリング・ディスクリプタと呼ばれている相対アドレスを示しています。ストリング・ディスクリプタは、メモリの文字変数領域の先頭アドレスから何番地目に文字列が入っているのかを示す値で、直接文字列の入っているアドレスを示すものではありません。文字列の入っているアドレスは、DEレジスタに入っています。

引数の値は、HLレジスタが指しているアドレスから次のような形式で格納されています。

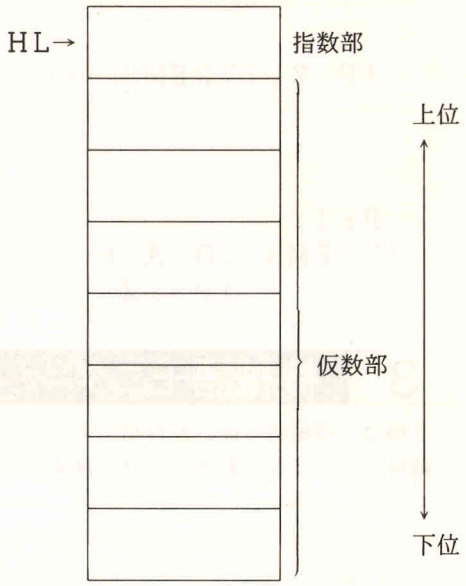
〔整数型〕……2バイト



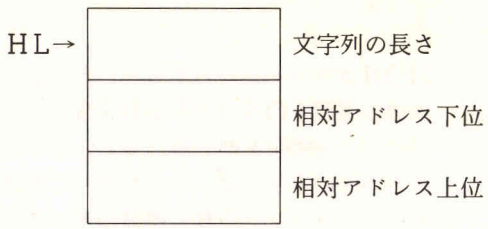
〔単精度型〕……5バイト



〔倍精度型〕……8バイト



〔文字型〕……3バイト



・DEおよびBレジスタ

DEとBレジスタは、Aレジスタが3のとき、すなわちUSR関数の引数が文字変数のときのみ意味をもちます。

このとき、DEレジスタにはその文字データが格納されているアドレス（絶対アドレス）、Bレジスタには文字データの長さが入っています。

USR関数の引数が文字変数だけのときは、引数の格納されているアドレスとUSR関数実行後の引数のアドレスが同じなので、引数の値が変わってしまう可能性があります。したがって、文字変数を引数にするときは、

```
USR（A$+""）          ※""はヌルストリングです。
```

というように、ヌルストリングを連結した形にするようにしてください。これで、引数A$の値がUSR関数実行前と実行後とで変わることはなくなります。

・USR関数内でのエラー処理

USR関数の中でエラーが発生した場合、BASICにエラーの発生をしらせることができます。このとき、BASIC内であらかじめON ERROR GOTOの処理をしていれば、エラー処理ルーチンへジャンプすることもできます。

その方法は、エラー番号をAレジスタに入れIXレジスタの示すアドレスにジャンプすることにより行なうことができます。

（例）機械語サブルーチン中にエラーが発生したら、Type mismatch（エラー番号13）の処理へジャンプする場合、

```
      :
      :
      :
   JP  Z,TYPEMS……エラーが発生したら分岐します。
      :
      :
      :
   RET
TYPEMS:LD  A,13……エラー番号13をAレジスタへいれます。IXレジスタ
       JP（IX）          の示すアドレスへジャンプします。
```

3 機械語モニタ

本機は、機械語の扱いを容易にするため、簡単な機械語モニタを備えています。

機械語モニタを起動するには、BASICのMONコマンドをダイレクトに入力してください。

```
Ok
MON↵
*
```

MONコマンドを入力すると、上のように＊印が表示され、機械語モニタが起動します。

本機の電源をONにするかBASICでBOOT↵を実行して、IPLを起動しているとき SHIFT ＋ BREAK を押し続けると、図の画面となります。

この画面のとき、M キーを押しても機械語モニタを起動することができます。ただし、この場合はモニタのコマンドのRで機械語モニタから抜け出ることができません。

```
Make your device ready

Press Selected Key to start driving:

        F:Floppy
        R:ROM
        C:CMT
        T:Timer
```

機械語モニタのコマンド

機械語モニタには、次に示す14種類のコマンドが用意されています。

コマンド	働き
D	Dump memory（メモリダンプ）
M	set Memory（メモリセット）
F	Find data（データサーチ）
P	Printer switch（プリンタスイッチ）
G	Gosub
T	Transfer data（データ転送）
S	Save
L	Load
V	Verify
R	Return
!	BIOS ROM/MAIN RAM access mode switch
#	WIDTH40/80 switch、画面の初期化など
W	Write（各デバイスへ）
Y	read（各デバイスから）

D （dump memory）

働き：メインメモリの内容を表示します。

文法：＊D〔XXXX〔YYYY〕〕

XXXX：先頭アドレス。16進数4けた以内。0～FFFF。

YYYY：最終アドレス。16進数4けた以内。0～FFFF。

説明：ＸＸＸＸからＹＹＹＹまでのメインメモリの内容を表示します。
ＹＹＹＹを省略すると、ＸＸＸＸから１２８バイト分（８バイト×１６行）表示します。ＸＸＸＸもいっしょに省略すると、それまで表示した次のアドレスから１２８バイト分表示します。
画面の１行分の表示形式は、次のようになります。

```
:XXXX=HH HH HH HH HH HH HH HH/CCCCC.C.
↑    ↑  ↑                  ↑     ↑     ↑
(1)  (2) (3)                (4)   (5)   (6)
```

(1) 編集可能なことを示すマーク
(2) 先頭データのアドレス。１６進数１～４けた。０～ＦＦＦＦ。
(3) アドレスとデータ群の区切り記号
(4) １６進数のデータ８バイト。ＨＨが００～ＦＦの１６進数で、空白をはさんで８個並ぶ。
(5) データ群と対応文字の区切り記号。
(6) (4)のデータ群に対応する文字列。ＨＨをキャラクタ・コード（１６進数）とする文字を表示します。ＨＨがコントロール・コードのときは、ピリオド（.）が表示されます。

この状態のとき、次の〔編集書式〕に従って、メインメモリの内容を書き換えることができます。
表示のストップは、BREAKキー、コマンド待ちに戻るにはSHIFT＋BREAKキーを押します。

〔編集書式〕

機械語モニタもＢＡＳＩＣと同様にスクリーン・エディタを備えており、メインメモリのダンプ後、あるいはサーチ後、次の書式に従って内容を書き換えることができます。

〔書式〕　：アドレス＝データ　データ　……

```
↑    ↑  ↑   ↑↑          ↑  ↑
(1)  (2) (3) (4)(5)      (4) (5)
```

(1) コロン：ＢＡＳＩＣの行番号と(2)のアドレスを間違えないための記号で、編集可能な行であることを示します。
(2) アドレス：先頭データのアドレス。１６進数４けた以内で指定します。０～ＦＦＦＦ。
(3) イコール：アドレスとデータ群の区切り記号。
(4) データ：メモリに書き込まれるデータで、次の４通りの指定の仕方があります。
ａ）ＨＨ⇨１６進数２けた。
ｂ）；Ｘ⇨Ｘは１文字。
ｃ）"ＸＸ……Ｘ"
ｄ）'ＸＸ……Ｘ'
ａ）の場合、１６進数２けたの値が書き込まれます。
ｂ）の場合、セミコロン（；）に続く１バイトに対応するキャラクタ・コードが書き込まれます。
データとデータの間は、原則として、空白で区切ります。
ｃ）またはｄ）の場合、ＸＸ……Ｘの部分にメッセージを書くことができます。ただし、ｃ）の場合、メッセージ中に引用符（"）を含めることができません。また、ｄ）の場合、メッセージ中にアポストロフィ（'）を含めることができません。

（例）：８０００="ＣＡＮ'Ｔ　ＣＯＮＴＩＮＵＥ"
：９０００='ＬＯＡＤＩＮＧ"ＢＡＳＩＣ"'

書き換えた後、その行で↵キーを押すと、その行のデータがメモリに記憶されます。

M（set memory）

働き：メインメモリの内容を1バイト表示します。

文法：＊M〔 XXXX〕

XXXX：先頭アドレス。16進数4けた以内。0～FFFF。

説明：MXXXX↵と入力すると、そのアドレスの内容を表示して、データの入力待ちとなります。

```
＊MFE00↵
：FE00=□H
        ↑
      カーソル
```

このとき、前述の〔編集書式〕に従って、データを書き換えることができます。書き換えた後、↵キーを押すと、データがメモリに記憶され、記憶されたデータ数から次のアドレスが計算されて、改行されます。

```
＊MFE00↵
：FE00=3E 41 13 00 C9↵
：FE06=□H
        ↑
      カーソル
```

この状態から抜け出すには、SHIFT＋BREAKキーを押します。

F（find data）

働き： データを探し出します。

文法： ＊F XXXX YYYY HH HH ……

XXXX：先頭アドレス。16進数4けた以内。0～FFFF。

YYYY：最終アドレス。16進数4けた以内。0～FFFF。

HH：データ。16進数2けた以内。0～FF。

説明： XXXXからYYYYまでのメインメモリから、HH HH ……で指定された一続きのデータ群を探し出し、見つかったときは、そのアドレスとデータを表示します。

```
＊F1000 2000 CD 41 00↵
：10C3=CD 41 00 ／ヘA．
＊□
  ↑
 カーソル
```

上の例は、「CD 41 00」という3バイトのデータを、メモリの1000～2000番地（16進数）から探して、10C3番地にそれが見つかったことを示しています。

このコマンドを実行すると、該当するアドレスとそのデータを、あるだけ何行でも表示します。途中で、このコマンドから抜け出たいときは、SHIFT＋BREAKキーを押してください。

P（printer switch）

働き： 画面表示とプリンタ表示の切り換えをします。

文法： ＊P

説明：　Pコマンドを実行すると、DおよびFコマンド実行後の表示を、プリンタにしたり、画面にしたり、切り換えることができます。

機械語モニタを起動した時点では、画面に表示するようになっていますが、このコマンドを入力するたびに、プリンタ表示／画面表示、と表示するデバイスが反転します。

プリンタ正しく接続されていない場合は、しばらくしてから「Ｅｒｒ　７３」と画面に表示されてコマンド待ちになります。そのときは、プリンタをチェックするか、再度Pコマンドを実行して表示を画面に戻してください。

G（gosub）

働き：　機械語サブルーチンを呼び出します。

文法：　＊G　HHHH

　　HHHH：　アドレス。１６進数４けた以内。

説明：　メインメモリ上のHHHHから始まるサブルーチンを呼び出します。

T（transfer data）

働き：　データの転送をします。

文法：　＊T　XXXX　YYYY　ZZZZ

　　XXXX：　データの先頭アドレス。１６進数４けた。

　　YYYY：　データの最終アドレス。１６進数４けた。

　　ZZZZ：　データの転送先のアドレス。１６進数４けた。

説明：　メインメモリ上のＸＸＸＸからＹＹＹＹまでのデータをＺＺＺＺを先頭とする場所に転送します。

S（save）

働き：　メインメモリの内容をカセット・テープに記録します。

文法：　＊S　XXXX　YYYY　ZZZZ：ｆｉｌｅ　ｎａｍｅ

　　XXXX：　先頭アドレス。１６進数４けた。０～ＦＦＦＦ。

　　YYYY：　最終アドレス。１６進数４けた。０～ＦＦＦＦ。

　　ZZZZ：　実行開始アドレス。１６進数４けた。０～ＦＦＦＦ。

説明：　ＸＸＸＸからＹＹＹＹまでのメインメモリの内容をカセット・テープに記録します。

再びＬコマンドでロードするとき、先頭アドレスが省略されると、ＸＸＸＸからロードされます。

ＺＺＺＺは、ＩＰＬから機械語プログラムを走らせたり、ＢＡＳＩＣからＬＯＡＤＭコマンドのＲオプションによって走らせるときに、実行の開始アドレスとなります。

```
＊S　0000　1FFF　0000：TEST.BIN
　　↑　　↑　　↑　　↑
　　(1)　(2)　(3)　(4)
```

(1)　ロード時の先頭アドレス

(2)　ロード時の最終アドレス

(3)　実行開始アドレス

(4)　ファイル名

ｆｉｌｅ　ｎａｍｅは、１３文字以内のファイル名と３文字以内のエクステンションからなり、：の後ろに続けて書きます。

L（load）

働き：　カセットのデータをメインメモリに読み込みます。

文法：　＊L〔XXXX〕〔：ｆｉｌｅ　ｎａｍｅ〕

　　XXXX：　ロード先頭アドレス。16進数4けた。

説明：　カセット・テープに記録されているデータや機械語プログラムをメインメモリに読み込みます。

　XXXXを指定すると、そのアドレスから読み込み、省略すると、Sコマンドで記録した通りの形で読み込みます。

　ｆｉｌｅ　ｎａｍｅを省略すると、最初に見つけたファイルを読み込みます。

　読み込んでいる途中で、SHIFT＋BREAKキーを押したり、その他のエラーが生じたときは、「Ｅｒｒ　２９」と表示されコマンド待ちの状態に戻ります。

このコマンドは機械語プログラムを読み込むだけで実行は行ないません。

このコマンドによって、読み込んだ機械語プログラムを実行するには、Gコマンドを使います。

V（verify）

働き：　カセットの内容とメインメモリの内容を比較します。

文法：　＊V〔：ｆｉｌｅ　ｎａｍｅ〕

説明：　指定したファイルをカセット・テープから読み込み、メインメモリの内容と比較します。

　Sコマンドで機械語プログラムやデータが正しく記録されたかどうか調べるときに使います。もしカセット・テープから読み込んだデータとメモリの内容と一致していなければ、「Ｅｒｒ　２９」を出して止まります。

　「Ｅｒｒ　２９」が出たときは、再度Sコマンドで記録しなおしてください。

R（return）

働き：　機械語モニタからシステムに戻ります。

文法：　＊R

説明：　機械語モニタを起動したＢＡＳＩＣに戻ります。

　このとき、ＳＰ（スタック・ポインタ）およびＨＬレジスタの内容は保存されているので、MONの次のコマンドやステートメントに移ります。次の命令がないときは、コマンド待ちの状態となります。

```
＊R
Ok
□
↑
カーソル
```

！（change access mode）

働き：　ＩＰＬ　ROMとメインメモリのアクセスの切り換えをします。

文法：　＊！

　　）！

説明：　モニタのコマンド（D、M、F、G、T）でアクセスされるメモリ、バンクをＩＰＬ　ROMにするか、メインメモリにするかを切り換えることができます。

　機械語モニタを起動した時点では、メインメモリがアクセスされており、コマンド待ちのとき「＊」が表示されますが、

　　＊！↵

　を実行すると、ＩＰＬ　ROMがアクセスされるようになり、コマンド待ちのとき）！が表示されます。

また、その状態で

）！

を実行すると、「＊」が表示されメインメモリのアクセスに戻ります。

ＩＰＬ　ROMのアクセス時、Ｄコマンドによる読み込むメモリの表示はＩＰＬ　ROMになっていますが、書き込むメモリはあくまでもメインメモリになるので注意してください。

＃（set　screen　mode）

働き：　画面モードの切り換え、ＰＡＬＥＴの初期化、グラフィック画面の消去をします。

文法：　〔1〕　＊＃

〔2〕　＊＃Ｐ

〔3〕　＊＃Ｃ

〔4〕　＊＃ｋｄｍ

ｋ：　0または1。

ｄ：　0～2。

ｍ：　0～5の整数。

説明：　〔1〕　＊＃↵

ＷＩＤＴＨ４０とＷＩＤＴＨ８０の切り換えをします。

〔2〕　＊＃Ｐ↵

ＰＡＬＥＴの初期化をします。

ＰＡＬＥＴ０，０：ＰＡＬＥＴ１，１：ＰＡＬＥＴ２，２：ＰＡＬＥＴ３，３：ＰＡＬＥＴ４，４：ＰＡＬＥＴ５，５：ＰＡＬＥＴ６，６：ＰＡＬＥＴ７，７↵またはＰＡＬＥＴ@０、１、２、３、４、５、６、７↵と同じ

〔3〕　＊＃Ｃ↵

グラフィック画面の消去をします。

（＝ＣＬＳ０↵）

〔4〕　＊＃ｋｄｍ↵

ｋ、ｄ、ｍの値によって次のような設定ができます。

kの値	コ　ー　ド　の　指　定
0	16進数表現のキャラクタコード８０～９Ｆ、Ｅ０～ＦＦを半角文字（1バイトコード文字）のコードとします。（＝KMODE０↵）
1	キャラクタコード８０～９Ｆ、Ｅ０～ＦＦを全角文字（2バイトコード文字）のコード（シフトＪＩＳコード）の第1バイトとします。（＝KMODE１↵）

dの値	使　用　モ　ニ　タ
0	専用ディスプレイテレビの標準／高解像度切換えスイッチの状態に従います。
1	標準ディスプレイの選択
2	高解像度ディスプレイの選択

mの値	設定される画面
0	テキスト画面25行／グラフィック画面200ライン
1	テキスト画面12行／グラフィック画面192ライン
2	テキスト画面20行／グラフィック画面なし
3	テキスト画面10行／グラフィック画面なし
4	テキスト画面25行／グラフィック画面400ライン
5	テキスト画面12行／グラフィック画面384ライン

W（device Write）

働き：　メインメモリの内容を指定デバイスの指定レコードに書き込みます。

文法：　＊WXd：nnnn　rr　aaaa

X：　デバイス名。

M……グラフィック・メモリ（d＝0または1）

E……外部メモリ（d＝0～9）

D……3or5インチ版ディスク（d＝0～3）

F……8インチ版ディスク（d＝0～3）

H……5インチ10Mハードディスク（d＝0～3）

d：　ドライブ番号。（デバイス名Xによって範囲が異なります。）

nnnn：　書き込む先頭のレコード番号。最大4けたの16進数。

rr：　書きこむ**レコード長**1）。1～FFの16進数。

aaaa：　メモリの先頭アドレス。0～FFFFの16進数。

1）　レコード長　1単位が1セクタ（標準256バイト）。

説明：　Xで指定されたデバイスのd番ドライブのレコード番号nnnnに、メインメモリ内のアドレス&Haaaaから始まるレコード長rrのデータを書き込みます。

デバイスXにDかFでディスクを指定するときは、Mコマンドを実行して、デスクのタイプを示すデータをメモリのワーク・エリアに書き込んでおく必要があります。

次に、使用するディスクのドライブ番号と書き込むアドレスの関係とフロッピー・ディスクと書き込むデータの関係を示します。

使用するドライブ番号		ディスクのタイプを表わすデータを書き込むアドレス
3or5インチ版	ドライブ0	FAB4
	1	FAB5
	2	FAB6
	3	FAB7
8インチ版	ドライブ0	FAB8のビット1、0
	1	FAB8のビット3、2
	2	FAB8のビット5、4
	3	FAB8のビット7、6

フロッピーディスクのタイプ	書き込むデータ
3or5インチ版2D（320K）	0
5インチ版2DD（640K）	1
5インチ版　2HD（1M）	2
5インチ版＊2HD（1M，標準フォーマット）	3
8インチ版　2D−256（1M）	00
8インチ版　＊2D−256（1M，標準フォーマット）	01
8インチ版　＊1S−128（240K）	10

（8"ディスクのデータは2ビット毎のビットパターン1バイトから成っています。）

（例）　3or5インチ版のドライブ0で2DDタイプのフロッピーディスクを使用する場合

```
*MFAB4↵
:FAB4=01↵
:FAB5=00 SHIFT + BREAK
*WD0:1  08  C000↵
```

