SHARP

New-Life People

USER'S MANUAL

パソコンテレビ X1 turbo Z
パーソナルコンピュータ
形名

CZ-880C

上手に使って上手に節電



製造番号は、品質管理上重要なものですから商品本体に表示されている製造番号と、保証書に記載されている製造番号とが一致しているか、お確かめください。

はじめに

このマニュアルはＢＡＳＩＣの初歩から、本機の特徴であるグラフィック機能、日本語処理機能、テロッパ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。

入門編…初めての方のために、本機を使う上での予備知識を説明します。

基礎編…プログラミングするための基本的なことを説明します。

応用編…上級者向けです。パソコンの機能を最大限に使いこなしてください。

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。

◎ディスクＢＡＳＩＣ（ＣＺ－８ＦＢ０２、２ＨＤタイプ）を起動させた場合

ディスク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われます。

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

ディスク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) ディスクＢＡＳＩＣ起動時の初期状態

```
ＩＮＩＴ
ＷＩＤＴＨ　８０，２５
ＣＯＮＳＯＬＥ　０，２４，０，８０
（高解像度ディスプレイモードのとき）
ＷＩＤＴＨ　８０，１２
ＣＯＮＳＯＬＥ　０，１１，０，８０
（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）

ＫＬＩＳＴ　１

ＦＵＮＣＴＩＯＮ　ＫＥＹの設定

フロッピーディスクの２ＨＤ(Ｘ１フォーマット)の設定

内蔵時計の「年」の項の設定

(iii) ＮＥＷＯＮの設定

ＮＥＷＯＮ命令はＢＡＳＩＣのコマンドやステートメントなどを削り、フリーエリアを増やします。

ディスクBASICを起動した直後、

```
NEWON■ ← カーソル
```

と画面に表示されます。

0～9までの数値を入力するか、または数値を省略してリターンキー↵を押してください。希望のレベルのNEWONが設定できます。

NEWON ↵

とキー入力すれば、すべてのコマンド、ステートメントが使用できます。

NEWONのレベル（0～9）については、基礎編『第5章　フリーエリアについて』を参照してください。

◎X1用BASIC（CZ－8CB01，CZ－8FB01）とX1 turbo用BASIC（CZ－8FB02、2HDタイプ）との初期設定の相違点

CZ-8CB01、CZ-8FB01	CZ-8FB02［2HD］
WIDTH 40,25	WIDTH 80,25（高解像度ディスプレイモードの時）
WIDTH 40,25	WIDTH 80,12（標準ディスプレイモードの時）
CONSOLE 0,25,0,40	CONSOLE 0,24,0,80（高解像度ディスプレイモードの時）
CONSOLE 0,25,0,40	CONSOLE 0,11,0,80（標準ディスプレイモードの時）
KLIST 0の状態	KLIST 1
KMODE 0の状態	KMODE 1
CLEAR &HFF00	CLEAR &HF400
DEVICE"0:0"の状態	DEVICE"0:2"
DEVICE"1:0"の状態	DEVICE"1:2"
DEVICE"2:0"の状態	DEVICE"2:2"
DEVICE"3:0"の状態	DEVICE"3:2"

X1 turbo用BASICは、以上の点で、X1用BASICと異なっていますので、X1 turbo用BASICで従来のX1用のソフトウェアを実行する際には、次の操作を行ってください。

DEVICE"0:0":DEVICE"1:0";DEVICE"2:0";DEVICE"3:0" ↵

CONSOLE　0，25：KLIST0：KMODE0：CLEAR　&HF400 ↵

ただし、X1用のソフトで&HF400以降を使用しているソフトは使用できません。

なお、本機にはX1用BASIC（CZ－8FB01　V1．0，CZ－8CB01　V1．0）が搭載されています。これの起動法については『アプリケーションソフト説明書』の「はじめに」の項をお読みください。

目次

第1部 入門編 …… 1

第1章 プログラミングを始めよう …… 2
1.1 コンピュータへのメッセージ …… 2
1.2 プログラムって何だろう …… 5
1.2.1 行番号を付けてみる …… 5
1.2.2 直接モードと間接モード …… 6
1.2.3 プログラムって何だろう …… 7
1.2.4 変数を使う …… 7
1.3 プログラムの作成(その1) …… 8
1.3.1 プログラムの入力 …… 8
1.3.2 1行 挿入 …… 10
1.3.3 1行削除 …… 10
1.3.4 1行変更 …… 11
1.3.5 行番号の付け直し …… 11
1.3.6 複数行 削除 …… 13
1.4 プログラムの作成(その2) …… 14
1.4.1 カーソルの移動と文字の修正 …… 14
1.4.2 効率のよい入力方法 …… 15
1.4.3 文字の挿入 …… 17
1.4.4 文字の削除 …… 18
1.4.5 困ったときに助けてくれる CTRL + D シ キー …… 18

第2章 プログラムの保存と再生 …… 20
2.1 フロッピーディスクに保存、再生する場合 …… 20
2.1.1 フロッピーディスクにセーブする …… 20
2.1.2 フロッピーディスクからロードする …… 21
2.1.3 フロッピーディスクの内容を見る …… 22
2.2 カセットテープに保存、再生する場合 …… 22
2.2.1 カセットテープにセーブする …… 22
2.2.2 カセットテープの内容を確認する …… 23
2.2.3 カセットテープからロードする …… 24
2.2.4 カセットテープの内容を見る …… 24

第3章 テレビコントロール …… 26
3.1 キーボードからテレビをコントロールする …… 26
3.2 テレビタイマコントロール …… 26
3.2.1 テレビタイマコントロールの呼び出し …… 26
3.2.2 テレビタイマコントロールの表示内容 …… 27
3.2.3 クロック(時計)を現在時刻に合わせる …… 28
3.2.4 タイマで番組予約をする …… 28

3.2.5 タイマでスイッチを OFF にする……29
3.2.6 タイマを取り消す……30
3.3 プログラムでテレビをコントロールする……31
3.3.1 現在時刻の設定……31
3.3.2 タイマコントロール……31

第2部 基礎編……34

第1章 プログラムの編集……34
1.1 基本的な編集機能……34
1.1.1 カーソルの移動……34
1.1.2 文字の削除……35
1.1.3 文字の挿入……35
1.2 知っていると便利な編集機能……36
1.2.1 カーソルの移動……36
1.2.2 文字の削除……36
1.2.3 文字の挿入……36
1.2.4 行の分割……36
1.2.5 行の結合……37
1.2.6 コピー……38
1.2.7 行間への新しい行の追加……38
1.2.8 EDIT 機能……38
1.2.9 行番号の整理……39
1.3 全角文字の編集……39
1.3.1 全角文字の表示……40
1.3.2 全角文字上のカーソルの移動……40
1.3.3 全角文字の削除……40
1.3.4 全角文字の挿入……40
1.3.5 全角文字の修正……40
1.4 コントロールコード表……42

算2章 ディスプレイ……44
2.1 ディスプレイモード……44
2.1.1 テキスト画面……44
2.1.2 グラフィック画面……45
2.1.3 グラフィック VRAM の構成……47
2.1.4 カラーモード……50
2.1.5 マルチページモード……51
2.1.6 カラーコード……52
2.1.7 パレットコード……53
2.1.8 中間色コード……54

2.2 テキスト……56
2.2.1 テキスト画面……56
2.2.2 テキストの属性……60
2.2.3 グラフィックと文字表示……64
2.3 グラフィック……65
2.3.1 画面の初期化……65
2.3.2 グラフィック座標系……65
2.3.3 ユーザ座標系と画面座標系……67
2.3.4 グラフィック命令の色指定……70
2.3.5 点を打つ……71
2.3.6 線を描く……72
2.3.7 正多角形を描く……76
2.3.8 円を描く……78
2.3.9 色を塗る（中間色の指定）……79
2.3.10 色を瞬時に変える……80
2.3.11 テキスト画面とグラフィック画面の優先順位を決める……82
2.3.12 文字パターンの描画……82
2.3.13 GET@とPUT@……84
2.3.14 グラフィックパターンの描画……84

第3章 日本語処理……86
3.1 全角文字とは……86
3.2 全角文字の入力……87
3.2.1 漢字の入力……87
3.2.2 ひらがなの入力……91
3.2.3 カタカナ・英数字の入力（全角文字）……91
3.3 全角文字の入力方式……96
3.3.1 ひらがな・カタカナ・英数字などの全角文字の入力方式……96
3.3.2 漢字への変換機能……96
3.3.3 日本語入力モードに入るための条件……97
3.4 入力モードの選択……98
3.4.1 HELP について……98
3.4.2 各入力モードの選択……99
3.5 漢字変換方式について……100
3.5.1 コード入力方式……101
3.5.2 一字変換方式……103
3.5.3 音訓変換方式……105
3.6 ミニ・ワープロ機能……108

第4章　ファイルについて……110
4.1　ファイル管理……110
4.1.1　階層ディレクトリ……110
4.1.2　カレントディレクトリ……112
4.1.3　ディレクトリの一覧……114
4.1.4　ディレクトリの作成……115
4.1.5　ディレクトリの削除……117
4.1.6　ファイルディスクリプタ……117
4.1.7　デバイス名の省略……120
4.1.8　ファイルの削除……121
4.1.9　ファイル名の付け換え……121
4.1.10　ファイルの属性指定……121
4.2　プログラムファイル……123
4.2.1　プログラムの保存……123
4.2.2　プログラムの再生……126
4.2.3　プログラムのベリファイ……127
4.2.4　プログラムのマージ……128
4.2.5　プログラムのチェイン……130
4.3　シーケンシャルアクセスファイル……132
4.3.1　シーケンシャルアクセスファイルの作成……132
4.3.2　シーケンシャルアクセスファイルの読み込み……134
4.3.3　シーケンシャルアクセスファイルへの追加……136
4.4　ランダムアクセスファイル……137
4.4.1　ランダムアクセスファイルの作成……138
4.4.2　レコードの割り付け……138
4.4.3　ランダムアクセスファイルへの書き込み……139
4.4.4　ランダムアクセスファイルの読み出し……140
4.4.5　ランダムアクセスファイルへのデータの追加とデータの変更……143

第5章　フリーエリアについて……146
5.1　グラフィックVRAMとフリーエリア……146
5.1.1　BASICの起動直後……146
5.1.2　プログラムロード後……146
5.1.3　プログラム実行後……146
5.2　オプションスクリーンとフリーエリア……148
5.3　グラフィック描画と外部記憶……150
5.4　日本語入力とフリーエリア……150
5.5　NEWONとフリーエリア……151
5.6　VDIMとフリーエリア……153
5.7　フリーエリアの大きさを確認するには……154

第6章　機械語サブルーチンとモニタ……155
6.1　機械語モニタ……155
6.2　機械語サブルーチンの入力……165
6.2.1　機械語サブルーチンの記憶領域の設定……165
6.2.2　機械語サブルーチンの入力方法……166
6.2.3　BASICプログラムから機械語サブルーチンを呼び出す方法……167

第3部 応用編……173

第1章　ミュージック……174
1.1　ブザー音を出す……174
1.2　楽譜を演奏する……174
1.3　PSGのコントロール……176

第2章　ディスクの使い方……183
2.1　ディスクの使用準備……183
2.2　ディスクドライブ接続構成……183
2.3　システムプログラム起動方法……184
2.4　ディスクモードスイッチの設定……187
2.5　接続ドライブのディスクタイプ設定……189
2.6　ディスクのフォーマット構成……192
2.7　ディスクファイル管理方法……194
2.7.1　ディレクトリ……195
2.7.2　FAT……197

第3章　RS-232Cインターフェイスの使い方……199
3.1　接続方法……199
3.1.1　接続例……199
3.1.2　RS-232Cインターフェイス信号端子表……199
3.1.3　接続方法……201
3.2　RS-232Cインターフェイスのデータ信号フォーマット……202
3.3　BASICでRS-232Cインターフェイスを扱う方法……202
3.4　入出力命令と割り込み処理……205
3.4.1　RS-232Cインターフェイスに関する入出力命令……205
3.4.2　割り込み処理……208
3.5　ユーザが機械語でRS-232C用ソフトを作成する場合の使用法……209
3.5.1　CTC（Z80A-CTC）の設定……210
3.5.2　SIO（Z80A-SIO）の設定……211
3.6　割り込み処理の使用……213

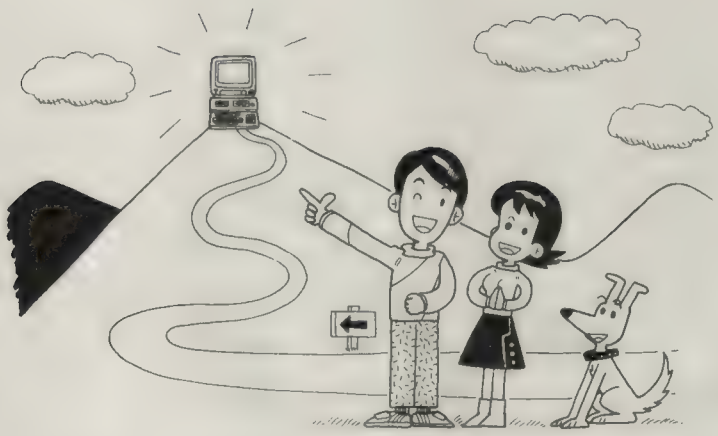
第4章　マウスの使い方……214
4.1　マウスとは……214
4.2　マウスを使う……214
4.2.1　マウス関数の書式と機能について……214
4.2.2　マウス関数を動かす……218
4.2.3　マウスカーソルを表示させる……219
4.3　注意事項……220

第5章　デジタルテロッパーの使い方……221
5.1　特徴……221
5.2　各部名称……221
5.3　VTR録画モードスイッチ……222
5.4　使用方法……223
5.5　黒抜き表示……225
5.6　注意事項……225

付録……232
A1　テキスト画面(VRAM)へのアクセス……232
A2　組み込み関数以外の数学的関数……235
A3　数値精度の変換……236
A4　数値データの誤差……237
A5　メモリマップ……239
A6　ユーザ定義文字の作り方……241

索引……247

第1部入門編



第1章 プログラミングを始めよう

1.1 コンピュータへのメッセージ

まず、本体に電源をいれてディスクBASICを起動します。起動の方法が分からなかったり、忘れてしまった場合には、『取扱説明書』の「第2章 動かしてみる」を参照してください。

ディスクBASICが立ち上がると、画面にNEWON■と表示されます。

```
SHARP HuBASIC  CZ-8FB02 Version1.0 [2HD]
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P


NEWON■
```

ここで [↵] (キャリッジリターンキー) を押してください。するとコンピュータからOkというメッセージが返ってきます。

画面上で四角く点滅しているものを**カーソル**といいます。カーソルは、コンピュータがあなたからのメッセージを待っている状態を表しています。

```
SHARP HuBASIC  CZ-8FB02 Version1.0 [2HD]
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P
74493 Bytes free

NEWON
Ok
■
```

では、手始めにキーを [C] [L] [S] の順に押してください。1文字押すごとにカーソルが1つずつ右へずれていきますね。図では大文字で表示されていますが、小文字を使ってもかまいません。

```
SHARP HuBASIC  CZ-8FB02 Version1.0 [2HD]
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P
74493 Bytes free

NEWON
Ok
CLS■
```

次に、[↵] を押してみてください。押した瞬間に、画面に書かれていた文字がすべて消えます。それから一番上の行にOkと表示され、その次の行にカーソルが点滅します。

```
Ok
■
```

"CLS"には「画面をきれいにする」という意味があります。このようにコンピュータが理解できる言葉を**キーワード**といいます。キーワードは全部で200以上あり、それぞれが、コンピュータに何かをさせる意味を持っています。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。

もしキーワードでない、でたらめな文字を打ち込んだらどうなるでしょう。

↵の記号は、「その位置で↵を押す」という意味です。

```
HSDAUZ↵
Syntax error
Ok
■
```

↵を押したとたんに「ピッ」と鳴って、上のように表示されました。これはコンピュータが、「"HSDAUZ"はキーワードではないので、何をやっていいのか分かりません」と言っているのです。

キーワードを打ち込んでコンピュータに何かをさせたいときには、必ず最後に↵を押してください。↵を押したときに初めてコンピュータに言葉が伝わるのです。↵を押すまではコンピュータは何も実行しません。このことを忘れないようにしましょう。

次にPRINTというキーワードを試してみましょう。

```
PRINT"X1"↵
X1
Ok
■
```

次の行にX1と表示されましたね。

では、

```
PRINT"23+44"↵
```

ではどうでしょう？　画面は下図のようになっています。

```
PRINT "23+44"
23+44
Ok
■
```

これでPRINTが、「引用符（"）で囲まれた文字をそのまま画面に表示しなさい」という意味を持っていることがお分かりいただけるでしょう。

それでは、続けて次のように打ち込んでください。

```
PRINT 23+44↵
```

今度は引用符で囲まなかったのに６７という数字が表示されました。よく見ると６７は２３＋４４という足し算を行った結果であることがわかります。

これは、ＰＲＩＮＴには「引用符（"）で囲まれた文字をそのまま表示しなさい」という意味のほかに「計算した結果を表示しなさい」という意味もあるからです。

なお、"ＰＲＩＮＴ"の代わりに"？"（疑問符）を用いてもよいことになっています。キーボードから [P]，[R]，[I]，[N]，[T] と５文字入れなければならないのが、[?] の１文字で済むのですからたいへん便利です。では試してみましょう。

```
?23+44 ↵
 67
Ok
■
```

同じ結果が出ましたね。

これまで見てきたように、キーワードはその名のとおり、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言葉です。キーボードから正しく打ち込んで ↵ を押せば、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るのです。

それでは、

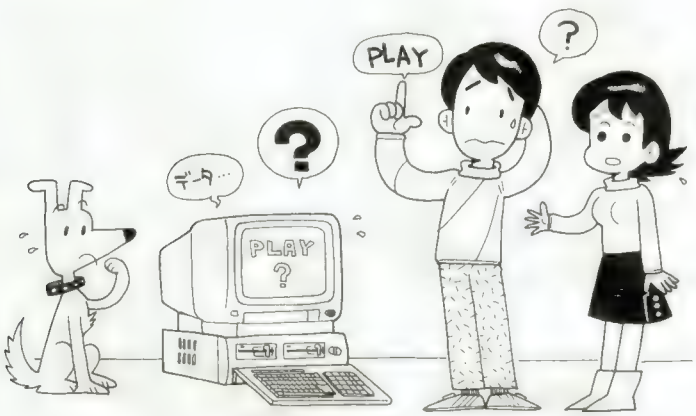
```
PLAY ↵
```

と打ち込んでください。

```
Missing operand
Ok
■
```

というメッセージが返ってきました。

コンピュータは「"ＰＬＡＹ"というキーワードの意味はわかるけれども、"ＰＬＡＹ"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。" Syntax error "のときもそうですが、間違ったキーワードを打ち込んで実行させようとしたり、必要なデータを書かなかったりすると、コンピュータはメッセージを表示して「間違っていますよ」と教えてくれるのです。



では、必要なデータをいっしょに打ち込んでみましょう。

```
PLAY"BG"
```

さあ、どうなりましたか？　何も起こらないという人は、音量調整を上げてもう一度打ち込んでください。〝シ〟の音と〝ソ〟の音で「シー・ソー」というメロディーが出るはずです。

ＰＬＡＹは音を出すためのキーワードです。音を表す文字を引用符（"）で囲んで指定すると、その音を出すことができます。音を表す文字は〝ＣＤＥＦＧＡＢ〟で、これが〝ドレミファソラシ〟に対応しています。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。これは先ほどの〝ＰＲＩＮＴ〟と使い方が似ています。



1.2　プログラムって何だろう

1.2.1　行番号を付けてみる

まず、次のように打ち込んでください。

```
PRINT"CHIME"
```

を押した瞬間に、引用符で囲まれた文字ＣＨＩＭＥが表示されます。
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くるわけでもありません。
今度は、

```
20 PLAY"BG"
```

と打ち込んでください。やはり何ごとも起こりません。でも、次のように打ち込むとどうなるでしょう。

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。

もう一度RUNと打ち込んで↵を押すと、やはりCHIMEと表示されて「シー・ソー」という音がでます。さらにRUN↵とすれば、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。

それでは

LIST↵

と打ち込んでみてください。

```
10 PRINT "CHIME"
20 PLAY "BG"
Ok
■
```

と画面に表示されましたね。コンピュータは先に入力したものをちゃんと覚えていたのです。ここでコマンドの前に付けた、10とか20の番号を行番号といいます。

実は、PRINTやPLAYの前に行番号を付けるか付けないかで、コンピュータの反応が大きく違ってくるのです。

1.2.2 直接モードと間接モード

行番号を付けないでキーワードを打ち込んで↵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。たとえば、

PRINT"CHIME"↵

と打ち込めば、その場でCHIMEと表示され、

PLAY"BG"↵

と打ち込めば、やはりその場で「シー・ソー」と音を出します。

このようにキーボードから打ち込むと、すぐに実行されるものを直接モード(ダイレクトモード)といいます。

しかし、行番号をつけると事情が変わってきます。コンピュータはそのときすぐには実行せず、後からRUN↵と打ち込んだときに実行します。

これは、コンピュータにはメインメモリという記憶装置があって、そこに行番号を付けた命令を覚えてくれるからです。

このように、いったんコンピュータに覚えさせておいて、あとから何度でも実行させることができるものを、間接モード（プログラムモード）といいます。

ここで、行番号の付いた

```
10 PRINT "CHIME"
20 PLAY "BG"
```

の2行のことを**プログラム**といい、行番号の後ろのPRINT"CHIME"やPLAY"BG"のことを**ステートメント**といいます。また、ダイレクトモードで使用できるキーワードを**コマンド**（または命令）といいます。コマンドは、プログラムをどうしたらいいのかをコンピュータに指示するときなどに使います。

1.2.3　プログラムって何だろう

プログラムというのは、コンピュータにどういう順序で仕事をさせるかを書いた手引書のようなものです。コンピュータはプログラムに書かれているステートメントを、行番号の小さい順に実行していきます。

たとえば、"MORNING"，"EVENING"，"NIGHT"を順に画面に表示させるプログラムを書いてみると、次のようになります。

```
10 PRINT "MORNING"
20 PRINT "EVENING"
30 PRINT "NIGHT"
40 END
```

ENDは、「そこでプログラムの実行をおしまいにします」という意味の命令です。

ここで、RUN ↵（プログラムを実行せよ）を入力すると、コンピュータは行番号10，20……の順に実行していきます。40行を実行し終わったときにプログラムの実行が終了します。

```
RUN ↵
MORNING
EVENING
NIGHT
Ok
■
```

各行を入力する順序を変えてもかまいません。行30，20，40，10の順番で入力しても、いざ実行するときには、ちゃんと行番号の小さい順に実行されるからです。

1.2.4　変数を使う

プログラムの中では、いろいろな数値を扱いますが、値の決まっていないものや、途中で値が変わるものは変数というかたちで扱うと便利です。

次の例は足し算をさせるプログラムですが、この中のA，B，Cの3つが変数です。変数は値を入れておくための箱のようなもので、別の値を入れ直さない限りいつまでもその値を覚えています。

変数にはアルファベット、またはアルファベットと数字の組合わせで名前を付けます。ただし、変数の名前は240字以下で、最初の1文字はアルファベットでなければならない、という約束になっています。

```
10 A=3
20 B=5
30 C=A+B
40 PRINT C
50 END
```

このプログラムを実行すると、行１０で、変数Aには３という値が入れられます。行２０では変数Bに５という値が入れられます。プログラムに出てくる〝＝〟は、数学に出てくる〝＝〟とは少し意味が違って、右辺の計算式を計算してその結果の値を、左辺の変数に代入するという意味になります。

行３０ではA＋Bを計算した結果の値、つまり８が変数Cに入れられます。そのCの値（つまり８）が、行４０のPRINT命令で表示されることになります。

1.3　プログラムの作成(その1)

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。

先ほど入力したプログラムはたった５行でしたが、大きなプログラムになると、何十行、何百行（あるいは何千行）にもなることがあります。ところが、大きくて複雑なプログラムを１回も間違わずに入力することなどできません。必ず途中で追加、変更、削除などの編集を行わなければならなくなります。

ここでは小さなプログラムを例に、プログラムを編集する方法を具体的に説明しましょう。

まずNEWという命令を使って、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去りましょう。

NEW ↵

試しに、

LIST ↵

と打ち込んでください。

```
Ok
■
```

と表示されるだけです。プログラムがきれいに消えていることが確認できましたね。

次は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

てください。

```
10 PRINT"X1" ↵
```

プログラムは、1行入力するごとに正しく入力されたかどうか見直しましょう。間違いが見つかったら、面倒でももう一度入力し直してください。もっと簡単に直す方法もあるのですが、それについては後で説明します。

正しく入力されたことを確かめたら実行してみましょう。

```
RUN↵
X1
Ok
■
```

と表示されたらOKです。もし、

```
Syntax error in 10
Ok
```

とメッセージがでたら打ち込んだ中に間違いがあったということですから、再び入力し直します。

ここでもう1行入力してみます。

```
20 PLAY"BGADDR" ↵
```

その後で、LIST↵と入力すれば

```
10 PRINT "X1"
20 PLAY "BGADDR"
```

と表示されるはずです。

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「シ・ソ・ラ・レ・レ」というメロディーが流れます。

もう1行増やしてみましょう。

```
30 PLAY"DE#FGGR" ↵
```

LIST↵とすれば次のように表示されます。

```
10 PRINT "X1"
20 PLAY "BGADDR"
30 PLAY "DE#FGGR"
Ok
■
```

これで、プログラムが3行になりました。

RUN↵と入力して、このプログラムを走らせる（プログラムを実行させることを「プログラムを走らせる」ともいいます）とＸ１と表示して、先ほどより長いメロディーが流れます。うまく流れましたか？

1.3.2　１行挿入

Ｘ１と表示してからメロディーを流すまでの間に、ＣＨＩＭＥと表示させることにします。次のように入力してください。

```
15 PRINT"CHIME"↵
```

ＬＩＳＴ↵とすると、

```
10 PRINT "X1"
15 PRINT "CHIME"
20 PLAY "BGADDR"
30 PLAY "DE#FGGR"
```

行１０と行２０の間に新しい行１５が挿入されています。プログラムが４行に増えました。

それでは実行してみましょう。

```
RUN↵
X1
CHIME
```

と表示されてから、メロディーが流れましたね。

このように行を挿入したい場合は、その前後の行番号の間の数字を行番号として命令を入力すればよいのです。たとえば、１０と２０の間に挿入するのであれば、１１から１９までの行番号を用いることができます。

1.3.3　１行削除

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でしょうか。最も簡単な方法はプログラムから行１０を取ってしまえばよいのです。そのためには、消したい行番号のみを入力します。

```
10↵
```

ここでＬＩＳＴ↵とすれば、

```
15 PRINT "CHIME"
20 PLAY "BGADDR"
30 PLAY "DE#FGGR"
Ok
■
```

行１０が削除されて３行のプログラムになっています。走らせてみましょう。

```
RUN↵
CHIME
Ok
■
```

Ｘ１を表示しないでメロディーが流れました。

このように、プログラムから１行削除するには、その行番号のみを入力して↵を押せばよいのです。

1.3.4　１行変更

行３０のＰＬＡＹのメロディーを変えてみましょう。そのためには、行３０を新たに入力し直します。

```
30 PLAY"DABGGR"↵
```

ＬＩＳＴ↵とすれば、

```
15 PRINT "CHIME"
20 PLAY "BGADDR"
30 PLAY "DABGGR"
Ok
■
```

行３０がいま入力した内容に変更されています。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↵
CHIME
Ok
■
```

1.3.5　行番号の付け直し

行の挿入や削除、変更を繰り返したので行番号が１０番おきではなくなってしまいました。この場合、次のように入力してみましょう。

```
RENUM↵
Ok
■
```

あれっ、何をしたのかな？　ＬＩＳＴ↵としてみましょう。

```
10 PRINT "CHIME"
20 PLAY "BGADDR"
30 PLAY "DABGGR"
Ok
■
```

プログラムの中身そのものは変わっていません。しかし、よく見るとプログラムの行番号が変わっています。

```
RUN
```

と入力して走らせると、前と同じことをやってくれます。

RENUMは、「プログラムの行番号を付け直しなさい」という意味を持つ命令なのです。たとえば１０という行と１１という行があったとしたら、この間には、新しい行を挿入することができません。そんなときにこの命令を使います。

さて、またプログラムを増やしてみましょう。

```
40 PLAY"GBADDR"
50 PLAY"DABGGR"
60 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 PRINT "CHIME"
20 PLAY "BGADDR"
30 PLAY "DABGGR"
40 PLAY "GBADDR"
50 PLAY "DABGGR"
60 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、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

```
RENUM 100
```

と入力します。LIST とすると、

```
100 PRINT "CHIME"
110 PLAY "BGADDR"
120 PLAY "DABGGR"
130 PLAY "GBADDR"
140 PLAY "DABGGR"
150 PRINT "END"
Ok
■
```

プログラムの中身は変わらず行番号だけが変わっています。走らせてみると先ほどと同じチャイムのメロディーが流れます。

1.3.6　複数行削除

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という命令を使えばよいのですが、もし行番号１１０～１５０の行だけを消したいときはどうすればよいでしょうか。

消したい行の行番号をひとつひとつ入力していくのもよいでしょうが、もっと簡単な方法があります。それはDELETEという命令を使う方法です。この令令を使えば、プログラムのいらなくなった行をまとめて消すことができます。

１１０～１４０の行を消したいときは、次のように入力します。

```
DELETE 110-140 ↵
```

これで、コンピュータは１１０～１４０の行をすべて消しました。確認のためプログラムを見てください。

```
100 PRINT "CHIME"
150 PRINT "END"
Ok
■
```

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

プログラム編集方法のまとめ（その１）

- １行挿入するには
  挿入したい行の上と下の行の、それぞれの行番号の間の行番号を選ぶ。
  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。
- 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。
- １行変更するには
  変更したい行と同じ行番号で新しく入力する。
- 何行かまとめて削除するには
  DELETE命令を使う。
  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。

1.4 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おきましょう。

```
NEW ↵
Ok
CLS ↵
Ok
```

1.4.1 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 PLEY "CRCRGRGRARARGGRR" ↵
```

ここでPLAYと入力するところを、誤ってPLEYと入力したことに気付いたとします。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…」と行番号から入力し直すのも1つの方法です。しかし、もっと簡単な方法があります。

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描いてあるキーで、⇧ ⇩ ⇦ ⇨ の4つがあります。カーソルコントロールキーを用いると画面上のカーソルを、つまり、次に文字を打ち込む位置を上下左右の好きな方向に動かすことができます。

PLEYの"E"を"A"に直すためには、まずカーソルコントロールキーを使って、カーソルをPLEYの"E"のところまで持っていきます。次の状態で⇧を1回、⇨を5回打ってください。

```
10 PLEY "CRCRGRGRGARARGGRR"
■ ←カーソル
```

ちょうど"E"のところに来たらOKです。もし、"E"を通り過ぎてしまっても、あわてないで4つのカーソルコントロールキーをうまく使って、"E"のところに持っていってください。

```
10 PLEY "CRCRGRARARGGRR"
```

ここでAを押すと"E"が"A"に変わり、PLAYとなります。カーソルは"Y"のところにきていますが、そのまま↵を押してください。これで修正が完了しました。

プログラムを見てみましょう。

```
LIST ↵
10 PLAY "CRCRGRGRGARARGGRR"
Ok
■
```

"PLAY"に書き換わっていますね。

さて、今度は1行を入力している最中、↵を押す前に入力の誤りに気付いた場合について見てください。

```
20 PLEY■
```

ここまで打って、PLAYの"A"が"E"になっている誤りに気づいたとき、それを直すには2通りの方法があります。

まず第1の方法は、⇦ を2回打ってカーソルを"E"のところに持って行き、 A と打ちます。

第2の方法は、INS DEL を2回打って、

```
20 PL■
```

"EY"の2文字を消してしまい、A , Y と打って修正します。

```
20 PLAY■
```

どちらかの方法で修正したら、再びプログラムの入力を続けることにしましょう。

1.4.2 効率のよい入力方法

引き続きプログラムを入力します。

```
20 PLAY "FRFRERERDRDRCCRR" ↵
30 PLAY "GRGRFRFRERERDDRR" ↵
```