……3or5インチ版ディスクドライブ0のフロッピーディスク上のレコード番号1以降に、メインメモリ内のアドレス&HC000から8レコード分のデータを書き込みます。

Y（device read）

働き：　指定デバイスの指定レコードから指定のメインメモリアドレスにデータを読み込みます。

文法：　*YXd:nnnn　rr　aaaa

X：　デバイス名。

M……グラフィック・メモリ（d＝0または1）

E……外部メモリ（d＝0～9）

D……3or5インチ版ディスク（d＝0～3）

F……8インチ版ディスク（d＝0～3）

H……5インチ10Mハードディスク（d＝0～3）

d：　ドライブ番号。（デバイス名Xによって範囲が異なる）

nnnn：　書き込む先頭のレコード番号。最大4けたの16進数。

rr：　書き込む**レコード長**1）。1～FFの16進数。

aaaa：　メモリの先頭アドレス。0～FFFFの16進数。

1）レコード長　1単位が1セクタ（標準256バイト）。

説明：　Xで指定されたデバイスのd番ドライブのレコード番号nnnnから、メインメモリ内のアドレス&Haaaaから始まるエリアに、レコード長rrのデータを読み込むます。

デバイスXにDかFでディスクを指定するときは、Mコマンドを実行して、ディスクのタイプを示すデータをメモリのワーク・エリアに書き込んでおく必要があります。書き込むデータについてはWコマンドを参照してください。

12章
テレビコントロール

本機では専用ディスプレイテレビを接続したときにキーボードやＢＡＳＩＣの命令で自由にテレビをコントロールできます。
さらに本機にはカレンダー付タイマー機能がついていますから、これらを組み合わせて、専用ディスプレイテレビをつけたり、消したりのテレビコントロールが可能になります。ここではあらかじめ内蔵されているテレビタイマーコントロールを使ったものと、ＢＡＳＩＣの命令を使ったものとの２通りの説明をしています。
ここで使うＢＡＳＩＣの命令は次のとおりです。

ASK、TIME$、DATE$、DAY$、TVPW、CRT、CHANNEL、VOL、SCROLL

1 テレビタイマーコントロール(TV Timer Control)

1.1 TV Timer controlの呼びだし

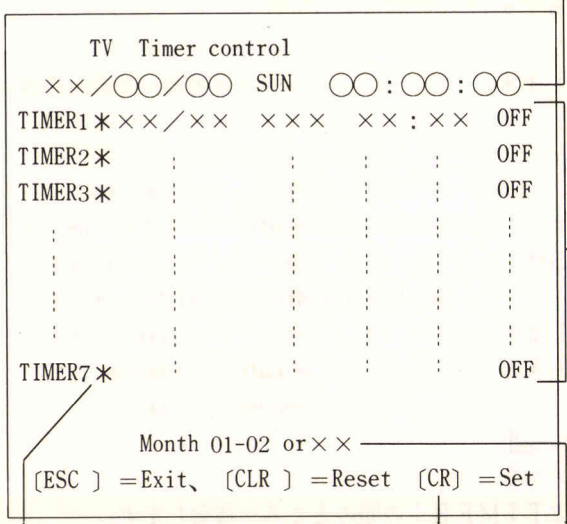
クロックおよびタイマーの設定は本機の電源を入れた直後、ＢＡＳＩＣを読ませない状態では次のようにして行ないます。
ＢＡＳＩＣを読み込まない場合、
専用ディスプレイテレビの画面上に次のメッセージが表示されますので、Ｔキーを押してください。

```
Make your device ready
Press selected key to start driving:
F:Floppy
R:ROM
C:CMT
T:Timer
```

ＢＡＳＩＣを読み込ませてある場合はＡＳＫ↵を入力してください。

1.2 TV Timer control の表示内容

画面は下図のようにかわり、カーソルがＴＩＭＥＲ ××／××…の最初のＸのところで点滅しています。



●クロック表示部（白文字）
××／○○／○○ SUN ○○：○○：○○
年 月 日 曜日 時 分 秒

＊はじめて電源を入れたときは表示が現在の時刻と合っていない場合がありますが、これは故障ではありません。クロックの設定手順通り設定してください。また、メイン電源スイッチを切った時も再度設定ください。

●タイマー設定表示部（赤色）
タイマーの設定が行なわれると
＊（アスタリスク）が表示されます。

●タイマー表示部（白文字）
×× ／ ×× ××× ××：×× ＯＦＦ
月 日 曜日 時 分 ＴＶの入・切表示

●入力モード表示Ａ（黄または赤文字）
カーソルの位置に対応した入力モード内容が黄文字で表示され、入力ミスした場合、赤文字でエラーを知らせる表示がでます。

●入力モード表示部Ｂ（緑文字）

〔ＥＳＣ〕 キーはクロック・タイマー表示を解できます。

注）ＢＡＳＩＣからＡＳＫ↵によってＴＶＴｉｍｅｒ ｃｏｎｔｒｏｌを呼びだした場合はＢＡＳＩＣにもどります。
ＩＰＬより直接 キーを入力した場合はＩＰＬのメニュー画面にもどります。

〔ＣＬＲ〕 CLR HOME キーはタイマー設定した内容を取消せます。

〔ＣＲ〕 ↵ キーは入力した内容でタイマーマーセットできます。

1.3 クロック（時計）を現在時刻に合わせる

■現在時刻にコンピュータ内蔵のクロックを合わせる必要があるときは次の手順に従って設定してください。

ここでは１９８５年 1月１２日土曜午後３時３０分２０秒と仮定して説明します。

1、カーソルをカーソルコントロールキーでクロック表示部の左端に移動させます。

2、西暦１９８５年の末尾２桁[8][5]をキー入力します。

3、１月の[0][1]をキー入力します。

4、１２日の[1][2]をキー入力します。

5、土曜日の[S][A][T]をキー入力します。

6、午後３時（１５時）の[1][5]をキー入力します。時刻は２４時間表示ですから午後３時は１５時になります。

7、午後３時３０分の分の単位[3][0]をキー入力します。

8、午後３時３０分２０秒の秒の単位[2][0]をキー入力します。

9、設定時刻の３時３０分２０秒になった瞬間に[↵]キーを押すとその時点からクロックはカウントをはじめ、クロックの設定は終了しカーソルはＴＩＭＥＲ１の頭のところへ移動します。

入力モード表示内容	
⇩	
Year	00- 99
Month	01-12 or××
Day	01-31 or××
SUN MON TUE WED THU FRI SAT	or×××
Hour	00-23 or××
Minute	00-59
Second	00-59

1.4 タイマーで番組予約をする。

好きな時刻にスイッチをＯＮにする

クロックが動作を始めたところで今度は専用ディスプレイテレビの番組予約をするためのタイマー設定を行なってみましょう。

例１　ここでは、先程クロックを設定した翌日つまり１９８５年 1月１３日、日曜の午前９時３０分からはじまる５チャンネルの１時間番組を番組予約すると仮定して説明します。

１、カーソルはタイマー表示部のＴＩＭＥＲ１　××…の最初の×のところにきているはずですが、そうでないときは最初の×のところへカーソルコントロールキーで移動します。

2、1月の[0][1]をキー入力します。

3、１３日の[1][3]をキー入力します。

4、日曜日の[S][U][N]をキー入力します。

5、午前９時の[0][9]をキー入力します。

6、午前９時３０分の分の単位[3][0]をキー入力します。

7、カーソルはＯＦＦのＯのところで点滅しています。この例では専用ディスプレイテレビをＯＮさせたいわけですから入力モード表示にこたえるかたちで　（ＹＥＳ）をキー入力します。

8、ＯＦＦに変り表示された"ＯＮ　ＣＨ"のうしろでカーソルが点滅していますので予約するチャンネルの[5]をキー入力します。

最後に[↵]キー入力するとこの行のタイマーがセットされたことを示す赤色の＊（アスタリスク）

入力モード表示内容	
⇩	
Month	01-12 or××
Day	01-31 or××
SUN MON TUE WED THU FRI SAT	or×××
Hour	00-23 or××
Minute	00-59
TV Poewr ON？ (Y or N)	
TV Channel 1-12	

がＴＩＭＥＲ１のあとに表示されカーソルは次の行の頭へ移ります。

ＴＩＭＥＲ１＊０１／１３　ＳＵＮ　０９：３０　ＯＮ　ＣＨ５

ここまでが専用ディスプレイテレビを指定された時刻とチャンネルでONされるタイマーセットです。このようにタイマーセットがなされるとコンピュータ本体インジケータ部の**TIMER**表示ランプが点灯し、タイマーが動作中であることを表示します。

1.5 タイマーでスイッチをOFFにする

こんどは専用ディスプレイテレビを指定された時刻にOFFさせるタイマーセットするため、さきほどの手順と同じように

TIMER2*01/13 SUN 10:30 OFF

と入力してもよいのですが「1時間後にOFFさせる」のなら次の方法が簡単です。ただし、この場合毎日午前10：30には専用ディスプレイテレビをOFFするようにタイマーが働きますので、ご注意ください。

9、カーソルはTIMER2 ××…の最初のXのところにきています。

10、カーソルキーでカーソルを順次右へ送り、午前10時30分の部分のみ入力します。

TIMER2××/×× ×××10:30 OFF

11、[N]キーまたは、[↵]キーの入力でセットされカーソルは次の行の頭に移ります。

TIMER2××/×× ×××10:30 OFF

これで専用ディスプレイテレビを指定された時刻にタイマーセットできました。

入力モード表示内容

⇩

Month 01-12 or××

TV Poewr ON? (Y or N)

応用例

例2 毎週火曜日の午前11時から午後2時まで専用ディスプレイテレビをONさせ、10チャンネルを番組予約するときのタイマー表示部がどうなっているかをごらんください。

TIMER3 * ××/×× TUE 11:00 ON CH10

TIMER4 * ××/×× TUE 14:00 OFF

■例1～3の内容を見て、お気付きのように、このタイマーは×（つまり内容を規定しない）を使用することで、プログラマーの思いのままに専用ディスプレイテレビをタイマーコントロールすることができるわけです。

■一度タイマーセットされた内容は一度メイン電源を切るかまたはクリア（タイマーの取消し）をしない限り、働き続けますので、毎日繰返すようなタイマー設定の場合は便利です。

1.6 タイマーを取消す

先程セットしたタイマー内容を取消す場合は次の手順で行ないます。TIMER1からTIMER6を取消すことを例にして説明します。

1、カーソルをTIMER1の行へ移動させます。この場合カーソルはTIMER1の行のどの位置でもかまいません。

2、[SHIFT]キーを押しながら[CLR HOME]キーを押すとTIMER1の内容は取消され表示はタイマー設定前の状態にもどります。

3、カーソルはTIMER2の行に移っていますので同様に[SHIFT]キーを押しながら[CLR HOME]キーを押しTIMER2を取消します。

同じ操作でTIMER3以降、TIMER6まで全部取消すとコンピュータ本体の**TIMER**表示は消えタイマー設定されていないことを示します。

ご注意

クロック表示とメイン電源の入・切について

・はじめてメイン電源を入れたときは、表示が現在時刻とちがっていますが、これは故障ではありません。クロックの設定手順を参照して、正確な時刻に合わせてください。
・また、メイン電源を切った状態で長時間経過すると、クロック機能が働かなくなりますので、再設定が必要です。
・本機のクロックは、年号の自動切換え表示を行ないませんので、年度が改まるごとに新しい年号を設定してください。
・ウルウ年（2月29日のある年）にあたる場合は、月／日の再設定が必要です。
・メイン電源を切りますと年号は××に変わりますので再設定してください。

2 プログラムでテレビをコントロールする

2.1 タイマー制御ステートメント

次のステートメントで内蔵タイマーの設定・表示が行なえます。

TIME$＝"時間：分：秒"

DATE$＝"年／月／日"

DAY$＝"曜日"

次のように入力することで設定・表示が行なわれます。

```
TIME$="12:34:56"↵
Ok
DATE$="85/12/07"↵
Ok
DAY$="FRI"↵
Ok
PRINT TIME$,DATE$,DAY$↵
12:36:03  85/12/07  FRI
Ok
```

2.2　テレビ制御ステートメント

BASICでテレビをコントロールする命令はつぎのとおりです。

TVPW	ON	テレビの電源をつけます
	OFF	テレビの電源を消します

CRT	0	テレビ放送を表示します
	1	コンピュータ画面を表示します
	2	テレビ放送のコントラストを下げてコンピュータ画面を重ねて表示します
	3	テレビ放送とコンピュータ画面を重ねて表示します

＊[SHIFT]＋[+]の状態はCRT2と同じです。

CHANNEL　1～12　チャンネルを設定します

VOL　－62（小さい）～63（大きい）音量を変化させます

SCROLL N　スーパーインポーズ画面のとき（CRT2、CRT3）、コンピュータ画面の上下方向のスクロールを行ないます。
Nの値が正のときは上方向に、負のときは下方向にスクロールします。Nの値が0、省略したときストップします。
ただし　N＝－3から3までの整数

＊ステートメントの詳しい説明は「BASIC　リファレンス　マニュアル」を参照してください。

ではBASICプログラムでテレビをコントロールしてみます。次のプログラムを入力してみてください。

〔例1〕
```
10 TVPW OFF
20 IF TIME$ <> "07:00:00" THEN 20
30 TVPW ON:CHANNEL 1
```

朝7時になるとテレビの電源が入って1チャンネルになります。20行の時刻、30行のチャンネルを変えることで好きな時刻に好きなチャンネルをつけることができます。
これだけでは TV Timer control で設定するのと同じですから、TV Timer control ではできない機能を使って少し工夫したプログラムにしてみます。

〔例2〕
```
10 INIT:WIDTH 40,25,0,1:CLS4
20 CHANNEL 5
30 CRT 2
40 CSIZE3:LOCATE 0,0:PRINT#0 TIME$
50 GOTO 40
```

20行で指定したチャンネルに時間を大きく表示するようにしたプログラムです。

13章
RS-232Cインターフェイスの使い方

ＣＺ－８５１Ｃ／ＣＺ８５２ＣはＲＳ－２３２Ｃインターフェイスを内蔵しており、ＲＳ－２３２Ｃインターフェイスを持っている機器との間でシリアルデータの送・受信ができます。ＣＺ－８５０ＣはＲＳ－２３２Ｃインターフェイスを内蔵していないため、ＲＳ－２３２Ｃボードが必要です。

1 接続方法

(1) 接続例

ＲＳ－２３２Ｃインターフェイス機能を使用して、以下に示すようなシステムのもとで、ＲＳ－２３２Ｃインターフェイス機能を持つ他のコンピュータあるいは周辺機器との間でデータの送・受信ができます。

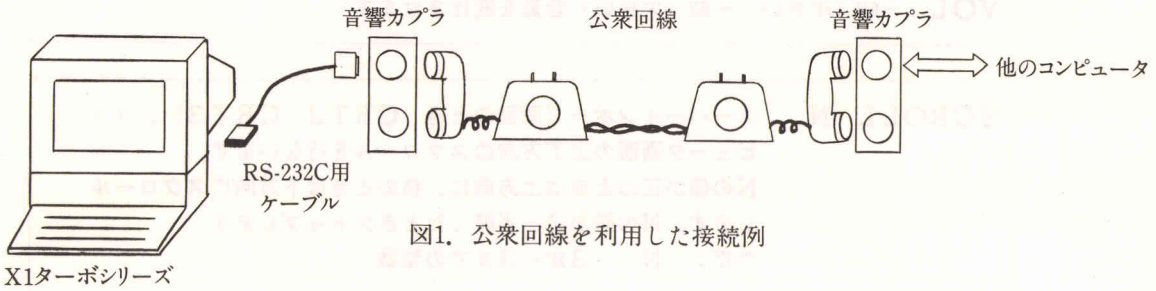


図1. 公衆回線を利用した接続例

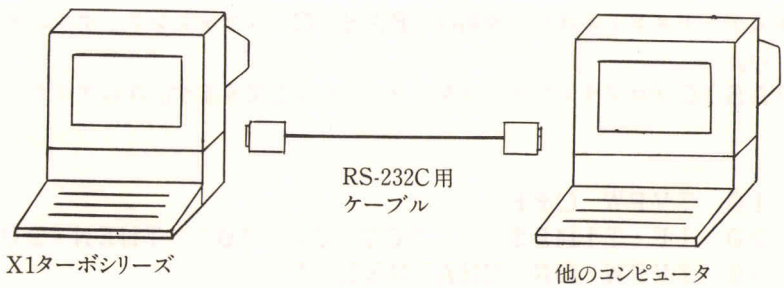


図2. 他のコンピュータとの接続

(2) ＲＳ－２３２Ｃインターフェイス信号端子表

ＲＳ－２３２Ｃインターフェイス用コネクタには２５ピンＤ－sub コネクタのメス形を使用しており、そのピン配置と信号端子は次のとおりです。

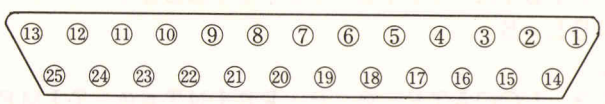


図３．RS-232Cインターフェイスコネクタピン配置

RS－232Cコネクタ端子表

端子番号	信　　号　　名	信号方向	機　　能
1	FG（保安用アース）	←→	保安用アース
2	T×D （送信データ）	→	送信データ
3	R×D （受信データ）	←	受信データ
4	RTS（送信要求）	→	ONにより相手を送信可能とする
5	CTS（送信可）	←	ONの時送信可能と判断する
6	DSR（データセットレディ）	←	ONの時相手が送・受信の準備ができていると判断する
7	SG（信号用アース）	←→	信号用アース
8	CD（キャリア検出）	←	ONの時受信データ有効と判断する
9～14	NC		
15	ST2（送信信号エレメントタイミング）	←	同期式通信時の送信信号エレメントタイミング信号を入力する
16	NC		
17	RT（受信信号エレメントタイミング）	←	同期式通信時の受信信号エレメントタイミング信号を入力する
18，19	NC		
20	DTR（データターミナルレディ）	→	ONにより送・受信の準備ができたことを知らせる
21	NC		
22	CI（被呼表示）	←	ONにより相手から呼び出されていると判断する
23	NC		
24	ST1（送信信号エレメントタイミング）	→	同期式通信時の送信信号エレメントタイミングを出力する
25	NC		

→ OUT
← IN

(3) 接続方法

相手機器との接続には両端に２５ピンＤ－sub コネクタのオスを持つＲＳ－２３２Ｃ用ケーブルを使用します。相手機器がモデム（音響カプラ含む）かコンピュータかによりコネクタ信号の方向が異なります。相手機器によっては使用コネクタまたは信号配置が異なるものもあるため、相手機器の説明書を熟読の上接続してください。以下に一般的な接続例を示します。

図4　モデムや音響カプラと接続する場合のケーブル配線例

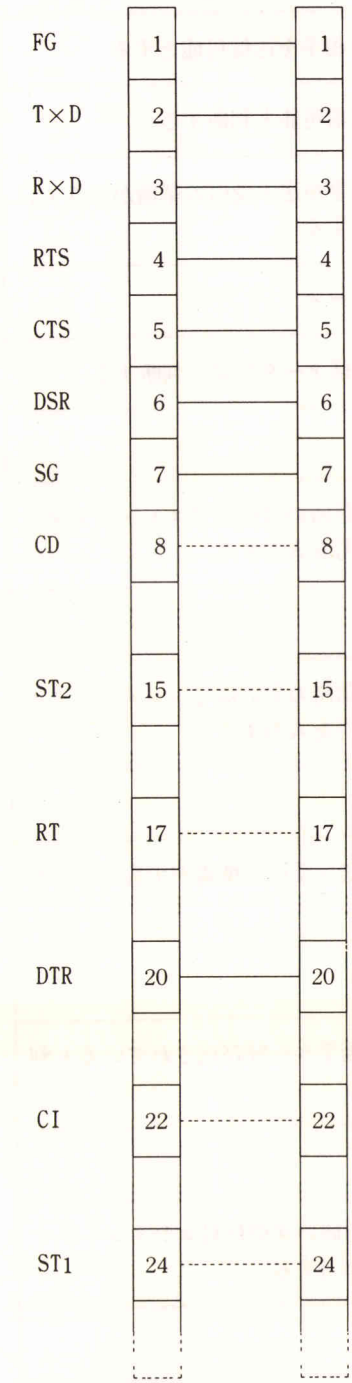
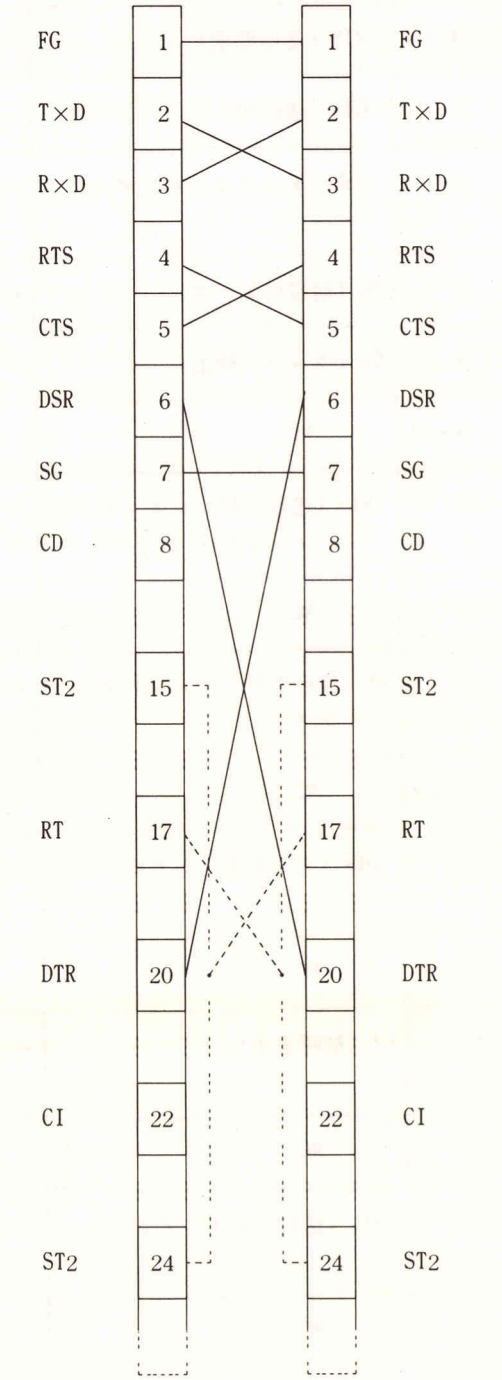


図5　他のコンピュータ（本機を含む）と接続する場合のケーブル配線例



…で示す信号線はＢＡＳＩＣではサポートしておらず特に接続する必要はありません。外部周期を使用する場合など、ユーザーがマシン語レベルでソフトを作成して使用する時に必要に応じて接続します。

2 RS-232インターフェイスのデータ信号フォーマット

ＲＳ－２３２Ｃインターフェイスは、コンピュータからの８ビットパラレルデータをシリアルデータに変換してＲＳ－２３２Ｃ規格で規定された電圧レベル（±12Ｖ）に変換して送信したり、逆に受信したシリアルデータをパラレルデータに変換するものです（図６参照）。シリアルデータの形式は同期式、非同期式通信方式によって異なります。本機のＢＡＳＩＣでは非同期式通信方式をサポートしています。この場合のデータ信号フォーマットを図７に示します。機器間でデータの送・受信を行なう場合にはこのデータ信号フォーマット及びボーレート（信号の伝送速度）を一致させる必要があります。設定方法については次項のＢＡＳＩＣ上でのＲＳ－２３２Ｃインターフェイス取扱方法で説明します。

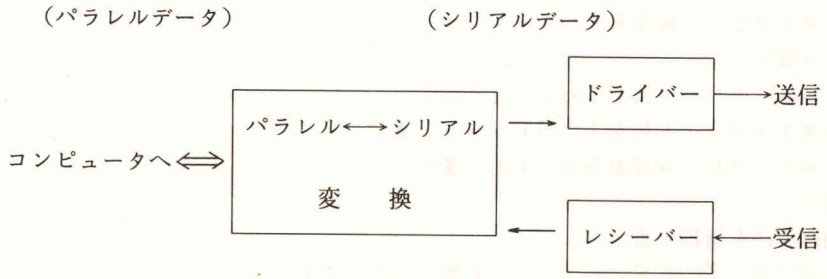


図６　**RS-232C 概念図**

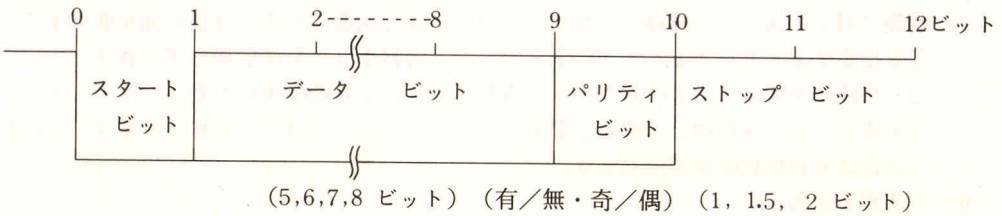


図７　**シリアルデータ　フォーマット**

3 BASICでRS-232Cインターフェイスを取り扱う方法

ＲＳ－２３２Ｃインターフェイスで扱うデータはファイルという概念で取り扱い、ＢＡＳＩＣの入出力命令を使って、一般の入出力装置として使用します。ＲＳ－２３２Ｃインターフェイスのファイルディスクリプタは次の形式で表わされます。

"COM：通信パラメータ"

通信パラメータは次のとおりです。

〈ボーレート〉〈パリティ〉〈データビット長〉〈ストップビット長〉〈通信制御指定〉〈カナの表現方法指定〉〈CR、ＬＦコードの送信処理〉〈CR、ＬＦコードの受信処理〉〈日本語文字列の表現方法指定〉〈エンドコード指定〉

〈ボーレート〉

０～６の数値によりボーレートを設定します。

数　　値	0	1	2	3	4	5	6
ボーレート（ボー）	150	300	600	1200	2400	4800	9600

〈パリティ〉

E……偶数パリティ・チェックを使用します。
O……奇数パリティ・チェックを使用します。
N……パリティ・チェックを使用しません。

〈データビット長〉

5……データビット長を５ビットとします。
6……データビット長を６ビットとします。
7……データビット長を７ビットとします。
8……データビット長を８ビットとします。

〈ストップビット長〉

1……ストップビット長を１ビットとします。
2……ストップビット長を１．５ビットとします。
3……ストップビット長を２ビットとします。

〈通信制御指定〉

通信制御方法を選択します。

X……XON／XOFFコードによる制御を行ないます。
R……RTS制御信号のON／OFFによる制御を行ないます。
{N 省略}…どちらの制御も行ないません。

この通信制御とは、データ送受信時に受信側がデータの受信処理が間に合わないときに、送信側に対して送信の一時停止を要求し、受信できる状態になると送信再開を要求することです。Xを指定するとデータとしてXOFFコード（&H13）を送信側へ送り返すことにより送信の一時停止を要求し、XONコード（&H11）により送信再開を要求します。また、Rを指定すると、RTS制御信号線をOFFにすることにより送信の一時停止を要求し、ONにすることにより送信再開を要求します。

〈カナの表現方法指定〉

S……データ７ビットモードでカナの送受信ができます。
{N 省略}…データ７ビットモードでカナの送受信ができません。

データ７ビットモードでカナの送受信は、シフトアウトコードSO（&H0E）があるとそれ以後のデータはカナとみなし、シフトインコードSI（&H0F）があるとそれ以後のデータを英数字とみなします。

〈CR，LFコードの送信処理〉

C……一連の文字列送信後、CRコード（&H0D）を復帰＋改行コードとして送信します。
{L 省略}…一連の文字列送信後、CRコード（&H0D）＋LFコード（$H0A）を復帰＋改行コードとして送信します。

〈CR　LFコードの受信処理〉

C……１つのCRコード（&H0D）の受信で復帰＋改行コードとして処理します。
{L 省略}…CRコード（&H0D）のみ受信すると、それはデータとして処理され、CRコード（&H0D）とLFコード（&H0A）を連続して受信すると復帰＋改行コードとして処理します。

〈日本語文字列の表現方法指定〉

日本語文字列の表現方法を選択します。

J……漢字インコードKI（&H1B4B）で日本語文字列の始まりを示し、漢字アウトコードKO（&H1B48）で日本語文字列の終わりを示します。

N
省略……シフトJIS漢字コードを使用します。

〈エンドコード指定〉

データ転送の終了を判断するためのエンドコードを指定します。エンドコードを設定するとSAVE命令を実行した場合プログラムデータの送信後指定されたエンドコードを送信してSAVE動作を終了します。LOAD命令を実行した場合、指定されたエンドコードを受信した時点でLOAD動作を終了します。RS-232CファイルをOPEN命令によりアウトプットオープンした場合、CLOSE命令の実行によりエンドコードが送信されます。また、インプットオープンした場合、EOF関数はエンドコードを受信すると真の値となります。

このエンドコードとしてコントロールコード（&H00～&H1F）の中から任意の1つを選択できます。パラメータ値としてはキャラクタコード（&H40～&H5F）対応する文字を使用します（&H60～&H7Fも可）。例えば、D（またはd）と指定すると、エンドコードとしてCTRL-D（&H04）が使用されます。またこのパラメータを省略するとエンドコードの送受信処理は行ないません。

注意　通信パラメータの〈ボーレート〉から〈ストップビット長〉までは省略できませんが、それ以降のパラメータは省略できます。ただし、途中のパラメータを省略して次のパラメータを指定することはできません。

日本語文字の送・受信はデータ長8ビットのときに可能です。ただし、7ビットモードでもSI／SOコード制御及びKI／KOコード制御を使用することにより可能です。この場合のデータ形式は次のとおりです。

SI	KI	漢字	KO	SO	カナ	SI	KI	漢字	KO	SO	

送／受信方向 ⟶

また、KI／KOコード制御にて日本語文字を送・受信する場合は、漢字表示モードKMODE1に設定してください。

4 入出力命令と割り込み処理

4.1 RS-232Cインターフェイスに関する入出力命令

SAVE"COM:通信パラメータ"

メインメモリ上のBASICプログラムをアスキー形式にて、RS-232Cインターフェイスを介して相手機器へ送信します。

LOAD"COM:通信パラメータ"

相手機器より送られてきたアスキー形式のBASICプログラムをRS-232Cインターフェイスを介してメインメモリへ読み込みます。

SAVEM"COM:通信パラメータ",〈開始アドレス〉,〈終了アドレス〉

メインメモリ上のマシン語のプログラムをRS-232Cインターフェイスを介して相手機器への送信します。

この場合の送信データ形式は次のとおりです。

開始アドレス		終了アドレス		
LOW	HIGH	LOW	HIGH	メモリ上のデータ

LOADM"COM：通信パラメータ"

相手機器より送られてきたマシン語プログラムをRS－232Cインターフェイスを介してメインメモリへ読み込みます。

この場合送られてきたデータはSAVEM命令の項に示すデータ形式になっている必要があります。

OPEN"{I/O/C}",〔#〕ファイル番号,"COM：通信パラメータ"

I……インプットオープン（受信用）

O……アウトプットオープン（送信用）

C……コミュニケーションオープン（送・受信用）

RS－232Cファイルを開き（通信パラメータの設定を行うこと）、PRINT# INPUT#命令等でデータの送・受信ができる状態にします。コミュニケーションオープンはRS－232Cファイルについてのみ有効です。コミュニケーションオープンすると同一ファイ

PRINT# ファイル番号,〔X_1,X_2,……〕

X_1,X_2……出力する式または文字列

RS－232Cファイルがアウトプットオープンされている場合、式または文字列を相手機器へ送信します。文字列送信後通信パラメータの〈CR、LFコードの送信処理〉でCが設定されている場合にはC コードがまたLが設定されている場合C +L コードが送信されます。

INPUT# ファイル番号,X_1〔,X_2,……〕

X_1、X_2……入力したデータを入れる変数

RS－232Cファイルがインプットオープンされている場合、相手機器より受信したデータを変数に読み込みます。

LINPUT# ファイル番号, 文字変数

LINE INPUT# ファイル番号, 文字変数

文字変数……入力したデータを入れる文字型変数

RS－232Cファイルがインプットオープンされている場合、相手機器より受信したデータ（255文字まで）を文字変数に読み込みます。通信パラメータの〈CR、LFコードの受信処理〉でCが設定されているとCRコードが、またLが設定されているとCR+LFコードがくるとLINPUT動作を完了します。

WRITE# ファイル番号,〔X_1,X_2,……〕

PRINT#文と同様ですが、データの区切りにはカンマ（,）を送信し、文字列データのときはダブルクォーテーション（"）をつけて、詰めて送信します。

INPUT$（n,〔#〕ファイル番号）

n……読み込む文字数

RS－232Cファイルがインプットオープンされている場合、相手機器より受信したn個の文字がこの関数の値になります。

LOC（ファイル番号）

受信バッファ（64バイト）にたまっている文字数がこの関数の値となります。

EOF（ファイル番号）

受信動作開始時は偽の値（0）がこの関数の値となり、相手機器よりエンドコードが送られてくると真の値（－1）がこの関数の値となります。

CLOSE〔〔#〕ファイル番号1，〔#〕ファイル番号2，……〕

OPENによって開かれたファイルを閉じます。RS－232Cによる送・受信を終了します。アウトプットオープンされていた場合にはエンドコードを送信します。

DEVICE"COM:"

デバイス名を省略をしたときに用いられるデバイス名をCOM:に設定します。LOAD、SAVE等のコマンドにおいてデバイス名を省略すると、デバイス名はCOM:と判断します。

（使用例）DEVICE"COM:"↵
SAVE"6N83XNLLJD"↵
の命令を実行するとRS－232Cポートへプログラムを送出します。

4.2 割り込み処理

INPUT#、INPUT$命令を実行すると、受信データ待ち状態となり、データを受信するまでプログラムは停止します。 ON COM GOSUB命令を使用することにより、この受信待ち時間には他の仕事を実行させておき、データを受信した時点で指定された行番号からの処理ルーチンの実行を開始することができます。この処理ルーチンからの復帰はRETURN命令によって行なわれます。

ON COM GOSUB {行番号 / "ラベル名"}

RETURN [{行番号 / "ラベル名"}]

行番号、ラベル名を省略すると、中断した所から処理を再開します。

また、COM ON／OFF／STOP命令によりRS－232Cからの割り込みの許可、禁止、停止を設定できます。

COM ON ……割り込み許可
COM OFF ……割り込み禁止
COM STOP……割り込み一時保留

〔サンプルプログラム〕

ディスクにあるアスキー形式のプログラムを読み込みRS－232Cインターフェイスより送信します。

```
10 INPUT "File name:",F$
20 F$=LEFT$(F$+SPACE$(13),13)
30 OPEN"I",#1,"1:"+F$
40 OPEN"O",#2,"COM:6N83XNCCND"
50 IF EOF(1)=-1 THEN 100
60 LINPUT #1,D$
70 PRINTD$
80 PRINT#2,D$
90 GOTO 50
100 CLOSE #1,#2
110 END
```

RS－232Cインターフェイスより受信したプログラムをグラフィックメモリに書き込みます。

```
10 INPUT "File name:",F$
20 F$=LEFT$(F$+SPACE$(13),13)
30 OPTIONSCREEN4
40 INIT"MEM:"
50 OPEN"O",#1,"MEM:"+F$
60 OPEN"I",#2,"COM:6N83XNCCND"
70 IF EOF(2)=-1 THEN 120
80 LINPUT #2,D$
90 PRINTD$
100 PRINT#1,D$
110 GOTO 70
120 CLOSE #1,#2
130 END
```

ライン命令実行中にRS－232Cインターフェイスよりデータを受信すると割り込みがかかり、そのデータを画面に表示して前の処理を再開します。

```
10 INIT:CLS4
20 OPEN"I",#1,"COM:6N83XN"
30 ON COM GOSUB 100
40 COM ON
50 FOR I=0 TO 99
60 LINE(0,100+I)-(319,199),PSET,I,BF
70 NEXT
80 COM OFF:CLOSE #1
90 END
100 PRINT INPUT$(LOC(1),1);
110 RETURN
```

5 ユーザーがマシン語でRS-232C用ソフトを作成する場合の使用法

図8にRS－232C周辺のシステム図を示します。データ伝送におけるボーレート（伝送速度）を決定する基本クロックの作成には、CTC（カウンタ／タイマ用LSI）を使用し、シリアルデータ通信用LSIにはSIOを使用しています。RS－232Cインターフェイスを使用してデータ通信を行なう場合にはこのCTCとSIOを通信形体に合わせて設定します。以下、CTC並びにSIOの設定方法について説明します。

(1) CTC（Z－80ACTC）の設定

CTCはカウンタ／タイマ機能を持ち、内部に4個の独立したチャンネル（チャンネル0～3）を持っています。このうちチャンネル1と2の出力をSIOに入力し、データ通信におけるボーレートを決定する基本クロックとしています。（ただし、チャンネル2の出力はマウス用に使用しているのでRS－232Cインターフェイスで使用するのはチャンネル1の出力のみです）CTCの各チャンネルのI／Oアドレスは次のとおりです。

チャンネル0……1FA0（HEX）
チャンネル1……1FA1
チャンネル2……1FA2
チャンネル3……1FA3

CTCはカウンタまたはタイマモードとして使用でき、モードにより分周機能が異なります。ボーレート決定用基本クロックを作るにはカウンタモードで使用し、2MHZ基本クロックを1／1～1／256 まで分周できます。

ＣＴＣを動作させるにはまず、チャンネル制御レジスタにそのチャンネルの使用モード等を決定する制御語（ここでは４５（ＨＥＸ））を設定し、つづいて時間定数レジスタに分周比として０～ＦＦ（ＨＥＸ）を設定します。

```
LD  BC,1FA1H
LD  A,45H
OUT (C),A
LD  A,0DH
OUT (C),A
```

これによりＣＴＣのチャンネル１がカウントを開始し、ボーレート決定用基本クロックをＳＩＯへ出力します。ボーレートと時間定数レジスタ値とは一義的には定まらずＳＩＯの設定により変わります。これについて次のＳＩＯの項で説明します。