ここで、たとえば次に入力する行40が、行30とまったく同じでよい場合を考えてみましょう。同じものを入力し直すのは面倒です。

そんなとき、カーソルを行30の"3"のところに持って行って 4 を打ち、↵ を押します。

```
20 PLAY "FRFRERERDRDRCCRR"
40 PLAY "GRGRFRFRERERDDRR"
```

これで30とまったく同じ行が、行40としてもう1つ作られました。

プログラムを見てみましょう。

```
10 PLAY "CRCRGRGRARARGGRR"
20 PLAY "FRFRERERDRDRCCRR"
30 PLAY "GRGRFRFRERERDDRR"
40 PLAY "GRGRFRFRERERDDRR"
```

どうです。40の行がきちんとできているでしょう。

次の行50は、行10とまったく同じにしたいのですが、どうすればよいしょう？ そうです。行10の"1"のところにカーソルを持ってきて、 5 と打って↵を押せばよいのです。

```
50 PLAY "CRCRGRGRARARGGRR"
20 PLAY "FRFRERERDRDRCCRR"
30 PLAY "GRGRFRFRERERDDRR"
40 PLAY "GRGRFRFRERERDDRR"
```

次の行60も行20とまったく同じにしてみます。いまちょうど、カーソルが行20の"2"のところにきていますね。その位置で 6 と打って↵を押してください。

```
50 PLAY "CRCRGRGRARARGGRR"
60 PLAY "FRFRERERDRDRCCRR"
30 PLAY "GRGRFRFRERERDDRR"
40 PLAY "GRGRFRFRERERDDRR"
```

するとカーソルは行３０の"3"の上にきましたね。この位置で何か入力すると、またプログラムが書き変わってしまいます。もうプログラムを変えたくないので画面を消してしまいましょう。SHIFT を押しながら CLR HOME を押してください。画面がすっかりきれいになりました。

ここで、プログラムを見てみましょう。

```
LIST
10 PLAY "CRCRGRGRARARGGRR"
20 PLAY "FRFRERERDRDRCCRR"
30 PLAY "GRGRFRFRERERDDRR"
40 PLAY "GRGRFRFRERERDDRR"
50 PLAY "CRCRGRGRARARGGRR"
60 PLAY "FRFRERERDRDRCCRR"
```

この曲は有名な「キラキラ星」です。このままこのプログラムを実行すると、「キラキラ星」が非常にゆっくりと演奏されます。

```
RUN
```

音楽を途中で止めたいときは、SHIFT を押しながら BREAK を押してください。

曲のテンポを速くするには、曲のテンポを指定する命令を入れます。

```
5 PLAY 300
```

ついでに、画面を消す命令と、タイトルを表示する命令も入れましょう。

```
3 CLS
4 PRINT"キラキラボシ"
```

カタカナ文字を入力するときには カナ を押して、カタカナ入力モードにしてください（再度 カナ を押すと、元のローマ字モードに戻ります）。

ＬＩＳＴをとってみます。

```
LIST
3 CLS
4 PRINT "ｷﾗｷﾗﾎﾞｼ"
5 PLAY 300
10 PLAY "CRCRGRGRARARGGRR"
20 PLAY "FRFRERERDRDRCCRR"
30 PLAY "GRGRFRFRERERDDRR"
40 PLAY "GRGRFRFRERERDDRR"
50 PLAY "CRCRGRGRARARGGRR"
60 PLAY "FRFRERERDRDRCCRR"
Ok
■
```

行番号をそろえて、

```
RENUM
Ok
■
```

再びＬＩＳＴをとってみましょう。

```
10 CLS
20 PRINT "ｷﾗｷﾗﾎﾞｼ"
30 PLAY 300
40 PLAY "CRCRGRGRARARGGRR"
50 PLAY "FRFRERERDRDRCCRR"
60 PLAY "GRGRFRFRERERDDRR"
70 PLAY "GRGRFRFRERERDDRR"
80 PLAY "CRCRGRGRARARGGRR"
90 PLAY "FRFRERERDRDRCCRR"
```

これを走らせると、今度はちょうどよい速さで「キラキラ星」が演奏されます。

1.4.3　文字の挿入

タイトルの「キラキラﾎﾞシ」を画面中央に表示したい場合、タイトルを書く位置をＬＯＣＡＴＥという命令を使って指定します。

画面の左上を基準にして、右に１２文字目、下に５行目の位置を指定するときにはＬＯＣＡＴＥ１２，５とします。次のように

```
15 LOCATE12,5
```

と１行挿入しても、もちろんよいのですが、２０の行番号とＰＲＩＮＴの間に「ＬＯＣＡＴＥ１２，５：」と挿入することもできるのです。″：″はステートメントの区切りを表します。１つの行には″：″で区切って、２つ以上のステートメントを書くことができるのです。これを、**マルチステートメント**といいます。

まず、カーソルを行２０のＰＲＩＮＴの″Ｐ″のところに持って行きます。そして SHIFT ＋ INS DEL を１２回押すと（ SHIFT を押したまま INS DEL を１２回打つ）、カーソルのあった″Ｐ″以降が１文字ずつ計１２文字分右にずれて、スペースがあきます。

```
20 ■                    PRINT"ｷﾗｷﾗﾎﾞｼ"
```

ここで″ＬＯＣＡＴＥ１２，５：″と入力して、 を押すと、行２０が修正されます。

```
20 LOCATE12,5:PRINT"ｷﾗｷﾗﾎﾞｼ"
```

プログラムを確かめてみましょう。

```
LIST
10 CLS
20 LOCATE12,5:PRINT "ｷﾗｷﾗﾎﾞｼ"
30 PLAY 300
40 PLAY "CRCRGRGRARARGGRR"
50 PLAY "FRFRERERDRDRCCRR"
60 PLAY "GRGRFRFRERERDDRR"
70 PLAY "GRGRFRFRERERDDRR"
80 PLAY "CRCRGRGRARARGGRR"
90 PLAY "FRFRERERDRDRCCRR"
```

さっそく実行してみましょう。画面中央に「キラキラホ゛シ」と表示されて音楽が流れます。

1.4.4 文字の削除

"LOCATE12，5："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 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、↵ を押すと行20が修正されます。

```
20  PRINT"キラキラホ゛シ"
```

SHIFT ＋ CLR HOME を押していったん画面を消してから、プログラムを確かめましょう。

```
LIST↵
10 CLS
20 PRINT "キラキラホ゛シ"
30 PLAY 300
40 PLAY "CRCRGRGRARARGGRR"
50 PLAY "FRFRERERDRDRCCRR"
60 PLAY "GRGRFRFRERERDDRR"
70 PLAY "GRGRFRFRERERDDRR"
80 PLAY "CRCRGRGRARARGGRR"
90 PLAY "FRFRERERDRDRCCRR"
```

これまでは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。プログラムを作るということがどのようなことか、ある程度お分かりいただけたかと思います。

1.4.5 困ったとき助けてくれる CTRL ＋ D シ キー

本機を使用されていく途中、次表のようなことで困ったことが生じるかもしれません。

1	キー入力された文字がすべて点滅文字になってもとにとどらないとき。
2	キー入力された文字がすべて反転文字になってもとにとどらないとき。
3	キーから入力された文字の内容が、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とき。
4	キーから入力した文字が標準の大きさでなく、倍文字となるとき。
5	キーから入力した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色と合わないとき。
12	LIST，FILESコマンドなどによって、グラフィック画面が表示されなくなったとき。

これらは 無意識のうちにコンピュータをそのような状態にさせた結果ですから、次のような命令で正常に戻してください。

CTRL ＋ D シ　そのあと　SHIFT ＋ CLR HOME

プログラムの編集方法のまとめ（その2）

●カーソルはカーソルキーを使って自由に移動できる

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

●文字を挿入するには

カーソルを挿入したい部分のすぐ後ろに持ってきて、SHIFT ＋ INS DEL を必要な回数だけ押す。

●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ろに持ってきて、INS DEL を必要な回数だけ押す。

●画面を消す

SHIFT ＋ CLR HOME　(=CLS↵)

●実行中のプログラムを止める

SHIFT ＋ BREAK

コマンドのまとめ

いままで出てきた基本的なコマンドをまとめてみましょう。

CLS↵	画面をきれいにする。
NEW↵	記録されたプログラムを消す。
LIST↵	記録されているプログラムを画面に表示する。
RUN↵	プログラムを実行する。
RENUM↵	プログラムの行番号を付け直す。
DELETE n−m↵	プログラムのn番からm番までを消す。

第2章 プログラムの保存と再生

前章ではプログラムの入力と実行の方法を学びました。しかし、そのまま電源を切ってしまうと、コンピュータはせっかく入力したプログラムをきれいさっぱり忘れてしまいます。

ですから、どこかにプログラムを保存しておかないと、同じプログラムをまた最初から打ち込まなければならなくなってしまいます。

プログラムを保存させるためのものを外部記憶装置といいます。コンピュータに覚えさせたプログラムを、外部記憶装置に保存することをプログラムのセーブといいます。また、セーブしたプログラムを再びコンピュータに再生させることを、プログラムのロードといいます。ここでは外部記憶装置として、フロッピーディスクとカセットテープを用いたセーブとロードの方法について説明します。

なお、カセットテープを使用する場合は、オプションの専用データレコーダ(CZ－8RL1)が必要です。

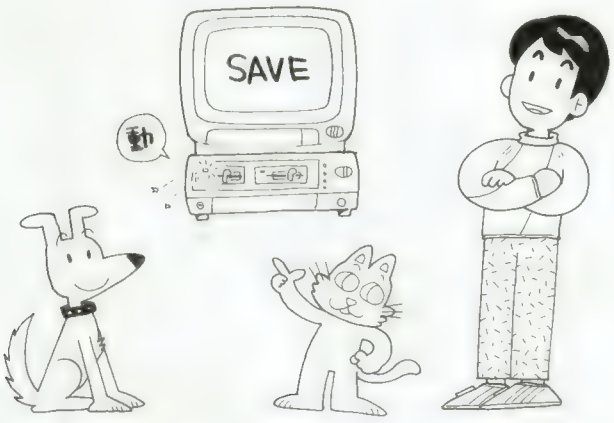
2.1 フロッピーディスクに保存、再生する場合

2.1.1 フロッピーディスクにセーブする

次のプログラムを入力したとします。

```
10 INPUT "A=";A
20 INPUT "B=";B
30 C=A+B
40 PRINT "A+B=";C
50 END
```

プログラムをフロッピーディスクに保存することをセーブといいます。それではセーブしてみましょう。



まず、フォーマット[注1]されたフロッピーディスクを本体のドライブ０にセットします。ただし、ライトプロテクト[注2]のかかったフロッピーディスクにはセーブすることはできません。

プログラムをセーブするにはＳＡＶＥ命令を使います。プログラムには適当な名前（ただし１３文字以内）を付けなければなりません。このプログラムの名前を〝タシサン ノフ゜ログ〟ラム〟としましょう。

次のように入力します（ここで、〝０：〟というのはデバイス名で、フロッピーディスクドライブのドライブ０のことです。

```
SAVE"0:タシザンノプログラム"
```

保存するデバイス名（フロッピーディスク０番目）　保存するプログラムの名前

すると、フロッピーディスクドライブのインジケータが点灯し、書き込みを始めます。そして、画面にＯＫと表示されれば、プログラムがフロッピーディスクにセーブされたことになります。

（注１）フォーマット

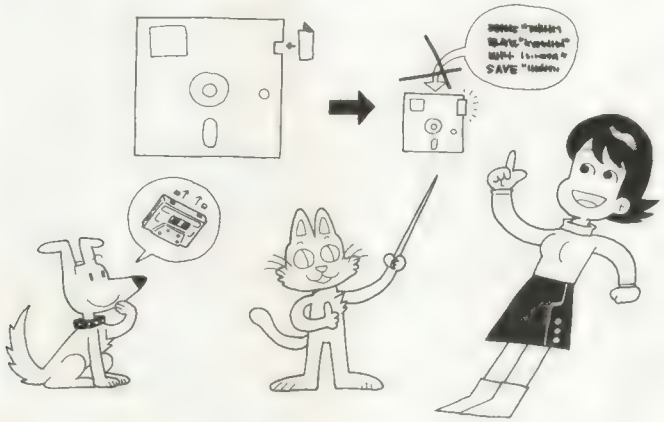
フロッピーディスクを初期化して、使用可能な状態にすることです。

フォーマットの仕方については『アプリケーションソフト説明書』の「ディスクユーティリティ」を参照してください。

（注２）ライトプロテクト

フロッピーディスクへの書き込み禁止。

ライトプロテクトノッチに保護用紙（シール）を貼っておくと、書き込みはできなくなります。



2.1.2　フロッピーディスクからロードする

フロッピーディスク等の外部記憶装置にセーブしていたプログラムを、メインメモリに読み込んで実行できる状態にすることを、ロードするといい、ＬＯＡＤ命令を使います。

先ほどセーブした〝タシサ〟ンノプ゜ロク〟ラム〟をロードしてみましょう。

```
LOAD"0:タシザンノプログラム"↵
```

再生するプログラムの名前

これによって、先ほど入力したプログラムが再生され、実行させることができるようになります。

ただし、この命令を実行すると、今まで入力していたプログラムはすべて消えてしまいますので注意してください。システム起動直後にロードする場合はかまいませんが、すでにプログラムを打ち込んでいる場合には、いったんプログラムをセーブしてから新たにプログラムをロードするようにしてください。

2.1.3 フロッピーディスクの内容を見る

フロッピーディスクにはいくつものプログラムをセーブすることができます。そしてフロッピーディスクの中に、どんなプログラムがセーブしてあるかを見たいときは、ＦＩＬＥＳという命令を使います。

プログラムの入ったフロッピーディスクをドライブ０にセットして、

```
FILES"0:"↵
```

と入力します。

この命令を実行すると、フロッピーディスクに入っているプログラムの名前（ファイル名）の一覧表を画面上に表示します。

プリンタにファイル名を出力するには

```
LFILES"0:"↵
```

と入力します。

なお、ディスクＢＡＳＩＣ起動後の初期状態ではファンクションキー［F1］を押すことによって、ＦＩＬＥＳ"０："↵を実行することができます。

2.2 カセットテープに保存、再生する場合

2.2.1 カセットテープにセーブする

前述の、〝タシサ〟ンノプ゜ロク〟ラム〟というプログラムをカセットテープに保存するには、やはり、ＳＡＶＥ命令を使います。

ツメを折っていないカセットテープを専用データレコーダにセットし、テープカウンタを〝０００〟としておきます。

次のように入力します。

```
SAVE"CAS:タシザンノプログラム"
```

プログラムを保存するデバイス名(カセット)　保存するプログラムの名前

するとデータレコーダが自動的に動作を開始します。
画面には

```
Writing "ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ"
```

と表示されます。
数秒後にデータレコーダは自動的に停止し、画面にはOkと出ます。これで"タシザンノプログラム"はカセットテープにセーブできました。

2.2.2　カセットテープの内容を確認する

以上でプログラムのセーブは終了したわけですが、うまくセーブされてなかったら……と考えると心配になります。せっかくのプログラムが、正しくセーブされていなければ困りますね。
LOAD?という命令を用いて、確実にセーブできているかどうかを確認することができます。

まず、カセットテープをテープカウンタ"000"まで巻き戻して、次のように入力します。

```
LOAD?"CAS:タシザンノプログラム"
```

確かめるプログラムの名前

すると自動的にデータレコーダが作動し、画面には

```
Found   "ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ"
```

と表示されます。
正しくセーブされていれば、数秒後にOkが表示されます。
しかし、もしカセットテープなどに問題があってうまくセーブされていないときは、次のメッセージが出ます。そのような場合は、カセットテープを交換してもう一度セーブをし直してください。

```
Tape read error
```

2.2.3 カセットテープからロードする

プログラムをカセットテープから再生するときには、次のように入力してください。

```
LOAD"CAS:タシザンノプログラム"
```

再生するプログラムの名前

データレコーダが自動的に作動し、画面に次のようなメッセージが出力されます。

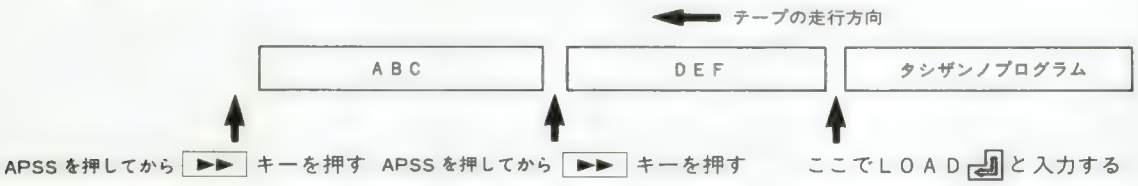
```
Found  "ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ"
```

数秒後にデータレコーダの動作が止まり、Ｏｋの表示が出てロードが完了したことを知らせます。

なお、１本のカセットテープに何種類かのプログラムが入っていて、その後ろに先ほどの〝タシサ〟ンノフ゜ロク〟ラム〟をセーブした場合、ロードの際には目的のプログラムに到達するまで、関係のないプログラムを読み飛ばします。これをスキップといいます。

```
LOAD "CAS:ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ"
SKIP "ABC"
SKIP "DEF"
Found  "ﾀｼｻﾞﾝﾉﾌﾟﾛｸﾞﾗﾑ"
```

上の例のように、〝タシサ〟ンノフ゜ロク〟ラム〟が３番目に入っていることがわかっているのであれば、専用データレコーダのＡＰＳＳボタンを押してから [►►] (ＦＦ)キーを押すことによって、次のプログラムの頭出しができます。これを２回繰り返すことで、〝タシサ〟ンノフ゜ロク〟ラム〟の頭の部分に到達することができます。



2.2.4 カセットテープの内容を見る

プログラムを記録したカセットテープの内容を見たいときにはＦＩＬＥＳ命令を使います。

プログラムの入ったカセットテープをデータレコーダにセットし、テープを巻戻してから

```
FILES"CAS:"
```

と入力します。

すると、カセットテープに入っているプログラムの名前（ファイル名）の一覧表を、画面上に順に表示します。

なお、プリンタにファイル名を出力するには、

```
LFILES"CAS:" ↵
```

と入力します。

第3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本体に接続した場合、キーボードから直接、またはＢＡＳＩＣの命令を用いて自由にテレビをコントロールすることができます。

さらに、本機にはカレンダー付きタイマ機能が付いているので、これらを組み合わせることによって、専用ディスプレイテレビをコントロールすることが可能になります。

本章では、内蔵されているテレビタイマコントロールを使うものと、ＢＡＳＩＣの命令を使うものとの２通りの説明をします。

3.1 キーボードからテレビをコントロールする

キーボードからテレビをコントロールするには、SHIFT とテンキーを組み合わせて使用します。

SHIFT + 0 ：消音（一度押すとテレビの音が消え、もう一度押すと音が出ます）。
SHIFT + 1 ～ 9 ：チャンネル１～９を選局します。
SHIFT + / ：チャンネル10を選局します。
SHIFT + * ：チャンネル11を選局します。
SHIFT + − ：チャンネル12を選局します。
SHIFT + + ：スーパーインポーズ画面に切り換えます。
SHIFT + = ：テレビ画面に切り換えます。
SHIFT + . ：コンピュータ画面に切り換えます。
SHIFT + ⇦ ：１つ前のチャンネルに切り換わります。（12,11→１）
SHIFT + ⇨ ：１つ次のチャンネルに切り換わります。（１，２→12）
SHIFT + ⇧ ：音量が大きくなります。
SHIFT + ⇩ ：音量を小さくなります。

3.2 テレビタイマコントロール

3.2.1 テレビタイマコントロールの呼び出し

クロックおよびタイマの呼び出しは、ＢＡＳＩＣを読み込んでいるかどうかにより操作が異なってきます。

ＢＡＳＩＣを読み込んでいない場合

フロッピーディスクをセットしないで本機を立ち上げた場合、ディスプレイ画面上に次のメッセージが表示されます。

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

ここで、[T]を押して、Timerを選択してください。テレビタイマコントロール画面になります。

BASICを読み込ませてある場合

すでにBASICが起動している場合には、ASK命令を用います。

ASK [↵]

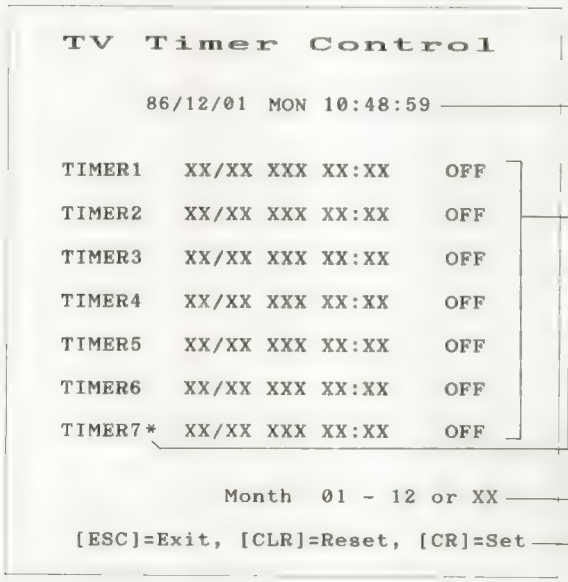
と入力してください。これでテレビタイマコントロールの画面になります。

3.2.2 テレビタイマコントロールの表示内容

画面は下図のように変わります。カーソルが

```
TIMER1    XX/XX XXX XX:XX    OFF
```

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）**
××／○○／○○ MON ○○：○○：○○
年 月 日 曜日 時 分 秒

●**タイマー表示部（白文字）**
××／×× ××× ××：×× OFF
月 日 曜日 時 分 TVの入・切表示

●**タイマ設定表示部（赤色）**
タイマの設定が行われると
＊（アスタリスク）が表示されます。

●**入力モード表示部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きます。

●**入力モード表示部B（緑文字）**
〔ESC〕[ESC]キーはクロック・タイマ表示を解除できます。
注）BASICからASK [↵] によってテレビタイマを呼びだした場合はBASICにもどります。
IPLより直接[T]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セットできます。

※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ので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してください。

3.2.3　クロック（時計）を現在時刻に合わせる

本機を購入後、初めて電源を入れたときや、コンピュータ内蔵のクロックを現在時刻に合わせる必要があるときには、次の手順に従って設定してください。

ここでは１９８７年１月１２日（月曜日）午後３時３０分２０秒に合わせる例を説明します。

入力モード表示内容

```
                Year      00-99
                Month     01-12 or xx
                Day       01-31 or xx
SUN MON TUE WED THU FRI SAT or xxx
                Hour      00-23 or xx
                Minute    00-59
                Second    00-59
```

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１９８７年の末尾２桁の [8] [7] 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 月曜日の [M] [O] [N] 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。
⑦ 午後３時３０分の分の単位の [3] [0] をキー入力します。
⑧ 午後３時３０分２０秒の秒の単位の [2] [0] をキー入力します。
⑨ 以上を設定し終えたら、設定時刻の３時３０分２０秒になるのを待ちます。その時刻になった瞬間に [↵] キーを押すと押した時点からクロックが作動を始めます。これでクロックの設定は終了し、カーソルがＴＩＭＥＲ１の最初の×のところへ移動します。

3.2.4　タイマで番組予約をする

クロックが作動しましたので、今度は専用ディスプレイテレビの番組予約をするために、タイマ設定を行ってみましょう。

ここでは先ほどクロックを設定した翌日、つまり、１９８７年１月１３日（火曜日）の午前９時３０分からはじまる５チャンネルの１時間番組を予約することにしましょう。

入力モード表示内容

```
                Month    01-12 or xx
                Day      01-31 or xx
SUN MON TUE WED THU FRI SAT or xxx
                Hour     00-23 or xx
                Minute   00-59
                 TV Power ON? (Y or N)
```

① カーソルはタイマ表示部の"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"×"のところまで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 火曜日の [T] [U] [E] 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 [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、" TV Power ON? （Y or N）"に答えて [Y] をキー入力します。
⑧ "ＯＦＦ"が"ＯＮ　ＣＨ"に変わり、その後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。
⑨ 最後に [↵] キー入力すると、この行のタイマがセットされたことを示す赤色の"＊"（アスタリスク）が"ＴＩＭＥＲ１"の後に表示され、カーソルは次の行の先頭に移ります。

```
TIMER1*  01/13 TUE 09:30   CH5
```

このようにタイマセットをすると、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、タイマが動作中であることを知らせます。

3.2.5　タイマでスイッチをＯＦＦにする

今度は、専用ディスプレイテレビを指定された時刻にＯＦＦさせるようにタイマをセットしてみましょう。

先ほどの手順と同じように、

```
TIMER2*  01/13 TUE 09:30   OFF
```

と入力してもよいのですが、次の方が簡単です。

⑩ カーソルは"ＴＩＭＥＲ２　　××……"の最初の"×"のところにきています。
⑪ カーソルを右に移動し、１０時３０分の部分のみ入力します。

```
TIMER2*  XX/XX XXX 10:30   OFF
```

[N] または [↵] の入力でＴＩＭＥＲ２はセットされます。

```
TIMER2   XX/XX XXX 10:30   OFF
```

ただし、この方法では日付けを省略しているために、毎日午前１０：３０に専用ディスプレイテレビがＯＦＦになるようタイマが働きますので注意してください。このように、日付と曜日を同時に省略することで「毎日」、また日付だけを省略して曜日を指定することで「毎週指定した曜日」にセットすることができます。

では、次の応用を考えてください。

[応用例] 毎週火曜日の午前11時から午後２時までの間、10チャンネルを番組予約します。
この場合のタイマ設定は、次のようになります。

```
TIMER3*   XX/XX TUE 11:00    ON CH10
TIMER4*   XX/XX TUE 14:00    OFF
```

なお、一度セットされたタイマの内容は一度メイン電源を切るか、またはクリア（タイマの取り消し）しない限りは働き続けます。したがって、毎日繰り返すようなタイマ設定の場合は便利です。

3.2.6 タイマを取り消す

先ほどセットしたタイマを取り消す場合は次の手順で行います。ＴＩＭＥＲ１からＴＩＭＥＲ３を取り消すことを例に説明します。

① カーソルを"ＴＩＭＥＲ１"のところへ移動させます。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 を押しながら CLR HOME を押します。するとＴＩＭＥＲ１の内容は取り消され、表示はタイマ設定前の状態に戻ります。
③ カーソルは"ＴＩＭＥＲ２"の行へ移っていますので、同様に SHIFT を押しながら CLR HOME を押し、ＴＩＭＥＲ２を取り消します。
④ カーソルは"ＴＩＭＥＲ３"の行へ移っていますので、同様に SHIFT を押しながら CLR HOME を押し、ＴＩＭＥＲ３を取り消します。

注意 クロック表示とメイン電源のON／OFFについて
・初めてメイン電源を入れたときには、表示が現在時刻と違っていますが、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はクロック機能が働かなくなりますので、再設定が必要になります。
・本機のクロックは年号の自動切換表示は行いませんので、年号が改まるごとに新しい年号を設定しなければなりません。
・うるう年（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。
・メイン電源を切ると、年号が"××"に変わりますので再設定してください。

3.3　プログラムでテレビをコントロールする

3.3.1　現在時刻の設定

BASICプログラムから現在時刻を設定するには、次のようにします。

時刻を設定する

```
TIME$="12:34:56"        ←12時34分56秒
```

日付を設定する

```
DATE$="87/01/11"        ←1987年1月11日
```

曜日を設定する

```
DAY$="SUN"              ←日曜日
```

日付、曜日、時刻を表示する

現在時刻を表示するには、次のようにします。

```
PRINT TIME$,DATE$,DAY$
13:32:21          86/12/01          MON
Ok
■
```

3.3.2　タイマコントロール

BASICプログラムからテレビをコントロールするには、次のようにします。

テレビの電源のON／OFF

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

テレビ画面のコントロール

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、コンピュータ画面を重ねて表示します（スーパーインポーズ）
　　 3 …… テレビ放送とコンピュータ画面を重ねて表示します（スーパーインポーズ）

※SHIFT＋＋ の状態はCRT 2と同じです。

チャンネルのコントロール

CHANNEL n	(n＝1～12) テレビのチャンネルを選択します

音量のコントロール

VOL n	(n＝−62 (小) ～64 (大)) 音量を変化させます

スクロールのコントロール

スーパーインポーズ画面（CRT 2，CRT 3）のときコンピュータ画面をスクロールさせます。

SCROLL n	(n＝−3～3の整数) nが正のときは上方向にスクロールします 負のときは下方向にスクロールします nが0、省略時はスクロールを止めます

では、BASICプログラムでテレビをコントロールしてみましょう。次のプログラムを入力してください。

（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。

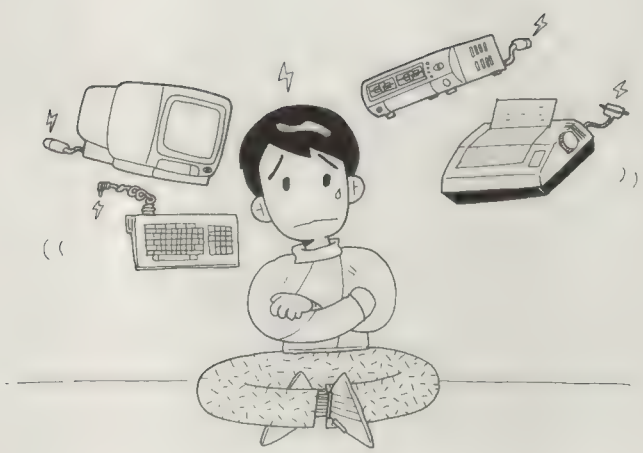
テレビタイマコントロールではできない機能を使って、少し工夫したプログラムを作ってみましょう。

（例2）

```
10 INIT:WIDTH 40,25,0,1:CLS 4
20 CRT 0:CHANNEL 5
30 CRT 2
40 CSIZE 3:LOCATE 0,0:PRINT #0,TIME$
50 GOTO 40
```

これは20行で設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。

第2部　基礎編



第1章　プログラムの編集

BASICプログラムを作成するときに、誤って隣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といった、打ち間違いをすることがよくあります。また、行やコマンドを新しくつけ加えたり、反対に削ったりしてプログラムを編集したいことがあります。このようなときのために、いろいろな画面編集（スクリーンエディット）機能が用意されています。

第1部第1章ではごく基本的な画面編集機能を紹介しましたが、この章ではより便利な機能を紹介していきます。

1.1　基本的な編集機能

1.1.1　カーソルの移動

●カーソルの移動

文字を修正するときは、まず修正したい文字の上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に移動します。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すと、一番上の行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行上へスクロールしカーソル位置は変わりません。

●リピート機能

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されて入力されます（リピート機能）。カーソルコントロールキーも、押している間カーソルが連続して移動します。リピート機能のON／OFFは、REPEAT命令で切り換えることができます。

REPEAT { ON ……リピート機能をONにします
　　OFF……リピート機能をOFF　にします

リピート機能をOFFにすると、1文字ずつしか受け付けなくなります。

●ホーム機能

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 を押します。また SHIFT + CLR HOME 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（上）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されます。

1.1.2 文字の削除

文字の削除を行う場合には、削除したい文字の次の文字の上にカーソルを移動させた後、INS DEL を押します。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。

1.1.3 文字の挿入

文字の挿入を行う場合には、挿入したい位置の次の文字の上にカーソルを移動させた後、SHIFT + INS DEL を押します。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。

以上のようなスクリーンエディット機能を用いてプログラムの訂正を行った場合には、各行の訂正が終わるたびに、必ず ↵ を押してください。何行分もまとめて一度に INS DEL で入力しても、↵ を押した行の内容しか訂正して記憶されません。必ず行番号単位で行ってください。

（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)■

← を押してカーソルを"L"の位置に移動させ、R を押します。

10 PRIINT CH$(65)

→ を押してカーソルを次の"I"の位置に移動させ、INS DEL を押して"I"を1文字削除します。

10 PRINT CH$(65)

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

10 PRINT CH■$(65)

R を押して"H"と"$"の間に文字 R を挿入します。

10 PRINT CHR$(65)

↵ を押してプログラムを記憶させます。

1.2 知っていると便利な編集機能

これまでに説明したスクリーンエディット機能を用いればプログラムの修正は可能ですが、さらに画面編集を効率よく、迅速に行うために便利な機能が用意されています。

1.2.1 カーソルの移動

●水平TAB

画面の初期状態では、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 を押すと、カーソルは現在位置から次のカーソル位置（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えます。