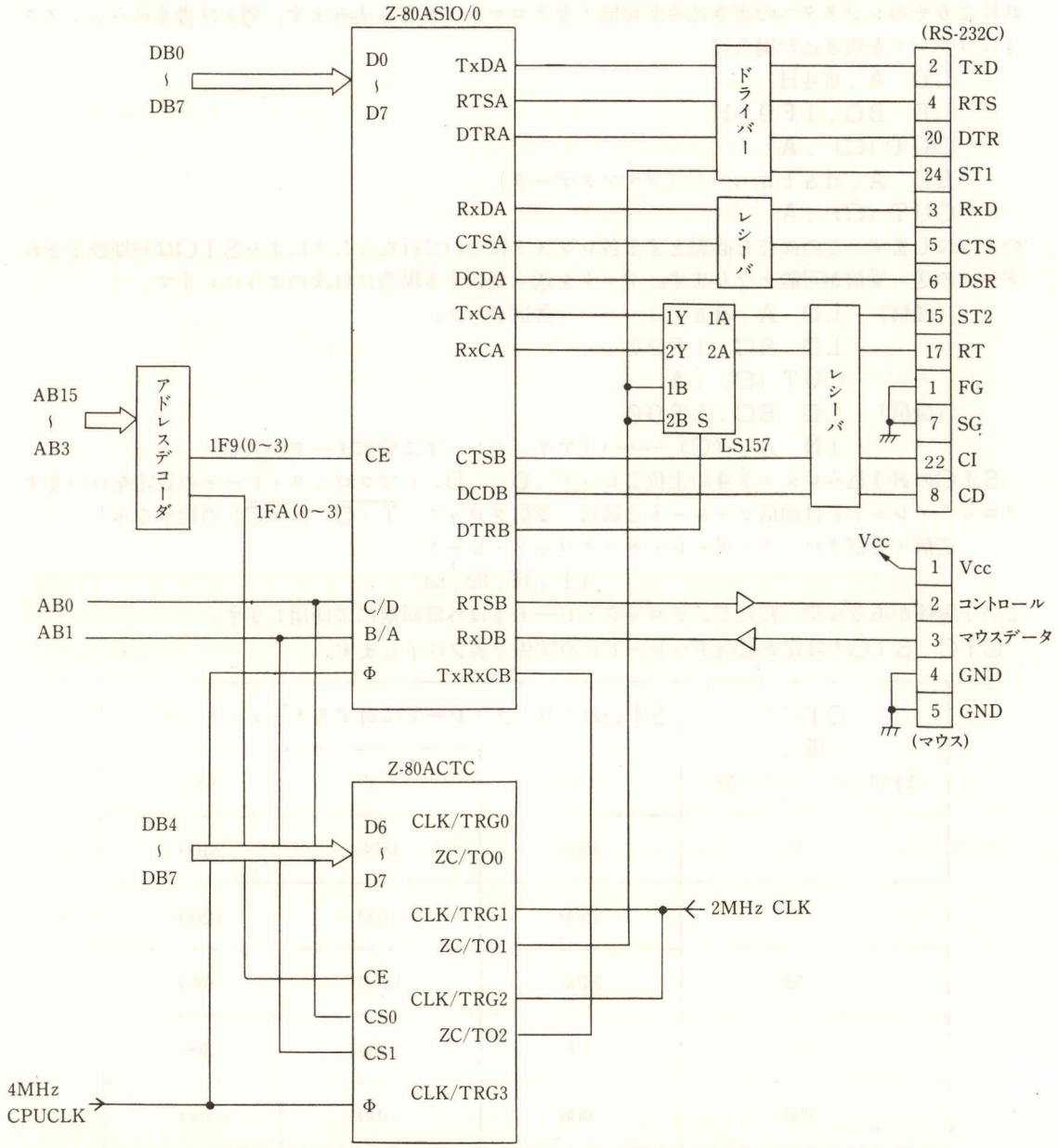


図8. RS-232C周辺回路図

(2) ＳＩＯ（Ｚ－８０Ａ　ＳＩＯ／０）の設定

ＳＩＯはシリアルデータ通信用チャンネルを２つ（チャンネルＡ、Ｂ）持っており、チャンネルＡをＲＳ－２３２Ｃインターフェイス用に、またチャンネルＢをマウス制御用に使用しています。ＳＩＯを動作させるにはまずコマンド書き込みにより通信モードを設定し、ＳＩＯを初期設定した後データの送・受信を行います。ＳＩＯのチャンネル、コマンド／データに対するＩ／Ｏアドレスは次のとおりです。

チャンネルＡのデータ……１Ｆ９０（ＨＥＸ）
チャンネルＡのコマンド……１Ｆ９１
チャンネルＢのデータ……１Ｆ９２
チャンネルＢのコマンド……１Ｆ９３

ＳＩＯにはコマンドの書き込みレジスタを７つ持っています。これらのレジスタにコマンドを書き込む場合には、まず書き込みレジスタ０により実際に書き込みたいレジスタの番号を指定します。これによりそのレジスタへの書き込みが可能となりコマンドを書き込みます。例えば書き込みレジスタ４にコマンドを書き込む場合は

```
LD　A，04H
LD　BC，1F91H
OUT（C），A
LD　A，ｄａｔａ……（コマンドデータ）
OUT（C），A
```

のようにします。この操作を必要とする各レジスタについて行なうことによりＳＩＯは初期設定されデータの送・受信が可能となります。データを送・受信する場合には次のようにします。

```
（送信）　LD　A，ｄａｔａ……（送信データ）
　　LD　BC，1F90
　　OUT（C），A
（受信）　LD　BC，1F90
　　IN　A，（C）……（アキュムレータに受信データが入る）
```

ＳＩＯの書き込みレジスタ４の上位２ビット（D_7，D_6）でクロック・レートの設定を行います。クロック・レートとは通信ボーレートと送信・受信クロック（$\overline{T\times C}$，$\overline{R\times C}$）の比率を示し

送信・受信クロック＝ボーレート×クロック・レート
（１，16，32，64）

という関係があります。ただし、クロック・レート１は外部同期にて使用します。

ＣＴＣ，ＳＩＯの設定と通信ボーレートとの関係を表２に示します。

ＣＴＣの分周比（時間定数レジスタ値）	ＳＩＯのクロック・レートに対するボーレート（ボー）		
	×16	×32	×64
13	9600	4800	2400
26	4800	2400	1200
52	2400	1200	600
104	1200	600	300
208	600	300	150

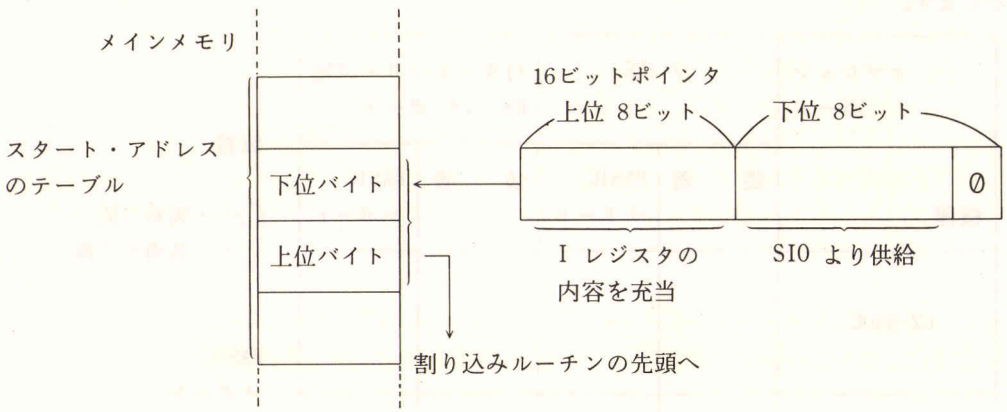
表２　ＣＴＣ、ＳＩＯの設定とボーレートとの関係

RS－232Cインターフェイスを使用してデータの送・受信を行なう場合、一般的にはデータ信号線（T×D，R×D）、コントロール信号線（RTS，CTS，DTR，DSR）とGND線を使用します。これらの信号線はSIOのチャンネルAによりすべて制御できます。本機はこれらの信号線の他にCI（被呼表示）、CD（キャリア検出）を用意しています。この2つの信号線の状態はSIOのチャンネルBの入力ポートにより読み出せます。また、本機は内部同期だけではなく、外部同期にも対応できるように、ST1（送信信号エレメントタイミング）、ST2（送信信号エレメントタイミング）、RT（受信信号エレメントタイミング）も用意しています。内部／外部同期の切り換えはSIOのチャンネルBの$\overline{\text{DTRB}}$出力ポートのHigh／Lowにより行ないます。SIOのチャンネルBはマウス用に使用しているため、$\overline{\text{DTRB}}$出力ポートをHigh／Lowに切り換える時に$\overline{\text{RTSB}}$出力ポートの状態が変化しないように注意してください。

6 割り込み処理の使用

CPUとSIOとの間で割り込み処理（ベクトル割り込み）を使用する場合について説明します。

この割り込み処理を使用する場合、まえもってプログラムにより割り込みルーチンのスタート・アドレス（2バイト）のテーブルをメモリーの適当な位置（偶数アドレスから下位バイト、上位バイトの順）に配置しておきます。CPUは割り込みを受け付けたとき16ビットのポインタで、必要な割り込みルーチンのスタート・アドレスをプログラムカウンタに読み込み、そのアドレスへジャンプします。このポインタ（割り込みスタート・アドレスのある位置を示している）として上位8ビットにはIレジスタの内容を充当し、下位8ビットは割り込みアクノリッジ期間にSIOより割り込みベクトル（プログラムにより設定されていること）が自動的に供給されます。



本機のBASICを起動すると、Iレジスタには&HF8が設定されF830～F83B番地を割り込みスタート・アドレスのテーブルとして使用できます。

7 RS-232Cボードについて

CZ－850CはRS－232Cボードを使用することによりBASIC命令でRS－232Cインターフェイス、マウスが使用できます。またこのボード上の切換スイッチをCZ－851C，CZ－852Cのモードに切り換えて、CZ－851C，CZ－852Cに装着するとRS－232Cインターフェイス、マウス機能を2チャンネル使用できます。ただし、この場合のオプション・ボード用のソフトはユーザーが作成する必要があります。この場合のI／Oアドレスは次のようになります。

本体内	CTC		SIO	
	チャンネル0	1FA0（HEX）	チャンネルAのデータ	1F90（HEX）
	チャンネル1	1FA1	チャンネルAのコマンド	1F91
	チャンネル2	1FA2	チャンネルBのデータ	1F92
	チャンネル3	1FA3	チャンネルBのコマンド	1F93
オプション				
ボード内	CTC		SIO	
	チャンネル0	1FA8（HEX）	チャンネルAのデータ	1F98（HEX）
	チャンネル1	1FA9	チャンネルAのコマンド	1F99
	チャンネル2	1FAA	チャンネルBのデータ	1F9A
	チャンネル3	1FAB	チャンネルBのコマンド	1F9B

X1ターボシリーズにX1用RS－232Cインターフェイスカード（CZ－8RS）を装着できますが、RS－232Cに関するBASIC命令でこのボードを動作させることはできません。各機種に対するRS－232Cインターフェイスボードの装着とBASICのサポートとの関係を図9に示します。

オプションボード / 機種	CZ-8RS		X1ターボシリーズ用 RS-232C ボード	
	装着	BASIC サポート	装着	BASIC サポート
CZ-850C	○	×	○	○
CZ-851C CZ-852C	○	×	○	× 内蔵RS-232Cのみサポート
X1シリーズ	○	×	○	×

装着

○……装着可能
×……装着不可能

BASIC
サポート

○……BASIC 命令にてサポートしている
×……BASIC 命令にてサポートしていないため、ユーザーがソフトを作る必要がある

図9　各機種に対するRS－232Cインターフェイスボードの装着とBASICのサポートとの関係

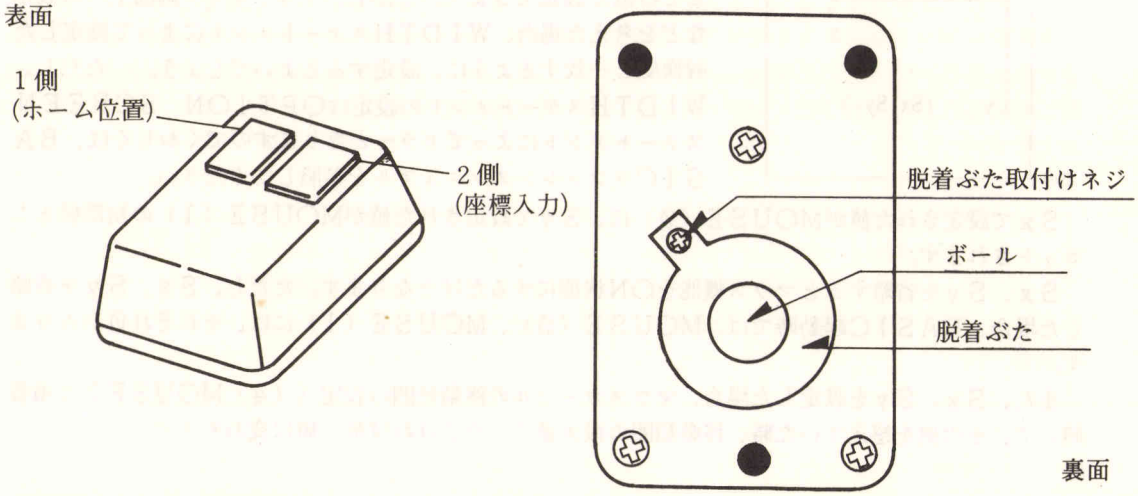
14章
マウスインターフェイスの使い方

本機（CZ－851C／CZ－852C）ではマウスインターフェイスが内蔵されており、コネクタもコンピュータ本体の前面トビラ内と後面の2ヵ所に用意されています。

1 マウスとは

本機用のマウスのハードウェアは、機械式に属します。

マウス裏面（ボタンのついた面の反対面）の中央に、大きな金属のボールがあり、そのボールがマウスの移動により回転し、その回転が本体内のローラに伝わり回転させます。そしてローラは、ロータリーエンコーダに回転を伝えます。これにより、ロータリーエンコーダではマウスの運転量に応じた信号を発生します。



マウスは、ローラとロータリーエンコーダがX軸（横）、Y軸（たて）方向用に1組ずつ用意されています。したがって、マウスの移動量が、X方向成分、Y方向成分に分けられ、X、Y座標の相対座標を表わして、そのため座標入力装置〈ポインティング　デバイス〉とも言われます。

また、マウスには2つのボタンがついています。

このボタンは、機能の選択や指示などセレクト用としてよく使われます。また、BASICでもマウス関数でサポートしています。ゲーム用として使う場合にはジョイスティックのトリガーボタンと同じ用途にも使えます。

2 マウスを使う

本機（CZ－851C／CZ－852C）では、DISK　BASICに、マウス関数が用意されています。以後この関数の使い方を中心に説明します。

2.1 マウス関数の書式と機能について

a．機能の設定ステートメント

(1)

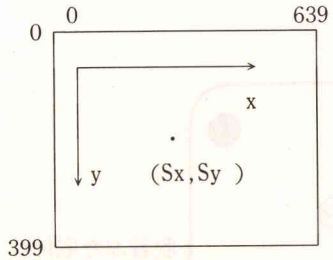
MOUSE 0

マウス機能を〇ＦＦ状態にします。マウスを使用しないのにマウス機能を〇Ｎ状態にしておくと、割り込み注１）の関係で処理スピードが遅くなります。ですからマウスを使わない時は、このステートメントを実行してください。

注１）ここでいう割り込みとは、ある仕事をしていてその途中で別の仕事をし、また元の仕事に戻る、といったことを指します。すなわちマウス機能が〇Ｎ状態だとすると、マウスを使う、使わないに関係なくマウスの状態を見に行くということです。別の実行をしているとき途中でマウスの状態を見に行くために実行が止まり、その分時間がかかってしまうのです。

(2)

MOUSE１，Ｓｘ，Ｓｙ



マウス機能を〇Ｎ状態にし、初期座標として、画面座標の座標（Sx,Sy ）を設定します。Ｓｘ、Ｓｙの値は、－32768 ～ 65535までの値が設定できます。しかし、グラフィック画面上への表示などを考えた場合、ＷＩＤＴＨステートメントによって設定した解像度と一致するように、設定するとよいでしょう。　ただし、ＷＩＤＴＨステートメントの設定は〇ＰＴＩＯＮ　ＳＣＲＥＥＮステートメントによってエラーとなりますのでくわしくは、ＢＡＳＩＣリファレンスマニュアルを参照してください。

Ｓｘで設定された値がＭＯＵＳＥ（０）に、Ｓｙで設定された値がＭＯＵＳＥ（１）の初期値としセットされます。

Ｓｘ、Ｓｙを省略するとマウス機能を〇Ｎ状態にするだけとなります。ただし、Ｓｘ、Ｓｙを省略した場合、ＢＡＳＩＣ起動時では、ＭＯＵＳＥ（０）、ＭＯＵＳＥ（１）には、それぞれ０が入ります。

また、Ｓｘ、Ｓｙを設定した場合、マウスカーソルの移動範囲の設定（（４）ＭＯＵＳＥ３の項参照）で、その値を超えていた時、移動範囲の最大値に、小さければ最小値に変わります。

(3)

MOUSE ２，ｄ，ｒ

Ｘ（横）方向、Ｙ（たて）方向のマウスカーソルの移動比率を設定します。ｄは、マウスカーソルの移動方向を指し、ｄ＝０では、Ｘ（横）方向、ｄ＝１では、Ｙ（たて）方向の移動比率の設定となります。移動比率ｒは、１～３２の値を設定できます。ｒの値が大きくなるほど、マウスの移動に対する座標の変化は小さくなります。それは、マウスの移動量（マウスから送られてくる値）を移動比率で割っているためです。つまりｒの値が大きくなるほど、ＭＯＵＳＥ（０）、ＭＯＵＳＥ（１）、に加えられる値が小さくなり、座標の変化が小さくなります。

ＢＡＳＩＣ起動時では、Ｘ、Ｙ方向の移動比率ｒはそれぞれ１０の値です。

(4)

MOUSE ３，Ｓｘ１，Ｓｙ１，Ｓｘ２，Ｓｙ２

マウスカーソルの移動範囲を設定します。

座標（Ｓｘ１，Ｓｙ１）と（Ｓｘ２，Ｓｙ２）を結ぶ線を対角線とする四角形で囲まれた領域をカーソル移動範囲とします。グラフィック画面との関係からＷＩＤＴＨステートメントによって設定した解像度と一致させるとよいでしょう。

ＢＡＳＩＣリファレンスマニュアルのＷＩＤＴＨ、ＯＰＴＩＯＮ　ＳＣＲＥＥＮを参照してください。

また、(2)の項のＭＯＵＳＥ１，Ｓｘ，Ｓｙで設定された座標（Ｓｘ，Ｓｙ）が、この移動範囲からずれていた場合、移動範囲にいちばん近い座標となります。実際は、移動範囲を出ていた方の値が移動範囲の最大値または最小値となります。

座標（Ｓｘ１，Ｓｙ１）と（Ｓｘ２，Ｓｙ２）を設定するときは、Ｓｘ１＜Ｓｘ２，Ｓｙ１＜Ｓｙ２でなくてはなりません。

ｂ．状態の読み出し関数

(1)

```
MOUSE（0）
```

マウスカーソルの現在のＸ（横）座標をこの関数の値として返します。

(2)

```
MOUSE（1）
```

マウスカーソルの現在のＹ（たて）座標をこの関数の値として返します。

(3)

```
MOUSE（2，b）
```

指定されたボタン番号ｂのボタンが押されているときに、－１の値をこの関数の値として返し、押されていないときは、0の値を返します。

ボタン番号ｂとは、マウスのケーブルの出ている方向を上にして、

ｂ＝１で左側のボタンが押されている時のみ関数は―１

ｂ＝２で右側のボタンが押されている時のみ関数は―１

となり条件を満たさない場合この関数は0となります。

(4)

```
MOUSE（3，b）
```

ボタン番号ｂのボタンが押された時のマウスカーソルのＸ（横）座標をこの関数の値として返します。

(5)

```
MOUSE（4，b）
```

ボタン番号ｂのボタンが押された時のマウスカーソルのＹ（たて）座標をこの関数の値として返します。

(6)

```
MOUSE（5，b）
```

ボタン番号bのボタンが離された時のマウスカーソルのX（横）座標を、この関数の値として返します。もちろんボタンは、押されていた必要があります。

(7)

MOUSE（6，b）

ボタン番号bのボタンが離された時のマウスカーソルのY（たて）座標をこの関数の値として返します。

(8)

MOUSE（7）

最後にこの関数が呼び出されてから、次にこの関数が呼び出されるまでにマウスカーソルが動いたX（横）方向の移動距離をこの関数の値として返します。つまり、最後にこの関数が呼び出された時の座標と、次のこの関数が呼び出された時の座標のX座標の差がこの関数の値となります。

(9)

MOUSE（8）

最後にこの関数が呼び出されてから、次にこの関数が呼び出されるまでにマウスカーソルが動いたY（たて）方向の移動距離をこの関数の値として返します。つまり最後にこの関数が呼び出された時のマウスカーソルの座標と次にこの関数が呼び出された時のマウスカーソルの座標のY座標の差がこの関数の値となります。

2.2 マウス関数を動かす

次のプログラムを入力してみてください。

```
10 INIT:WIDTH 80,25,0,0
20 MOUSE1,0,0
30 MOUSE2,0,5
40 MOUSE2,1,12
50 MOUSE3,0,0,639,199
60 X=MOUSE(0):Y=MOUSE(1)
70 S1=MOUSE(2,1):S2=MOUSE(2,2)
80 PRINT"X= ";X,"Y= ";Y,"SW1= ";S1,"SW2=
";S2
90 GOTO 60
```

RUN↵してみますと、下の様に表示されます。

```
X=  0   Y=  0   SW1=  0   SW2=  0
```

ここで、マウスを動かして見てください。

X＝　とY＝　の後の数値が変わります。

ではボタンを押したり離したりして見てください。

左側のボタンを押すと、SW1＝　－1の表示が、右側のボタンを押すと、SW2＝　－1の表示が現われ、離すとそれぞれ－1が0になることがわかります。

では、プログラムを止めてLIST↵を入力してください。

プログラムの止め方は、SHIFT＋BREAKキーとしてください。

プログラムの説明をします。10行目は、画面を初期化し、画面座標系を 640×200 ドットの画面に設定します。専用ディスプレイテレビ以外のモニターをお使いの方は本機の標準／高解像度切換スイッチで合せてください。プログラムを直す必要はありません。

20行目は、マウス機能をON状態にし、初期座標を（0，0）に設定しますので最初は、X＝0、

Y＝0を表示します。

30行目は、X（横）方向の移動比率を5と設定します。

40行目は、Y（たて）方向の移動比率を12と設定します。

30行目と40行目の移動比率（1～32の値）をいろいろ変えてみてください。違いがわかります。

50行目は、マウスカーソルの移動範囲を（0，0）－（639，199）に設定します。プログラムを実行させた時、マウスを動かしてみてください。移動の範囲の値しかとれません。大きくなったり、小さくなったりはしません。

60行目は、変数Xにマウスカーソルの X（横）座標の値を変数Yにマウスカーソルの Y（たて）座標の値を代入します。

70行目は、変数S1に左側のボタンの状態を変数S2に右側のボタンの状態を代入します。どちらもボタンが押されていると－1の値が、離されているとき0の値が入ります。

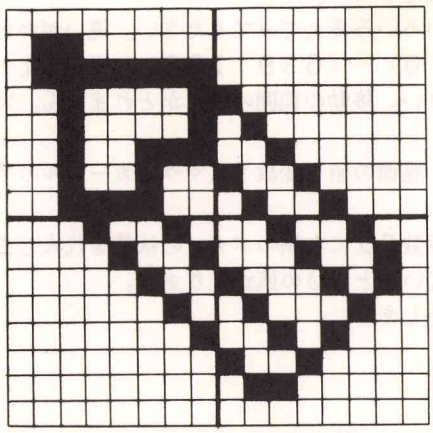
80行目は、それぞれの変数と変数の値の表示をします。

90行目は、60行目にジャンプします。

2.3 マウスカーソルを表示させる

本機ではマウスカーソルを表示するには、そのためのプログラムの入力が必要です。次に示すのがそのサンプルプログラムです。図1の様なエンピツ状のマウスカーソルを表示し、マウスの左のボタンを押しながらマウスを動かすと線を引き右のボタンで画面をクリアするというものです。

```
10 INIT:PALET:OPTION SCREEN0
20 WIDTH 80,25,1,0
30 LINE(0,0)-(7,7),XOR,BF,HEXCHR$("000000
606060 7F7F7F 212121 212121 272727 242424
3C3C3C")
40 LINE (8,0)-(15,7),XOR,BF,HEXCHR$("00000
0 000000 000000 808080 404040 202020 90909
0 484848")
50 LINE (0,8)-(7,15),XOR,BF,HEXCHR$("12121
2 090909 040404 020202 010101 000000 00000
0 000000")
60 LINE (8,8)-(15,15),XOR,BF,HEXCHR$("2424
24 121212 8A8A8A 444444 282828 909090 6060
60 000000")
70 DIM A(50)
80 GET@(0,0)-(15,15),A,7
90 X=0:Y=0:XX=0:YY=0
100 MOUSE1,0,0
110 MOUSE2,0,5
120 MOUSE2,1,4
130 MOUSE3,0,0,624,384
140 PUT@(X,Y)-(X+15,Y+15),A,XOR,7
150 X=MOUSE(0):Y=MOUSE(1)
160 IF MOUSE(2,1)=-1 THEN LINE(XX,YY)-(X,
Y),PSET
170 PUT@(X,Y)-(X+15,Y+15),A,XOR,7
180 XX=X:YY=Y
190 IF MOUSE(2,2)=-1 THEN CLS 0:GOTO 150
200 GOTO 140
```



データ

```
0 0   0 0
6 0   0 0
7 F   0 0
2 1   8 0
2 1   4 0
2 7   2 0
2 4   9 0
3 C   4 8
1 2   2 4
0 9   1 2
0 4   8 A
0 2   4 4
0 1   2 8
0 0   9 0
0 0   6 0
0 0   0 0
```

図1　マウスカーソルの形状例とデータ（16進数）

プログラムの説明をします。

10行目は、画面の初期化です。

20行目は、グラフィック画面の解像度を 640×400 ドットにします。ですから専用ディスプレイテレビまたは高解像度ディスプレイをお使いでない方はＷＩＤＴＨ　８０，２５，０，０に直してください。ただし、マウスカーソルの形状は画面のたて、横比の関係で少しゆがみます。

30～60行目は、マウスカーソルの形状をグラフィック画面の座標（０，０）と（15，15）を対角点とする四角形の中に表示します。実際は見えません。

70行目は、配列Ａを51個用意します。

80行目は、先ほどグラフィック画面に表示したマウスカーソルの形状を配列Ａに読み込みます。

90行目は、変数の初期値を０とします。

100 行目は、マウス機能をＯＮ状態にし、初期座標を（０，０）に設定します。

110 行目は、Ｘ（横）方向の移動比率を５と設定します。

120 行目は、Ｙ（たて）方向の移動比率を４と設定します。

130 行目は、マウスカーソルの移動範囲を（０，０）－（ 624，384 ）に設定します。ここで、注意して頂きたいのは、グラフィック画面の座標（０，０）－（ 639，399 ）と一致していない点です。これは、ＰＵＴ@ステートメントの有効範囲がこの場合、Ｘ方向 639、Ｙ方向 399となっており、140 行目のステートメントがＸ＋15、Ｙ＋15になっています。マウスの座標　Ｘ、Ｙ　が（ 625，385 ）を超えると、ＰＵＴ@の有効範囲を超えてエラーとなります。20行目でＷＩＤＴＨ　80，25，０，０とした方は、ＭＯＵＳＥ　３，０，０，624　，184 としてください。

140 行目、170 行目は、80行目で読み込んだ図形を表示します。

150 行目は、Ｘにマウスのｘ座標をＹにマウスのＹ座標を代入します。

160 行目は、マウスの左のボタンが押されていると線をひきます。

180 行目は、ＸＸにＸの値をＹＹにＹの値を代入します。

190 行目は、マウスの右のボタンが押されると画面をクリアします。

200 行目は、 140行目にジャンプします。

3　使用上の注意事項

本機にマウスを接続する場合、コンピュータ本体の前面トビラ内と後面の２箇所のコネクタに同時に取り付けて実行することはおやめください。正常な動作をしません。

15章
配列

コンピュータでデータを扱う場合、そのデータがプログラムの実行によって変化するような場合、変数というものを使います。

変数はプログラムで使われるデータの記憶場所で、その内容により数値変数とか文字変数と呼びます。

データの数が少ない場合には使われる変数の数は少なくてすみますが、100 個、1000個…というようにデータの数が増えると、いくつもの変数を用意しなくてはならず、プログラムが複雑になってしまいますし、メモリのムダも多くなります。

そこで、このように扱うデータ量が多くなった場合には、配列を使用すると便利です。

配列というのは一組のデータに番号を付けて規則的に並べたもので、以下に例をあげて説明します。

たとえば、郵便屋さんが、あるアパートに手紙を配達する場合を考えてみてください。そして、そのアパートが５階建てで、各階に10件ずつの家があるとします。

さて、ここで　各家庭に手紙を配達しようとしたとき、手紙の宛名をみながら一軒一軒手紙を配っていくと、階段を何度も上ったり降りたりして、たいへん効率の悪い配り方となります。ところが、この手紙は２階の３軒目の家の手紙というようにそれぞれの手紙を何階の何軒目という具合に整理してから配達すると、時間も短くて済み、効率が良く配達できます。

コンピュータでデータを扱う場合も同様で、データの数が増えるほど一組にまとめて、番号順に並べて整理をした方が、扱いやすいものとなります。

このようにデータに一連の番号を付けて、並べたものを配列といいます。

1 配列名と添字

配列は変数の集まりですので名前を付けて扱います。この名前を配列名といい、使用文字や長さは一般の変数と同様です。

配列で使用する一連の番号は（　）でくくり、その配列上の位置を示しており、添字と呼びます。

つまり、配列データを参照する場合は、配列名と添字を指定します。

また、配列の個々のデータを配列要素といいます。

配列データの書き方は次のようにします。

配列名（添字）

たとえばＡという配列名で、５個のデータを格納する場合次のようになります。

Ａ（５）

A(1)	A(2)	A(3)	A(4)	A(5)

さらに、添字はカンマで区切ってＢ（３，３）のように並べることができます。

この例では、３×３＝９個のデータを格納できます。

Ｂ（３，３）

B(1,1)	B(1,2)	B(1,3)
B(2,1)	B(2,2)	B(2,3)
B(3,1)	B(3,2)	B(3,3)

B(3,3)とした場合　添字が2個ありますので、2次元配列といいます。
添字が3個ある場合には3次元配列、n個ある場合はn次元配列といいます。
次元は一行で扱える文字数との関係から125 次元程度まで作成できます。

2 配列宣言

配列を使用する場合は、あらかじめ使用するデータ領域を確保する必要があります。
これを配列宣言といいDIMステートメントを使います。
DIMステートメントは次のように書きます。

DIM 配列名(大きさ、大きさ、・・・・・・・・・)

ここで大きさと書いたのは添字の最大値のことで、この値によって指定された大きさの記憶場所を確保します。
また配列宣言は省略することも可能です。
配列宣言を省略して配列変数を使った場合は

```
DIM 配列名(10)
```

と宣言したのと同じになります。

3 OPTION BASE

配列の添字の最小値は0か1です。
この値を決めるには OPTION BASEというステートメントを使用します。

OPTION BASE {0 / 1}

```
OPTION BASE 0
```

とすると添字の最小値が0となり

```
OPTION BASE 1
```

とすると添字の最小値は1となります。
OPTION BASEを実行する場合は配列変数があらかじめすべて消去されている必要があります。
もし配列変数が存在した状態で OPTION BASEステートメントを実行しようとすると
Duplicate definition のエラーとなります。

4 配列変数の消去

DIMステートメントで宣言した配列を消去するにはCLEARステートメントもしくは、ERASEステートメントを使用します。
CLEARステートメントを実行すると、変数はすべて消去されますが、このとき、配列変数もす

べて消去されます。
ERASEステートメントは指定した配列変数の内容のみ消去し、次のように使います。

ERASE 配列名、配列名、……

5 VDIM

DIMステートメントで宣言される配列変数は、メインメモリ上に記憶領域を確保します。
したがって、大量の配列を宣言すると、プログラムテキストとして使用する部分が少なくなり、out of memory エラーの原因となります。
そこで、大量のデータを扱う場合には、グラフィックVRAMの一部を変数領域として確保し、メインメモリになるべく大きなプログラムテキストを書く方法が便利です。
この場合、OPTION SCREEN 2 もしくは3で、グラフィックVRAMを、変数エリアとして使用するようにあらかじめ定義しておきます。
前記のスクリーンモードで、グラフィックVRAMに配列変数（もしくは通常の変数）領域を確保する命令がVDIMステートメントです。
VDIMステートメントは

VDIM 配列名（大きさ、大きさ……）

のように書きます。
VDIMステートメントで宣言された配列変数は記憶領域がグラフィックVRAMに確保されること以外はDIMステートメントで宣言された配列変数と同様の使い方をします。
ただし、VDIMステートメントで宣言された配列変数はCLEARステートメントでは消去されません。
消去するには、VDIM CLEARというステートメントを用います。
また、ERASEステートメントはDIMの場合と同様に、VDIMで宣言された配列変数を消去することができます。
配列を使用した例

```
10 INIT:CLS
20 OPTION SCREEN1
30 WIDTH 40,25
100 DIM A$(255,1)⇦配列データをメインメモリ上ではなくグラフィック
    RAM上にとる場合は、100行のDIM A$(255,1)をVDIM
    A$(255,1)に変更します。
110 FOR I=0 TO 255
115 CLS:LOCATE 0,0:PRINT HEX$(I)
120 FOR J=0 TO 1
130 A$(I,J)=CGPAT$((J+1)*256+I)
150 PRINT
160 PRINT #0,A$(I,J):PAUSE5
170 NEXT J
180 NEXT I
```

この例は2次元配列を利用したプログラムです。
100行で、A$という配列を宣言しています。130行でキャラクタゼネレータのデータを読み出し、配列の中に入れています。配列 A$(I,J)のうち J=0ときROMのデータが入り、J=1のときRAMのデータが入ります。160行で読み出した内容を表示しています。
以上の操作を256回くり返しています。

配列データをメインメモリ上ではなくグラフィックRAM上にとる場合は、
100行のDIM A$(255,1)をVDIM A$(255,1)に変更します。

16章
デジタルテロッパーの使い方

（CZ－851C、CZ-852C をお持ちの方のみお読みください）

1 概要

CZ－８５１C／CZ－８５２Cでは専用ディスプレイテレビ（CZ－８５０D）との組み合わせ、あるいは家庭用テレビとの組合わせにより、コンピュータ画像とテレビやビデオ画像との重ねあわせくスーパーインポーズ〉画像を表示でき、またそれらの映像を自由にコントロール編集しＶＴＲに録画することができます。

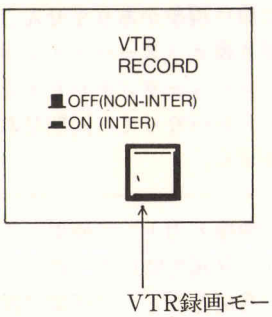
2 特長

①Ｒ．Ｇ．Ｂ．信号のみの信号およびスーパーインポーズ信号をＮＴＳＣの標準複合映像信号に変換するのでコンピュータ画像やスーパーインポーズ画像をＶＴＲに録画することができます。

②接続機器（テレビ、ＶＴＲ、ビデオディスク、ビデオカメラなど）の映像信号とＲ．Ｇ．Ｂ．信号を同期させ、一般家庭用テレビ（ビデオ入出力端子のついたもの）にスーパーインポーズ画像をさらに、コンピュータ画像（ＮＴＳＣ信号に変換されたもの）を表示することができます。

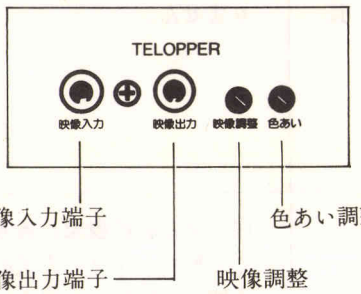
3 各部名称

前面トビラ内



VTR録画モードスイッチ

後面



映像入力端子

映像出力端子

映像調整

色あい調整

ＶＴＲ録画モードスイッチ

コンピュータ画像をＶＴＲに録画するときはスイッチをＯＮ（インターレースモード）にします。詳しくは「ＶＴＲ録画モードスイッチについて」を参照してください。

・映像入力端子

接続機器（テレビ、ＶＴＲ、ビデオディスク、ビデオカメラなど）からの映像信号を入力する端子

・映像出力端子

コンピュータ画像（ＮＴＳＣ信号に変換されたもの）、スーパーインポーズ画像、ビデオ画像の出力を行なう端子

・映像調整、色あい調整

映像出力端子より出力されるコンピュータ画像の調整ボリュームですが、あらかじめ　標準状態に設定してあります。調整したい場合は販売店にご相談ください。

4 VTR録画モードスイッチ

本機前面トビラ内にはＶＴＲ録画モードスイッチがありますが、これには２つのモードがありますがこのモードについて説明します。

このスイッチは　コンピュータ画面にしたとき使用し、その画像を録画する場合にはＯＮにし、録画しない場合は　ＯＦＦとします。

なお、テレビ（またはビデオ）画面、スーパーインポーズ画面にした場合は、自動的にインターレースモードになり、また高解像度ディスプレイモードにした場合、自動的に　ノンインターレースモードになりますのでスイッチ操作をする必要はありません。

	ＶＴＲ録画モードスイッチ　ＯＦＦ（ノンインターレースモード）	ＶＴＲ録画モードスイッチ　ＯＮ（インターレースモード）
使用目的	・コンピュータ画像をＶＴＲに録画しないとき（プログラム作成時など）はＯＦＦにしておきます。	・コンピュータ画像をＶＴＲに録画するときはＯＮにします。
出力信号形　態	・コンピュータ本体内部で発生する同期信号を使って出力するのでＮＴＳＣ標準複合信号から少しはずれた信号です。（内部同期）	・コンピュータ本体へ入力する映像信号の同期信号を使って出力するので正規のＮＴＳＣ標準複合信号です（外部同期）
映像入力	・本機の映像入力端子への入力とは無関係	・本機の映像入力端子へＮＴＳＣ信号の入力が必要。
特　長	・ノンインターレース走査のため映像の垂直ゆれ（フリッカー）がありません。	・正規のＮＴＳＣ標準複合信号だからＶＴＲに録画が可能です。 ・白文字に着色現象がありません。 ・モード切り換え（スーパーインポーズ／コンピュータ／テレビまたはビデオ）を行なっても同期乱れがありません。
	・白文字に着色現象があります。 ・くし形フィルター方式のテレビと本機とを接続した場合には、一般のテレビよりもコンピュータ画面に、にじみがでます。	・ＶＴＲを映像入力として使用した場合、ＶＴＲ再生時以外（停止、早送り、巻戻しなど）で画像に乱れが生じることがあります。