また、TAB位置は任意の桁に設定できます。まず、設定されているTAB位置を取り消す場合には、取り消す位置へカーソルを移動させ、CTRL + Y を押します。新しくTAB位置を設定する場合には、設定したい位置にカーソルを移動させ、CTRL + T を押します。

●1ワード単位でのカーソル移動

CTRL + F 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 を押すと、カーソルは現在位置より左（後ろ）にあるワードの先頭に移動します。ワードとは、空白、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。

1.2.2 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 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。

●カーソルのある位置以降の画面クリア

CTRL + Z 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。

1.2.3 文字の挿入

●インサートモード

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て CTRL + A を押します。するとインサートモードに設定され、それ以後文字が押されると、カーソルより右の文字列が右へ移動して、カーソル位置に挿入されます。インサートモードは、CTRL + S （BREAK と同じ）を押すか、または ↵ 、カーソルコントロールキーを押すと解除されます。なお、インサートモード時に INS DEL を押してもインサートモードは解除されません。

1.2.4 行の分割

1行を途中で区切って、2行に分けたい場合には、区切りたい位置へカーソルを移動させ CTRL + J を押します。すると、カーソルからその行の終わりまでが自動的に次の行へ移動します。移動

した行の先頭に行番号を付けて[↵]キーを押せば、新しい行ができます。

（例）次に示すプログラムをＣＩＲＣＬＥ命令とＰＡＩＮＴ命令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の"Ｐ"の位置に移動させ[CTRL]＋[J]を押します。するとＰＡＩＮＴ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]を押して、ＰＡＩＮＴ命令文の前に行番号を挿入するスペースを作り、行番号６０を入力します。

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

[↵]を押して、行番号５０，６０の文をプログラムとして記憶させます。

1.2.5　行の結合

２行にわたる文を１行に納めたい場合には、最初の行の上にカーソルを置いて[CTRL]＋[W]を押します。すると、その行とその次（下）の行が結合されます（画面上には変化が表われません）。これによって、この２行は同じ行として扱われます。[INS DEL]を用いて空白を削除すれば間を詰めることができます。

結合する２行のうち、前の行の文が画面の数行に渡るときには、カーソルを、その文の一番下の行にもっていって[CTRL]＋[W]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]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。

カーソルをＰＡＩＮＴの"Ｐ"の位置に移動させ[INS DEL]を押すと、ＰＡＩＮＴから右の部分が左に移動します。画面左端まで移動し、さらに押し続けると１行上の行番号５０行の画面右端よりＰＡＩＮＴ文が現われます。ＣＩＲＣＬＥ文の右にＰＡＩＮＴ文が並んだ時点で[↵]を押します。

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

これにより、行番号５０の文にＣＩＲＣＬＥ文とＰＡＩＮＴ文をプログラムとして記憶させることができました。しかし、行番号６０という行はまだ生き残っていますので、行番号６０の文を消

す必要があります。

60 ↵

行番号60のみ入力して↵を押すことにより、行番号60がメインメモリから削除されます。

1.2.6 コピー

●文字のコピー

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位置に、カーソルを移動し、[COPY]を押します。カーソルは点滅しなくなり、つきっぱなしの状態になります。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元の文字の位置まで移動させます。

↵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にコピーされ、両カーソルは1つ右へ移動します。↵を押すごとに1文字ずつコピーされていきます。コピーを終了するには再度[COPY]を押します。

ただし、画面上で2行以上に渡る文字を一度にコピーすることはできません。そのような場合は、以上のコピー操作を2回以上に分けて行ってください。

●ハードコピー

画面に表示されている文字やグラフィックを、プリンタにハードコピーします。

[SHIFT]＋[COPY]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]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]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。

ただし、以下の画面モードのときにはハードコピーはできません。

40文字×20行，80文字×20行，40文字×10行，80文字×10行

1.2.7 行間への新しい行の追加

表示画面の行と行の間を広げ、その間に新しく行を追加したい場合には、[ROLL UP]あるいは[ROLL DOWN]を使用します。

[ROLL UP]を押すと、カーソルのある行から上側が1行分上にスクロールアップします。

[ROLL DOWN]を押すと、カーソルのある行から下側が1行分下にスクロールダウンします。

これらによって行と行の間が広げられ、新しい行を追加できます。ただし、次の項に示すように、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。

1.2.8 EDIT機能

プログラムを実行させたとき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文を使用すれば、エラーの発生したプログラムの修正を能率よく行えます。

たとえば

EDIT ↵

と入力すると、エラーの発生した行を画面に表示してエディットモードに入ります。カーソルは行の先頭に置かれています。

エディットモードでは、ROLL UP あるいは ROLL DOWN を使用することにより、巻物を見るように画面に表示されている文の前後の文を表示できます。

エディットモードを解除するには SHIFT + CLR HOME あるいは SHIFT + BREAK を押してください。

エディットモードで新しく行を追加するには、CTRL + E または CTRL + Z を使用して画面に空白部分を作成してから、行番号を付けた文を入力し、最後に ↵ を入力してください。

また、EDIT機能はエラーが発生したときのみでなくプログラムの作成後、プログラムの見直しを行う場合に用いると便利です。LIST文の実行でもプログラムの見直しはできますが、LISTの場合には、画面上部へ消えていった行を表示することはできません。

EDIT文は次の形式で表します。

```
EDIT (n)  ..... nの行番号の行をエディットする
     (.)  ..... エラーが発生した行をエディットする
```

1.2.9　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命令を使用して行番号を整理することができます。RENUM命令は次の形式で表します。

```
RENUM [[i],[m],[n]]

                    i：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
                    n：増分．省略すると10
```

旧行番号mで指定した行以降の行番号を、新行番号iで始まる行番号に、増分nで付け変えます。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自動的に新しい行番号に書き換わります。

1.3　全角文字の編集

本機では、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、プログラム中でも日本語文字を使用することができます。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

ードのときには、日本語文字は表示できません。

一般の英数カナ文字は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。

1.3.1 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 SHIFT + XFER 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現われますので、表示したい全角文字を指定して ↵ を押すと、カーソルの位置に指定した全角文字が2桁の大きさで表示されます。

再度 SHIFT + XFER を押すと日本語入力モードが解除されます。日本語入力モードの詳しい説明は『第3章 日本語処理』の項を参照してください。

1.3.2 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の上に移動すると、カーソルは全角文字全体を覆う大きさに変わります。大きいカーソルは、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 ⇨ 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 ⇨ 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 ⇦ 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。

1.3.3 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前半でも後半でもかまいません）させた後、INS DEL 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、カーソルの右の文字列が左へ全角文字1文字分（2桁）移動します。

1.3.4 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の上にカーソルを移動（次の文字が全角文字の場合には、カーソル位置はその前半でも後半でもかまいません）させた後、SHIFT + INS DEL を2回押します。すると、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。

1.3.5 全角文字の修正

打ち間違えた全角文字を正しい全角文字に置き換えたい場合は、間違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るときに新しく全角文字を入力すると、次の文字を消してしまいますので注意してください。

（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。

１０　ＰＲＩＮＴ"本日語の文を使います。"■

⇦ でカーソルを〝本〟の位置まで移動し、COPY を押します。そして、点滅するカーソルを〝日〟の位置に移動させて ↵ を押して〝日〟を〝本〟の位置にコピーした後、再度 COPY 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 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、〝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。"

⇨ を４回押してカーソルを〝文〟の位置に移動させます。
INS DEL を１回押して〝の〟を削除します。

１０　ＰＲＩＮＴ"日本語文を使います。"

⇨ を２回押してカーソルを〝を〟の位置に移動させます。
SHIFT ＋ INS DEL を２回押して〝文〟と〝を〟の間に、全角文字１文字分のスペースを作ります。

１０"ＰＲＩＮＴ〝日本語文　を使います。"

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 ↵ を押します。
SHIFT ＋ XFER を押して日本語モードを解除します。

１０　ＰＲＩＮＴ"日本語文字を使います。"

↵ を押して修正したプログラムを記憶させます。

1.4 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 SHIFT + X は SHIFT を押しながら X 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移す。
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う。
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。(=↓)

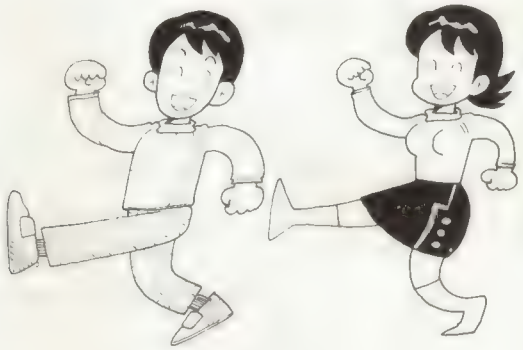
0		画面の背景色を黒(透明)にする。(=COLOR.0)
1		画面の背景色を青にする。 (=COLOR.1)
2		画面の背景色を赤にする。 (=COLOR.2)
3		画面の背景色をマゼンタにする。(=COLOR.3)
4		画面の背景色を緑にする。 (=COLOR.4)
5		画面の背景色をシアンにする。 (=COLOR.5)
6		画面の背景色を黄色にする。 (=COLOR.6)
7		画面の背景色を白にする。 (=COLOR.7)

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黄色をにする。 (=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。

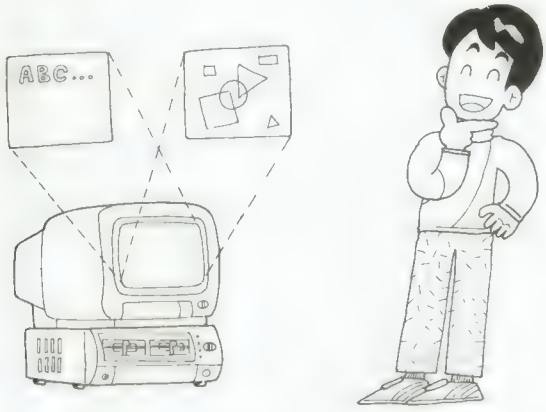
3）プログラムの実行中はキーを押している間だけ停止し、キーを離すと再び実行を再開します。



第2章　ディスプレイ

ＢＡＳＩＣからは図形文字（アルファベット、数字、漢字、カタカナ、ひらがな、セミグラフィック文字など）や、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。



2.1　ディスプレイモード

2.1.1　テキスト画面

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って異なります。

●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

●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

ディスクＢＡＳＩＣを起動した直後は次のように設定されています。
- ・標準ディスプレイ使用時には　　８０文字×１２行
- ・高解像度ディスプレイ使用時には　８０文字×２５行

テキスト画面の表示文字数を設定するには、ＷＩＤＴＨ命令を使います。

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、横４０文字×縦２５行の画面に変わります。

WIDTH命令を使って、次の８通りの設定ができます。

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可能。 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縦方向の解像度の設定

０：２００／１９２ドット

１：４００／３８４ドット

d：ディスプレイモードの指定

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）

・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。

・専用ディスプレイテレビでは、標準ディスプレイモードおよび高解像度ディスプレイモードのどちらでも使用可能です。切り換えはWIDTH命令の第４パラメータdによって行います。

2.1.2　グラフィック画面

横方向の解像度

WIDTH命令の第１パラメータによって決まります。

WIDTH　４０　ならば　３２０ドット

WIDTH　８０　ならば　６４０ドット

縦方向の解像度

グラフィック画面は、２つのグラフィックメモリ（４８ＫＢ×２）から構成されています。本機の専用ディスプレイテレビは、

標準ディスプレイモード　　（縦２００　ドット）
高解像度ディスプレイモード　（縦４００　ドット）

の切り換えができます。これはＷＩＤＴＨ命令の第３，４パラメータで指定します。

指定の方法は次のとおりです。

縦のドット数	第３パラメータの値	第４パラメータの値
２００	０	１
４００	１	２

ただし、縦４００ドットを指定する場合は、あらかじめ

```
OPTION SCREEN 0
```

を実行して、グラフィックＶＲＡＭの全部をグラフィック用に使用することを宣言しておかなければなりません。

なおＷＩＤＴＨ命令の第４パラメータを０にして実行すると、本機前面トビラ内の標準／高解像度切換えスイッチによって

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）　▮

に切り換えることができます。

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

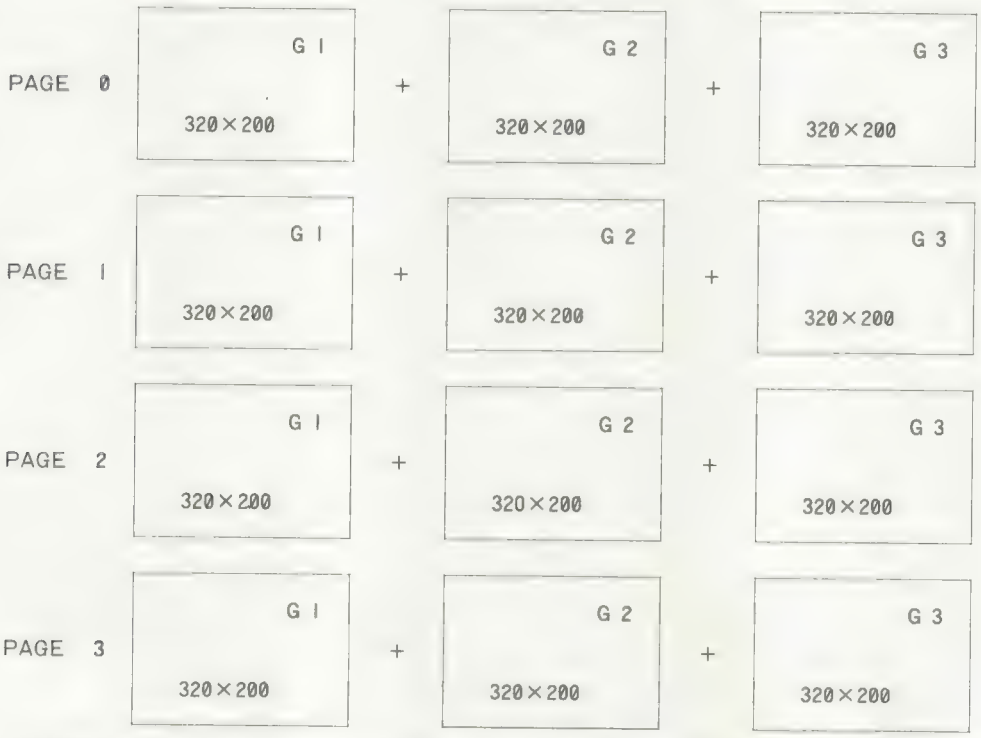
d：0または1。標準ディスプレイモードに設定。d＝0のとき、本体の標準／高解像度切換えスイッチは標準ディスプレイモード（▬）に設定されていること。

D：0または2。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／高解像度切換えスイッチは高解像度ディスプレイモード（▮）に設定されていること。

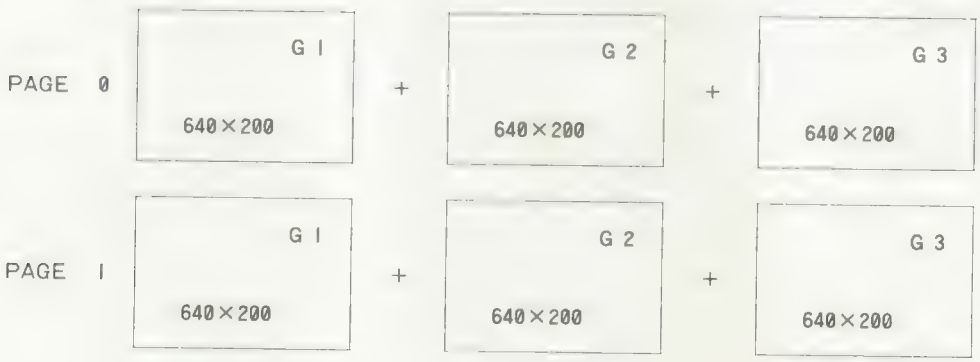
2.1.3　グラフィックVRAMの構成

グラフィック画面の基本的な構成を次に示します。1枚目のグラフィック画面をG1、2枚目の画面をG2、3枚目の画面をG3と呼びます。

〈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または1または2）〉



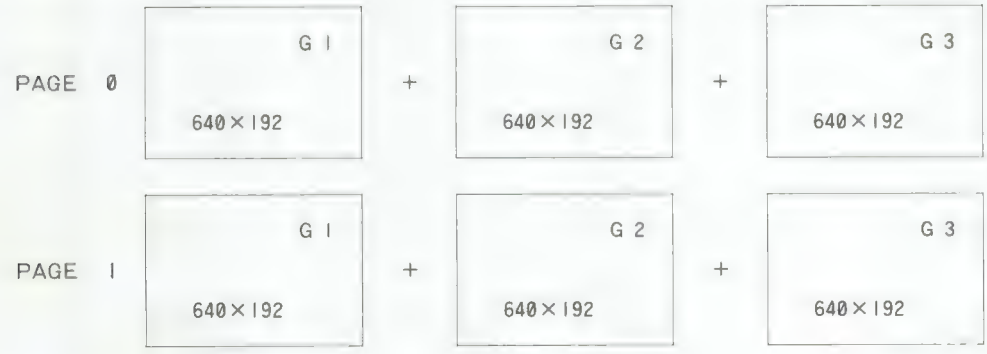
〈高解像度／標準ディスプレイモードで、WIDTH80，25,0,Dを設定（D＝0または1または2）〉



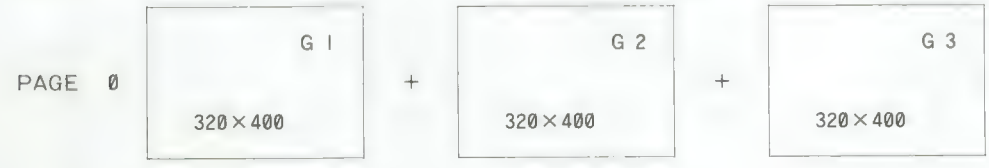
〈高解像度／標準デイスプレイモードで、ＷＩＤＴＨ４０，１２，０，Ｄを設定（Ｄ＝０または１または２）〉

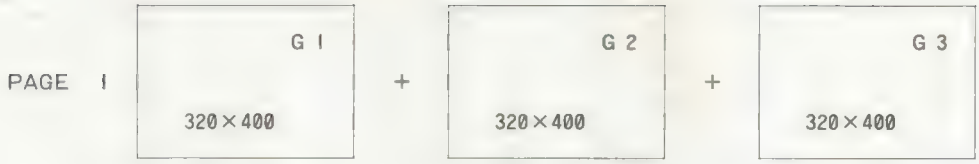


〈高解像度／標準デイスプレイモードで、ＷＩＤＴＨ８０，１２，０，Ｄを設定（Ｄ＝０または１または２）〉

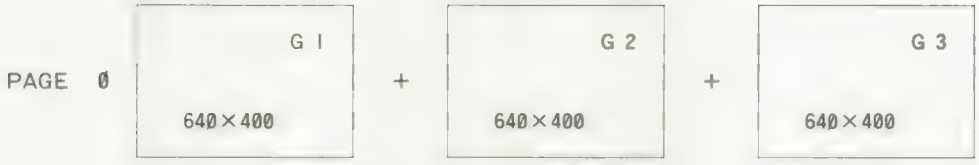


〈高解像度デイスプレイモードで、ＷＩＤＴＨ４０，２５，１，Ｄを設定（Ｄ＝０または２）〉

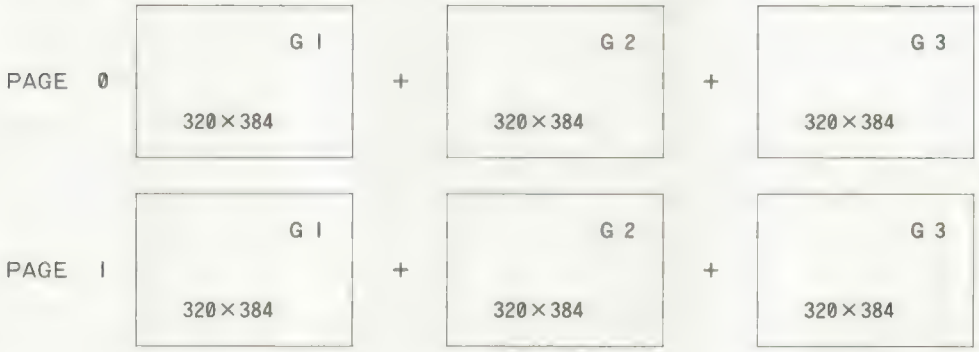




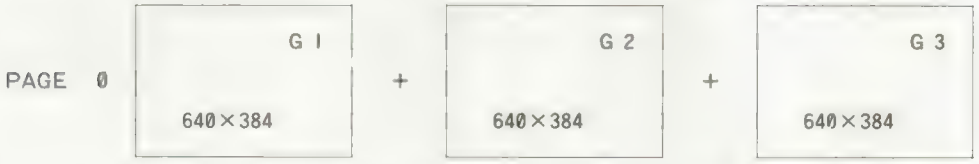
〈高解像度 デイスプレイモードで、ＷＩＤＴＨ８０，２５，１，Ｄを設定（Ｄ＝０または２）〉



〈高解像度 デイスプレイモードで、ＷＩＤＴＨ４０，１２，１，Ｄを設定（Ｄ＝０または２）〉



〈高解像度 デイスプレイモードで、ＷＩＤＴＨ８０，１２，１，Ｄを設定（Ｄ＝０または２）〉



※　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、初期状態では次の図のようにＧ１が青、Ｇ２が赤、Ｇ３が緑に設定されています。この３枚のグラフィック画面が合成されて、８色のカラー表示になります。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には、そのドットはマゼンタ（紫）で表示されます。

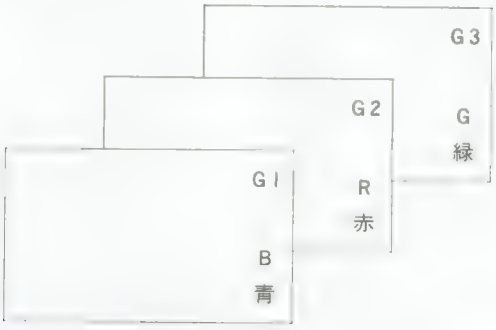


3枚のグラフィック画面の合成
（ＢＲＧ合成）

※グラフィックＶＲＡＭは、本来のグラフィック表示の目的以外に、データやプログラムを記憶するメモリとしても使用できます。グラフィックＶＲＡＭの使い方を指定するのがＯＰＴＩＯＮ　ＳＣＲＥＥＮ命令です。詳しくは『第５章　フリーエリアについて』を参照してください。

2.1.4.　カラーモード

カラーモードとは、全グラフィック画面がアクセスできる状態のことをいいます。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。



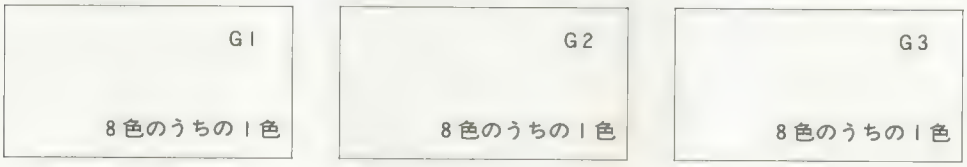
また、このような画面のことをカラー画面といいます。画面の解像度により、何ページかの独立したカラー画面を設定することができます。

カラーモードを選択するには、ＳＣＲＥＥＮ命令の第３パラメータに０を指定します。カラー画面が複数ページであるときは、ＳＣＲＥＥＮ命令の第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。

解　像　度	カラー画面のページ数
320×200、320×192	4（PAGE 0～PAGE 3）
640×200、640×192	2（PAGE 0，PAGE 1）
320×400、320×384	
640×400、640×384	1

2.1.5. マルチページモード

マルチページモードとは、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命令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また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のとき、本体の標準／高解像度切り換えスイッチはHIGH(■)に設定されていること）。

マルチページモードを選択するには、SCREEN命令の第3パラメータ(グラフィックモード)の指定が1，2，3のいずれかに指定します。

※SCREEN命令は、グラフィック画面の使用モードを指定します。

SCREEN 〔出力ページ〕,〔入力ページ〕,〔グラフィックモード〕

使用できるページ数は画面の解像度によって、次のようになっています。

画面解像度		使用できる画面の数	
横	たて	グラフィック画面	テキスト画面
40文字 (320ドット)	200又は 192ライン	4	2
80文字 (640ドット)	200又は 192ライン	2	1
40文字 (320ドット)	400又は 384ライン	2	2
80文字 (640ドット)	400又は 384ライン	1	1

第1および第2パラメータは出力ページと入力ページを指定します。出力ページとは実際に画面に表示され、目に見えるページ、入力ページとは目に見える見えないにかかわらずテキスト画面、グラフィック画面の各画面に対する命令が実行されるページをいいます。

詳細は『BASICリファレンスマニュアル』の2．6．1を参照してください。

2.1.6. カラーコード

カラーコードは0～7の整数で表され、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)	○	○	○	0 0 0	0
青	○	○	●	0 0 1	1
赤	○	●	○	0 1 0	2
マゼンタ	○	●	●	0 1 1	3
緑	●	○	○	1 0 0	4
シアン	●	○	●	1 0 1	5
黄	●	●	○	1 1 0	6
白	●	●	●	1 1 1	7

※　●はドットがある、○はドットがないことを示しています。

2.1.7.　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

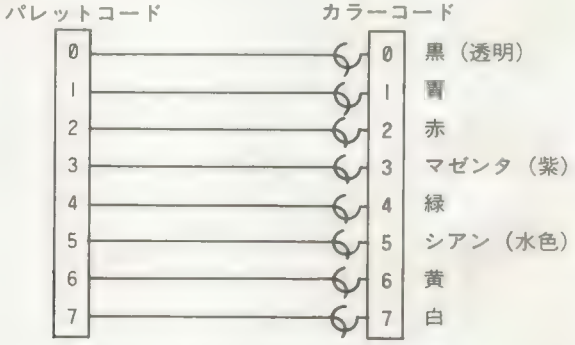
6＝黄、7＝白

というように数字と色が絶対的に対応しています。

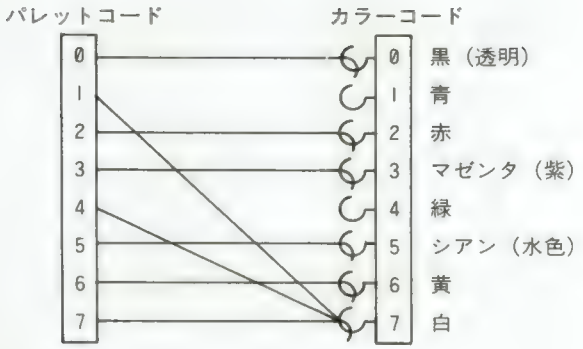
これに対してパレットコードも色を指定するコードで0～7の数字で表されますが、カラーコードのように数字に対する色が決まってませんので、自由に色を設定することができます。

パレットコードの色の設定はＰＡＬＥＴ命令を使って行います。

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様にパレットコード＝カラーコードとなっていますが、次の様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。



2.1.8　中間色コード

グラフィック命令のうち

LINEのBF（ボックスフィル）
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桁で表され、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またはnを単独で指定したときの色と同じです。また、黒と他の色の混合色

&H00, &H01, &H02……, &H07

はカラーコードの

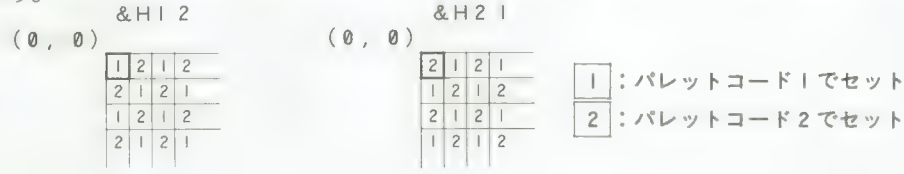
0，1，2……，7そのもので中間色を出すことができないので、黒と他の色の混合色は

&H08, &H09, &H0A……, &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異なります。



<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)	マゼンタ (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)	マゼンタ (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
```

＊標準ディスプレイモードではグラフィック画面のドットが粗いので、中間色として見にくい部分があります。

2.2 テキスト

2.2.1 テキスト画面

キーボードから入力された文字やプログラムなどのリストは、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である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。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、それぞれをグラフィック画面、テキスト画面といいます。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＝0または1、Ｄ＝0または1または2）のとき〉



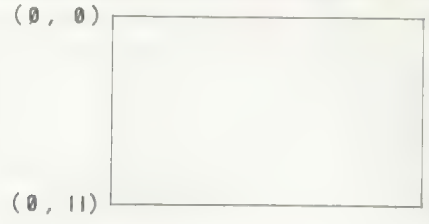
〈ＷＩＤＴＨ80,25,Ｇ,Ｄ（Ｇ＝0または1、Ｄ＝0または1または2）のとき〉



〈ＷＩＤＴＨ40,12,Ｇ,Ｄ（Ｇ＝0または1、Ｄ＝0または1または2）のとき〉



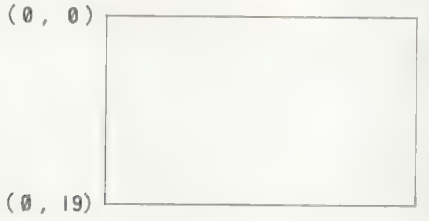
<WIDTH80,12,G,D（G＝0または1、D＝0または1または2）のとき>



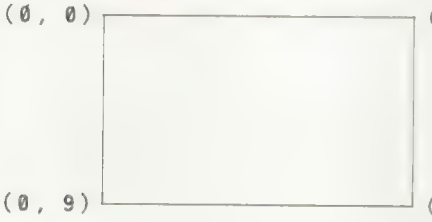
<WIDTH40,20,G,D（G＝0または1、D＝0または1または2）のとき>



<WIDTH80,20,G,D（G＝0または1、D＝0または1または2）のとき>



<WIDTH40,10,G,D（G＝0または1、D＝0または1）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＝0または1、D＝0または1）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命令は、前ページに表示した座標の範囲で設定することができます。

ＷＩＤＴＨ命令は画面に表示する文字数を設定します。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　縦方向の表示開始行，縦方向の表示行数， 横方向表示開始行，横方向表示桁数

たとえば、横方向の最初の桁から２０桁と縦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 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，縦方向の位置

たとえば、横方向１０桁目、縦方向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。

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命令により、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、ＫＥＹＬＩＳＴという命令を使って行います。

KEYLIST { 0　ファンクションキーの内容は表示されない 1　ファンクションキーの内容は表示される

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。この設定を行うにはＫＭＯＤＥという命令を使います。

KMODE { 0 …… 漢字表示モードにする
1 …… 漢字表示モードを解除する

たとえば、

```
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縦方向の行数が２５行、あるいは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「会社」という文字を表示するには

```
PRINT "会社"
```

あるいは

```
PRINT CHR$(&J3271,&J3C52)
```

と入力します。また、

```
PRINT CHR$(&H89EF,&H8ED0)
```

と入力しても

```
会社
```

と表示されます。

漢字コードは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で始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されません。そこで漢字のかわ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$(&H89EF)
```

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

2.2.2 テキストの属性

テキストは最大８０×２５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（ブリンク）も同様に１文字単位で行えます。

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。

色の指定

文字の色はCOLORという命令で行えます。

```
COLOR  n        n＝0：黒
                   1：青
                   2：赤
                   3：マゼンタ
                   4：緑
                   5：シアン
                   6：黄
                   7：白
```

この命令が実行されると、以後画面に表示される文字は指定された色になります。また、CTRL 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。

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）…… 標準モード
     { 1                …… 反転モード
```

反転モードにするには

CREV　1 ↵

と入力します。

また、CTRLを押しながらテンキーの－を押すことによって、反転モード、標準モードの切換えができます。

点滅（ブリンク）文字

点滅モードにすると、表示される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）	…… 標準モード
	1	…… 点滅モード