5 使用方法

本機内蔵のデジタルテロッパーにより映像出力端子から次の３種類の信号が出力されます。

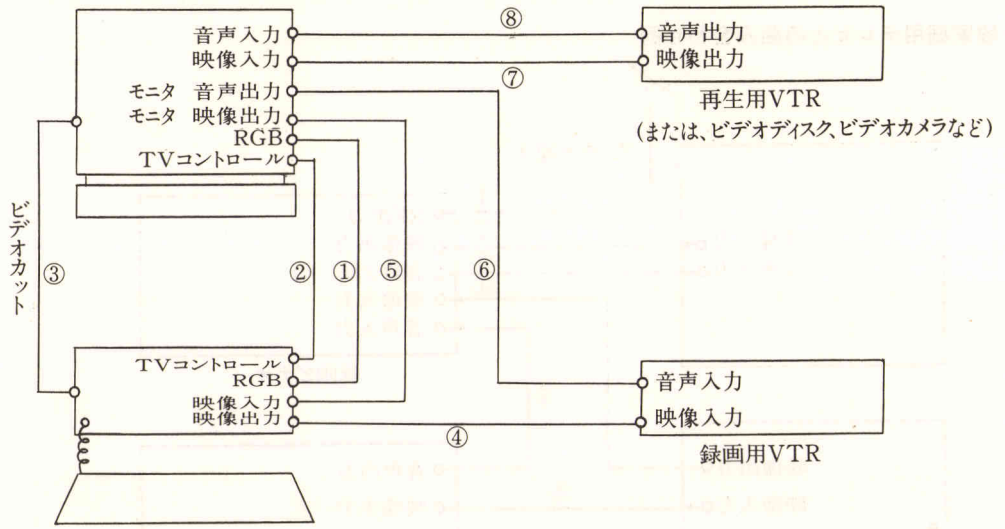
①ＮＴＳＣ信号に変換されたコンピュータ画像の出力。（ノンインターレースモードとインターレースモードの２種類あります。）

②映像入力端子に入力されたＮＴＳＣ信号とコンピュータのＲＧＢ信号を一つのＮＴＳＣ信号に合成・変換したスーパーインポーズ画像の出力

③映像入力端子に入力されたＮＴＳＣ信号をそのまま出力

接続については、ご使用になる接続機器（ＶＴＲ、ビデオディスク、ビデオカメラ、ビデオ入出端子付テレビなど）の組み合わせによってさまざまな接続方法がありますが、ここでは標準的な接続方法２通りの説明をします。
接続する時には各接続機器の電源をかならず○ＦＦ（切）にしてください。なお接続用ケーブルは最寄りの販売店でお買い求めください。

①専用ディスプレイテレビ（ＣＺ－８５０Ｄ）との組み合わせ例



接続方法

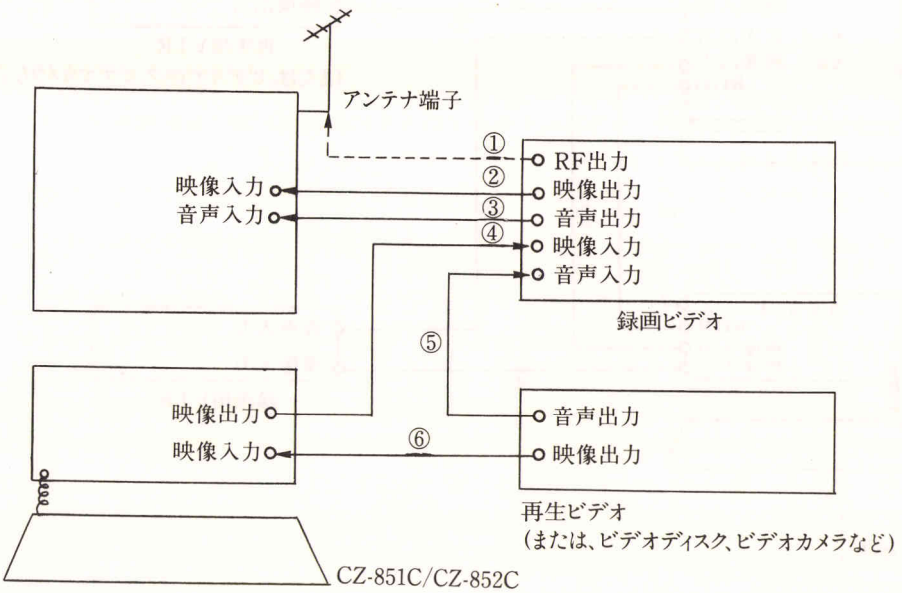
①～③本機とディスプレイテレビとをＲ．Ｇ．Ｂ．信号用ケーブル、テレビコントロールケーブル、ビデオカット用ケーブルで接続します。
④本機の映像出力端子と録画用ＶＴＲの映像入力端子とを接続します。
⑤本機の映像入力端子とディスプレイテレビのモニタ出力端子（映像）とを接続します。
⑥ディスプレイテレビのモニタ出力端子（音声）と録画用ＶＴＲの音声入力端子とを接続します。
⑦ディスプレイテレビの映像入力端子と再生用ＶＴＲの映像出力端子とを接続します。
⑧ディスプレイテレビの音声入力端子と再生用ＶＴＲの音声出力端子とを接続します。

操作方法

接続ができましたら、各接続機器の電源を○ＮＮ（入）にしてください。なお、各接続機器の取扱い操作については、それぞれの取扱説明書にしたがってください。
①コンピュータ本体前面トビラ内のＶＴＲ録画モードスイッチを○ＮＮ（入）にしてください。
②テレビ放送を録画したいときは、ディスプレイテレビＣＺ－８５０Ｄをテレビ画面にし、再生用ＶＴＲの映像を録画したいときは、ビデオ録画（画面表示はＰ．）にします。　（コンピュータ画像のみを録画したい場合、テレビ画像またはビデオ画像などのＮＴＳＣ信号をコンピュータの映像入力端子に入力する必要があります。）
③本機のキーボードで[SHIFT]キーを押しながらテンキーの [-] キーを押すとディスプレイテレビにはコンピュータ画像が映し出され、録画できる状態になります。
＜スーパーインポーズ画像のモニタ・録画をする場合＞
④本機のキーボードで[SHIFT]キーを押しながらテンキーの [+] キーを押すとディスプレイテレビにはスーパーインポーズ画像が映し出され、録画できる状態になります。

〈ビデオ画像のモニタ・録画をする場合〉
⑤本機のキーボードで[SHIFT]キーを押しながらテンキーの [=] キーを押すとディスプレイテレビにはテレビまたはビデオ画像が映しだされ、録画できる状態になります。
〈プログラムを組む場合〉
⑥前面トビラ内のＶＴＲ録画モードスイッチを〇ＦＦ（切）にし、本機のキーボードで[SHIFT]キーを押しながらテンキーの　・を押します。

(2)一般家庭用テレビとの組み合わせ例



＊映像入力端子付テレビの場合は②、③の接続を行ない、端子がない場合には①の接続を行ないます。

接続の方法

①テレビに映像入力端子がない場合、この接続を行ないます。録画用ＶＴＲのＲＦ出力とテレビのアンテナ端子を接続します。
②テレビに映像入力端子がある場合、この接続を行ないます。録画用ＶＴＲの映像出力端子とテレビの映像入力端子を接続します。
③テレビに音声入力端子がある場合、この接続を行ないます。録画用ＶＴＲの音声出力端子とテレビの音声入力端子を接続します。
④本機の映像出力端子と録画用ＶＴＲの映像入力端子を接続します。
⑤再生用ＶＴＲの音声出力端子と録画用ＶＴＲの音声入力端子を接続します。
⑥再生用ＶＴＲの映像出力端子と本機の映像入力端子を接続します。
以上の接続ができましたら、各接続機器の電源を〇Ｎ（入）にして次のように操作してください。
なお、各接続機器の取り扱いについては、各々の取扱説明書にしたがってください。

操作方法

(1) 専用ディスプレイテレビとの組み合わせ例の操作方法を参照してください。
ただし、②項については、次に従ってください。
② 接続の手順で、①を行なった場合は　チャンネルを１ｃｈまたは２ｃｈに合わせます。
②③を行なった場合は、ビデオ画面にします。

6 黒抜き表示

本機では　テキスト画面、グラフィック画面ともに黒抜き表示が可能です。つまり、テキスト画面はグラフィック画面、テレビ画面に対する黒抜き表示が行なえ、グラフィック画面ではテキスト画面、テレビ画面に対する黒抜き表示が行なえます。

黒抜き表示は、専用ディスプレイテレビ（CZ-850D ）との組合わせでは双方のビデオカット端子を接続することによって　RGB出力で直接行なうことができます。それ以外のモニターでは本機の映像出力端子から出力されるＮＴＳＣ信号によって、黒抜き表示を行なうことができます。

7 注意事項

①このテロッパー機能は標準ディスプレイモードのときのみ使用可能で高解像度ディスプレイモードでは使用できません。

②コンピュータ画像またはスーパーインポーズ画像をＶＴＲに録画する際はＮＴＳＣ信号という帯域制限を受けるため、コンピュータ画面を下のいずれかに設定してください。

40字×10行、40字×12行、40字×20行、40字×25行

また、タイリングペイントの様な細いドット構成の画面はちらつき、色変化を生ずることがあります。さらに　ＶＴＲの性能により録画画像に色ずれ、画質の劣化を生ずることがあります。

（理由）

RGB信号の周波数特性は　10―14MHｚであるのに対し、ＮＴＳＣ信号は下図に示すように輝度信号の後にある色信号と交錯しないようにするため　3～3.2 MHｚ程度の帯域巾となっています。このため　横80字（水平640 ドット）のように細い指定をしても　見にくかったり、色つきが悪かったりします。

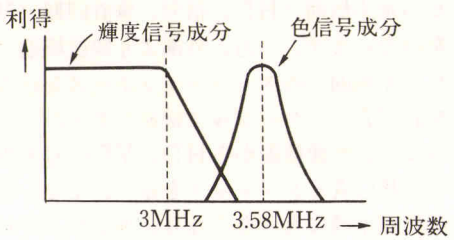
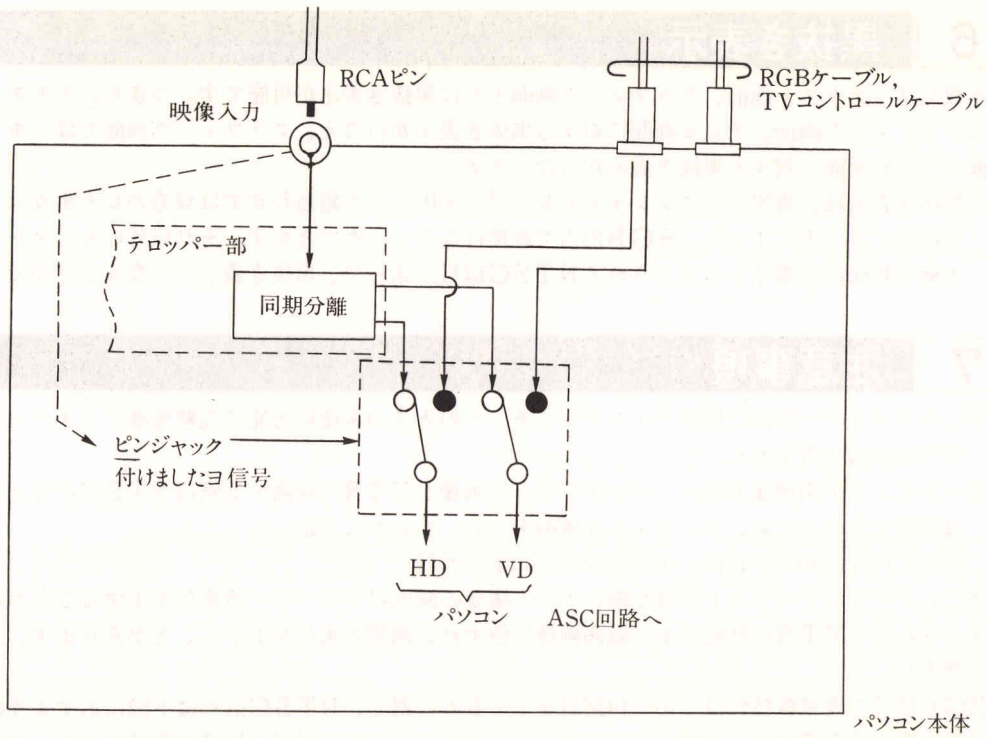


図. NTSC信号の周波数スペクトラム

③本機の映像入力端子にＲＣＡピンケーブルを接続し、そのケーブルを他の映像機器（ＶＴＲ、ビデオカメラ、ビデオディスク等）の映像出力端子に接続されていない場合や、接続されていても、映像信号が本機に入力されているい場合、本機の映像出力信号はもとより、モニターの画像も乱れます。従ってデジタルテロッパー使用の際は、必ず映像入力端子に他の機器より映像信号を入力してください。もし、デジタルテロッパーを用いない場合にはＶＴＲ録画モードスイッチをＯＦＦ（切）にするか、もしくは、本機の映像入力端子からＲＣＡピンケーブルをはずしてください。



（理由）

スーパーインポーズは映像信号にRGB信号を同期させて行なっています。専用ディスプレイテレビと本機との組み合わせでは、ディスプレイテレビの水平周期（HD）信号、垂直同期（VD）信号にRGB信号を同期させてスーパーインポーズを行ないます。一方、外部より映像機器（VTR、ビデオディスク、ビデオカメラなど）を接続し、ビデオ画面とのスーパーインポーズを行なう場合、本機の映像入力端子と映像機器の映像出力端子とをRCAピンケーブルで接続しますが、このとき本機内部では、RCAピンジャックを差し込んだ時より、映像機器からHD、VD信号を供給することになります。したがって、本機の映像入力端子にRCAピンジャックを差し込んで、映像機器に接続されないままで放置されると、HD、VD信号が本機に入らなくなり、画面がみだれて何も見えなくなります。

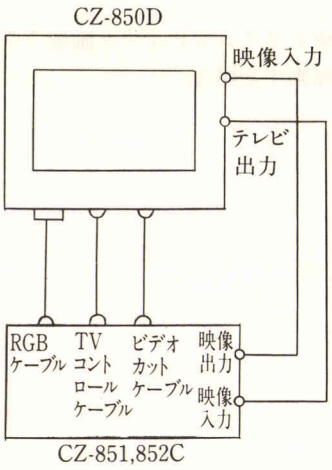
④映像入力端子にモニターの映像信号と異る、映像信号が入力されている場合スーパーインポーズモードにしますとモニター画面が乱れます。　モニターのスーパーインポーズ画面を使用するときは映像入力端子にはモニターと同一の映像信号を入れるかまたは何も接続しないでください。

⑤本機の映像入力端子へ入力される映像が白黒であれば、スーパーインポーズモード時またはコンピューターモード時（インターレースモード）のときのコンピュータ画像は白黒になります。ただし映像が白黒であってもカラーバースト信号がついていれば　カラーになります。

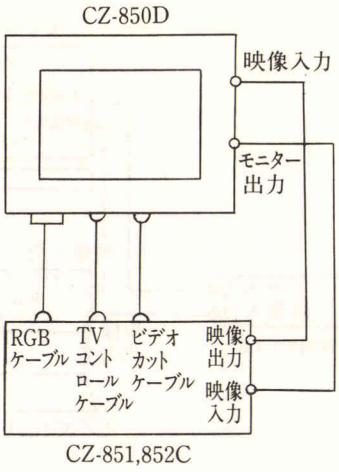
⑥本機に入力される映像信号の同期部分が劣化している場合　コンピュータ画像が乱れることがあります。例えば　テレビのアンテナ入力信号が弱い場合のテレビ出力や幾度もダビングされたVTRテープの再生信号等。

⑦下記のような接続では、画像が異常になるため使用しないでください。二重スーパーインポーズとなります。

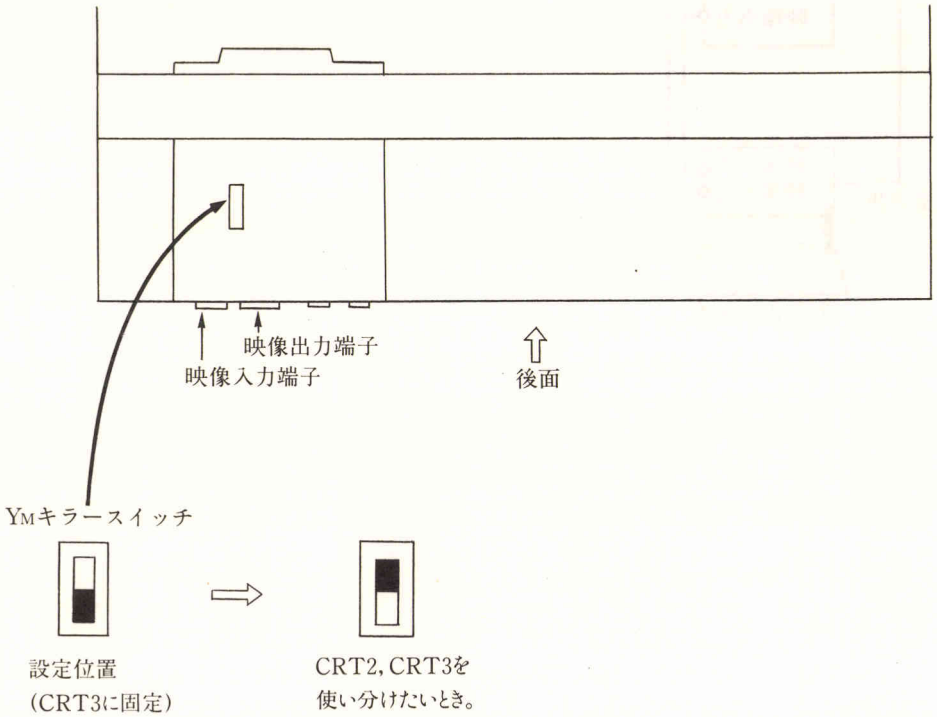
(i) 二重スーパーインポーズとなります。



(ii) 専用ディスプレイテレビ(CZ-850D)をビデオモードにしたとき、同期流れがおこります。



⑧本機の映像出力端子より出力される　スーパーインポーズ画像は必ずＢＡＳＩＣステートメントＣＲＴ３（テレビ放送とコンピュータ画面を同時に重ねて表示）を実行した状態になっています。プログラムにて映像出力を　ＣＲＴ２（テレビ放送のコントラストを下げて、コンピュータ画面を重ねて表示）状態と　ＣＲＴ３の状態とを使い分けたい場合は、上ぶたを開けて、ＹＭキラースイッチをコンピュータ前面側へスライドさせてください。

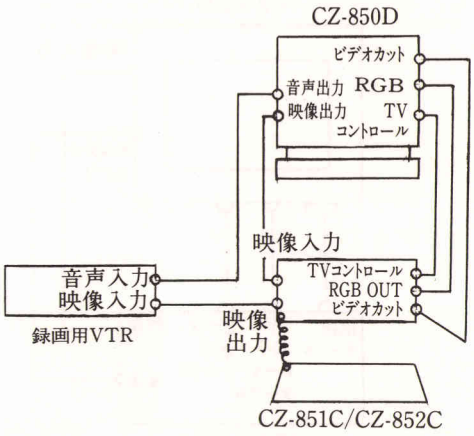


参考

接続応用例

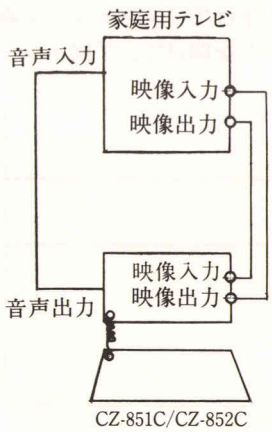
接続応用例①

・専用ディスプレイテレビ（ＣＺ－８５０Ｄ）を使用して＂コンピュータ画像＂＂テレビとのスーパーインポーズ画像＂＂テレビ画像＂をＶＴＲに録画する場合。



接続応用例②

・家庭用テレビ（ビデオ入出力端子付）に接続し、テレビ画像とコンピュータ画像を重ね合わせ、スーパーインポーズ画像を楽しむ場合。



付録

A1 テキスト画面(V-RAM)へのアクセス

テキスト画面とその属性エリアは共にI／Oポートにあり、

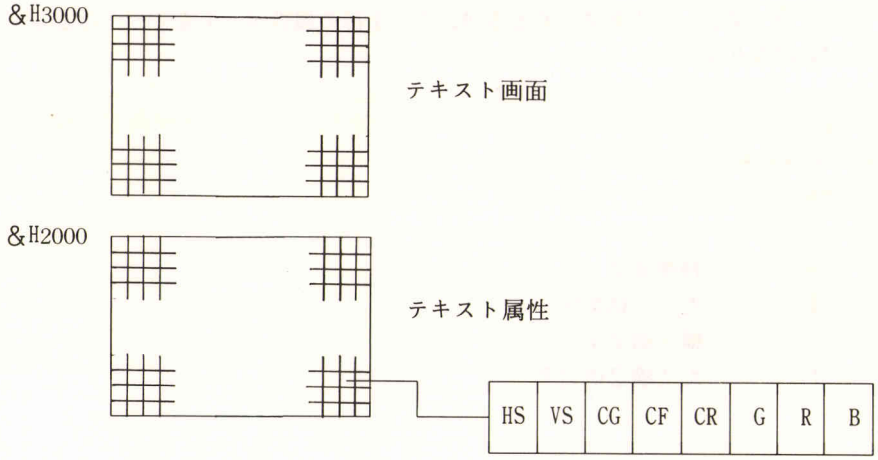
テキスト画面が　&H3000～&H37FF（番地）
その属性が　　&H2000～&H27FF（番地）
漢字の属性が　　&H3800～&H3FFF（番地）

の範囲で、それぞれ2KB（2048バイト）の容量をもっています。

テキスト画面とその属性エリアとはちょうど1対1に順序よく対応していて、次に示すステートメントや関数を使って1バイト単位でアクセスすることができます。

出力用：　POKE@、OUT
入力用：　PEEK@、INP

テキスト画面の属性1バイトのビット構成は下図の通りです。



以下、下位ビットから順次説明して行きます。

（1）　B、R、G ····· 色の指定をする（⇨COLOR）

テキスト画面の文字の色はB（青）、R（赤）、G（緑）の3ビットにより次のように決定されます。

ビット	2	1	0	
	G	R	B	
	0	0	0	····· 黒
	0	0	1	····· 青
	0	1	0	····· 赤
	0	1	1	····· マゼンタ（紫）
	1	0	0	····· 緑
	1	0	1	····· シアン（水色）
	1	1	0	····· 黄
	1	1	1	····· 白

（2）　CR…… 文字の反転モードの指定（⇨CREV）

テキスト画面の文字の反転の指定はビット３で行ないます。

ビット３が０のとき標準モード、１のとき反転モードとなります。

（3）　CF…… 文字の明滅モードの指定（⇨CFLASH）

テキスト画面の文字の明滅の指定は４ビットで行ないます。

ビット４が０のとき標準モード、１のとき明滅モードとなります。

（4）　CG…… ROMCG／RAMCGの切り換えの指定（⇨CGEN）

キャラクタ・ジェネレータには、ROMCGとRAMCGの２種類があり、それぞれ２５６文字まで表示させることができます。

テキスト画面にROMCGで生成した文字を表示するか、RAMCGで生成した文字を表示するかの指定は、ビット５で行ないます。

ビット５が０のときROMCG、１のときRAMCGを指定します。

（5）　VS、HS…文字の拡大モードを設定します（⇨CSIZE）

テキスト画面の文字のサイズの指定はビット６と７で行ないます。

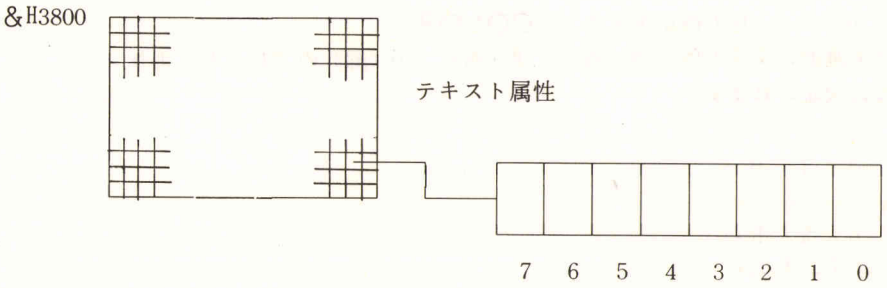
ただし、ビット６（VS）に１を立てるときは、その１行の属性ポートをすべてのビット６を立てなければなりません。

ビット　7　6

HS	VS	

0　0 …… 標準文字
0　1 …… たて２倍文字
1　0 …… 横２倍文字
1　1 …… たて横２倍文字

漢字の属性１バイトのビット構成は次のようになっています。



以下、テキスト属性同様、下位ビットから順次説明していきます。

（1）　下位４ビット（ビット３，２，１，０）…漢字データ部の上位４ビットコードを示す。
　全角文字（漢字）が表示されているとき、そのデータ部を示しています。これは、ＢＡＳＩＣリファレンスマニュアルの付録「非漢字およびＪＩＳ第一水準漢字一覧表」のＶＲＡＭ（３８）に当たります。

（2）　ビット４…… 漢字ＲＯＭの第１水準か第２水準かの指定
　または、ＰＣＧのキャラクタモードか外字モードかの指定
テキスト画面の属性１バイトのビット５（ＣＧ）が０のとき漢字ＲＯＭの指定です。
１のときＰＣＧの指定となる。そして、このビット４が０のとき、JIS 第一水準または、キャラクターモードを、１のときJIS 第２水準または、外字モードを指定します。

（3）　ビット５…… アンダーラインの指定
　アンダーラインの指定は、ビット５で行ないます。
ここが６のときアンダーライン消去、１のときアンダーラインを表示します。

（4）　ビット６…… 全角文字のサイド指定
　全角文字（漢字）が表示されて、それが外字でないとき、
ここがゼロならば文字の左側の部分、１ならば文字の右側部分であることを示しています。

（5）　ビット７…… ＣＧＲＯＭと漢字ＲＯＭの選択の指定
　０のときＣＧＲＯＭを選択し、１のとき漢字ＲＯＭを選択することを指定します。

A2 組み込み関数以外の数学的関数

組み込み数値関数にない三角関数と双曲線関数およびそれらの逆関数を定義する公式を示します。
使用する場合には各関数の定義域に注意する必要があります。

関数名	BASICの形式による公式	関数
secant	A (X) = 1／COS (X)	$\sec x$
cosecant	B (X) = 1／SIN (X)	$\mathrm{cosec}\, x$
cotangent	C (X) = 1／TAN (X)	$\cot x$
arcsine	D (X) = ATN (X／SQR (1-X ^ 2)	$\sin^{-1} x$
arccosine	E (X) =-ATN (X／SQR (1-X ^ 2) + π／2	$\cos^{-1} x$
arcsecant	F (X) =-ATN (SQR (X ^ 2-1)) + (SGN (X) -1) ＊π／2	$\sec^{-1} x$
arccosecant	G (X) = ATN (1／SQR (X ^ 2-1)) + (SGN (X) -1) ＊π／2	$\mathrm{cosec}^{-1} x$
arccotangent	H (X) =-ATN (X) + π／2	$\cot^{-1} x$
hyperbolic sine	I (X) = (EXP (X) -EXP (-X)) ／2	$\sinh x$
hyperbolic cosine	J (X) = (EXP (X) -EXP (-X)) ／2)	$\cosh x$
hyperbolic tangent	K (X) =-EXP (-X) ／ (EXP (X) + EXP (-X)) ＊2+1	$\tanh x$
hyperbolic secant	L (X) = 2／ (EXP (X) +EXP (-X))	$\mathrm{sech}\, x$
hyperbolic cosecant	M (X) = 2／ (EXP (X) -EXP (-X))	$\mathrm{cosech}\, x$
hyperbolic cotangent	N (X) = EXP (-X) ／ (EXP (X) -EXP (-X)) ＊2+1	$\coth x$
arc-hyperbolic sine	O (X) = LOG (X+SQR (X ^ 2+1))	$\sinh^{-1} x$
arc-hyperbolic cosine	P (X) = LOG (X+SQR (X ^ 2-1))	$\cosh^{-1} x$
arc-hyperbolic tangent	Q (X) = LOG ((1+X) ／ (1-X)) ／2	$\tanh^{-1} x$
arc-hyperbolic secant	R (X) = LOG ((SQR (1-X ^ 2) +1) ／X)	$\mathrm{sech}^{-1} x$
arc-hyperbolic cosecant	S (X) = LOG ((SGN (X) ＊ SQR (X ^ 2) +1) +1) ／X)	$\mathrm{cosech}^{-1} x$
arc-hyperbolic cotangent	T (X) = LOG ((X+1) ／ (X-1)) ／2	$\coth^{-1} x$

以上の各関数を使うときは、「DEF FN」ステートメントであらかじめプログラム中に定義しておくと便利です。これらの関数は、パラメータXの定義域に十分注意して使ってください。

```
(例)  100 DEF FNA(X)=1／COS(X)
      ･･･ ･･･････
      190 Y=･･･
      200 H=FNA(Y)
      ･･･ ･･･････
```

A3 数値精度の変換

数値は、整数の場合、−32768～32767、実数の場合、約−1.7×10^{38}～1.7×10^{38}で、単精度型のとき有効数字8けたの精度、倍精度型のとき有効数字16けたの精度で表現されます。

数値は、変数の属性文字（%、!、#）、型変換の関数（CINT、CSNG、CDBL）によって、その精度を変換することができます。

精度の変換は、次の規則に従って行われます。

(a) 数値が、別の精度の数値変数に代入されると、それは代入先の変数名の表わす精度で記憶されます。

（例） A%=3.14 ····· A%は整数型変数なので、A%の値は3となります。

(b) 数値をそれより低い精度の変数に代入すると、数値は丸められて低い精度で記憶されます。

（例） A%=3.1459265358# ··· Aは単精度型変数なので、Aの値は3.1415927に丸められます。

(c) 数値をそれより高い精度の変数に代入すると、数値は高い精度の表示になりますが、より正確になることはありません。

（例） A#=3.14 ····· A#は倍精度型変数なので、A#の値は3.139999999664724と表示されます。

(d) 式の値を求めるときは、最も精度の高いオペランドと同じ精度に変換されます。

（例） A#=4#/7 ····· 4#が倍精度型数値なので、計算の結果は倍精度型となり、A#の値は、.5714285714285714となります。

(e) 関数の値は、単精度8けたが保証されますが、引数に倍精度型の数値が含まれると倍精度16けたまで求めることができます。

（例） A=SQR（3） ····· Aの値は、1.732050

A#=SQR（3#） ····· A#の値は、1.7320508075688877

(f) 数式の値の型変数を行なうには、CINT、CSNG、CDBLの各関数を使います。

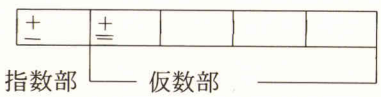
CINTは値を整数型に、CSNGは単精度型に、CDBLは倍精度型にそれぞれ変換します。

（例） A=CINT（R/100） ····· Rの値が314のとき、Aの値は3になります。

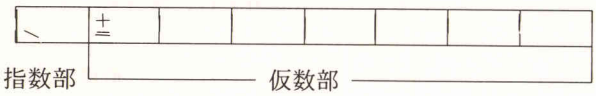
A4 数値データの誤差

BASICで扱う単精度型および倍精度型の数値は、コンピュータ内部では2進浮動小数点形式で表現され、演算や比較もこの形式のまま行なわれます。

単精度型（5バイト）



倍精度型（8バイト）



この内部表現上の制約のため、数値は必ずしも正確な値として記憶されるとは限らず、このために起こる、演算結果および関係式の比較における誤差、画面やライン・プリンターなどの出力装置に表示される値とのずれ、などに注意する必要があります。

たとえば、単精度型や倍精度型の数値を使った演算の結果が、数学では整数となる場合でも、必ず

しも整数が得られるとは限りません。

数値データの内部表現の誤差が表示される値に及ぼす影響について、簡単な例をあげて説明してみましょう。

（例1）

```
10 X=0
20 FOR I=0 TO 1500
30 PRINT X;
40 X=X+.1
50 NEXT
```

注）.1＝0.1

（例1）は、変数Xの初期値を0として0、1ずつ加えては、その値を表示するというプログラムです。

このプログラムを実行すると、はじめは、0、.1、.2 …… と増えて行きますが、.46.1を過ぎると、46.1、46.199999、46.299999、…というように誤差が現れます。

これは、内部表現の誤差がたまったために生ずる現象です。

この不合理を解消する方法として、次の3つが考えられます。

（1） 整数の1を加え、それを10で割る。

```
10 X=0
20 FOR I=0 TO 1500
30 PRINT X/10;
40 X=X+1
50 NEXT I
```

（2） 10倍した値をCINTを使って整数型に変換してから10で割る。

```
10 X=0
20 FOR I=0 TO 1500
30 PRINT X;
40 X=X+.1
50 X=CINT(X*10)/10
60 NEXT I
```

（3） STR$を使って文字型に変換してからVALを使って数値型に戻す。

```
10 X=0
20 FOR I=0 TO 1500
30 PRINT X;
40 X=X+.1
50 X=VAL(STR$(X))
60 NEXT I
```

（例2）

```
10 FORI=0 TO 1 STEP .1
20 PRINT I;
30 NEXT I
```

（例2）は、FOR、NEXTループにおいてループ変数Iの初期値を0、増分を.1として、終了値1まで回すというプログラムです。

このプログラムを実行すると、0、.1、.2 …… 、 .9となって、最後の1までループしません。

これも、内部表現の誤差によって生ずる現象です。

これを防ぐためには、次のように、ループ変数の増分を整数にします。

```
10 FOR I=0 TO 10
20 PRINT I/10;
```

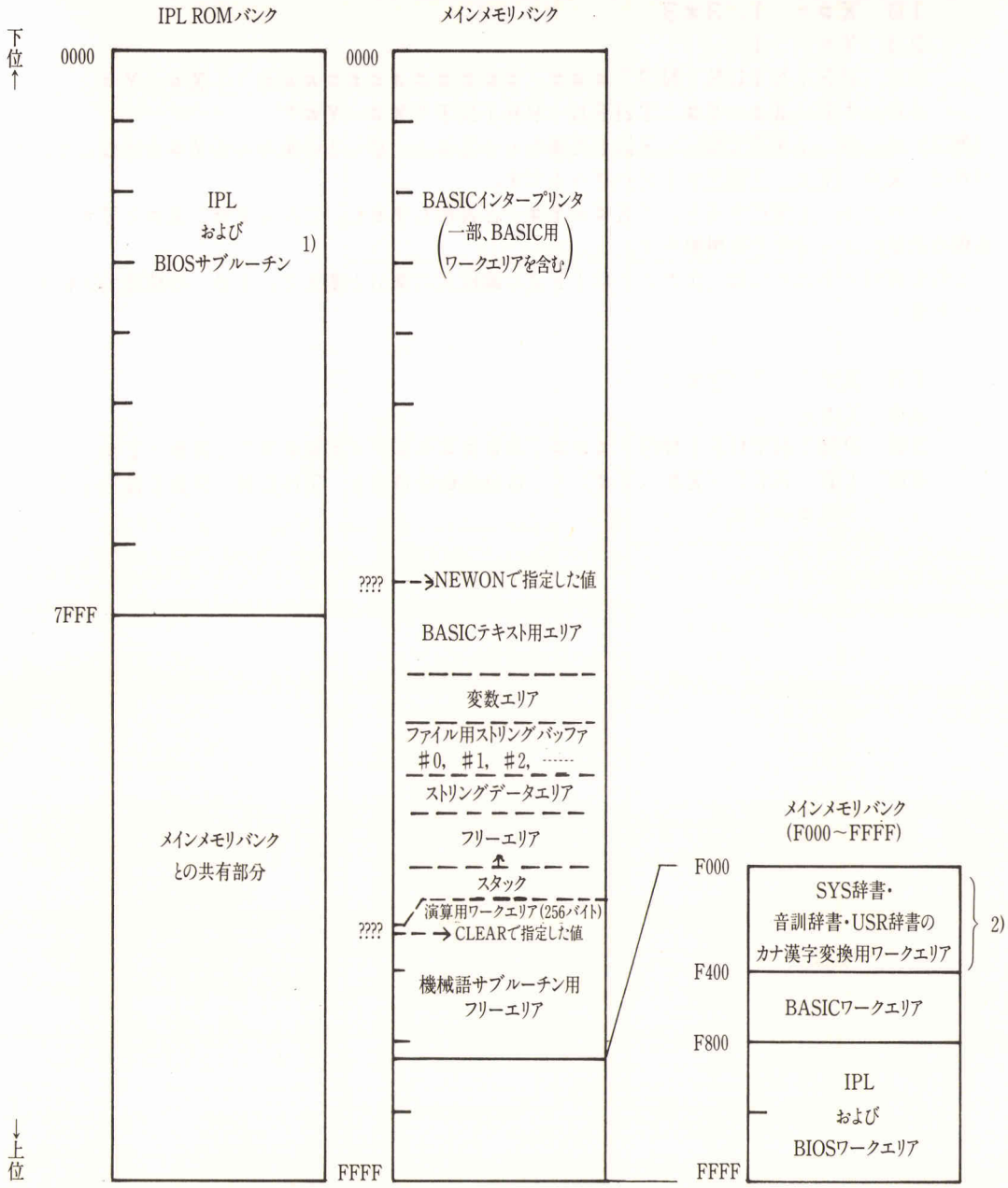
```
    30 NEXT I
```
（例3）
```
    10 X#=.1/3*3
    20 Y#=.1
    30 PRINTUSING"###.##########";X#;Y#
    40 IF X#=Y# THEN PRINT"X#=Y#"
```
（例3）は、0．1を3で割って3倍した値をもつX#と、0．1の値をもつY#を比較して、等しければ「X#＝Y#」と表示するプログラムです。

このプログラムを実行すると、「X#＝Y#」と表示しません。これもまた、X#とY#との内部表現の誤差によって生じる現象です。

これを解消するためには、次のようにIF文の論理式の部分を変えて、10　の精度で比較するようにします。

```
    10 X#=.1/3*3
    20 Y#=.1
    30 PRINTUSING"###,##########";X#;Y#
    40 IF ABS(X#-Y#)<.00000001 THEN PRINT
       "X#=Y#"
```