点滅モードにするには

CFLASH 1 ↵

とします。

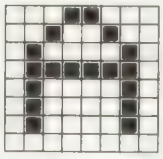
また、CTRLを押しながらテンキーの＊を押すことによっても、点滅モード、標準モードの切換えができます。

倍文字

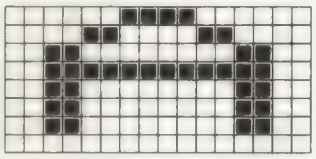
本機では4種類の文字を扱うことができます。これは、グラフィックで1ドットずつ文字を作るのではなくPRINT文で使うことができます。この4種類を区別して使い分ける命令はCSIZEで、書式は次のとおりです。

CSIZE n	n＝0：標準の8×8ドットの文字です
	1：縦方向に2倍の文字です
	2：横方向に2倍の文字です
	3：縦方向、横方向ともに2倍の文字です

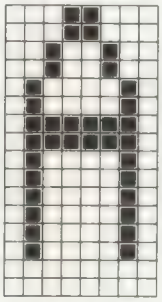
CSIZE 0



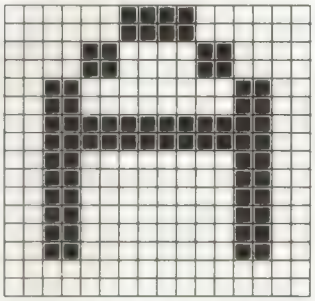
CSIZE 2



CSIZE 1



CSIZE 3



倍文字を書く場合、まずCSIZE命令で、サイズを指定します。次にPRINT命令を使いますが、ここで使うPRINT命令には#0というファイル番号を用います。

```
PRINT#0, "〔文字列〕"
```

たとえば、

```
10 CSIZE2
20 PRINT#0, "ABC"
```

とすると、ABCという文字が横2倍の文字で書かれます。

文字を表示する座標の指定はLOCATE命令を使用します。文字の表示座標を指定する場合、次の点に注意してください。

① CSIZE1（縦2倍）またはCSIZE3（縦横2倍）を指定した場合の縦方向の座標に関する注意点

表示する文字より上にある他の文字との間隔は偶数行空けなくてはなりません。もし、上に何も文字がない場合には、偶数座標にしか表示できません。奇数座標に表示したい場合には、その座標より上方向に偶数行あけたスペースを挿入することにより可能となります。また、縦2倍あるいは縦横2倍文字が表示されている2行内に標準文字および横2倍文字を表示することはできません。

② CSIZE2（横2倍）またはCSIZE3（縦横2倍）を指定した場合の横方向の座標に関する注意点

横方向の座標を奇数座標に指定しても、指定した奇数座標に1を加えた偶数座標に文字は表示されます。よって、横2倍文字または縦横2倍文字を表示する場合の横方向の座標は偶数座標を指定してください。

サンプルプログラム

ちょっとコマーシャル

文字の大きさを変えて"SHARP PERSONAL COMPUTER X-1"と表示するものです。

```
50 REM CSIZE ヲ ツカウ
100 WIDTH 40:INIT
110 CSIZE 3                          ←縦横2倍文字にします。
120 CLS
200 LOCATE 5,8
210 COLOR 4
220 PRINT #0,"SHARP "                ←縦横2倍文字で"SHARP"
240 LOCATE 10,12                       を表示
250 COLOR 2
260 CSIZE 1                          ←縦2倍文字にします。
270 PRINT #0,"PERSONAL COMPUTER"     ←縦2倍文字で"PERSONAL
300 LOCATE 20,16                       COMPUTER"を表示
310 COLOR 6
320 CSIZE 2                          ←横2倍文字にします。
330 PRINT #0,"X-1"                   ←横2倍文字で"X-1"を表示
400 END
```

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）には、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、標準状態でキー入力をすると、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命令が**DEFCHR$**です。またRAMCGとROMCGとの表示を切り換える命令が**CGEN**です。

例をあげて説明します。

```
CGEN 0 ↵
PRINT "A" ↵
```

とすると、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ASCIIコードを示しています。ASCIIコード65はROMCGの"A"にあたります。

RAMCGは8×8のパターンで256個まで定義できます。

また、RAMCGとROMCGとの切り換えは、CTRL 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。

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、行と行と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**命令を使用します。

KSEN { 0（または省略）……標準モード
　　 1〔, 色〕　　……アンダーラインモード

たとえば、

```
KSEN 1
```

としますと、この命令を実行した後、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
```

2.2.3 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)を結ぶ線を対角線とするBOXを、文字〝A〟で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。

これらの命令の応用として、属性のみの変更ができます。

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つける場合

```
KSEN 1, 7
LINE(0, 2)-(79, 2), ""
```

とします（このとき、WIDTH命令でアンダーラインの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、その時点での属性で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
```

として、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。

2.3　グラフィック

本機の特徴のひとつに豊富なグラフィック処理機能があげられます。96KB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。ここでは、多くのプログラム例をまじえながら、これら命令群の使い方を説明します。

2.3.1　画面の初期化

まず最初に画面を初期化する命令について説明します。命令の中には、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が描けない場合もあります。そのようなとき、次の命令を実行することによって、画面状態はもとの設定に戻ります。

```
INIT
```

また CTRL + D とダイレクトに入力することによりＩＮＩＴ命令と同様の結果が得られます。

ＩＮＩＴ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には次の命令を実行します。

```
CLS  n        n＝省略：テキスト画面のみをクリア
                      （SHIFT + CLR HOME と同じ）
              0：グラフィック画面Ｇ１、Ｇ２、Ｇ３をクリア
              1：グラフィック画面Ｇ１のみをクリア
              2：グラフィック画面Ｇ２のみをクリア
              3：グラフィック画面Ｇ３のみをクリア
              4：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
```

ＩＮＩＴおよびＣＬＳ命令によって、画面状態の初期化とクリアが行われます。したがって、これからいろいろな命令を実行していくなかで、画面状態を元に戻したいときは、上記の命令を実行してください。

2.3.2　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る座標系です。

グラフィック画面の解像度によって、次のような座標の範囲が設定されます。

〈WIDTH40,25,0,D（D＝0または1または2）のとき〉



〈WIDTH40,25,1,（D＝0または2）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＝0または1または2）のとき〉



〈WIDTH80,25,1,D（D＝0または2）のとき〉

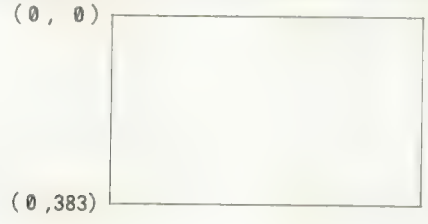


※D＝0のとき本体の標準／高解像度切換スイッチは高解像度ディスプレイモード（■）にセットされていること。

〈WIDTH40,12,0,D（D＝0または1または2）のとき〉

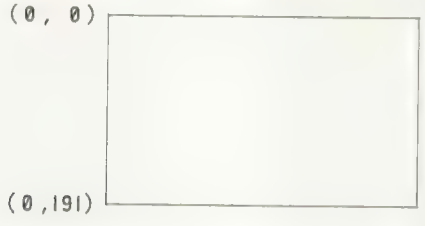


<WIDTH40,12,1,D（D＝0または2）のとき>



※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。

<WIDTH80,12,0,D（D＝0または1または2）のとき>



<WIDTH80,12,1,D（D＝0または2）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。

※WIDTH 40（80），20，G，D（G＝0，1、D＝0，1，2）
WIDTH 40（80），10，G，D（G＝0，1、D＝0，1）
のときグラフィック画面は使用できません。したがって、グラフィック座標系は設定されません。

2.3.3 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、前項の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に対応しています。

ユーザ座標系とは、BASICのWINDOW命令（『BASICリファレンスマニュアル』の2．8．2 WINDOW参照）によってユーザが指定した座標系で、縦方向、横方向ともー1．701411 8E+38 ～ 1．701411 8E+38（指数部ー38 ～ +38）の単精度の範囲で指定できます。PSET，PRESET，LINE，POLY，CIRCLE，PAINTのグラフィック命令およびPOINT関数はこのユーザ座標で使用されます。次の図は、ユーザ座標系と画面座標系です。





(GX, GY) =(319, 199)
または (319, 191)
または (319, 399)
または (319, 383)
または (639, 199)
または (639, 191)
または (639, 399)
または (639, 383)

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

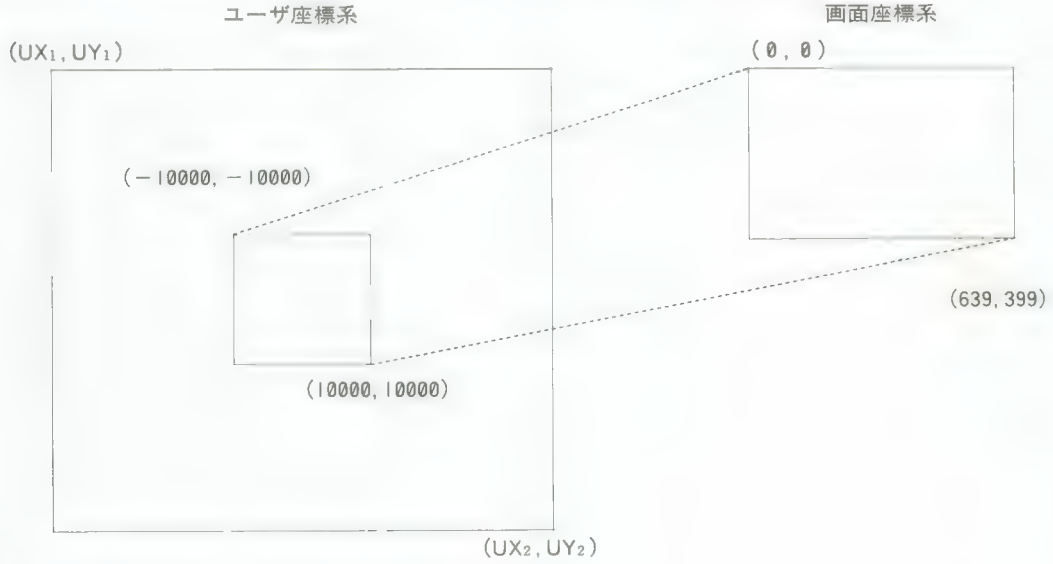
画面座標系は実際に目で見ることはできますが、ユーザ座標系の広大な座標平面は目で見ることができません。

したがって、BASICのWINDOW命令を使って、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、WIDTH 80, 25, 1, 2が設定されているとき、

```
WINDOW (0, 0)−(639, 399), (−10000,−10000)
−(10000, 10000)↵
```

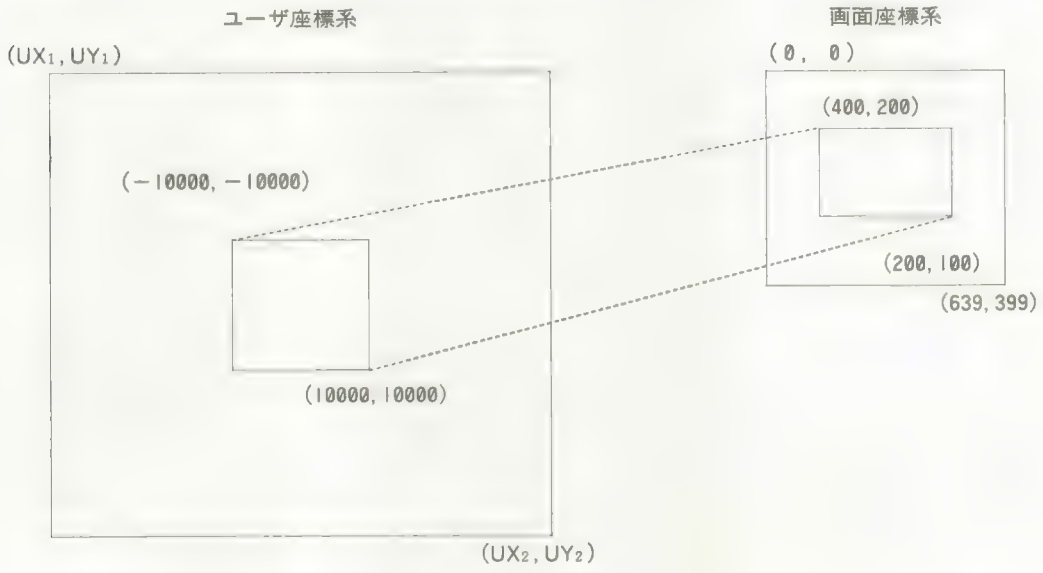
という命令を実行すると、次のようにユーザ座標系の(−10000,−10000) から(10000, 10000)を対角線とする長方形の領域を画面座標系全域（グラフィック画面全体）に表示することができます。



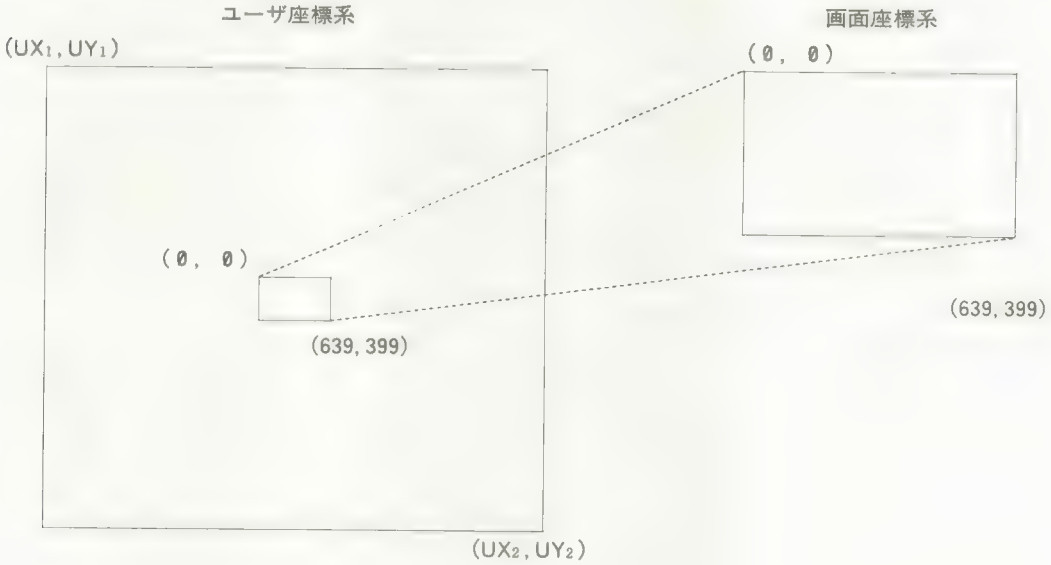
また、

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

を実行すると、次のようにユーザ座標系の（－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。



これから述べる各種グラフィック命令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命令によって座標系が異なりますので、WINDOW命令を使用した場合は、まずその命令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うようにしてください。ただしBASIC起動時には、ユーザ座標系は、画面座標系とまったく同じものですので、WINDOW命令でユーザ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、つまりユーザ座標系を画面座標系と同一にするには、INIT命令または CTRL + D を実行します。

2.3.4 グラフィック命令の色指定

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命令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ゆく各種グラフィック命令の色指定がどちらのコードで行われているか、あらかじめ知っておく必要があります。次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命令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命令またはPALET命令、または CTRL + D を実行します。　また、グラフィック命令において色指定（パレットコード）を省略した場合は、自動的にCOLOR命令の第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命令のパレットコードとみなして実行します。

COLOR命令の設定は、次のとおりです。

COLOR　〔C_1〕〔,C_2〕

C_1：テキスト文字の表示色（カラーコード）
C_2：画面の背景色（カラーコード）

これより、グラフィック命令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。

たとえば、COLOR命令の第1パラメータをカラーコード1に設定した場合、パラメータ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命令は青色として実行されます。また、PALET命令によって、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命令は緑色として実行されます。なお、COLOR命令は、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命令の代わりに、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命令の設定方法については、後述「2.3.10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述べます（カラーモードについては、『2.1.4 カラーモード』の項を参照してください）。

2.3.5 点を打つ

PSET命令を使うことによって画面上の任意の場所に点（ドット）を打つことができます。またPRESET命令によって、任意の点を消すこともできます。

点を打つ場合、消す場合のPSET，PRESETのパラメータは次のようになります。

```
PSET (x, y [, c])
PRESET (x, y [, c])

                    x, y：ドットの位置（ユーザ座標系）
                    c：パレットコード
```

PSET命令は、グラフィック画面に指定のパレットコードでドットをセットします。パレットコードを指定しないときは、テキスト画面の文字の色に対応したパレットコードの色でドットをセットします。

たとえば、INIT↵としたあと

```
PSET (20, 20, 6) ↵
```

とすると、ユーザ座標系の（20，20）の位置に黄色のドットを表示します。

一方，PRESET命令はグラフィック画面のドットを指定の色でリセットします。たとえば

```
PRESET (20, 20, 2)↵
```

とすると、先程黄色だった点を赤色でリセットしますので、この点は緑色になります。

つまり、PRESET命令は正確にいうと「点を消す」のではなく「指定した色を消す」命令です。この場合には赤と緑からなる黄色の点から赤を消したので残るのは緑色というわけです。

2.3.6 線を描く

LINE命令は、ただ単に線を引くだけでなく、パラメータの設定によって次の動作を行うことができます。

①ラインスタイルを設定することによって、点線、破線、一点鎖線など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。

線を引く

線を引く場合、LINE命令のパラメータは次のようになります。

LINE〔(x_1, y_1)〕－(x_2, y_2)〔, mode, c, ls〕

x_1, y_1：始点座標（ユーザ座標系）
x_2, y_2：終点座標（ユーザ座標系）
mode：PSET, PRESET, XOR
c：パレットコード
ls：ラインスタイル

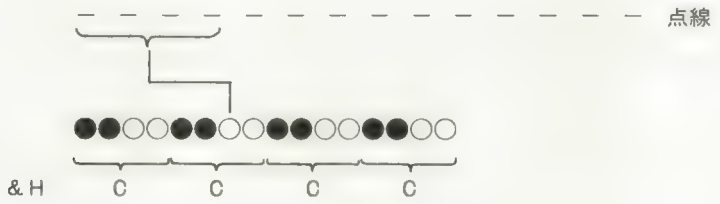
ただ単に実線を引くだけならば、たとえば次の命令を実行するだけでOKです。

```
LINE (0, 0)－(319, 199), PSET
```

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命令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『2．3．5　点を打つ』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ところで線の色が反転します。たとえば、すでに白（＝青＋赤＋緑）でペイントされているところへXORを使って青の線を引くと、重なったラインの色は黄（＝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modeの影響を受けます。

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。たとえば、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、ラインスタイルには、16ドットの範囲内で任意の繰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。以下におも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、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）を結ぶ一点鎖線を黄色で引きます。

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

どうですか、うまく引かれましたか？　もし、思いどおりの直線が引かれなければ 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？　今まで一点鎖線が表示されていたところが黒（透明）になり、いままで黒（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論理演算が行われ、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,VAL(A$)
90 NEXT
100 DATA &HFFFF,&HFF00,&HCCC,&HAAAA,&HFFCC,&HE4E4,&HFCCC,&HFF24
```

長方形を描く

長方形を描く場合、ＬＩＮＥ命令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まります。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。

長方形を塗りつぶす

長方形を塗りつぶす場合、ＬＩＮＥ命令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：長方形を塗りつぶす指定

長方形を描いてその中を塗りつぶすには、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。

たとえば、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で塗りつぶす場合には次の命令を実行します。

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

ただし、BFを指定する場合は、同時に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のタイリングパターンを定義する場合には、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とき、LINE命令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。

たとえば、横8ドットのパターンを1ドット1色にして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

タイリングパターンt$は次のように表わされます。

```
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命令を実行します。

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

折れ線を描く

LINE命令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
x2, y2, x3, y3：折れ線の頂点座標
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ができます。たとえば、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）
```

2.3.7　正多角形を描く

正多角形に限り、POLY命令を使って簡単に描くことができます。

```
POLY（x，y），r〔，c，Δθ，θs，θe〕
```

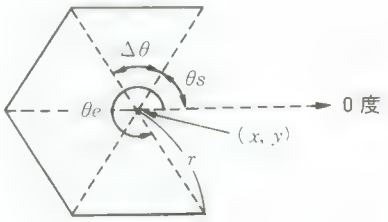
x，y：正多角形の中心座標
r：中心から頂点までの距離
c：正多角形の辺の色（パレットコード）
Δθ：ステップ角（０～３６０度）
θs：初期角（０～３６０度）
θe：終了角（０～３６０度）

多角形は、前述のLINE命令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なければなりません。しかし、POLY命令では、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『２．３．８　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ります）。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ます。

```
INIT
CLS0
```

正6角形を描いてみましょう。

```
POLY(100, 100), 50, 6, 60, 0, 360
```

点（100，100）を中心とした黄色の6角形が描けました。

次に、左側半分だけの半円を描いてみましょう。まずは画面をクリアします。

```
CLS0
```

左側だけの半円ですから初期角は90度、終了角は270度と指定します。またステップ角は1度でやってみましょう。

```
POLY(100, 100), 50, 6, 1, 90, 270
```

これで黄色の半円が描けました。

（注）POLYおよびCIRCLE命令における頂点距離または半径は、画面モードによっては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命令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
```

2.3.8　円を描く

円および楕円を描く命令はＣＩＲＣＬＥ命令です。

```
CIRCLE〔@〕(X, Y), r〔, c, f, θs, θe〕
                    f：偏平率（＝縦径／横径）
```

このように、ＣＩＲＣＬＥ命令のパラメータは、ＰＯＬＹ命令と似ています。違いは、ＰＯＬＹ命令のステップ角が、ＣＩＲＣＬＥ命令では偏平率に変っている点だけです。また、ＣＩＲＣＬＥ命令では、@記号が付くか付かないかで半径および偏平率の扱いが異なってきます。

@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基準にしています。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。偏平率を使用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。

@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半径は異なってきます。ですから偏平率を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命令およびＣＩＲＣＬＥ@命令の項を参照してください。

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を示すプログラムです。

```
10 'CIRCLE
20 INIT:CLS4:OPTIONSCREEN0
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
```

2.3.9　色を塗る（中間色の指定）

グラフィック曲線で囲まれた任意の形の図形を塗るには、ＰＡＩＮＴ命令を使います。ここでは中間色を指定します。

本機では赤、青、緑の３色を１ドット単位に自由な組み合わせで基本の８色以外の中間色を作ることができます。これを**タイリングペイント**といいます。

(1)　PAINT〔x, y〕, c, b〔, b_1, b_2……b_6〕
(2)　PAINT〔x, y〕, t$, b〔, b_1, b_2……b_6〕

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, b_1, b_2, ……b_6：境界色（パレットコード）

(1) の書式で、中間色を指定するときはｃを次のように表わします。

&Hmn　（またはｍ＊１６＋ｎ）

ｍ＝０～７のパレットコード

ｍ＝０～Ｆのコード

こうすれば、ｍとｎを混合した中間色を出すことができます。

赤色の円を描いてその中を中間色（紫）を使って塗ります。

```
10 WIDTH 40,25,0:INIT
20 CLS 4
30 CIRCLE(200,100),80,2
40 PAINT(200,100),&H12,2
```

(2) の書式で、中間色を指定するときは、タイリングパターンｔ$はCHR$関数やHEXCHR$関数などによって表わします。たとえば黄緑色は黄色と緑色のドットを1ドット単位で交互に打つことによって表示することができます。

```
CHR$(&H00)+CHR$(&HAA)+CHR$(&HFF)
またはHEXCHR$("00AAFF")

青→0  0  0  0  0  0  0  0   →&H00
赤→1  0  1  0  1  0  1  0   →&HAA
青→1  1  1  1  1  1  1  1   →&HFF
   黄 緑 黄 緑 黄 緑 黄 緑
```

このデータをPAINT文に入れると

```
PAINT(x, y), CHR$(&H00)+CHR$(&HAA)+CHR$(&HFF)
```

または

```
PAINT (x, y), HEXCHR$("00AAFF")
```

となります。

2.3.10 色を瞬時に変える

PALET命令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（0または1）
```

PALET命令は色の変換を行いますが、実際に塗り変えるわけではありませんので瞬時に色の変換ができます。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」で行います。

たとえば、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ようにします。

```
PALET 1, 2 ↵
```

また、PALET命令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, 8またはPALET1, 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画面では透明と黒の区別はつきません。透明と黒との違いが現れ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。

第3パラメータはボーダーカラーに関するパラメータです。このパラメータが1のときは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抜きにすることもできます。

複数のパレットコードを一度に変換する場合はPALET@命令を用います。

PALET@ $C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8$

C_1, C_2 …C_8：カラーコード

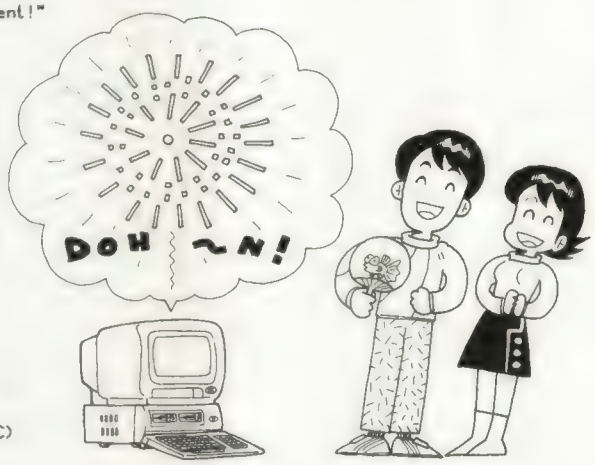
パレットコードはパラメータの位置に対応しています。すなわち、第1パラメータがパレット0に、第2パラメータがパレット1に、……第8パラメータがパレット7にそれぞれ対応しています。たとえば、パレットコード1を赤に、パレットコード7を緑に変換するためにはつぎの命令を実行します。

```
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 SCREEN 1,0: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 SCREEN 1,1:CLS4
170 SOUND 7,&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:SOUND 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 NEXT J
290 NEXT I
300 FOR I=1 TO 7
310 PALET I,0
320 PAUSE 1
330 NEXT I
```



2.3.11　テキスト画面とグラフィック画面の優先順位を決める

PRW命令は、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とき、どちらを優先して表示させるかを決める命令です。

```
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。



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持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ます。たとえば、グラフィックの青と緑を優先させて表示するには、次の命令を実行します。

```
PRW &B00010010 ↵    (=PRW 18)
```

PRW命令はパレットコードに対する命令ですので、たとえば

```
PALET 1, 2 ↵
PRW &B00010010 ↵
```

とするとどうなるでしょうか。パレットコード1と2は両方とも赤色ですが、パレットコード1の赤色はテキストより優先されます。しかし、パレットコード2の赤はテキストに隠れます。このように、PRW命令を利用すると見かけは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戻すにはPRW ↵ を実行します。

2.3.12　文字パターンの描画

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

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。値の設定によっては、文字が画面からはみだ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「2.3.9 色を塗る」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度左に回転したもの

θ＝3では270度左に回転したもの

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〝A〟という文字を青色で縦横1倍で表示します。

```
SYMBOL (100, 100), "A", 2, 2, 4, 1, PSET
```

これはグラフィック画面に〝A〟という文字を緑色で縦横2倍の大きさで90度左に回転したものを表示します。

```
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。

2.3.13 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命令です。

また、PUT@命令はGET@の逆で、配列変数に読み込んだ情報を画面に描く命令です。

通常これらの2つの命令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することができます。

これらの命令は、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塗り、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=1 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@(x_1, y_1)－(x_2, y_2), a, c
PUT@(x_1, y_1)－(x_2, y_2), a, mode, c

x_1, y_1：読み込む領域の左上隅の座標（画面座標系）
x_2, y_2：読み込む領域の右下隅の座標（画面座標系）
a：配列名
c：パレットコード
mode：PSET, PRESET, XOR, OR, AND, NOT

PUT@命令では表示のときにPSET, PRESET, XOR, OR, AND, NOTのモードがあり、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命令になります。

2.3.14 グラフィックパターンの描画

グラフィック画面にドットパターンを描く際には、まず描画開始位置を指定します。それには、POSITION命令を使用します。

POSITION x, 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命令の実行により描画開始位置を指定した後、PATTERN命令でドットパターンを描画します。

PATTERN n, x$(, y$……)

n：ドットパターンの縦方向の段数
n＜0のとき下の方向に重なる
n＞0のとき上の方向に重なる
x$, y$：ドットパターンを表わす文字式

たとえば、

```
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）

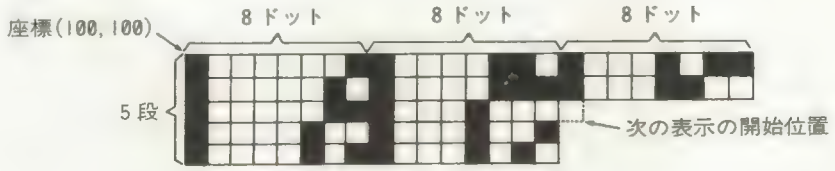
が表示されます。

また、ドットパターンの積み重ねは段数を指定するnの値によって積みかさ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かさねられます。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かれます。

たとえば、

```
10 WIDTH 40:INIT
100 A$=HEXCHR$("818283848586878889898A8B8C")
110 POSITION 100,100
120 PATTERN -5,A$
```

を実行すると下の様に表示されます。



第3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（ＣＺ－８ＦＢ０２）が標準装備されています。この章ではこの漢字ＢＡＳＩＣを使用して漢字、ひらがなどの全角文字と呼ばれる文字の扱いについて説明します。

3.1 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ドットで表示された文字のことを指します。それに対して文字のフォントが８×８または１６×８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ＢＡＳＩＣでは、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・ひらがなどの全角文字はプログラム中ではコード（ＪＩＳ漢字コード、区点コードなど)としてしか扱うことができませんでした(たとえば〝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表わすことができます。それは、次のような場合に可能です。

・ＰＲＩＮＴ文等のダブルクォーテーション（"）で囲まれた部分
・ファイル名およびディレクトリ名
・ＲＥＭ文
・ＤＡＴＡ文
・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バイトのコードで処理されています。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よばれています。

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。たと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（呼）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。(たたし、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、ＫＭＯＤＥ命令によって決められます。

ＫＭＯＤＥ	０	半角文字のみ使用可（セミグラフィック文字使用可）
	１	全角文字も使用可（セミグラフィック文字使用不可）

また、全角文字の場合、その文字フォントが縦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ときには、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
```

3.2 全角文字の入力

全角文字は日本語入力モードに入ってから入力します。日本語入力モードと通常モードの切換えには次の2つのうちどちらかを使ってください。

(a) SHIFT + XFER

または、

(b) CTRL + XFER

日本語入力モードに入ると、画面の最下行に変換フィールドが現れます。



3.2.1 漢字の入力

では、実際に漢字を入力してみましょう。

例1

「日本」と表示してみよう。

① N I C H I とタイプします。



② XFER を押して漢字に変換します。このとき、「にち」という読みを持つ漢字は、音訓辞書中には"日"しかありませんので漢字グループは表示されません。直接テキストエリア上へ"日"が表示されます。



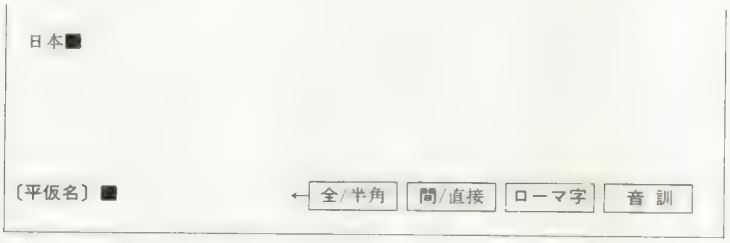
③次に H O X とタイプします。(X は「ん」に対応しています。詳しくは後述の「カナ一ローマ字変換一覧表」を参照してください。)



④ XFER を押して漢字変換すると、「ほん」の読みを持つ漢字グループが最下行に表示されます。



⑤「本」がありましたね。↵ またはテンキーの 1 を押して「本」を選びます。



以上で漢字「日本」が入力されました。

例2

では H TAB を使って「日本」と表示してみましょう。

① N I C H I とタイプします。



②H TAB を押します。変換フィールドに記号 " ^ I " が現れます。



③続けて H O X とタイプします。



④ XFER を押します。すると、最初の「にち」が「日」に変換されてテキストエリアに現れます。



⑤続けて、XFER を押すと、次の「ほん」の読みを持つ漢字グループが現れます。



⑥漢字グループの中から「本」を選びます。



以上で漢字「日本」が入力されました。

このように H TAB を用いると、一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。このとき、「H TAB」は漢字の読みと読みの間に入力します。後は、XFER を押して漢字変換を繰り返します。

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"^"は半角文字として入力されますので、一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。

ここで、漢字入力の方法をまとめます。

①変換フィールド内に入力されたひらがなまたはカタカナは XFER 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大９文字ずつ表示され、それがテンキーの 1 ～ 9 に対応しています。

②目的の漢字が見つからない場合は、XFER 、スペースキーまたは ⇩ を押します。次の漢字グループが表示されます。

また、前のグループに戻すには、⇧ を押します。

③目的の漢字が見つかれば、その漢字を取り出します。その方法には次の2通りがあります。

（a）テンキーの [1] ～ [9] 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] のテンキーに対応していますので、目的の漢字に対応したテンキーを押します。

（b）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、[↵] を押します。

④以上の操作によって、指定された文字がテキストエリア内のカーソルの位置に表示されます。

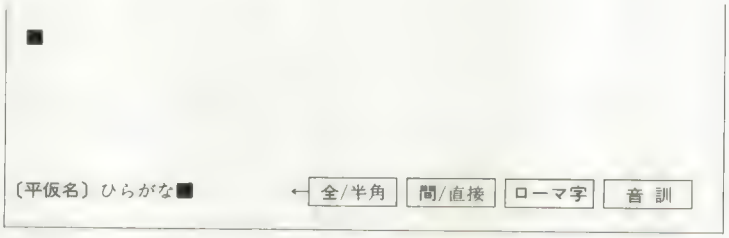
⑤なお、漢字グループを表示した状態で、入力誤りに気付いたり、目的の漢字がなくて変換を中止したい場合などは [ESC] を押します。すると、漢字グループの表示が消え、変換フィールド内にカーソルが戻りますので、ひらがなまたはカタカナの訂正ができます。変換フィールド内で文字を訂正するとき、間違って入力した場合は [BS DEL] を押してタイプし直してください。カーソルコントロールキー[←]は使えません。

3.2.2 ひらがなの入力

例3

「ひらがな」と表示してみよう。

① [H][I][R][A][G][A][N][A] とタイプします。



②[↵]を押せばテキストエリアに「ひらがな」と表示します。

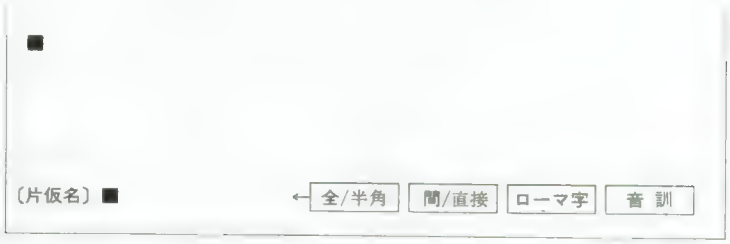


3.2.3 カタカナ、英数字の入力（全角文字）

例4

「カタカナ」と表示してみよう。

① F3 を押すと、［平仮名］の表示が［片仮名］に変ります。



② K A T A K A N A とタイプします。



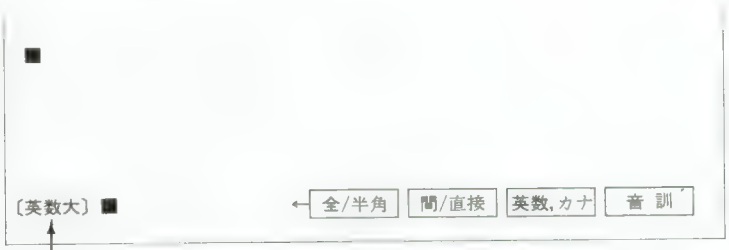
③ ↵ を押せばテキストエリアに「カタカナ」と表示します。



例5

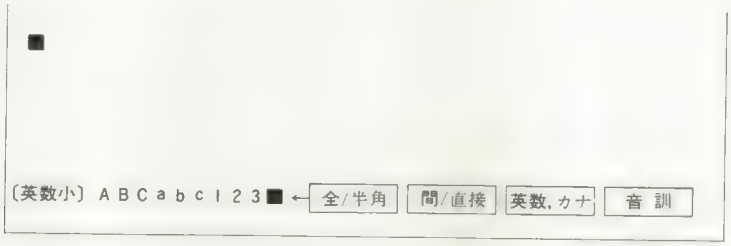
「ABCabc123」と表示してみよう。

① F4 を押すと、表示が［英数大］に変ります。

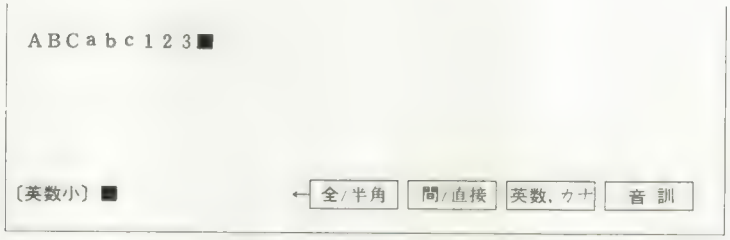


キャピタルロックキー CAPS LOCK をロック状態にすると〔英数大〕になり、ロックを解除すると〔英数小〕になります。

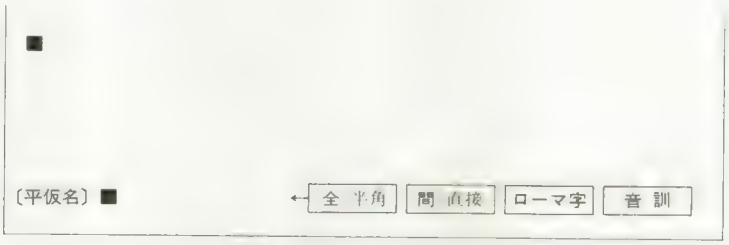
②［英数大］で [A] [B] [C] とタイプし、CAPS LOCK のロックを解除して［英数小］で [a] [b] [c] とタイプし、さらにテンキーの [1] [2] [3] をタイプします。



③[↵] を押せば、テキストエリアに「ＡＢＣａｂｃ１２３」と表示します。



なお、この状態で次に漢字、平仮名などを表示するには [F5] 続いて [F3] を押してください。次の画面になり、漢字、ひらがなが入力できます。



ファンクションキー [F3] ～ [F5] の機能を次の表にまとめます。

[F3]：平仮名／片仮名……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。
[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。
[F5]：ローマ字―カナ変換方式……アルファベットを入力すると、ひらがなまたはカタカナに変換します。

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				

●小文字は SHIFT 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。

「あ」…… SHIFT ＋ A
「い」…… SHIFT ＋ I
「う」…… SHIFT ＋ U
「え」…… SHIFT ＋ E
「お」…… SHIFT ＋ O
「や」…… SHIFT ＋ Y A
「ゆ」…… SHIFT ＋ Y U
「よ」…… SHIFT ＋ Y O
「っ」…… SHIFT ＋ Z または SHIFT ＋ T U または SHIFT ＋ T S I または SHIFT ＋ ｱｬﾔ

小文字の「っ」は上記入力のほかに、子音字を2度連続して入力することによって表示させることができます。

〔例〕ローマ字入力「 I R A S S Y A I」
→カナ変換出力「いらっしゃい」

●「ん」は次のようにも入力できます。

「ん」……「 SHIFT ＋ N」または「 N、 SHIFT ＋ ﾜﾔﾔ 」

●「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ができます。

〔例〕「きゃ」……「 K Y A」
または「 K I , SHIFT ＋ Y A」
↑ ↑
大文字「き」入力　小文字「や」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。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 を入力した後、SHIFT を押しながら ﾜﾔﾔ を押すことを示します。

※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。

●濁点「゛」および半濁点「゜」を入力するには次のキーを押します。

「゛」…… @
「゜」…… [「

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また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。

逆に直接モード（F2）を選択した場合、カナと半濁点は各1文字分のスペースを取りますので2文字として扱われます。

●読点「、」および句点「。」の入力は次のキーを押します。

「、」……SHIFT ＋ ・ね
「。」……SHIFT ＋ ・る

●始めかぎカッコ「「」および終わりかぎカッコ「」」の入力には、次のキーを押します。

「「」……SHIFT ＋ (
「」」……SHIFT ＋) む

3.3　全角文字の入力方式

全角文字の入力方式は先ほども触れましたが、ここで改めて詳しく説明します。

3.3.1　ひらがな・カタカナ・英数字などの全角文字の入力方式

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。

●英数字直接入力方式

アルファベット、数字等の入力を簡単に行える。

●ローマ字ーカナ変換方式

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。

●カナ入力方式

カナ を ロックしてカナ表記で入力し、それをひらがなまたはカタカナに変換する。

3.3.2　漢字への変換機能

次の2通りの方法があります。

●コード入力方式

漢字のコード（ＪＩＳ漢字コードまたは区点コード）を入力して、直接、漢字、ひらがな等の全角文字に変換する。

●カナー漢字変換方式

ローマ字ーカナ変換方式、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する。この方式には次の2つの方式があります。

・一字変換方式

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、その読みを持つ漢字をＪＩＳ漢字コードの配列通りに羅列して変換する。

音訓変換方式
漢字の音読み、訓読みを入力して変換する。

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などは一切不要ですが、音訓変換方式は、システムディスクの中の音訓辞書が必要です。

3.3.3　日本語入力モードに入るための条件

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。
（高解像度ディスプレイモードの場合は制限がありません。）

```
WIDTH 80,12,0,{1
                0
WIDTH 40,12,0,{1
                0
WIDTH 80,10,0,{1
                0
WIDTH 40,10,0,{1
                0
```

②CONSOLE命令によって、表示画面の最下行が変換フィールドエリアとして確保されていること。
〔例〕25行モードならば、CONSOLE0，24
12行モードならば、CONSOLE0，11
20行モードならば、CONSOLE0，19
10行モードならば、CONSOLE0，9
ただし、CONSOLE命令だけでなく、INIT命令やCTRL＋D入力よっても、変換フィールドは確保されます。

③KMODE1が実行してあること。

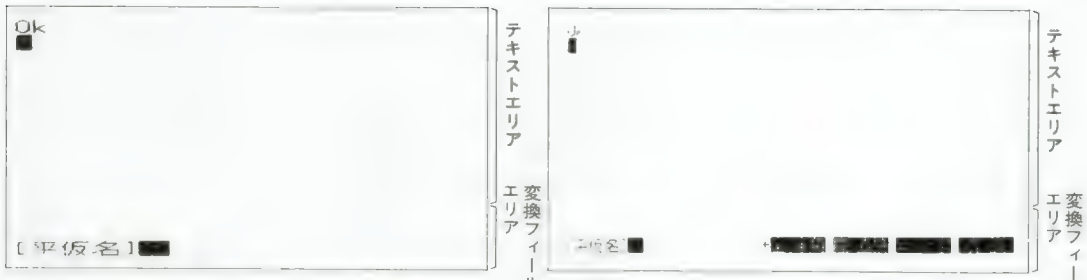
以上の条件が満たされているならば、

（a）SHIFT＋XFER

または、

（b）CTRL＋XFER

によって、日本語入力モードに入ることができます。
なお、BASICの起動時は、上記の条件①～③を満たしています。
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れます。全角文字への変換は、すべてこの変換フィールド内で行われます。また、日本語入力モードから通常の状態へ戻るには再び（a）または（b）を実行します。



(a) 4 0 桁モードの場合の入力画面　　(b) 8 0 桁モードの場合の入力画面

日本語入力モードにおける表示画面例

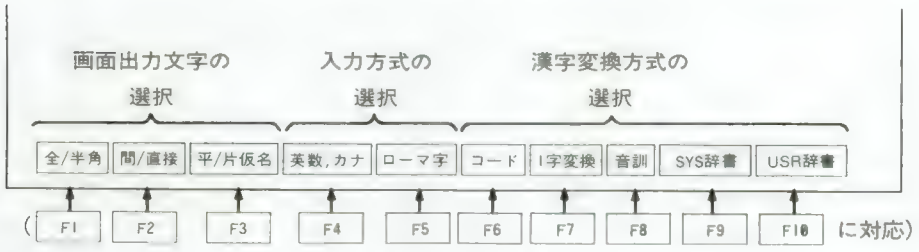
〔注〕日本語入力モードにおける表示画面は、４０桁と８０桁においては若干異なりますが、動作には関係ありません。

3.4 入力モードの選択

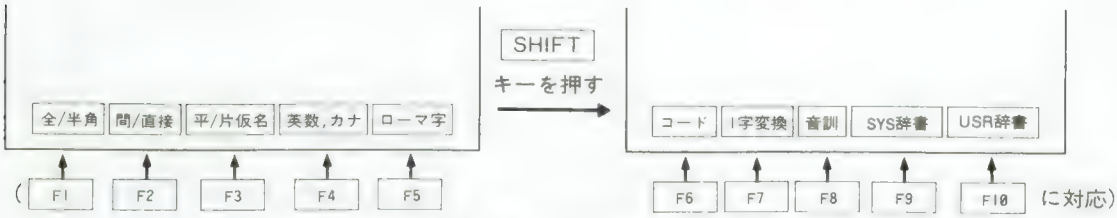
日本語入力モードでは、画面に出力される文字、入力方式、漢字変換方式をファンクションキー F1 ～ F10 によって選択することができます。

3.4.1 HELP について

日本語入力モードで、HELP を押すと、最下行の変換フィールドは次のように変ります。



８０桁モードの場合



４０桁モードの場合

このように入力モードの確認は HELP によって行います。上記の画面で現在のモードは赤字で表示されます。これらを表示したままの状態で F1 ～ F10 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 を押します。

3.4.2　各入力モードの選択

●画面出力文字の選択（F1　F2　F3 についてそれぞれ選択）

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……ローマ字ー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。

●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：ローマ字ーカナ変換方式……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 の英数字または F5 のローマ字ー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。

●漢字変換方式の選択（F6 ～ F10 の中から１つ選択）

F6：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り換わります。
（SHIFT ＋ F1）

F7：一字変換方式……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。
（SHIFT ＋ F2）

F8：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。
（SHIFT ＋ F3）

F9：システム辞書変換方式…システム辞書ディスクを使用するときの方式です。
（SHIFT ＋ F4）

F10：ユーザ辞書変換方式……ユーザ辞書ディスクを使用するときの方式です。
（SHIFT ＋ F5）

●一字変換、音訓変換、システム辞書、ユーザ辞書変換方式について

・一字変換機能

コンピュータの内部プログラムとして常駐しています。

・音訓変換方式

これを利用するためにはシステムディスク中の音訓辞書が必要です。

・システム辞書、ユーザ辞書変換方式

それぞれシステム辞書・ユーザ辞書 が必要です。

なお、音訓変換方式、システム辞書、ユーザ辞書の方式を用いるためには、変換エリアをメモリ内部に確保するために、ＬＩＭＩＴ命令（またはＣＬＥＡＲ命令）でエリアの確保を行う必要があ

ります。指定方法はそれぞれ次の通りです

LIMIT&HF000　または　CLEAR&HF000

※入力モードの選択は、ファンクションキー F1 ～ F10 の代わりに GRAPH を押しながら数字キー 1 ～ 0 を入力することによって行うことができます（ただし、テンキーを数字キーとして用いることはできません）。

〔例〕　全角／半角の切換え方法… F1 または GRAPH ＋ 1 を入力する。
○音訓変換方式への切換え方法… F8 （ SHIFT ＋ F3 ）または
GRAPH ＋ 3 を入力する。

3.5　漢字変換方式について

漢字変換方式には前項でも説明したように、次の5種類があります。

1．コード入力方式
2．一字変換方式
3．音訓変換方式
4．システム辞書変換方式
5．ユーザ辞書変換方式

1，2，3について説明します。4，5についてはアプリケーションソフト説明書の4章システム・ユーザー辞書の説明をお読みください。

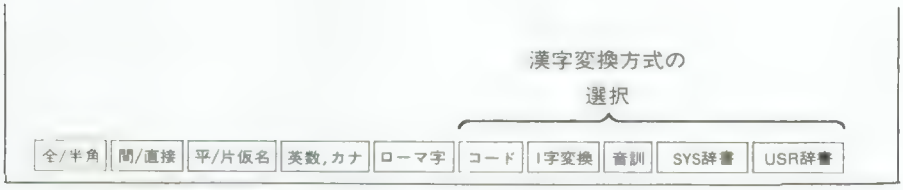
なお、上記1～5の漢字変換方式を選択するには次のように行なってください。

①ディスクBASIC（CZ－8FB02）を起動します。

②次に、日本語入力モードに入ります。

（ SHIFT ＋ XFER または CTRL ＋ XFER をキー入力します）

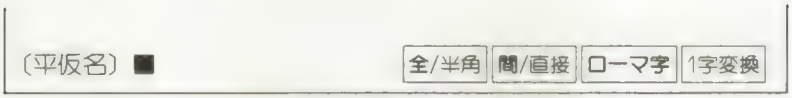
③ HELP キーを押すと、



BASIC起動時は音訓変換方式になっています。

・コード入力方式を選択するときは、　F6 （ SHIFT ＋ F1 ）
・一字変換方式を選択するときは、　F7 （ SHIFT ＋ F2 ）
・音訓変換方式を選択するときは、　F8 （ SHIFT ＋ F3 ）
・システム辞書変換方式を選択するときは、
ドライブのにシステム辞書ディスクを入れて　F9 （ SHIFT ＋ F4 ）
・ユーザー辞書変換方式を選択するときは、　F10 （ SHIFT ＋ F5 ）

ドライブのにユーザー辞書ディスクを入れて
のキーを押してください。
④次に HELP キーを押すと、日本語入力モードに戻っています。



一字変換入力方式にした例

3.5.1　コード入力方式

コード入力方式を選択するには F6 または SHIFT ＋ 6 を押します。
コードには、ＪＩＳ漢字コードと区点コードの２種類があり、どちらで入力するかは同じく
F6 または GRAPH ＋ 6 を押すことによって切り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、変換フィールド内では数字の０～９およびアルファベット大文字のＡ～Ｆ以外を入力することができなくなります。

●コードの入力は４桁で行い、（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

とみなされます。

●コードと全角文字とは１対１に対応しており、その対応関係の詳細は『ＢＡＳＩＣ　ＲＥＦＥＲＥＮＣＥ　ＭＡＮＵＡＬ　付録Ａ．３．３』を参照してください。ここでは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
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（＊は任意の数字を表します）。

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 を押す必要がありません。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。



〔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。



●全角文字には、漢字，非漢字文字以外にユーザ登録文字（外字）があります。
外字はPCGに定義され、次の２種類があります。

外字全角文字…PCG４キャラクタ分に定義
外字半角文字…PCG２キャラクタ分に定義

（外字の定義方法については、『アプリケーションソフト説明書』の「デフチャーツール」の項を参照してください。）

ここでは外字全角文字の表示方法について述べます。

コード入力方式では、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一水準および第二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（６４文字分）

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します。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、『アプリケーションソフトの説明書』の「デフチャーツール」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&HFEまでのPCGキャラクタコードです。

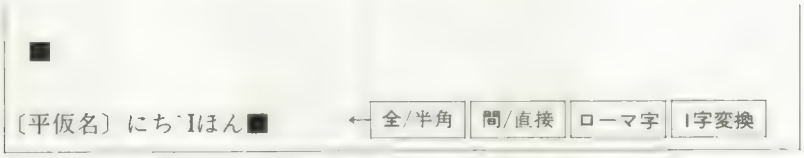
3.5.2　一字変換方式

変換フィールド内で入力したカナの最初の１文字をもとに、先頭にその読みを持つ漢字をＪＩＳ漢字コードの順番通りに羅列します。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次々と表示します。

例１

一字変換方式によって、漢字〝日本〟を表示します。

①入力モードを全角文字出力（ F1 ）、間接出力（ F2 ）、ひらがな出力（ F3 ）、ローマ字入力（ F5 ）、一字変換方式（ F7 ）に設定します。
②変換フィールドに N I C H I H TAB H O X　とタイプします。



③ XFER を押します。すると、〝にち〟の先頭文字〝に〟の読みを持つ漢字が羅列されます。目的の漢字が見つかるまで XFER か ⇩ かスペースキーを押し続けます。





④目的の漢字〝日〟を取り出します（ ↵ または数字キー 1 を押します）。



⑤同様に、 XFER を押して〝本〟の先頭文字〝ほ〟の読みを頭に持つ漢字を羅列します。



⑥目的の漢字〝本〟を取り出します。



以上で漢字〝日本〟が表示されます。

例2

キャラクタコードによる一字変換。

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。たとえばキャラクタ〝！〟（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れる全角文字は、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～O	&H３０～&H４Ｆ	３０２１～４Ｆ７Ｅ	第１水準　漢字
P～u	&H５０～&H７５	５０２１～７５７Ｅ	第２水準　漢字

羅列された漢字グループから目的の漢字を取り出す方法は、通常の方式とまったく同じです。

（注）入力方式としてローマ字ー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。

3.5.3　音訓変換方式

カナー漢字変換の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。ディスクＢＡＳＩＣを起動させたとき、日本語入力モードにすると音訓変換方式が初期状態として設定されます。



この音訓変換方式の利用については『３．２　全角文字の入力』で説明していますので、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。

（a）　〝キゴウ〟……各種記号を羅列します。
（b）　〝ギリシア〟…各種ギリシア文字を羅列します。
（c）　〝ロシア〟……各種ロシア文字を羅列します。
（d）　〝ガイジ〟……外字全角文字を羅列します。

変換フィールド内で上記カナ文字列を入力した後 XFER を押すと、変換フィールドエリアに全角文字が羅列されますので、目的の文字を取り出します。

●音訓辞書の学習機能について

カナー漢字変換の音訓変換方式では、直前に使用した漢字を同一グループ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を先頭に羅列することが可能です。

この機能は、数字キーによる漢字の選択を行った場合のみ行われ、カーソルコントロールキーによる選択時には行われません。また同梱の音訓辞書ディスク（システムディスク）が書き込み禁止状態にあるときには働きません。

●辞書機能を利用する前に

音訓辞書機能やシステム辞書機能、ユーザ辞書機能を利用して漢字変換を行うためにそれぞれ音訓辞書、システム辞書ディスク、ユーザ辞書ディスクが必要です。 これらの辞書機能を利用する場合は、辞書ディスクを挿入したデバイス名をコンピュータに教えなければなりません。そのために、次の操作を行います。

①DEVICE命令を使って、辞書ディスクを挿入したデバイス名をデフォルトデバイス*に指定します。

・フロッピーディスク…0：～3：
・グラフィックメモリ…MEM0：, MEM1：

②日本語入力モードにして漢字変換方式を選択します。選択のしかたは本章の3．5を参照してください。

(注) デフォルトデバイス…デバイス名の指定を省略したときに用いられるデバイス

これらの操作によって、コンピュータはそれ以降、辞書ディスクをアクセスする際に指定したデバイスに対してアクセスを行います。一度この操作を行うと指定を変更しない限り、アクセスするデバイスは変りません。また、各辞書機能を変更する際には再び上記①、②の操作を繰り返してください。

〔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

①DEVICE"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「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。

こ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VRAMに転送することによって行います。そのためには次の操作を行います。

①グラフィックVRAMの使用目的を外部記憶用に設定します。

```
OPTION SCREEN 2 ↵
```

②外部記憶用に設定したグラフィックVRAM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るので、 Y 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VRAMにコピーします。

```
COPY "0：音訓　　変換.DIC"AS "MEM0：" ↵
```

●日本語入力モードにおける特殊キーの動き

キー入力	動　　作
SHIFT ＋ XFER または CTRL ＋ XFER	日本語入力モードに切り換えたり、 日本語入力モードから抜け出したりする。
XFER	変換を行なう。
HELP	入力モードの確認。
F1 ～ F10 または GRAPH ＋ 1 ～ 0	入力モードの選択。

●変換フィールドエリア内に入力された状態での働き

キー入力	動　　作
↵	変換フィールドエリア内の文字をテキストエリアに移す。
INS DEL	変換フィールドエリア内の文字を抹消する。
CLR HOME	変換フィールドエリア内の文字をクリアする。
カーソルコントロールキー	無効。
HTAB	連続して漢字を変換する。

●漢字グループを表示している状態

キー入力	動　　作
← →	漢字グループ内のカーソルを移動する。
↑ ↓	漢字グループの次候補、前候補を選択する。
ESC	漢字グループの表示を消し、交換フィールド内にカーソルを戻す。
↵	目的の漢字をテキストエリアに表示する。
テンキー 1 ～ 9	目的の漢字に対応したテンキーを押せばその漢字をテキストエリアに表示する。

④デフォルトデバイスをグラフィックVRAMに指定します。

```
DEVICE "MEM0:"↵
```

⑤日本語入力モードに切り換えます。

SHIFT ＋ XFER

⑥音訓変換方式を選択します。

SHIFT ＋ F3

⑦デバイスの指定を④以前に戻します。

以上で、以後の音訓変換は、グラフィックVRAMに対して行われますので、非常に高速な漢字変換ができます。

システム辞書およびユーザ辞書についてはグラフィックVRAMを利用した高速変換はできません。

3.6 ミニ・ワープロ機能

本機のBASIC（CZ－8FB02）では、漢字変換機能だけでなく簡単なワードプロセッサとしての機能を持っています。この機能を利用すれば日本語文章を画面に表示させたり、プリンタに印字したりすることが簡単に行えます。本機のBASICでは、そのために次の命令を用意しています。

LIST*	……画面表示命令
LLIST*	……プリンタ印字命令
CONSOLE#	……印字エリア設定用命令
AUTO*	……REM付き行番号自動発生命令

●LIST*

まずアポストロフィ「'」（REM文の省略形）を使って次の命令を入力してください。

```
10 'パソコンテレビ　X1turboZ
20 '　パーソナル　コンピュータ
30 '　　CZ－880C　新発売
```

ここで、LIST*↵を実行すると次の画面のように、行番号や「'」もない普通の文章になっています。

```
パソコンテレビ　X1turboZ
　パーソナル　コンピュータ
　　CZ－880C　新発売
```

また、LIST*の終わりに行番号を指定することにより、任意の行番号だけの表示も可能です。

●LLIST*

LLIST*命令はアポストロフィ「'」付きの文章だけをプリンタに印字させる働きをします。したがってメッセージや説明文、手紙などの文章をアポストロフィ「'」を使ってプログラム中に作成することが可能となります。

●CONSOLE#

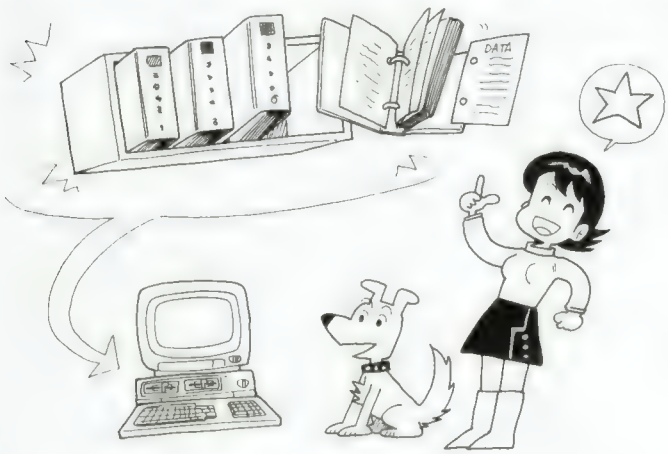
CONSOLE#命令を使えばプリンタの印字エリアを設定することができますので、ハガキ大のものから大判サイズのものまで自由に印字することができます。

●AUTO*

アポストロフィ「'」を使ったプログラムの作成には、AUTO*命令を使用すると便利です。AUTO*命令は、自動的にアポストロフィ「'」付きの行番号を発生させますので、プログラムということを意識することなく文章を作成することができます。ただし、1つの行番号に入力できる文字数には限りがありますので注意してください。(半角文字の場合250文字程度、全角文字の場合125文字程度)

第4章　ファイルについて

一般的にファイルというと、「書類などの情報をバインダ等にとじたもの」という意味で用いられています。ＢＡＳＩＣで使用するファイルという言葉もこれと同じ意味を持っていますが、情報を記録しておくものがバインダではなく、デバイスと呼ばれる記憶装置であるという点が異なります。



4.1　ファイル管理

書棚にしまっておくファイルに見出しを付けておくように、ＢＡＳＩＣで扱うファイルにも名前が付けられます。

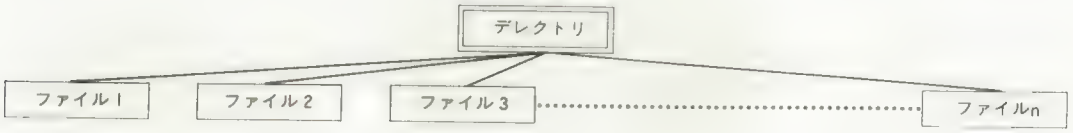
さらに、ファイルを探しやすいように、ファイルの名前といった、ファイルについての情報は、ファイル本体とは別の場所に保管されています。フロッピーディスクのように、ランダムアクセスファイルを扱うことのできるデバイスでは、名前などの情報を**ディレクトリ**と呼ばれる場所にまとめて記録しています。これに対して、カセットテープのように、シーケンシャルアクセスファイルしか扱うことのできないデバイスでは、名前などの情報を１ヶ所にまとめずに、各ファイルの直前に記録しています。このため、以下で述べるディレクトリについては、カセットテープなどには適用できませんので注意してください。

4.1.1　階層ディレクトリ

ディレクトリというのは、デバイスの中に記憶されているファイルの登録簿のことで、本にたとえれば目次に当たります。ファイル名、ファイルの種類、属性、記憶されている位置、ファイルサイズ、作成日時などが記録されています。

ＦＩＬＥＳ命令は、このディレクトリを参照することによってファイル名の一覧を表示することができるのです。

従来の多くのパーソナルコンピュータで行われてきたファイル管理の方法は、ディスク上にただ１つのディレクトリ、つまり下図のような単層のディレクトリを置くことにより行われていました。



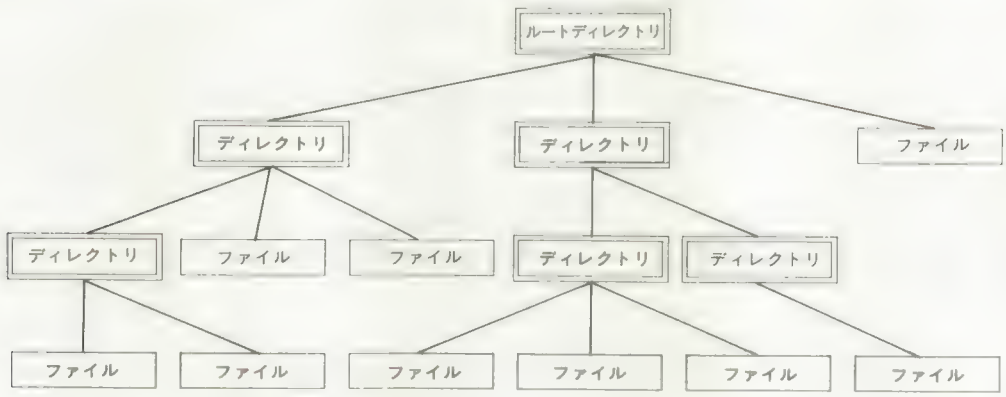
ＦＩＬＥＳ命令を実行すると、ファイル１、ファイル２、……、ファイルｎの、ディスクに格納されているすべてのファイルの名前が一度に表示されることになります。

単層ディレクトリのもとでは、保存するファイルの数が増えると管理が難しくなってきます。たとえば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はって区別するなどの方法を用いるしかありません。

ところが、ハードディスクのように、１台の中にフロッピーディスク何十枚分もの大量のデータを扱えるデバイスではもはやこの方法は使えません。

ＦＩＬＥＳ命令でファイル名一覧を表示させる場合を考えましょう。このとき、ファイル数が１０や２０であれば問題は生じません。しかし、ファイル数が１００、２００となった場合、その中から特定のファイルを探し出すことが容易に行えるでしょうか。

このような問題を解決するために考え出されたのが、階層ディレクトリと呼ばれるファイルの管理方法です。



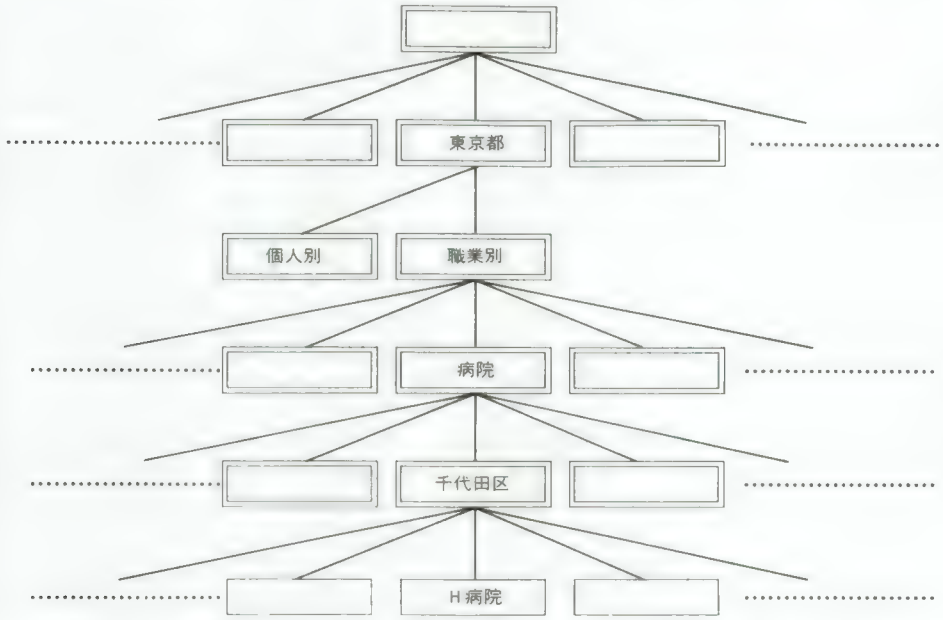
たとえば、国内のＨという病院の電話番号を、電話帳を使って調べる場合を考えます。Ｈ病院が東京都千代田区にあったとすると、まず東京都23区の電話帳が必要となります。電話帳は個人別のものと職業別のものとに分かれていますが、この場合は職業別のものが必要になります。次に、電話帳の目次から病院の項のページを調べ、そのページを開きます。すると、病院名と住所、電話番号が列記されていますが、これらは各区ごとに分類されています。そこで、今度は千代田区の項目を探し出し、さらにその項に列記されている病院の中からＨ病院を探し出し、電話番号を知ることができます。

H病院の電話番号をみ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」がファイルに対応します。



ディレクトリの区切りを表す文字として〝　〟（スラッシュ）を使用します。すると、先ほどの例は次のように表されます。

／東京都／職業別／病院／千代田区／H病院

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を**ルートディレクトリ**と呼びます。これは、先頭の〝／〟に対応します。

4.1.2　カレントディレクトリ

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にファイルの管理を行います。このため、別のディレクトリの下であれば、同一の名前のファイルを存在させることが可能です。

先ほどの例では、千代田区ではなく港区にもH病院という名前の病院があってもよいということです。

／東京都／職業別／病院／港区／H病院

千代田区のH病院や港区のH病院を表すには、ルートディレクトリからのパス名を書いてやればよいわけですが、いつでもこのような長い名前を書かなければならないのでは面倒です。そこで、あるディレクトリに注目し、そこからたどってゆくことにすればファイルを簡単に指定することができるようになります。現在注目しているのが

／東京都／職業別／病院／千代田区

であるとすれば、

H病院

とするだけで、

／東京都／職業別／病院／千代田区／H病院

を表すことができるのです。このような、「現在注目しているディレクトリ」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は任意のディレクトリに変更することができます。これには、CHDIR命令を使います。

CHDIR"〔デバイス名：〕ディレクトリ名"

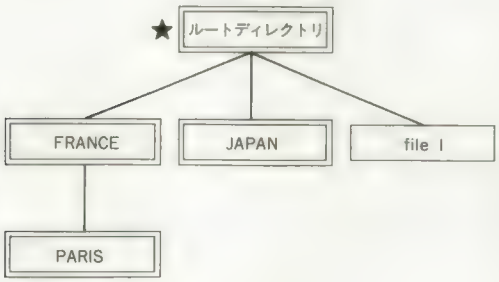
デバイス名を指定すると、そのデバイスのカレントディレクトリが変更されます。

たとえば、現在のカレントディレクトリがルートディレクトリで、これをディレクトリFRANCEへ変更する場合、次の操作を行います。