A5 メモリマップ



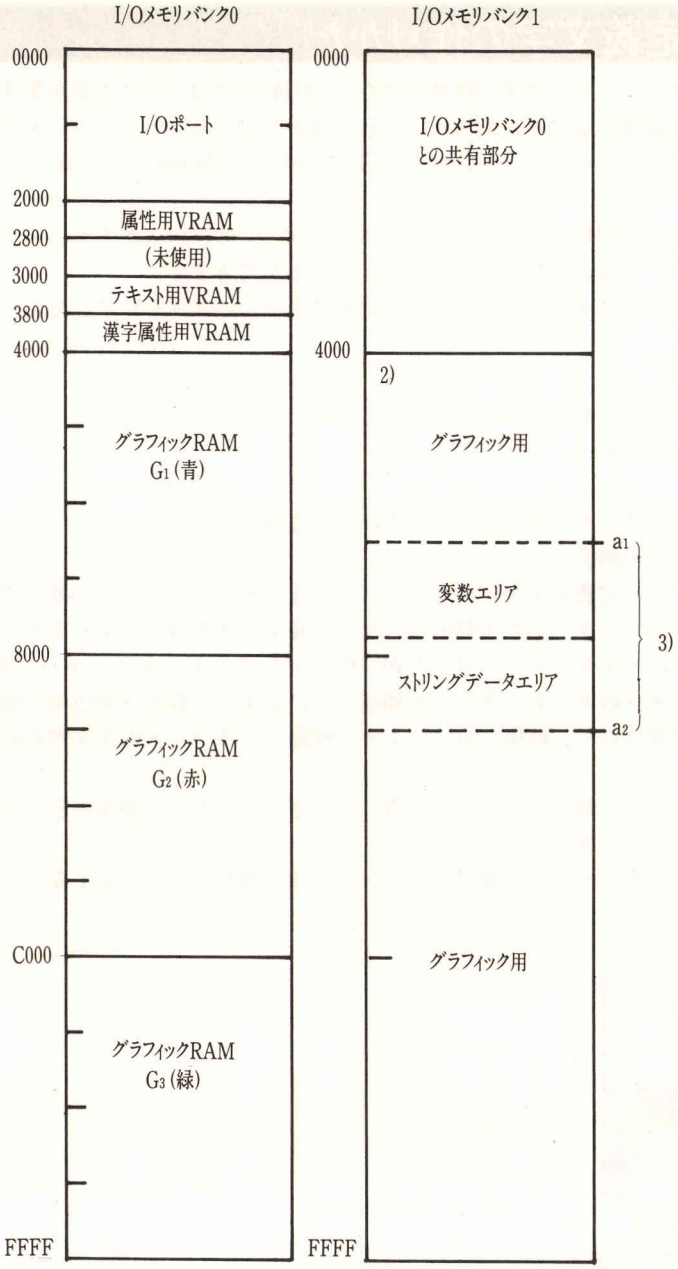
1) BIOSサブルーチンの機能

- キー入力
- 画面表示
- プリンタ出力
- カセット出力
- ディスク、HDのリードライト
- グラフィックサブルーチン
- CGの定義・読み込み
- 四則演算および関数
- その他

2) 辞書を使用しない場合は、

CLEAR &HF400

を実行することにより、機械語サブルーチン用フリーエリアとして使用できます。



2) OPTION SCREEN1および2の場合、VDIM変数エリアとして使用することができる。
OPTION SCREEN0が設定されている場合は、バンク0と同様グラフィック表示用エリアとして使用できる。

3) VDIM変数エリアとして使用する場合、VDIM CLEARでその範囲を指定し、その外をグラフィック用として使用することができる。
a1は変数エリアの先頭アドレス、a2はエンドアドレス。VDIM CLEARを実行していなければ 4000 ～FFFFが変数エリアとして使用できる。

A6 ユーザー定義文字の作りかた

英数字、カナ文字、セミグラフィック文字、特殊文字などの図形文字は、小さな点が集まって構成されています。1つの文字を構成する点は、半角文字で8×8または16×8ドット、全角文字で16×16ドットあって、このドットパターンがROMCG（Read-Only Memory Character Genertor）に記憶されています。

これに対して、RAMCG（Random Access Memory Character Generator）があり、ユーザーが好きなドットパターンをデザインして、これに記憶させておくことができます。

ユーザーがデザインできる図形文字（以下、ユーザー定義文字といいます）には、そのサイズによって、

〔1〕　たて8×横8ドット
〔2〕　たて16×横8ドット
〔3〕　たて16×横16ドット

の3つのタイプがあります。

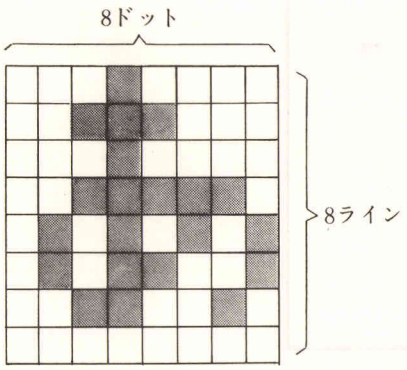
ここでは、その各々について作りかたと表示のしかたについて説明します。

〔1〕　たて8×横8ドットの場合

たて8×横8ドットのユーザー定義文字を定義するには、24バイトの文字列が必要です。なぜならば、その文字列の表わすビットパターンが1個のユーザー定義文字を形成するからです。すなわち、1バイトのキャラクタコードのビットパターン8けたが、横8×たて1ドットのパターンを表わし、8バイトの文字列でたて8ラインのドットパターンを構成しています。これが3原色の合成原理により、青、赤、緑の3画面分必要なので、結局、8バイト×3画面＝24バイトの文字列が必要となります。

この24バイトの文字列のうち、最初の8バイトが青、次の8バイトが赤、最後の8バイトが緑の部分のドットパターンを表わしています。

たとえば、ひらがなの「あ」という文字を青地にシアン色で作る場合についてみてみましょう。



■はドットがセットされていることを示す。

ユーザー定義文字は、8×8ドットパターンで表わされ、ひらがなの「あ」は上のようなパターンになります。これを2進数のパターンで表わし、さらに16進数表現に変換すると、次のようになります。

（2進数表現）		（16進数表現）	
00010000	⇨	10	……①
00111100	⇨	3C	……②
00010000	⇨	10	……③
00111110	⇨	3E	……④
01010101	⇨	55	……⑤
01011001	⇨	59	……⑥
00110010	⇨	32	……⑦
00000000	⇨	00	……⑧

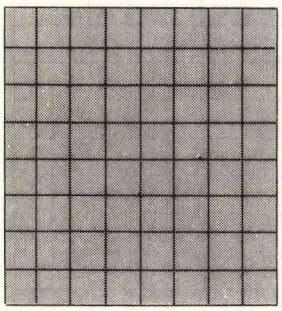
したがって、このドットパターンは、

```
HEXCHR$("10 3C 10 3E 55 59 32 00")
          ①  ②  ③  ④  ⑤  ⑥  ⑦  ⑧
```

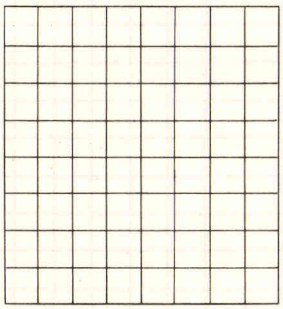
となります。

シアン色は青と緑を合成して得られ、青地にシアン色の「あ」という文字を作るためには、青の画面、赤の画面、緑の画面に対するパターンをそれぞれ次のように定義しなければなりません。

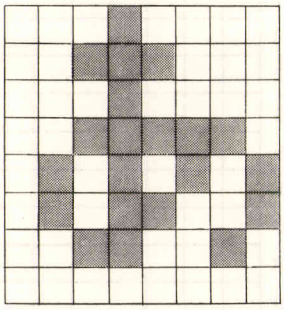
（青の画面）



（赤の画面）



（緑の画面）



（青の画面）		（赤の画面）		（緑の画面）	
11111111	⇨FF	00000000	⇨00	00010000	⇨10
11111111	⇨FF	00000000	⇨00	00111100	⇨3C
11111111	⇨FF	00000000	⇨00	00010000	⇨10
11111111	⇨FF	00000000	⇨00	00111110	⇨3E
11111111	⇨FF	00000000	⇨00	01010101	⇨55
11111111	⇨FF	00000000	⇨00	01011001	⇨59
11111111	⇨FF	00000000	⇨00	00110010	⇨32
11111111	⇨FF	00000000	⇨00	00000000	⇨00

したがって、青地にシアン色の「あ」を定義するのに必要な文字列は、

```
HEXCHR$("FFFFFFFFFFFFFFFF0000000000000000103C103E55593200")
```

となります。

このサイズのパターンを定義できるコード（キャラクタコード）は0～255で、256個の文字を定義することができます。たとえばこれをコードの32に定義するにはDEFCHR$ステートメントを使って、

```
DEFCHR$(32)=HEXCHR$("FFFFFFFFFFFFFFFF0000000000000000103C103E55593200")
```

と書きます。

こうして定義した文字は、CGEN1とPRINT#0ステートメントを使って画面に表示することができます。

（例）
```
100 KMODE0
110 CGEN1
120 A $="FFFFFFFFFFFFFFFF0000000000000000103C103E55593200"
130 DEFCHR$(32)=HEXCHR$(A$)
140 PRINT #0,CHR$(32)
150 CGEN0
```

青の画面、赤の画面、緑の画面の３枚とも同じデータの場合は、８バイトの文字列ですませることができます。

たとえば、黒地に白の「あ」を定義するには、

DEFCHR$(32)=HEXCHR$("103C103E55593200103C103E55593200103C103E55593200")

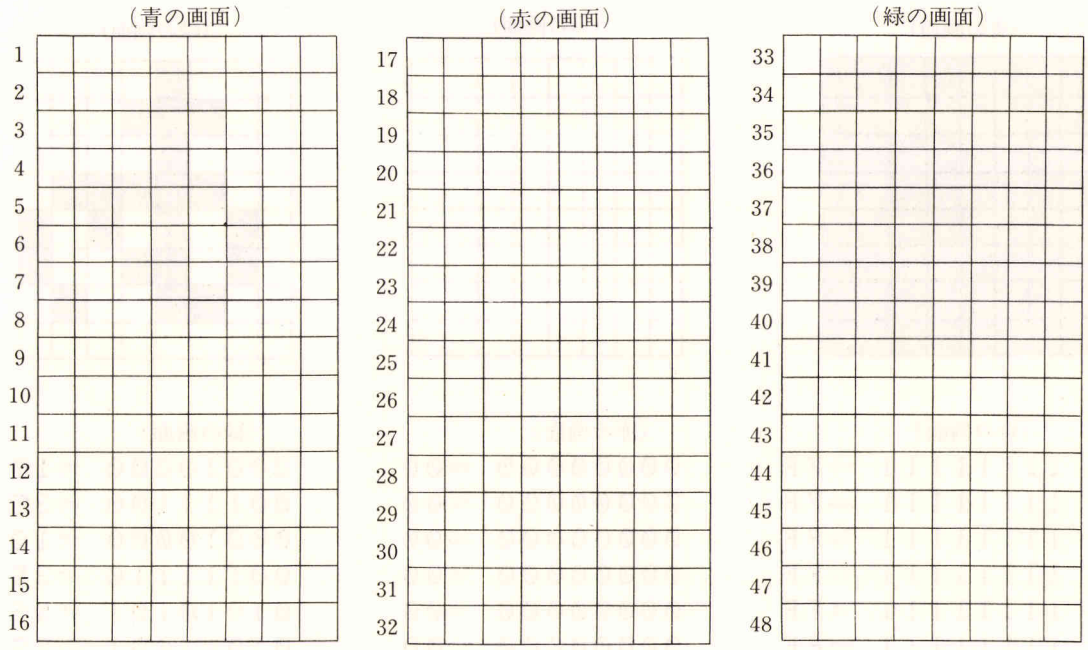
としないで、

DEFCHR$(32)=HEXCHR$("103C103E55593200")

とするだけですみます。

〔2〕　たて１６×横８ドットの場合

たて１６×横８ドットのユーザー定義文字の定義には、８×８ドットの文字がたて２倍にのびたパターンなので、４８バイトの文字列が必要です。



１６×８ドットのユーザー定義文字は、１６×８ドットパターンで表わされます。ここにデザインしたパターンを、８×８のときと同様に２進数のパターンで表わし、さらに１６進数表現に変換してコードの文字列にするのです。

このサイズのパターンを定義できるコードは&Ｈ１００、&Ｈ１０２、&Ｈ１０４、‥‥‥、&Ｈ１ＦＥの１２８文字分です。たとえばあるパターンをコードの&Ｈ１００に定義するには、

```
DEFCHR$(&H100)=HEXCHR$("HHH ---------- HHH ")
                        96けた（48バイト）
```

と書きます。

こうして定義した文字は、ＣＧＥＮとＰＲＩＮＴ＃０ステートメントを使って画面に表示することができます。

（例）

```
100  KMODE0
110  CGEN2
120   A$＝"HHH ---------- HHH "     →  ４８バイトの任意の文字列
130  DEFCHR$ (&H100) =HEXCHR ( A$)
140   PRINT#0, CHR$ ( 0)
150  CGENO
```

※　&H１００のコードは　ＰＲＩＮＴ#０,CHR$（０）
　　&H１０２のコードは　ＰＲＩＮＴ#０,CHR$（&H2）
　　&H１０４は　　ＰＲＩＮＴ#０,CHR$（&H4）

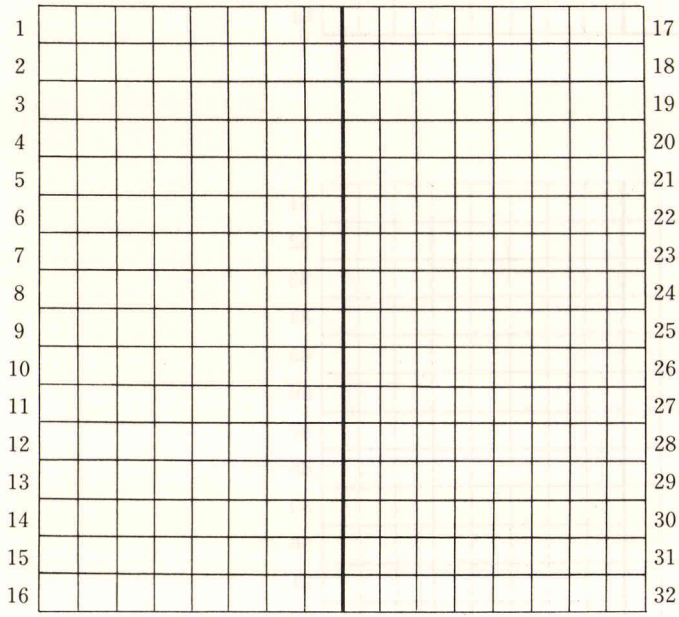
というように書いて表示します。

なお、青の画面、赤の画面、緑の画面の３枚とも同じデータの場合は、１６バイトの文字列ですませることができます。

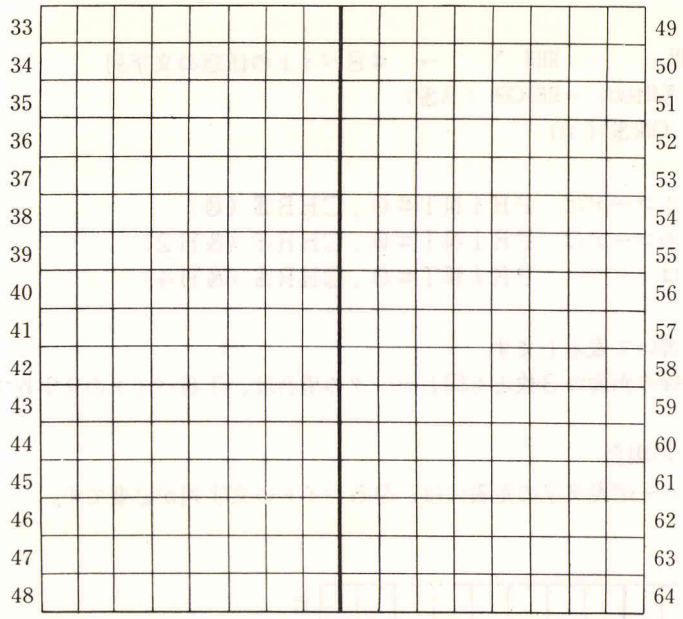
〔3〕　たて１６×横１６ドットの場合

たて１６×横１６ドットのユーザー定義文字の定義には、９６バイトの文字列が必要です。

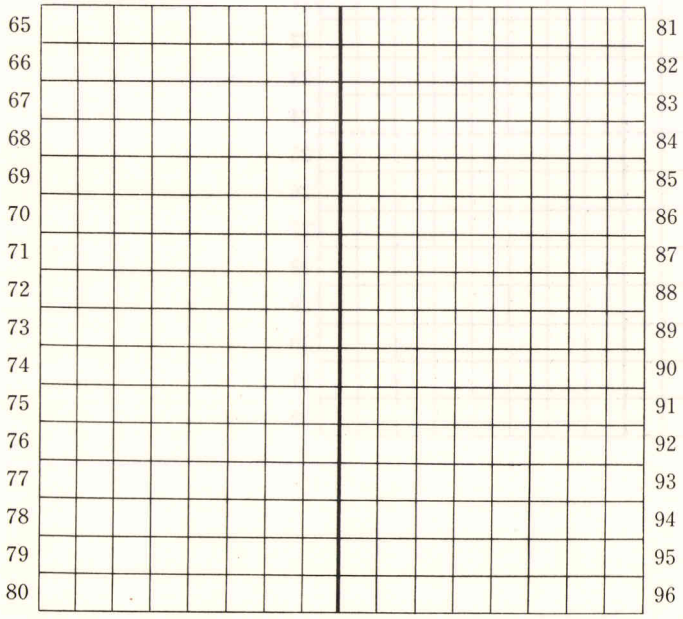
（青の画面）



（赤の画面）


33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64

（緑の画面）


65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96

１６×１６ドットのユーザー定義文字は、１６×１６のドットパターンで表わされます。ここにデザインしたパターンを、８×８のときと同様に２進数のパターンで表わし、さらに１６進数表現に変換してコードの文字列にします。 1)

このサイズのパターンを定義できるコードは＆Ｊ７６２１～＆Ｊ７６６０の６４文字分です。

たとえば、あるパターンをコードの＆Ｊ７６２１に定義するには、

```
DEFCHR$（&J7621 ）＝HEXCHR$（" HHH ---HHH "）
                                    192 けた（96バイト）
```

と書きます。

こうして定義した文字は、KMODE１を実行後、普通のＰＲＩＮＴ文を使って画面に表示することができます。

```
KMODE１
PRINT　CHR$（&Ｊ７６２１）
```

なお、青の画面、赤の画面、緑の画面の３枚とも同じデータの場合は、３２バイトの文字列ですませることができます。

注１）＆Ｊ７６２１～＆Ｊ７６６０のＪＩＳ１６進定数に定義される文字を特に外字と呼びます。

シャープ株式会社

本　　社　〒５４５　大阪市阿倍野区長池町22番22号
電話 06（621）1221（大代表）
電子機器事業本部　〒329-21　栃木県矢板市早川町174番地
電話 02874（3）1131（大代表）

お客様へ……お買いあげ年月日、お買いあげ店名を記入されますと、修理などの依頼のときに便利です。

お買いあげ年月日	5□年 4月 1日
お買いあげ店名	
	電話番号
もよりのお客様ご相談窓口	
	電話番号

TMAN-3580CEZZ
T0569-A③ 4A