```
CHDIR "FRANCE"
```

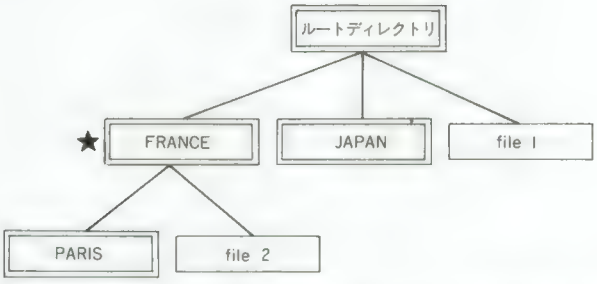
このときにFILES命令を実行すると次のように表示されます。

```
xx Clusters free
Path name    "x:/FRANCE/"
Dir*         "x:PARIS      DIR"
```



ここで、ＢＡＳＩＣプログラムを″ｆｉｌｅ２″という名前で作成する場合を考えましょう。これには、次の操作を行います。

```
SAVE "file2"
```



ここからさらに下のディレクトリＰＡＲＩＳへ移るには次の操作を行います。

```
CHDIR "PARIS"
```

カレントディレクトリをルートディレクトリへ変更するには以下の操作を行います。

```
CHDIR "／"
```

4.1.3　ディレクトリの一覧

外部ファイルにセーブされたファイル名の一覧表を表示するにはＦＩＬＥＳ，ＬＦＩＬＥＳ命令を使います。

```
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：，
```

指定したデバイス上のファイル名一覧が、ＦＩＬＥＳの場合は画面へ、ＬＦＩＬＥＳの場合はプリンタに表示されます。

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

Ｂａｓ……ＢＡＳＩＣプログラムファイル
Ａｓｃ……アスキー形式で記録されたファイル
Ｂｉｎ……機械語プログラムファイル
Ｄｉｒ……ディレクトリファイル

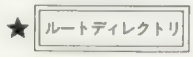
［注意］カセットテープなど一部のデバイスでは、ディレクトリファイルを扱うことはできません。また、カセットテープに対してＦＩＬＥＳおよびＬＦＩＬＥＳを実行すると、デバイス名が〝ＣＡＳ０：〟と表示されますが、〝ＣＡＳ：〟と〝ＣＡＳ０：〟とはまったく同じものです。

4.1.4　ディレクトリの作成

ディスクのフォーマットを行うと、ディレクトリが初期化され、ファイルはすべて削除されます。この状態でＦＩＬＥＳ命令を実行すると、以下のように表示されます。

```
xx Clusters free
Path name "x: "
```

ディスク上には、ルートディレクトリのみが存在していることがわかります。

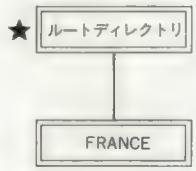


ディレクトリを新しく作成するにはＭＫＤＩＲ命令を使用します。

```
MKDIR"〔デバイス名：〕ディレクトリ名"
```

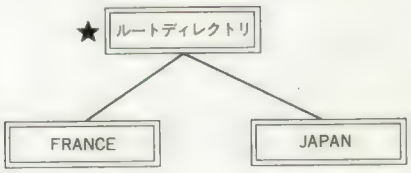
たとえば、ルートディレクトリの下にＦＲＡＮＣＥというディレクトリを作成する場合は、次のようにします。

```
MKDIR "FRANCE"
```



さらに、ＪＡＰＡＮというディレクトリを作成する場合、次の操作を行います。

```
MKDIR "JAPAN"
```

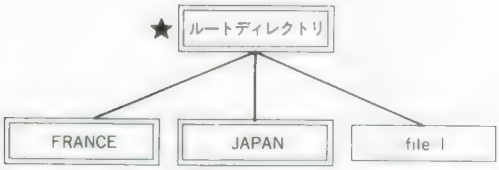


もちろんここに、ディレクトリではなく、プログラムファイルやデータファイルを作成してもよいのです。たとえば、ＢＡＳＩＣプログラム〝ｆｉｌｅ１〟を作成する場合、次のようにします。

```
SAVE "file1"
```

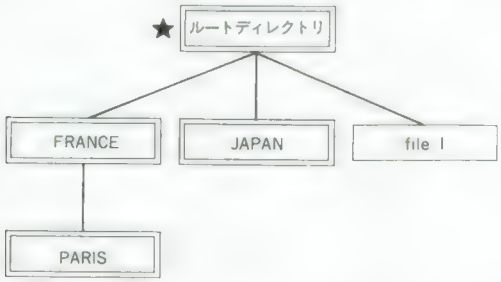
ディレクトリがどのようになっているかを見るために、ＦＩＬＥＳ命令を実行してみましょう。

```
xx Clusters free
Path name     "x:/"
Dir*          "x:FRANCE    .DIR"
Dir*          "x:JAPAN     .DIR"
Bas           "x:file1     .    "
                :
                :
```



ディレクトリFRANCEの下にＰＡＲＩＳというディレクトリを作成するには、次のように操作を行います。

```
MKDIR "FRANCE/PARIS"
```



なお、カレントディレクトリを変更して、次のようにしてもやはりＦＲＡＮＣＥディレクトリの下にＰＡＲＩＳディレクトリが作成されます。

```
CHDIR "FRANCE"
MKDIR "PARIS"
```

さらに下の階層を作成する場合も同様に／で区切って指定します。

```
MKDIR "..... ..... ..... ....."
```

4.1.5　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命令を使用します。

```
RMDIR "〔デバイス名：〕ディレクトリ名"
```

ディレクトリJAPANの下にあるTOKYOというディレクトリを削除する場合は、次の操作を行います。

```
RMDIR "／JAPAN／TOKYO"
```

このとき、ディレクトリTOKYOの下にあるファイルおよびディレクトリはすべて削除されていなければならず、もし、ひとつでもファイルやディレクトリが残っているとディレクトリの削除は行えません。KILLコマンドで削除してから行ってください。

4.1.6　ファイルディスクリプタ

ここでは、BASICで扱うファイルの正確な指定方法を説明します。

ファイルの指定は、ファイルディスクリプタという文字列を用いて行います。ファイルディスクリプタは次の形式で表現されます。

```
"［デバイス名：］［パス名］［ファイル名］"
```

たとえば、次のようなものは正しいファイルディスクリプタです。

```
"1：／JAPAN／TOKYO／file1"

"　FRANCE　PARIS　file2"

"HD0：file3"
```

デバイス名

デバイス名はファイルの存在する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（：）で現されます。

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（3または5インチ版） ｜ ディスク3（3または5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として使用することができます。

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どこにあるかを示します。

また、パス名の指定はフロッピーディスクなどのようにランダムファイルを扱えるデバイスに対してのみ有効です。

パス名は以下の規則に従って付けられます。

（1）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ます。

／ディレクトリ／ディレクトリ ……

（2）1つのディレクトリは半角文字で1～13文字の文字列で表されます。全角文字は2文字分と数えます。

（3）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の機密を保持するために指定され、いったん指定すると、以後パスワードをつけずに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／……／

（4）デバイス名、ディレクトリ、セミコロン、パスワードを含めて半角文字128文字以内でなければなりません。

（5）パス名の先頭のスラッシュ"　"はルートディレクトリを示しています。先頭の"／"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。

（6）パス名の指定に誤りがあると"Bad file descripter"のエラーとなります。

ファイル名

ファイル名は、デバイス名で指定したデバイス上の、どのファイルであるかを指定するためのものです。ファイル名は以下の規則に従って付けられます。

（1）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〕

（2）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半角文字で1～3文字で表されます。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（；）、パスワードを合わせて、31文字以内でなければなりません。全角文字は2文字分と数えます。
この規定を超えた文字数を使用した場合、"Bad file descripter"のエラーとなります。

（3）パスワードはファイルの機密を保持するために付けられ、いったんパスワードを指定すると、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。

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（合言葉）を指定してファイルを作成します。すると、そのファイルを読み出すときには、指定したパスワードを入力しないと読み出せなくなります。パスワードを指定してファイルのアクセスを行うには、

"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指定しなければなりません。

パスワードには、ピリオド（.）、引用符（″）、コロン（：）、セミコロン（；）スラッシュ（／）は使用できません。また、パスワードを含めたファイル名の全体の長さは３１文字以内でなくてはなりません。

〔例〕メインメモリ上にあるBASIC　プログラムを、ファイル名″TEST″、パスワード″ABC″でフロッピーディスク（ドライブ０）にセーブするには、

```
SAVE "0:TEST;ABC" ↵
```

とします。そのとき、

```
FILES"0:" ↵
```

と、ファイル名のリストを表示させてもパスワードは表示されません。しかし、

```
LOAD "0:TEST" ↵
```

と入力して読み出そうとしても、

```
No Password
```

とエラーが表示され、読み出すことができません。読み出すためには、

```
LOAD "0:TEST;ABC" ↵
```

とパスワードを指定しなければなりません。

以上のようにファイルにパスワードを指定すると、パスワードを知っている人でなければファイルの操作ができなくなります。

4.1.7　デバイス名の省略

ファイル操作を行うプログラムでは、同じデバイス名を何度も指定しなくてはいけない場合があります。このような場合、DEVICE命令によって、省略したときのデバイス名を指定することができます。

```
DEVICE ″デバイス名″
```

ファイル操作命令の使用時に、ファイルディスクリプタ中のデバイス名を省略した場合、DEVICE命令で指定したデバイス名が用いられます。なお、DEVICE命令がまだ一度も使用されていない状態では、BASICを起動したデバイスが対象となっています。

［例］

```
DEVICE "1:" ↵
Ok
LOAD "X1-WORLD" ↵
Ok
```

この例では、フロッピーディスクのドライブ１上に存在する〝Ｘ１－ＷＯＲＬＤ〟をロードします。

4.1.8　ファイルの削除

不要になったファイルを削除するにはＫＩＬＬ命令を使用します。

```
KILL"〔デバイス名：〕ファイル名"
```

指定したデバイス内の指定したファイルを削除します。削除したプログラムは復元することができませんので、十分に注意してください。

デバイス名としては、０：～３：，ＭＥＭ０：～ＭＥＭ１：，ＥＭＭ０：～ＥＭＭ９：，Ｆ０：～Ｆ３：，ＨＤ０：～ＨＤ３：が使用できます。

〔例〕

```
KILL "1:X1-WORLD"
Ok
```

フロッピーディスク（ドライブナンバー１）に記録された〝Ｘ１－ＷＯＲＬＤ〟というファイルを削除します。

〔注意〕デバイス名に〝ＣＡＳ：〟を指定して、カセットテープ上に記録されたプログラムを削除することはできません。

4.1.9　ファイル名の付けかえ

ファイルのファイル名を付けかえるにはＮＡＭＥ命令を使用します。

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

デバイス名としては、０：～３：，ＭＥＭ０：～ＭＥＭ１：，ＥＭＭ０：～ＥＭＭ９：，Ｆ０：～Ｆ３：，ＨＤ０：～ＨＤ３：が使用できます。

旧ファイル名で指定したファイルの名前を、新ファイル名に変更します。このとき、２つのデバイス名は同じでなければなりません。また、デバイス名に〝ＣＡＳ：〟は指定できません。

〔例〕フロッピーディスク（ドライブ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。

```
NAME "0:TEST" AS "0:SAMPLE"
```

4.1.10　ファイルの属性指定

ファイルの属性には次のものがあります。

（１）作成したファイルを誤って消去しないように書き込みを禁止する**ライトプロテクト属性**。

（2）ファイルに書き込みを行った後、自動的に書き込んだ内容と読み出した内容の比較を行い、書き込みの信頼性を上げる**リードアフターライト属性**

（3）ＦＩＬＥＳ命令を行ってもファイル名が表示されない**シークレット属性**。

ファイルの属性を設定または解除するにはＳＥＴ命令を使用します。ただし、ＳＥＴ命令は〝ＣＡＳ：〟には使用できません。

```
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：OPEN文で指定したファイル番号

デバイス名としては、0：～3：，MEM0：～MEM1：，EMM0：～EMM9：，F0：～F3：，HD0：～HD3：が使用できます。

属性を指定する文字として"P"，"R"，"S"を使用します。

"P"を使用すると、ライトプロテクト属性が設定されます。
"R"を使用すると、リードアフターライト属性が設定されます。
"S"を使用すると、シークレット属性が設定されます。
" "　を使用すると、すべての属性が解除されます。

（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"
```

ＦＩＬＥＳ命令を実行してください。〝SAMPLE〟というファイルの〝Bas〟の次に〝＊〟マークがついています。 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ています。

（例2）ファイル番号1でオープンしたファイルに、リードアフターライトの属性を設定します。

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

以後このファイルにデータを書き込む際には、自動的にベリファイ動作を行います。このファイルをクローズした時点でリードアフターライト属性は解除されます。

（例3）すでにライトプロテクトをかけたファイルに、さらにシークレット属性を設定します。

```
SET "0:SAMPLE","S"
```

これでシークレット指定されました。ＦＩＬＥＳ"0:"としても〝0：SAMPLE〟と

いうファイル名が表示されなくなりましたが、"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わないというだけです。ためしに、

LOAD "SAMPLE"↵

として、

LIST↵

とやってみましょう。ちゃんとプログラムリストが表示されます。

4.2 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記憶装置にプログラムファイルとして保存・再生することが可能です。また、プログラムにはBASICファイルと機械語ファイルの2種類があり、扱いが少し異なります。

4.2.1 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）といいます。セーブはSAVE，LIST，SAVEM命令によって実行できます。

```
SAVE 〔"〔デバイス名：〕ファイル名"〕〔，A〕
```

デバイス名として、0：～3：、CAS：，MEM0：～MEM1：，EMM0：～EMM9：，F0：～F3：，HD0：～HD3：，COM：を使用することができます。

SAVE命令は、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でセーブします。なお、ファイル名にはパス名を付けて、階層ディレクトリの任意のディレクトリにセーブすることができます。

また、最後に"A"オプションを指定すると、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LIST命令で画面に出力されるような文字でセーブすることです。"A"を指定しない場合は、プログラムは中間コードに変換されてセーブされますが。アスキーセーブは中間コードのセーブよりも多くのファイル領域が必要になり、その結果セーブやロードに時間がかかりますが、後で述べるMERGE命令で使用することができ、またシーケンシャルデータファイルとして用いることができるなど、有利な点があります。

デバイス名に"COM："を指定した場合、プログラムはRS－232Cポートへ送り出されます。この場合、"A"オプションを付けなくてもアスキー形式でセーブされます。

（例）次のプログラムを"X1－WORLD"という名前でセーブします。

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

RUNすると回数を聞いてきますから、数字を入力してください。入力した回数だけ〝X1−WORLD〟と表示する簡単なプログラムです。

上のプログラムを入力後、次のようにしてセーブします。

・カセットテープにプログラムをセーブする場合

```
SAVE "CAS:X1-WORLD"↵
Ok
```

・フロッピーディスク（ドライブ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 SCREEN 2↵
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〕〕〕

デバイス名として、SAVE命令で指定できるもの以外に〝SCR：〟，〝CRT：〟，（どちらも画面）、〝LPT：〟（プリンタ）が使用できます。

LIST命令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。なお、ファイル名にはパス名を付けて、階層ディレクトリの任意のディレクトリに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」とでは、セーブする行番号を指定できる点が異なります。ＬＩＳＴ命令は、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」までが、また、「開始行番号」だけを指定した場合は、その行番号の行だけがセーブされます。

また、デバイス名、ファイル名を省略すれば、プログラムを画面上に表示するという、今までに何度も出てきた使い方になります。

この命令と、後で述べるＭＥＲＧＥ命令を合せて用いることによって、あるプログラムのサブルーチンを他のプログラムに追加することなどが容易に行えます。

［例１］

```
LIST "CAS:EXP",30-50 ↵
Ok
```

これを実行すると、メインメモリにあるプログラムの行番号３０～５０までがアスキー形式で、カセットテープに〝ＥＸＰ〝というファイル名でセーブされます。

［例２］

```
LIST "0:TEST",10-40 ↵
Ok
```

これを実行すると、メインメモリ上にあるプログラムの行番号１０～４０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〝ＴＥＳＴ〝というファイル名でセーブされます。

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〕

デバイス名として、ＳＡＶＥ命令と同じものが指定できます。

ＳＡＶＥＭ命令は、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、なおファイル名にはパス名を付けて、階層ディレクトリの任意のディレクトリにセーブすることができます。

「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。これはファイルに記録され、ロードするときに使われます。

「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アドレスを指定するもので、省略すると「開始アドレス」が「実行開始アドレス」として使われます。

［例］

```
SAVEM "CAS:TEST",&H8000,&H80FF,&H800E
Ok
```

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。

4.2.2　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にセーブされたプログラムファイル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命令によって実行できます。

LOAD〔"〔デバイス名：〕ファイル名"〕

デバイス名として、SAVE命令と同じものが指定できます。

LOAD命令は、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へロードします。なお、ファイル名にはパス名を付けて、階層ディレクトリの任意のディレクトリのファイルをロードすることができます。

アスキー形式でセーブされたプログラムも同じ方法でロードします。

ファイル名はセーブしたときのものと同じでなければなりません。もし、指定したファイルがみつからなかった場合は〝File　not　found〝エラーになります。

なお、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。

デバイス名に〝COM：〝を指定した場合には、RS－２３２Cポートからプログラムを読み込みます。この場合、通信回線から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命令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。

・カセットテープからプログラムをロードする場合

まずプログラムの先頭までテープを巻戻し（または早送り）した後、次の命令を入力します。

```
LOAD "CAS:X1-WORLD"↵
Ok
```

・フロッピーディスク（ドライブ１）からプログラムをロードする場合

```
LOAD "1:X1-WORLD"↵
Ok
```

RUN〔"〔デバイス名：〕ファイル名"〕

デバイス名として、SAVE命令と同じものが指定できます。

RUN命令は、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へロードした後、直ちに実行します。なお、ファイル名にはパス名を付けて、階層ディレクトリの任意のディレクトリのファイルをロードすることができます。

また、デバイス名、ファイル名を省略すれば、メインメモリ中のプログラムを実行するという今までに何度も出てきた使い方になります。

（例）

```
RUN "1:X1-WORLD"↵
Ok
```

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

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

デバイス名として、ＳＡＶＥ命令と同じものが指定できます。

ＬＯＡＤＭ命令は、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へロードします。なお、ファイル名にはパス名を付けて、階層ディレクトリの任意のディレクトリのファイルをロードすることができます。

「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がロードされます。

「ロード開始アドレス」を省略した場合、ＳＡＶＥＭで指定した「セーブ開始アドレス」からロードされます。

最後のパラメータ〝Ｒ〟を指定した場合は、ロードした後で直ちにプログラムを実行します。

（例１）

```
LOADM "CAS:TEST",,R↵
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。

（例２）

```
LOADM "CAS:TEST"↵
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。

4.2.3　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比べて、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。このとき使うのが、ＶＥＲＩＦＹまたはＬＯＡＤ?です。この２つはまったく同じ意味の命令です。

```
VERIFY〔"〔CAS:〕ファイル名"〕
LOAD?　〔"〔CAS:〕ファイル名"〕
        ※この命令で扱えるデバイスは"CAS:"のみです。
```

この命令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ってください。

［例］ここではSAVE命令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、ベリファイを行う手順を示します。

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ドにセットします。テープカウンタを0にしておくか、カウンタ番号を記録しておけばカセットテープのどこに記録したかの目安になります。

```
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テープをセーブ開始位置まで巻戻し、次のように入力を行います。

```
VERIFY"CAS:X1−WORLD"
```

正しくセーブされているならば次のように表示されます。

```
Found "X1-WORLD"
```

正しくセーブされていない場合には次のように表示されます。

```
Tape read error
```

この場合、もう一度セーブしなおしてください。

4.2.4　プログラムのマージ

メインメモリ上に存在するBASICプログラムと、デバイス上から読み込んだBASICプログラムとを合成し、1つのプログラムにすることを、BASICプログラムのマージといいます。これにはMERGE命令を使います。

```
MERGE　"〔デバイス名:〕ファイル名"
```

デバイス名として、SAVE命令と同じものが指定できます。

指定したプログラムファイルが読み込まれ、メモリに存在するプログラムと組み合わさった1つのプログラムがメモリ上に作成されます。

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

でないファイルをマージしようとすると〝Bad file mode〟のエラーとなります。

なお、双方で同じ行番号の行が存在する場合には、ファイルから読み込んだ方が優先されます。

〔例〕MERGEの手順を示します。

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0）にアスキー形式でセーブします。

```
SAVE "0:TEST", A ↵
OK
```

③メインメモリのプログラムを消します。

```
NEW ↵
OK
```

④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ＬＩＳＴ↵として、リストの確認をします。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わかります。

```
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でその合計を計算し表示するというものです。

4.2.5　プログラムのチェイン

プログラムが長すぎるために、メインメモリで一度に処理できない場合があります。そんなときは、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。このことを「プログラムをチェインする」といい、ＣＨＡＩＮ命令を使用します。

```
CHAIN"〔デバイス名：〕ファイル名"
```

デバイス名として、ＳＡＶＥ命令と同じものが使えます。

ＣＨＡＩＮを実行すると、使用している変数を保存し、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。

なお、ロードする前にオープンしていたデータファイルはＣＨＡＩＮの際にクローズされません。

〔例〕２つのプログラムを用いてＣＨＡＩＮの例を示します。

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"
```

〔注〕カセットテープを使用するときは、行番号７０を次のように変えて入力してください。

```
70 CHAIN"CAS:SAMPLE2"
```

（例）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０）に〝ＳＡＭＰＬＥ２〟というファイル名でセーブします。

```
SAVE "0:SAMPLE2" ↵
Ok
```

〔注〕カセットテープを利用する場合は次のようにしてください。

```
SAVE"CAS:SAMPLE2" ↵
```

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絶対値を表示するプログラムです。

```
RUN
DATAノ コスウ3
 1 バンメ ノ DATA
? 143.3
 2 バンメ ノ DATA
? 12345.67
 3 バンメ ノ DATA
? .410

| 143.3   - 12345.67  |= 12202.37

| 12345.67 - .41      |= 12345.26
Ok
```

サンプル（1）の70行のCHAIN命令を実行することにより、フロッピーディスクドライブ0またはカセットテープからサンプル（2）のプログラムがロードされ、実行されます。

また、サンプル（2）は、CHAIN命令によってサンプル（1）の変数が保存されているため、サンプル（1）のプログラムで入力されたデータが処理され表示されます。

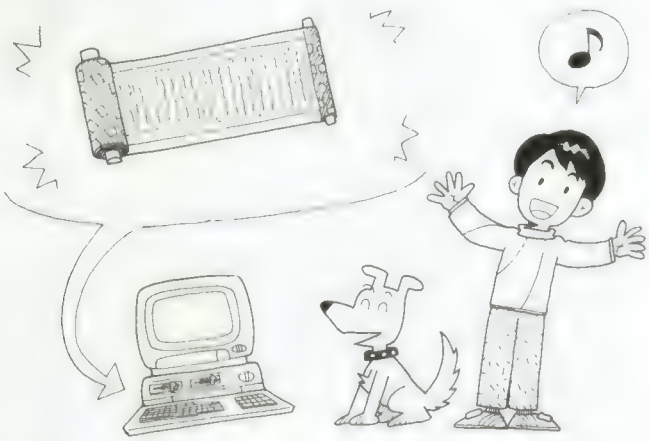
ただしCHAIN命令が実行されるとサンプル（1）のプログラムは消去されますので注意してください。

4.3　シーケンシャルアクセスファイル

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してセーブ・ロードすることができます。データファイルには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の２種類があります。

シーケンシャルアクセスファイルは、連続的に書かれたファイルのことで、次のような特徴があります。

・ファイルの中ではデータは書き込まれた順番に並んでおり、データをセーブ　ロードする際には先頭から順番に行います。
・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、必要な処理を行った後でまとめてファイルに書き込むことが必要です。
・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。
・ファイルの中のデータは順序よく並んでおり、無駄がありません。
・ファイルの作成が比較的簡単に行えます。



4.3.1　シーケンシャルアクセスファイルの作成

シーケンシャルアクセスファイルを作成したりデータの保存・再生を行うには、プログラムファイルのように１つの命令で行うことはできません。

シーケンシャルアクセスファイルを作成するには、ＯＰＥＮ命令を用いて次のようにします。

```
OPEN"O", #ファイル番号, "〔デバイス名：〕ファイル名"
```

ファイル番号は、プログラム中でそのファイルを扱うための識別番号で、１～１５の整数を指定します。

これによって新しいファイルが作成され、そのファイルに対してデータを書き込むことができるようになります。

データを書き込むには、ＰＲＩＮＴ#などを用います。すべてのデータを書き込んだら、その内容を保持するために、オープンしたファイルを閉じるという処理が必要です。そのための命令がＣＬＯＳＥです。

CLOSE　#ファイル番号

このファイル番号は、ＯＰＥＮの際に指定したものと一致していなければなりません。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。

シーケンシャルアクセスファイルへのデータの保存する方法を、実際のプログラム例を用いて説明します。なお、命令の詳細については、『BASIC リファレンスマニュアル』を参照してください。

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
```

〔注〕カセットテープを使用する場合は、行番号１０および１００を次のように書き換え、さらに行番号７０を新たに入力してください。

```
10 OPEN "O",#1,"CAS:DATA"
70 APSS -1
100 OPEN "I",#1,"CAS:DATA"
```

それでは、プログラムの説明を行います。

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

シーケンシャルアクセスファイルを作成し、それにデータを書き込むことを宣言するために、ＯＰＥＮ命令を使います。ＯＰＥＮ命令の各パラメータによって、次のことが設定されています。

″Ｏ″……ファイルへのデータの書き込み（ＯＵＴＰＵＴ）を設定します。

＃１……出力するファイル番号を＃１に設定します。

″１：ＤＡＴＡ″……出力デバイスを″１：″に、ファイル名を″ＤＡＴＡ″に設定します。

すなわち、行番号１０の実行によって、フロッピーディスク（ドライブ１）の″ＤＡＴＡ″というファイルに対して、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。シーケンシャルアクセスファイルにデータを出力する命令は、このＰＲＩＮＴ#の他にＷＲＩＴＥ#があります。この２つの命令の違いについては『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、ファイルをクローズします。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。

4.3.2　シーケンシャルアクセスファイルの読み込み

シーケンシャルアクセスファイルのデータを読み込むには、作成したときと同様にファイルをＯＰＥＮする必要があります。ただし、こんどは″Ｏ″の代わりに″Ｉ″（ＩＮＰＵＴ）を指定します。

```
OPEN "I", #ファイル番号, "〔デバイス名：〕ファイル名"
```

ファイル番号として、１～１５の整数を指定します。
これによって、そのファイルからデータを読み込むことができるようになります。
最後には、やはりＣＬＯＳＥをしなければなりません。

```
CLOSE  #ファイル番号
```

このファイル番号は、OPENの際に指定したものと一致していなければなりません。
再び例を用いて、シーケンシャルアクセスファイルからの読み込みを見てみることにしましょう。

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

シーケンシャルアクセスファイルからデータを読み出す場合、書き込む場合と同様OPEN文でファイルをオープンする必要があります。ただし、今度は読み出しですから、"I"（INPUT）に換えます。
行番号100のによって、フロッピーディスク（ドライブ1）のシーケンシャルアクセスファイル"DATA"からデータの読み出しが可能となります。

```
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。シーケンシャルアクセスファイルでは、データを読み出す場合、書き込んだ順に行う必要があり、一部のデータしか必要としない場合であっても、その前に書き込まれたデータをいったん読み出さなくてはなりません。

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

行番号130，140では、シーケンシャルアクセスファイルから読み出した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、読み出しの場合でも、その実行が終わったならばファイルを閉じる必要があります。
また、カセットテープを使用する場合、行番号70を追加してください。

```
70 APSS -1
```

これは行番号10～60で作成したファイルの頭出しを行うための命令です。

〔例〕このプログラム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
```

（「パソコンテレビ」は全角文字、「2」は半角文字で入力してください。）

4.3.3　シーケンシャルアクセスファイルへの追加

シーケンシャルアクセスファイルでは、ファイルの任意の位置への書き込み／読み出しは行うことはできませんが、ファイルの一番最後にデータを追加することができます。ただし、カセットテープではデータを追加することはできません。

データを追加するには、OPENの際に"A"（APPEND）を指定します。

```
OPEN"A", #ファイル番号, "〔デバイス名：〕ファイル名"
```

ここでは、先ほど作成したシーケンシャルアクセスファイルにデータを追加する例をあげて説明します。

次のプログラムを入力してください。

```
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"（APPEND）にします。行番号10ではフロッピーディスク（ドライブ1）の"DATA"というシーケンシャルアクセス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文で何回もデータを入力できるようになっています。追加されるデータは、それ以前のデータの後ろに、順に追加されます。行番号６０までででデータの追加は終わりです。「名前は？」のところで"ＥＮＤ"と入力するとデータの追加が終わります。

```
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アクセスファイルから、データの入力を行い、画面に表示します。行番号１０５のＩＦ文はシーケンシャルアクセスファイルの終わりをＥＯＦ関数（END　OF　FILE）によって検出し、もし終わりであれば行番号１５０に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という値（－１）を返し、そうでなければ偽という値（０）を返します。

〔例〕以下はこの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 0 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
```

4.4　ランダムアクセスファイル

シーケンシャルアクセスファイルに対して、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。

・ファイル中のデータは、インデックスによって直接任意のレコード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えます。
・ファイル中の一部のデータだけを取り出して変更することが可能です。

・記録したデータの最後にデータを追加することができます。
・データが少ない場合でもロード・セーブを1レコードごとに行うので無駄な部分が生じます。
・ファイルの作成のために多くの設定が必要なため、シーケンシャルアクセスファイルに比べてプログラムが複雑になります。

シーケンシャルアクセスファイルを用いればほとんどのデータ処理が行えますが、ランダムアクセス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価値が発揮できるといえます。

4.4.1　ランダムアクセスファイルの作成

シーケンシャルアクセスファイルでは、"I"などのパラメータを用いて、ファイルをどのようにアクセスするかを指定しなければなりませんでした。

これに対してランダムアクセスファイルは、どんな場合であっても"R"（RANDOM）と指定します。

```
OPEN"R", #ファイル番号, "〔デバイス名：〕ファイル名"
```

ここで、ファイル番号は、シーケンシャルアクセスファイルと同様に、ファイルを識別するための整数で、1～15で指定します。

もし、OPENしたときに指定したファイルが存在しなければ、新しいランダムアクセスファイルが作成されます。

ランダムアクセスファイルの場合も、ファイルの操作が終わった時点でクローズしなければなりません。

```
CLOSE  #ファイル番号
```

ファイル番号は、OPENの際に指定したものと同じでなければなりません。

4.4.2　レコードの割り付け

ランダムアクセスファイルは、決まった大きさ（256バイト）のレコードが集まったものと考えることができます。レコードを指定するには何番目のレコードであるかを言えばよく、この数のことをレコード番号、またはインデックスと言います。

ランダムアクセスファイルは、データの入出力を1レコード（256バイト）の大きさのファイルバッファ（メインメモリ上でデータを一時的に蓄える場所）を介して行います。データを読み込むときは、指定したレコード番号のレコードの内容をいったんファイルバッファに読み込み、データを書き込むときは、いったんファイルバッファにデータを格納してから指定したレコード番号のレコードに書き出すことになります。

ファイルバッファに格納するデータはすべて文字列に変換する必要があるために、シーケンシャルアクセスファイルに比べてよけいな手間がかかります。

レコードをどのように使うかを決めるためにFIELDを用います。

```
FIELD #ファイル番号，フィールド幅 AS 文字変数，……
```

ここでファイル番号は、OPENの際に指定したものと一致していなければなりません。

「フィールド幅 AS 文字変数」によって、レコードの最初から「フィールド幅」を「文字変数」に割り当てます。これを繰り返すことによって、その次の「フィールド幅」を、次の「文字変数」に割り当ててゆくことができます。

なお、FIELD文はファイルバッファに変数の割り付けを行うだけで、データをファイルバッファやファイルに入出力するわけではありません。

（例）

```
FIELD#1，10 AS A$，2 AS B$
```

ファイル番号1のファイルのレコードを、最初の10文字分をA$に、次の2文字分をB$にそれぞれ割り当てます。

4.4.3 ランダムアクセスファイルへの書き込み

ランダムアクセスファイルへデータを書き込むには、いったんファイルバッファに値を設定しておかなければなりません。

その際、数値データは文字型に変換しなければなりません。それには次の関数を用います。

MKI$(整数)
MKS$(単精度実数)
MKD$(倍精度実数)

その後で、FIELD文で割り当てた文字変数に値を格納するわけですが、これには普通の代入文を用いることはできず、次の命令を用いなければなりません。

```
LSET 文字変数＝文字式
RSET 文字変数＝文字式
```

ここで、「文字式」は、FIELD文で割り当てたものでなければなりません。
LSETは「文字変数」の表すフィールドの中に左詰めで格納しRSETは右詰めで格納します。
値を格納し終わったら、1レコードをファイルに書き出します。

```
PUT #ファイル番号，レコード番号
```

これで、ファイル番号のファイルのレコード番号のレコードに1レコードが書き込まれます。

4.4.4　ランダムアクセスファイルの読み出し

ランダムアクセスファイルからデータを読み出すには、いったんファイルバッファに１レコードを読み込まなければなりません。

```
GET　#ファイル番号，レコード番号
```

これで、ファイル番号のファイルレコード番号のレコードがファイルバッファに読み込まれます。

レコードはＦＩＥＬＤ文で割り当てられた形式でデータが格納されています。データはすべて文字列として格納されていますので、数値データの場合には値を変換しなければなりません。

ＣＶＩ（文字変数）　　整数に変換されます
ＣＶＳ（文字変数）　　単精度実数に変換されます
ＣＶＤ（文字変数）　　倍精度実数に変換されます

ここで、文字変数はＦＩＥＬＤ文で割り当てられた変数でなければなりません。
以上でレコード内のデータを読み込むことができました。

以上を簡単な例を挙げて説明することにしましょう。
まず、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します。

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

ランダムアクセスファイルをオープンする場合、"Ｒ"モード(Random　file　mode)指定を行います。なお、ランダムアクセスファイルの場合、入力、出力で別のモード指定を行う必要はなく、１つのＯＰＥＮ文でデータの入出力をすることが可能です。行番号１０では、フロッピーディスクドライブ１に"ＲＤＡＴＡ"という名のランダムアクセス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%)
```

ランダムアクセスファイルに出力するデータは文字型のものだけで、数値型は使用できません。そのため、数式を文字列に変換したあとで出力する必要があります。数式を文字列に変換する命令は、行番号４０のＭＫＩ＄の他にＭＫＳ＄、ＭＫＤ＄があります。

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

ランダムアクセスファイルのデータの入出力を行う場合、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レコード分ずつ入出力を行います。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なければなりません。それを行うのが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。

```
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ＬＳＥＴを用いてデータを格納しています。

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

ＰＵＴ文によってファイルバッファの内容がランダムアクセス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。ランダムアクセスファイルにはレコード番号が記録されるようになっているので、このレコード番号の指定を変えることによって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アクセスファイルを閉じています。ランダムアクセスファイルの場合、入出力を１つのＯＰＥＮ文、ＦＩＥＬＤ文で行うことができます。

〔例〕このプログラム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 ↵
Ok
```

※入力する文字「パソコンテレビ」は全角文字で入力してください。

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します。

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

行番号１０でランダムアクセスファイルをデータ入力のためにオープンしています。ランダムアクセスファイルの場合、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、変数名は出力時と変えてもかまいませんが、ファイルバッファの割り付け方は同じにする必要があります。

```
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
```

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バイト読み出しています。

ファイルバッファに読み込まれたデータはＦＩＥＬＤ文で割りつ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、いったん文字式に変換した数値は、ＣＶＩ，ＣＶＳ，ＣＶＤを用いてもとの数値に変換します。

以上は行番号４０～６０で行われており、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。

```
RUN↵
名前はパソコンテです。
歳は ２ 才です。
Ok
```

※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。

プログラム（1）で名前の入力をする際に全角文字で「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〝パソコンテ〟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。

4.4.5　ランダムアクセスファイルへのデータの追加とデータの変更

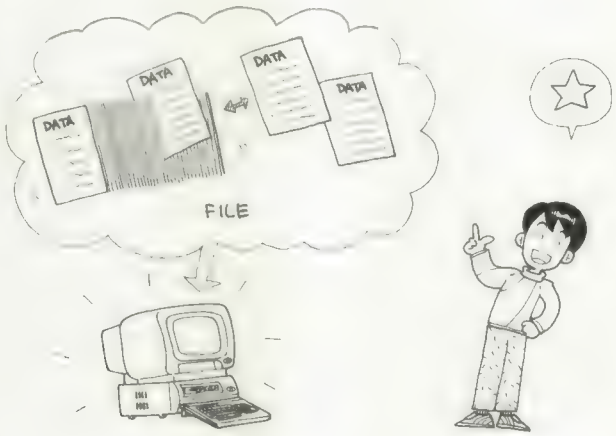
ランダムアクセスファイルはシーケンシャルアクセスファイルに比べて多くの手続きを必要としますが、その代わりデータの追加・変更を自由に行うことができます。すなわち、レコード番号を指定することにより、任意の場所のデータを任意の順序でアクセスすることができるのです。

ここでは、先ほど作成したランダムアクセスファイルにデータを追加・変更するプログラムを例にとって説明します。

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・変更プログラム

```
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って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・変更を繰り返すことができます。

〔例〕以下に実行例を示します。

```
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、次のプログラムを入力してください。このプログラムはプログラム（3）で作成したランダムアクセスファイルからデータを取り出すプログラムです。

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25でアクセスするレコード番号を入力し、行番号65によって繰り返しができ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
```

※ここでは、レコード番号として０を指定することによりプログラムが終了するようにしています。
　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）の実行によって「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。



第5章 フリーエリアについて

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64KBのメモリ空間を持っています。メモリはBIOS ROM、メインメモリ、グラフィックVRAMなどから構成されます。メインメモリにはBASICインタプリタが置かれ、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ます。BASICインタプリタが使用している領域を除いた残りの部分をフリーエリアといいます。

5.1 グラフィックVRAMとフリーエリア

プログラムやデータはフリーエリアの領域に格納されるわけですが、こ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。ところがメインメモリ64KBという制限の中で、フリーエリアを増やすには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。

グラフィックVRAMは、48KBのものが2枚ありそれぞれをバンク0、バンク1といいます。これらはそれぞれグラフィック描画、変数領域、外部記憶として使用することが可能です。

フリーエリアはプログラムの実行前や実行後や、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。

5.1.1 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エリアなどがとられるだけです。また、グラフィックVRAM1は変数領域として使用されるモードとなりますので、図1のようになります。

5.1.2 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、およびグラフィックVRAMからなり、図2のようになります。

5.1.3 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領域がとられます。

グラフィックVRAMを変数領域として使用する場合

またVDIM命令を使っていれば、グラフィックVRAMにも変数領域がとられます。

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の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。

図1　ＢＡＳＩＣ起動直後

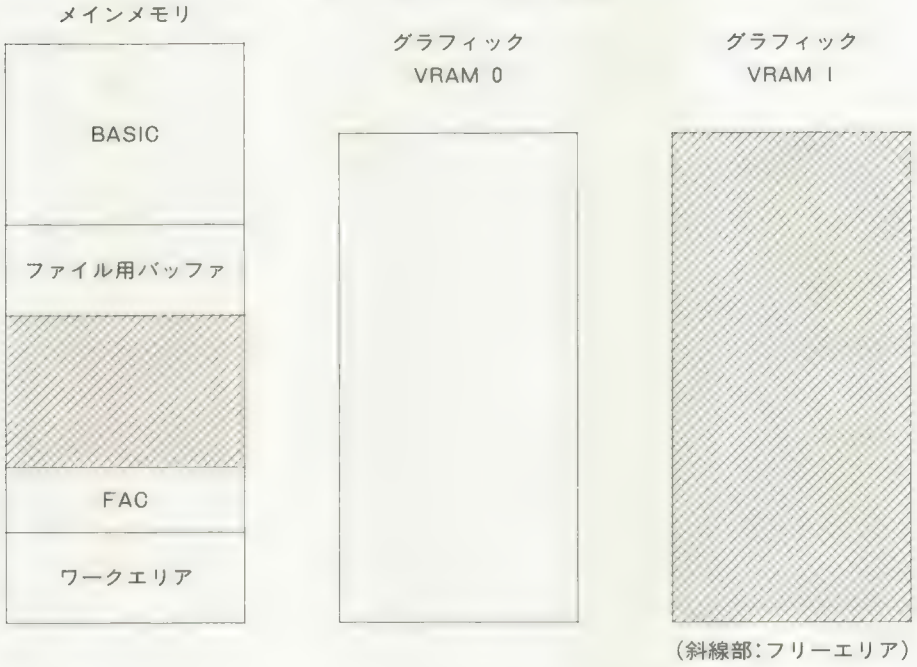


図2　プログラムロード後（実行前）

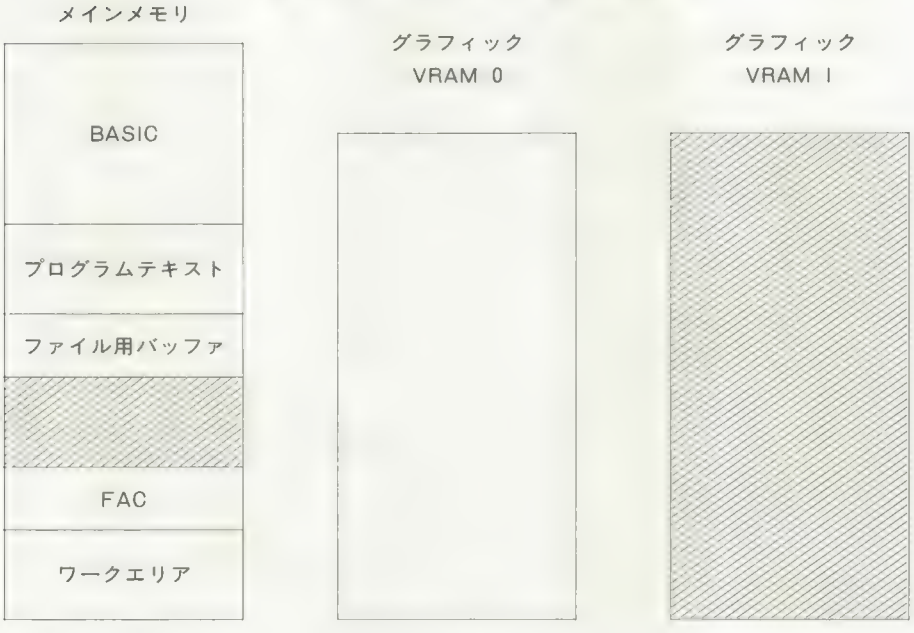
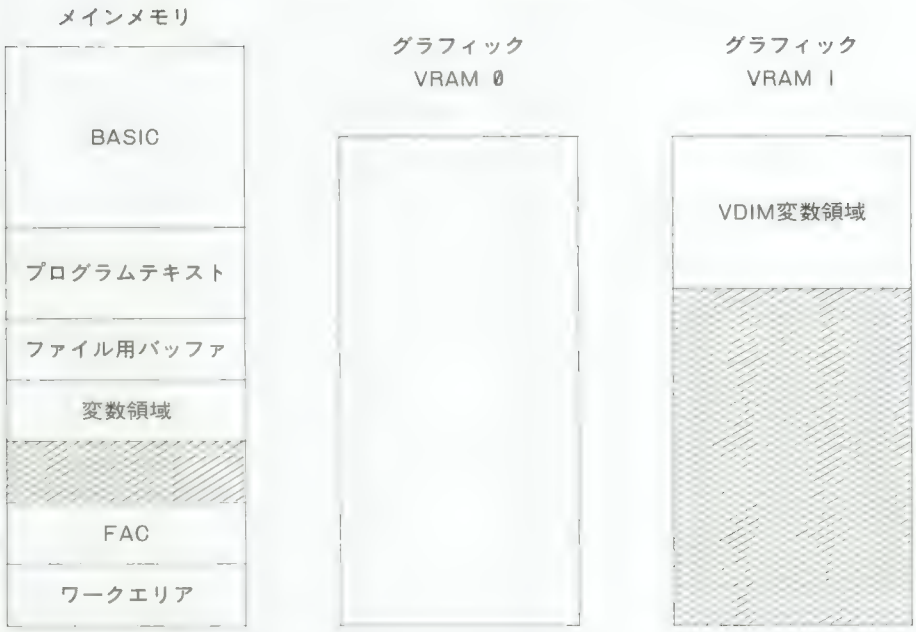


図3　プログラム実行後



・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。

グラフィックVRAMを変数領域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ができません。また、外部記憶装置として使用する場合もやはり変数を格納することはできません。

これらの場合、メインメモリの残りの部分のみがフリーエリアとなります。

5.2　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 n          (n=0～4)
```

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 0～4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することができるのは、OPTION SCREEN 1とOPTION SCREEN 2のみで、それ以外ではグラフィック描画用、または外部記憶として使用されます。

(1) OPTION 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。
そのため、グラフィックとして次の画面モードを使用することができます。

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。
さらに、640×400、320×400、640×384、320×384の各画面モードにおいては次の命令を実行することができません。

```
OPTION SCREEN 1
OPTION SCREEN 2
```

また、メインメモリ上にしか変数領域、プログラム領域を確保することができません。

(2) OPTION SCREEN 1

グラフィックVRAMのバンク0を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。そのため、画面モードとしては次のモードを使用することができます。

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

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VRAMバンク1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を合わせた部分となります。

(3) OPTION SCREEN 2

グラフィックVRAMのバンク0を外部記憶として使用し、バンク1を変数領域として使用します。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を合わ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。

(4) OPTION SCREEN 3

グラフィックVRAMのバンク0を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。

(5) OPTION 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。

5.3 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することができますが、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ができます。

また、48KB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といった使い方も可能となります。

(1) 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。

(2) 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。

(3) 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には注意が必要です。たとえばCLS命令などを用いるとファイルが壊されてしまいます。

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16KBにファイルを格納し、残り32KB（赤と緑）をグラフィックに使うということも可能です。

5.4 日本語入力とフリーエリア

ディスク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ディスクから読み込んで変換を行いますので、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、ワークエリアとして使用するため、あらかじめ、

```
CLEAR &HF000
```

を実行しておかなければなりません。

しかし音訓変換を使用しないのであれば、

```
CLEAR &HF400
```

として、フリーエリアを1Kバイト増加させることができます。

5.5　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ように、非常に多くの命令を含んでいます。そのため、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用のフリーエリアとして使用することができれば、より大きなプログラムやデータを扱うことができます。

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たいという場合に使うのが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。

NEWONn　　（n＝0～9）

ＮＥＷＯＮで削除される命令群には１０のレベルがあり、０～９の番号で指定します。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と数字（０～９）との間にスペースを入れないでください。

なお、ＢＡＳＩＣ起動時、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。

また一度ＮＥＷＯＮ（ｎ＝０～９）を実行したあと、削除されたＢＡＳＩＣのプログラムを実行しようとしても

Reserved　feature　　または　Error　34

というエラーメッセージが表示されて実行できません。

<参考>

① ｜HELP｜を押しながら、ディスクＢＡＳＩＣを起動すると、自動的にＮＥＷＯＮ４が設定されます。

```
SHARP HuBASIC  CZ-8FB02 Version1.0 [2HD]
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P
 80030 Bytes free

Ok
```

② ＢＡＳＩＣを起動後、希望するＮＥＷＯＮを自動的に設定し、すぐにプログラム入力ができる

ようにするには、" Start　up. Bas " のプログラムを次のように変更します（行番号５７０，６５０を変更）。

```
570 IF INKEY$(0)=CHR$(&HE2) THEN 650
```

↓

```
570 IF INKEY$(0)><CHR$(&HE2) THEN 650
```

```
650 COLOR0:KEY0,"NEWON4"+CHR$(13)+"COL.7 ・・・・・・・・・
```

↓

０～９の数値を入力するか、または数値を省略して、
希望するNEWONのレベルを設定する。

③ 変更したプログラムをシステムディスクに

SAVE "０：Start　up. Bas "

と入力してセーブします。

n	削除されるコマンド、ステートメント
省　略	すべてのコマンド、ステートメント、関数などを使用可
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, SAVEM, エラーメッセージ
1	PLAY, PLAY@, MUSIC, MUSIC@, TEMPO, SOUND, SOUND@, CBLACK
0	WINDOW, LINE, PSET, PRESET, CIRCLE, POLY, PAINT, PAINT@, SYMBOL, POINT, PRW, CLICK, NEWONn, PALET, PALET@

（例）NEWON（省略）に設定するとBASIC起動後は、次のような画面になります。

```
SHARP HuBASIC  CZ-8FB02 Version1.0 [2HD]
Copyright (C) 1984 by  SHARP/Hudson
 Printer : CZ-800P
 74495 Bytes free

Ok
```

5.6　VDIMとフリーエリア

DIM命令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でき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 1もしくは2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。このモードで、グラフィックVRAMに配列変数（もしくは通常の変数）領域を確保するのがVDIM命令です。

VDIM命令は

```
VDIM 配列名（大きさ1, 大きさ2……）
```

のように書きます。

VDIM命令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命令で宣言された配列変数と同様の使いかたをします。

ただし、VDIM命令はCLEAR命令では消去されません。その場合にはVDIM CLEARという命令を用います。

また、ERASE命令は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。

［配列を使用したサンプルプログラム］

```
10 WIDTH 40,25,0
20 INIT:CLS
30 OPTION SCREEN1
100 DIM A$(255,1)   ⇐配列データをメインメモリ上ではなくグラフィックRAM上にとる場合は、
110 FOR I=0 TO 255                  100の行のDIM A$(255, 1) をVDIM A$(255, 1)
115 CLS:LOCATE 0,0:PRINT HEX$(I)  に変更します。
120 FOR J=0 TO 1
130 A$(I,J)=CGPAT$((J+1)*256+I)
150 PRINT
160 PRINT#0,A$(I,J):PAUSE 5
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$という変数を宣言しています。１３０行でキャラクタジェネレータのデータを読み出し配列の中に入れています。配列Ａ$(Ｉ，Ｊ)のうち、Ｊ＝０のとき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回繰り返しています。

配列のデータをメインメモリ上ではなくグラフィックＶＲＡＭに取りたい場合は、１００行のＤＩＭ　Ａ$(２５５，１)を　ＶＤＩＭ　Ａ$(２５５，１)に変更してください。

※配列変数については『ＢＡＳＩＣ　ＲＥＦＥＲＥＮＣＥ　ＭＡＮＵＡＬ』2.2.8~2.2.12を参照してください。

5.7　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）	ｎ＝１のときメインメモリのフリーエリア ｎ＝２のときグラフィックＶＲＡＭのフリーエリア ｎ＝０のときそれらの合計

という書式で書くことができます。

第6章 機械語サブルーチンとモニタ

6.1 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命令をダイレクトに入力してください。

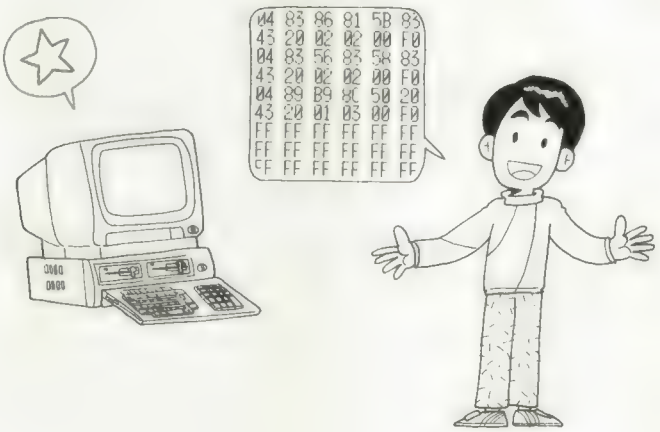
```
⋮
Ok
MON↵
*
```

MON命令を入力すると、上のように＊印が表示され、機械語モニタが起動します。
また、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、次の画面となります。
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〝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桁以内。０～ＦＦＦＦ。

YYYY：最終アドレス。１６進数４桁以内。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。

例として、４０字モード時の表示について説明します。

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4桁。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個または16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。

また、80文字モード時には、1行16バイト分のデータが表示され、YYYYを省略すると256バイト分のデータが一度に表示されます。

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、再開はスペースキーかを押します。また、コマンド待ちに戻るにはSHIFT＋BREAK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。

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、メーセージ中にアポストロフィ（'）を含めることができません。

メッセージは行の先頭から入力してください。

（例）：8000="CAN'T　CONTINUE"
　　：9000='LOADING"BASIC"'

(5) 空白：データとデータの間は、原則として、空白で区切ります。

書き換えた後、その行でを押すと、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桁以内。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=HH
       ↑
      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、↵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=HH
       ↑
      カーソル
```

この状態から抜け出すには、SHIFT＋BREAK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桁以内。0～FFFF。

YYYY：最終アドレス。16進数4桁以内。0～FFFF。

HH：データ。16進数2桁以内。0～FF。

説明：XXXXからYYYYまでのメインメモリから、HH HH ……で指定されたひと続きのデータ群を探し出し、み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）から探し出して、10C3番地にそれがみ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したいときは、SHIFT＋BREAK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、切り替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。
プリンタが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の入力待ちになります。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桁以内。0～FFFF。

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桁以内。0～FFFF。
YYYY：データの最終アドレス。16進数4桁以内。0～FFFF。
ZZZZ：データの転送先のアドレス。16進数4桁以内。0～FFFF。

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桁以内。0～FFFF。
YYYY：最終アドレス。16進数4桁以内。0～FFFF。
ZZZZ：実行開始アドレス。16進数4桁以内。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。
ファイル名は、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桁以内。0～FFFF。

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]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。
このコマンドは機械語プログラムを読み込むだけで、実行は行いません。

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を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が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コマンドで記録し直してください。

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 命令に移ります。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替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コマンドによって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］PALETの初期化をします。

PALET0，0：……：PALET7，7 ↵

または

PALET@0，1，2，3，4，5，6，7 ↵

と同じ。

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。（=KMODE 0 ↵）
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。（=KMODE 1 ↵）

dの値	使用モニター
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

〔例〕KMODE 1、標準ディスプレイ、テキスト画面10行に設定します。

*#113 ↵（パラメータ間にスペースを入れてはいけません。）

［3］グラフィック画面の消去をします。
＝（ＣＬＳ０↵）
［4］ｋ，ｄ，ｍの値によって 前ページのような設定ができます。

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……３または５インチディスク（ｄ＝０～３）
Ｆ……８インチディスク（ｄ＝０～３）
Ｈ……５インチ10ＭＢ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桁の１６進数。
ｒｒ：書き込むレコード長。※１～ＦＦの１６進数。
ａａａａ：メモリの先頭アドレス。０～ＦＦＦＦの１６進数。
※レコード長：１単位が１セクタ。

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。

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。

（例）３または５インチのドライブ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 ↵
```

３または５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。

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
3または5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
3または5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インチ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。）

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……3または5インチディスク（d＝0～3）
F……8インチディスク（d＝0～3）
H……5インチ10MB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桁の16進数。
rr：書き込むレコード長。※1～FFの16進数。
aaaa：メモリの先頭アドレス。0～FFFFの16進数。

※レコード長：1単位が1セクタ。

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。

6.2 機械語サブルーチンの入力

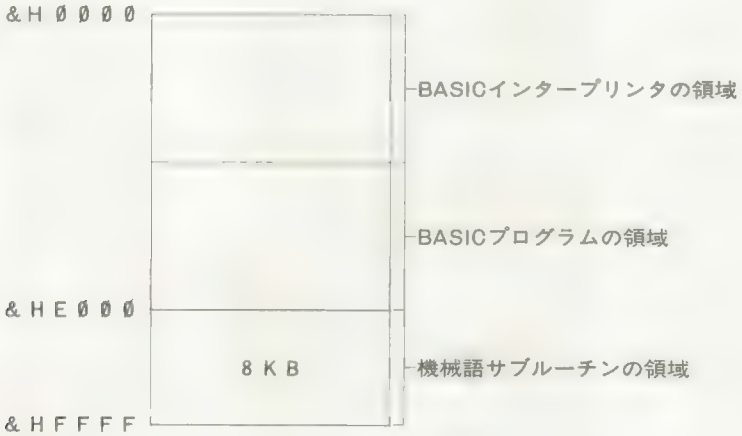
6.2.1 機械語サブルーチンの記憶領域の設定

機械語サブルーチンを入力するためには、まずCLEAR命令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（最上位の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
```

をプログラムの先頭に置くか、そのままダイレクトに実行します。



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３ＫＢ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ているので、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。

6.2.2　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通りがあります。

（１）ＰＯＫＥ命令を使う。

（２）ＭＯＮ命令で機械語モニタを起動する。

（１）ＰＯＫＥ命令による入力

ＰＯＫＥコマンドを使って、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文で逐次読み込んで、ＰＯＫＥ命令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使います。

次に簡単な例を示します。

```
100 'ＰＯＫＥ命令による入力
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命令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。

（2）機械語モニタによる入力
機械語モニタを用いて、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。

6.2.3 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命令を使う方法とUSR関数を使う方法の2通りがあります。

CALL命令

機械語サブルーチンは、BASICのCALL命令を使って呼び出すことができます。
CALL命令の形式は、次のとおりです。

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、もとに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命令を実行すると、プログラムカウンタの内容をスタックに退避し、CALL命令中に指示されたアドレスa、すなわちサブルーチンの入り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にくると、スタックに退避しておいたプログラムカウンタを元に戻し、元のBASICプログラムの実行を再開します。
CALL命令では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が、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。

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命令を使って次のように書きます。

```
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
200 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が指定されたものとみなされます。

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、再度定義し直すまで、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。

```
[1] y=USRn(x)
[2] y$=USRn(x$)
```

[1] 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$：文字変数。
n：USR関数識別番号。0～9の整数。
x$：文字式（引数）。

（例）
```
200 A=USR0(65)
210 A$=USR1(C$)
```

USR関数の後ろに付ける番号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$をもって、n番の機械語サブルーチンを呼び出し、リターン時にその値をyやy$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や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$は文字列を値に持つ文字式です。

USR関数で機械語サブルーチンを呼び出すとき、CPUの各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

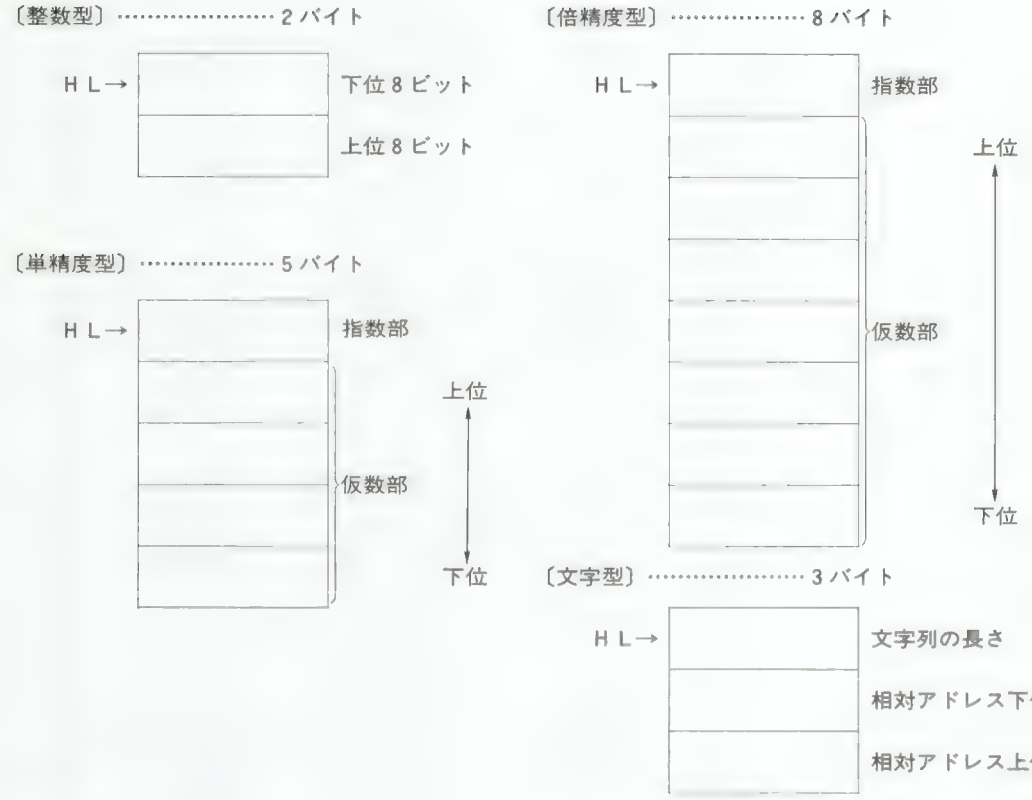
これらの値は、文字変数を除き、メモリ内において何バイトで表現されているかを表わしています。

●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。



●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持ちます。

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かたち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にエラーの発生を知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。

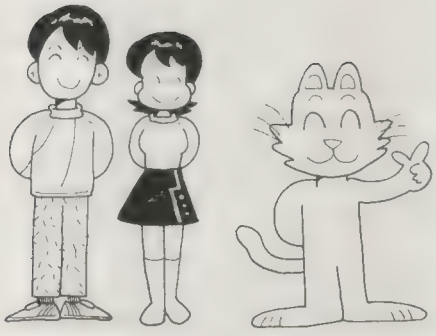
その方法は、エラー番号をAレジスタに入れ、IXレジスタの示すアドレスにジャンプすること

により行なうことができます。

（例）機械語サブルーチンの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。
            JP（ＩＸ）　　ＩＸレジスタの示すアドレスへジャンプします。
```

第3部 応用編



第1章 ミュージック

本機には、プログラムサウンドジェネレータ（PSG）という、疑似音声発生用LSIが内蔵されています。PSGを使うことにより、音楽演奏をしたりゲームの効果音を作ったりすることができます。ここでは、PSGをコントロールするサウンド制御コマンドの使い方を説明します。

1.1 ブザー音を出す

ブザー音を出すためには、BEEP命令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。 }

ただBEEPとすると、1回だけ「ポッ」と鳴ります。BEEP 1とするとブザー音が鳴りっぱなしになります。これはBEEP 0を実行するまでは止まりません。これらを組み合わせれば任意の時間だけブザーを鳴らしておくことができます。

1.2 楽譜を演奏する

MUSIC，PLAY，TEMPOの命令を使用して、楽譜の演奏（8オクターブ、3重和音）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うことができます。

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

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。音の長さは４分音符を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。

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（符点１６分音符）	２
1/2（８分音符）	３
3/4（符点８分音符）	４
１（４分音符）	５
1 1/2（符点４分音符）	６
２（２分音符）	７
３（符点２分音符）	８
４（全音符）	９

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。

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〝＋〟を付けることによって１オクターブ高い音を、そして、〝－〟を付けることによって１オクターブ下の音が指定できます。初期状態は〝Ｏ４〟です。

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。

和音

２重和音や３重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）をはさんで、〝チャンネルＡ：チャンネルＢ：チャンネルＣ〟のようにします。

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。

テンポは、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ます。PLAYの後ろに数式を置いた場合はTEMPO文と同様に、文字式を置いた場合はMUSIC文と同様になります。

（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、これを繰り返します。

```
10 'MUSIC or PLAY
20 WIDTH 40:INIT
30 CLS4
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:DEFGABAGFEDCB:EFGABAGFEDCBA"
90 NEXT O
100 NEXT T
```

MUSIC@，PLAY@命令は音の長さをCTCでコントロールしますので、音を出しながらグラフィック命令等を実行することができます。直接実行することはできません。

1.3　PSGのコントロール

本機で音を出すには、先に述べたPLAY命令などによって楽譜を文字列で表わ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ことができます。

なお、SOUND命令は、1つのデータを1つのレジスタ（8ビット）に書き込みますが、SOUND@命令は、1つのデータを連続したレジスタ（16ビット）に書き込むことができます。後で説明する周波数の設定の場合のように、連続したレジスタにデータを書き込む必要がある場合にSOUND@命令を使用します。

また、データをカンマ(,)で区切って並べて書くと、複数の連続したレジスタにデータを順番に書き込んで行くことができます。たとえば

SOUND 4，28，1

とすると、レジスタ4にデータ28が、レジスタ5にデータ1が書き込まれます。

PSGは、3つのトーンジェネレータ、ノイズジェネレータ、エンベロープジェネレータ、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。（次ページ図）

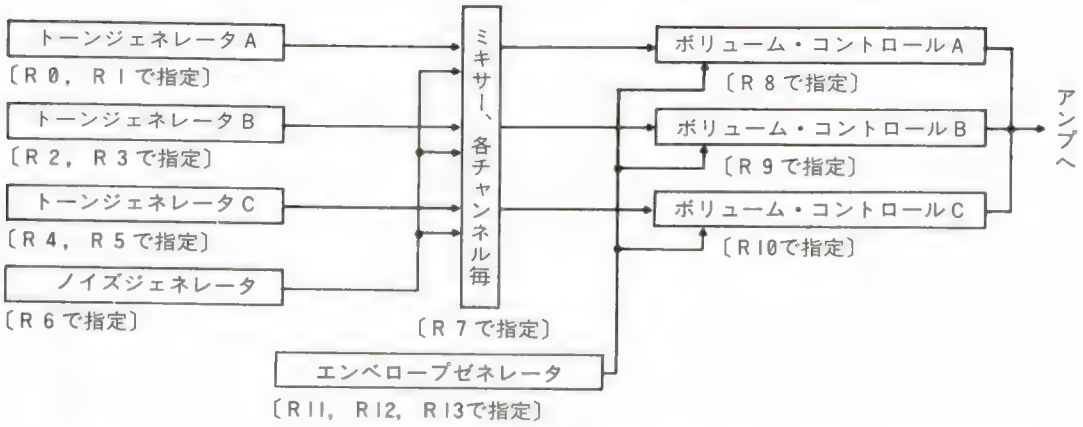
レジスタにデータを送ることによって様々な音が発生します。各レジスタの機能表を以下に示します。

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）番号
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 C ノイズ 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）
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		下位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

各チャンネルとも
M=0のとき（データ値0～15）
4ビットの値で0から15段階に音量設定可能
M=1のとき（データ値16～31）
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。

※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。



（注）図中のR0～R13はレジスタ番号です

トーンジェネレータ

レジスタ0～5は、A，B，C各チャンネルの周波数を決定します。周波数は音程にあたり、1オクターブ音が上が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。

次の表は、オクターブ4の各音の周波数を表しています。

音　階	C	#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	440.00

音　階	#A	B
周波数（Hz）	466.16	493.88

※音階はMUSICの文字列で表しています。

PSGはA，B，Cの3個のトーンジェネレータを内蔵していて、そのトーンの周波数は粗調整レジスタ（R1，R3，R5）の各4ビットと、微調整レジスタ（R0，R2，R4）各8ビットの計12ビットによって設定されます。

PSGには、2MHzのクロックが与えられており、この周波数は内部で1　16に分周（周波数を分割する操作）されています。この周波数がトーンジェネレータによって分周されて目的の周波

数になります。

目的の周波数をｆ〔ＨＺ〕、粗調整レジスタにセットするデータT_H　、微調整レジスタにセットするデータT_L　との関係は、次式のとおりです。

$$f=\frac{2\times10^6}{16\times(T_H\times256+T_L)}$$

この式を変形して、

$$T_H\times256+T_L=\frac{2\times10^6}{16\times f}$$

目的の周波数ｆからT_HおよびT_Lを求めるには、ＢＡＳＩＣの関数を利用して次式のように計算できます。

```
TH＝INT（（2000000！／16＊ f）／256）
TL　＝FRAC（（2000000！／16＊ f）／256）＊256
```

たとえば、チャンネルＡから４４０Ｈｚの音を発生させるには、

```
TH　＝INT（（2000000！／16＊440）／256）　　＝1
TL　＝FRAC（（2000000！／16＊440）／256）＊256　＝28.09
```

ですから、ＳＯＵＮＤ命令を使用して設定すると次のようになります。

```
SOUND　0，1，28
```

ＳＯＵＮＤ@命令を使用すると、レジスタ０とレジスタ１に同時にチャンネルＡの周波数を設定することができます。このとき設定する値Ｔは、

$$T=\frac{2\times10^6}{16\times f}$$

となります。この値は、$T_H\times256+T_L$と同じです。

ＳＯＵＮＤ@コマンドを用いて440Ｈｚの音を出すには、

$$T=\frac{2\times10^6}{16\times440}=284$$

ですから、

```
SOUND@　0，284
```

となります。

ノイズジェネレータ

ノイズとは周波数が不規則に変化するものをいいます。レジスタ6に設定するデータによってノイズの平均周波数が決まります。周波数を決める式は、先ほどの式をそのまま適用することができます。

たとえば、5 KHzのノイズを発生するには次の式を計算します。

$$T=\frac{2\times10^{6}}{16\times5000}\ 25$$

設定するレジスタは1つでよいため、SOUND命令を用います。

```
SOUND 6,25
```

ミキサー

レジスタ7は、A，B，C各チャンネルの音を同時に出力するかどうかを選択します。なおトーンとノイズは別々に選択します。たとえば、

```
SOUND 7,44  または  SOUND 7,&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。

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（ビット4）の値によりエンベロープを有効にするかどうかを選択します。M＝0とすると、4ビットデータにより0～15までの一定した音量が設定できます。M＝1とするとエンベロープがかかり、自動的に音量が0～15まで変化します。

たとえば、

```
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。

エンベロープジェネレータ

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。

周波数を決定するには先ほどの式が適用できます。たとえば、0.5 Hz、（エンベロープ周期2秒）を設定するには、次の式を計算します。

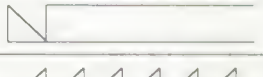
$$T=\frac{2\times10^{6}}{256\times0.5}=15625$$

この分周値を次のように指定することによって、0.5Hz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に設定します。

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/fε=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しょう。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 SOUND 6,25            ……ノイズ周波数設定
30 SOUND@ 11,15625       ……エンベロープ周期の設定
40 SOUND 13,8            ……エンベロープパターンの設定
50 SOUND 8,15            ……Aチャンネルの音量設定
60 SOUND 7,&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。音を止めるには CTRL + D を押してください。60行目を次のように変更してください。

60 SOUND7,&B110110……Aチャンネルのノイズとトーンを選択

これで、Aチャンネルからは音とノイズの両方が出力されます。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
```

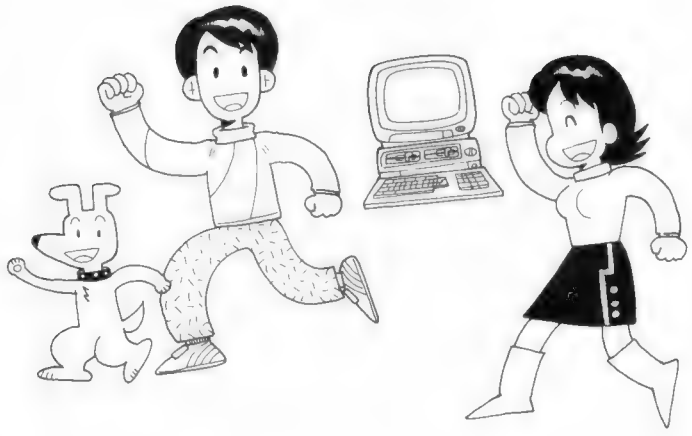
Aチャンネルの音量にエンベロープがかかり、おもしろい音になります。

また、次のようにSOUND命令でエンベロープを指定することによって、ＰＬＡＹ（またはMUSＩC）命令によるスタッカートをかけることができます。

```
PLAY１２０：PLAY"V１６O４C５CDDEED"
```

上のＰＬＡＹ命令を、次のように変えてください。ただし、文字式での音量指定は〝Ｖ１６〟にします。

```
１０　SOUND@１１，５２４５：SOUND１３，０：SOUND８，１６
２０　PLAY１２０：PLAY"V１６O４C５CDDEED"
```



第2章　ディスクの使い方

本機では、5インチフロッピーディスク（3インチフロッピーディスク）の他に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、これらのフロッピー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「第2部　第4章　ファイルについて」を参照してください。

2.1　ディスクの使用準備

新しいフロッピーディスクを使用する場合、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っておく必要があります。

フォーマッティングとは新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のディスクユーティリティの中に、フォーマットのためのプログラムが用意されています。これを使用してください。

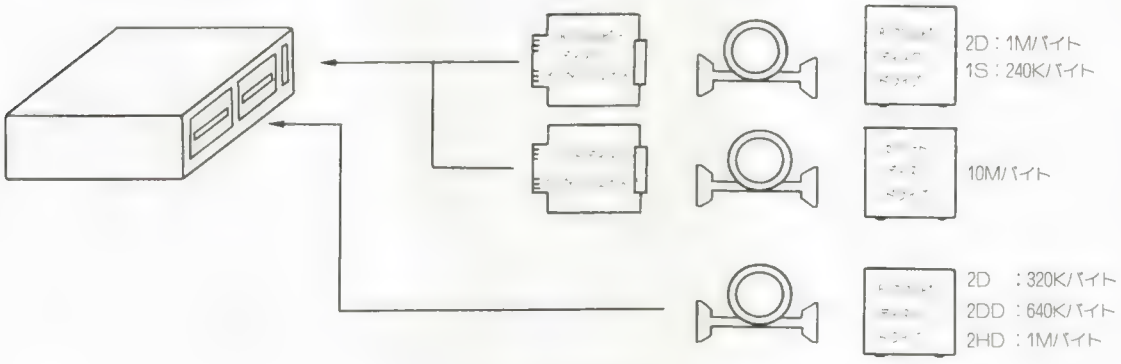
（別冊『アプリケーションソフトの説明書』「ディスクユーティリティ」参照）

2.2　ディスクドライブ接続構成

本機は、下図のように、3インチまたは5インチフロッピーディスクドライブ、8インチフロッピーディスクドライブ、および5インチハードディスクを接続することが可能です。

本機は、5インチフロッピーディスクドライブおよびインターフェイスを内蔵していますので、3インチまたは5インチフロッピーディスクドライブは直接増設可能です。

また、8インチフロッピーディスクドライブ、ハードディスク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。



ドライブ接続構成図

〈ドライブ接続台数〉

接続ドライブ		台　数	備　　考
3インチまたは5インチフロッピーディスクドライブ	内　蔵	2	最大接続台数はトータルで4台
	内部増設	—	
	外部増設	2	
8インチフロッピーディスクドライブ		4	専用インターフェイス使用
5インチハードディスクドライブ		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4 8Mバイトです。）

本機には外部5インチ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。

2.3　システムプログラムの起動方法

本機では、3 インチまたは5インチフロッピーディスクドライブをトータル4ドライブ、8インチ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）が一度に接続可能であり、8種類のディスクタイプ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

⑴起動ディスクタイプスイッチ（前面トビラ内）によるシステムディスクのタイプ設定
⑵ディスクモードスイッチ（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
⑶キー入力により、システムディスクのタイプ設定

の3種類が可能です。なお、本機内蔵のディスクドライブのみで使用する場合は、『⑵ディスクモードスイッチ』を設定し直す必要はありません。

ディスクモードスイッチは工場出荷時の初期設定のままで、『⑴起動ディスクタイプスイッチ』をディスクタイプに合わせ設定してください。

⑴起動ディスクタイプスイッチによるシステムディスクのタイプ設定

起動ディスクタイプスイッチは内蔵のフロッピーディスクに対し、ディスクタイプを2HD（×1フォーマット）と設定するか、2D（×1フォーマット）と設定するか選択するためのスイッチです。

この方法では、起動ディスクタイプスイッチにより起動するシステムディスクタイプを設定し、

システムの電源を投入することによって内蔵フロッピー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。

本スイッチは後述のディスクモードスイッチのＳＷ５がＯＮ時に有効となります。

⑵ディスクモードスイッチによるシステムディスクのタイプ設定

このスチッチは、内蔵ディスクドライブだけでなく、周辺機器を含めたディスクタイプの設定を行う時に使用します。

後面ディスクモードスイッチのうち、ＳＷ１、ＳＷ２、ＳＷ３を初期モードスイッチと呼びこの初期モードスイッチはディスクタイプ選択スイッチのＳＷ５がＯＦＦの時、有効となります。初期モードスイッチにより起動するシステムディスクタイプ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インチまたは５インチおよび８インチフロッピーディスクドライブ、ハードディスクドライブ）のドライブのから自動的にシステムプログラムが起動できます。

ディスクモードスイッチの詳細は、ディスクモードスイッチの設定を参照してください。

以上⑴、⑵の方法でシステムプログラムを起動したサポートドライブ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起動ディスクタイプスイッチまたは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。

⑶キー入力でシステムプログラムを起動する場合

起動ディスクタイプスイッチおよびディスク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

```
      Make your device ready
Press selected key to start driving:
          F:Floppy
          R:ROM                          ……(1)
          C:CMT
          T:Timer
```

という表示が画面に出ますので、キーボードから F 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  320kbytes   2D          ……0
or
3"FD  640kbytes   2DD         ……1
      1M  bytes   2HD         ……2
      1M  bytes  *2HD         ……3
                                ……(3)
8"FD  1M  bytes   2D—256      ……4
      1M  bytes  *2D—256      ……5
      240kbytes *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。表示画面は

```
IPL is looking for a program from FD[ NO ]……(4)
                                (NOは0～3のうちどれか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 CZ8FB02          ……(5)
```

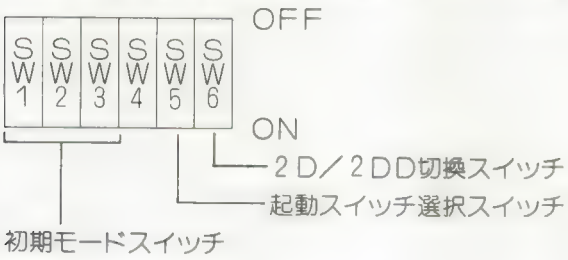
表示となり、BASICの読み出しを開始します。

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(1)の表示になるので、入出力装置の状態を確認の上、再操作してください。また、(2)または(3)の画面の時 SHIFT ＋ BREAK キーを押しますと、(1)の表示画面に戻すことができます。入力ミスをした場合などに利用してください。

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インチまたは5インチドライブの場合ディスクタイプ2、8インチドライブの場合はディスクタイプ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。

2.4 ディスクモードスイッチの設定

本機の後面にあるディスクモードスイッチの構成は次のようになっています。



(1)初期モードスイッチ（SW1、2、3）

SW1、SW2、SW3の組み合せによりシステムを起動するディスクタイプナンバーを選びます。

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			ディスクタイプナンバー	内　　容	記録方式	記録内容	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インチ または 5インチFD
OFF	ON	ON	1	2DD　X1 フォーマット	両面倍密度 倍トラック記録	640K	
ON	OFF	ON	2	2HD　X1 フォーマット	両　面 高密度記録	1M	
OFF	OFF	ON	3	*2HD　8インチ 標準フォーマット	両　面 高密度記録	1M	
ON	ON	OFF	4	2D　X1 フォーマット	両　面 倍密度記録	1M	8インチFD
OFF	ON	OFF	5	*2D　8インチ 標準フォーマット	両　面 倍密度記録	1M	
ON	OFF	OFF	6	1S　8インチ 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	10M	ハードディスク

※内蔵ディスクドライブを2DDモードで使用する場合はSW6をOFF（2DD）にする必要があります。

⑵起動スイッチ選択スイッチ（SW5）

起動ディスクタイプスイッチを有効とするか初期モードスイッチを有効とするかを選択するスイッチです。

ON　…前面トビラ内の起動ディスクタイプスイッチの設定を有効とします。

OFF…後面のディスクモードスイッチ内の初期モードスイッチの設定を有効とします。

⑶2D／2DD切換スイッチ（SW6）

内蔵のディスクドライブを2DDとして使用する場合のみ、OFF（2DD）に切り換えます。

まず、起動ディスクタイプスイッチで2Dモードにし、このスイッチ（SW6）をOFF（2DD）にしてください。

なお、2DDとして使用しない時はかならず、ON（2D）にしておいてください。

出荷時の設定は次のようになっています。



SW1	SW2	SW3	SW4	SW5	SW6
ON	OFF	ON	ON	ON	ON

SW1～SW3　初期モードスイッチ…2HD×1フォーマット設定
SW5　　起動スイッチ選択スイッチ…起動ディスクタイプスイッチを有効とする
SW6　　2D／2DD切り換えスイッチ…2Dタイプ設定

〈注意〉

ディスクモードスイッチの切り換えはシステムの起動前に行なってください。
起動後の設定変更はシステムの動作に悪影響をおよぼすことがあります。

2.5　接続ドライブのディスクタイプ設定

本機にはディスクドライブを最大12ドライブまで接続できます。

・5（3）インチ（2D，2DD，2HD）……4ドライブ　　0：～3：
・8インチ（2D，1S）　　……4ドライブ　F0：～F3：
・ハードディスク　　……4ドライブ　HD0：～HD3：

3インチ，5インチおよび8インチフロッピーディスクドライブに、ど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もの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あります。すなわちディスクタイプによってヘッド数、トラック数、セクタ数が異なり、またFM記録／MFM記録、3インチまたは5インチ（2D、2DD）、5インチ（2HD）、8インチ（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は

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

の3つの方法があります。

①については、『アプリケーションソフトの説明書』を参照してください。
②は、BASICのDEVICE命令で、

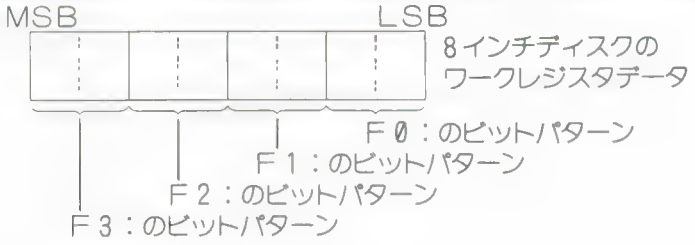
DEVICE　"デバイス名：ディスクタイプ番号"

とすることによって設定できます。ディスクタイプ番号については次表を参照してください。

	デバイス名	ディスクタイプNo.
3インチ または 5インチ FD	0： 〜 3：	0… 2D 1… 2DD 2… 2HD 3… 2HD
8インチ FD	F0： 〜 F3：	0… 2D 1… 2D 2… 1S

③はモニタまたはBASICから、ディスクの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、各デバイスについて5インチは1バイト、8インチは2ビットごとのビットパターン1バイトからなっています。

	デバイス名	ワークレジスタアドレス	データ
5インチ 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インチ標準フォーマット
8インチ FD	F0：	FAB8	00：2D X1フォーマット
	F1：		01：2D 8インチ標準フォーマット
	F2：		10：1S 8インチ標準フォーマット
	F3：		（ビットパターン）



たとえば、接続されているディスクが

5（3）インチ　0：　2HD　X1フォーマット
　　1：　2HD　X1フォーマット
　　2：　2D
　　3：　2D
8インチ　　F0：　2D　X1フォーマット
　　F1：　2D　X1フォーマット
　　F2：　2D　8インチ標準フォーマット
　　F3：　1S　8インチ標準フォーマット

の場合、ディスクタイプのワークレジスタは

FAB4：02
FAB5：02
FAB6：00
FAB7：00
FAB8：90

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）

●5（3）インチディスクで起動したとき

デバイス名	デフォルトタイプ
0：	BOOTしたディスクタイプ（Start upで2HDX1フォーマットに設定）
1：～3：	2HD×1フォーマット（※）
F0：～F3：	2D　X1フォーマット

●8インチディスクで起動したとき

デバイス名	デフォルトタイプ
0：～3	2HDX1フォーマット（※）
F0：	BOOTしたディスクタイプ
F1：～F3：	2D　X1フォーマット

●ハードディスクで起動したとき

デバイス名	デフォルトタイプ
0：～3：	2HDX1フォーマット（※）
F0：～F3：	2D　X1フォーマット

（※）Start upで設定されます。Start upで設定しないときは2Dとなります。

またキー入力によって起動した場合は、起動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5（3）インチ、8インチフロッピーディスクは

5（3）インチ……2HD（X1フォーマット）

8インチ　　……2D　X1フォーマット

に設定されます。

2.6　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や読み出しを行う場合には、ディスク上のアドレス情報（ヘッド番号，トラック番号，セクタ番号等）をもとにして、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つける必要があります。このアドレスの割り付けのことをフォーマット（1次フォーマット）といいます。

市販の3インチまたは5インチのディスク（2D，2DD，2HD）はフォーマットされていませんので、必ずフォーマットしなくてはなりません。一方、市販の8インチのディスクは8インチ標準フォーマットがかけられており、必ずしもフォーマットが必要ではありませんが、全セクタとも256バイト／セクタ、倍密度記録のX1フォーマット[注1)]にフォーマット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は次表のとおりです。

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インチまたは5インチ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インチFD 標準フォーマット
8インチ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インチFD 標準フォーマット
	1S	0	0～76	1～26	128[注3)]	8インチ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フォーマットは、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インチ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

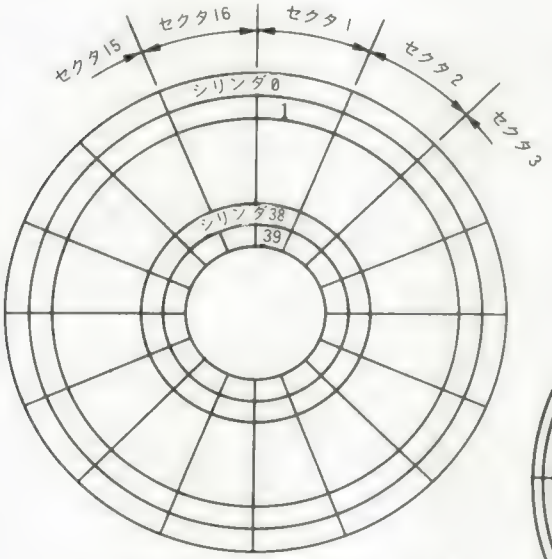
また、ディスクの管理を行うためには、ディスク上に、ある特別な領域、すなわちＦＡＴ領域およびディレクトリ領域を設け、システムに合わせてＦＡＴ領域とディレクトリ領域に初期データを書き込む必要があります。これをシステムフォーマット（２次フォーマット）といいます。本機ではディスク管理のため、どんなディスクタイプであっても必ずシステムフォーマットをする必要があります。

実際にディスクをフォーマットする場合には、アプリケーションソフト中のフォーマットプログラムを用いてください。このフォーマットプログラムでは、１次フォーマットと２次フォーマットの両方を行いますが、８インチ２Ｄ，１Ｓの市販ディスクは２次フォーマット（システムフォーマット）のみで使用可能です。

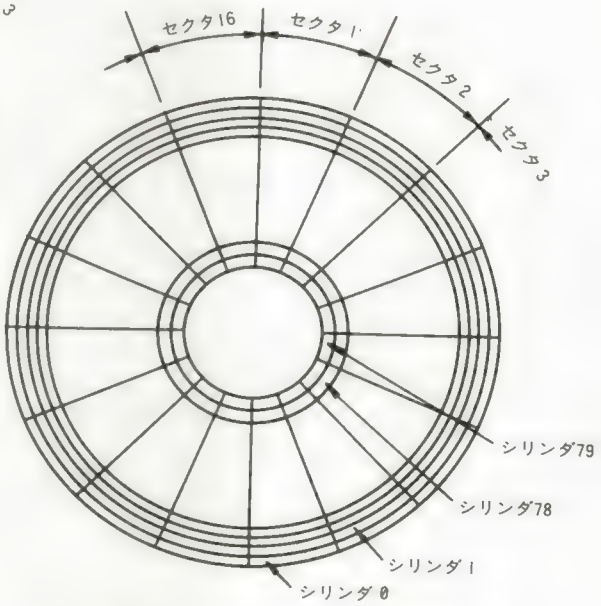
システムフォーマットを行う場合は、

ＩＮＩＴ〝デバイス名：〝

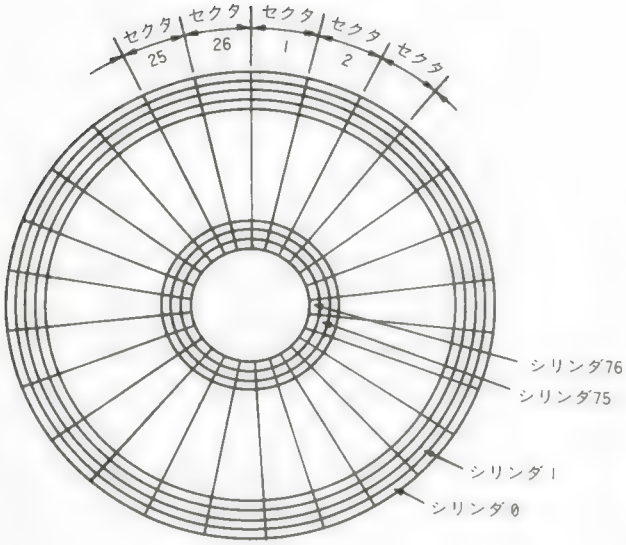
を実行してください。



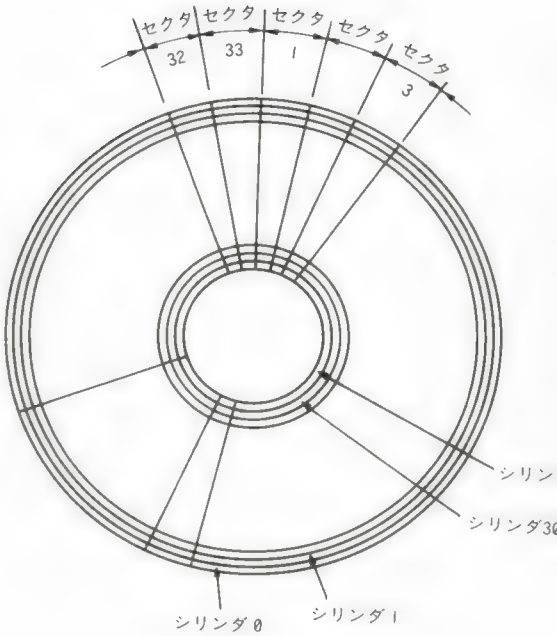
5インチFD　2Dタイプのディスク構造



5インチFD　2DDタイプのディスク構造



5インチFD　2HD
8インチFD　2D　IS　タイプのディスク構造



5インチハードディスクのディスク構造

2.7　ディスクファイル管理方法

3インチまたは5インチ、8インチフロッピーディスクおよびハードディスクのすべてのタイプのディスクへの書き込み　読み出しには、ヘッド番号、トラック番号、セクタ番号を指定します。また、これらの値をもとに連続した論理アドレスを求め、その最小単位のレコード（128　256バイト）ごとにも可能です。しかしBASICでのファイル管理は、クラスタ（1クラスタ＝16レコード＝4Kバイト）を最小単位として行っています。

ディスク上におけるファイルの記録状態は、FAT（File Allocation Table）内のクラスタ番号とディレクトリをもとに行っています。

各ディスクの記憶容量によって、FAT，ディレクトリ、データ領域は次表のようになっています。

なお8インチ片面単密度記録では、1レコード（128バイト）単位の書き込み　読み出しは可能ですが、BASICではサポートしていません。

また、ハードディスクの使用終了時には、ヘッドを使用領域外に移動することによって、ディスク上のデータやFAT、ディレクトリなどの破壊を防いでいます。

（単位＝レコード）

3インチまたは5インチフロッピーディスク				8インチフロッピーディスク	ハードディスク
ディスク / 項目	2D	2DD	2HD	2D	
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

2.7.1 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、日付等を記録し、1ファイルについて32バイトを使用します。

各タイプのディスクはともに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ファイル分）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、次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

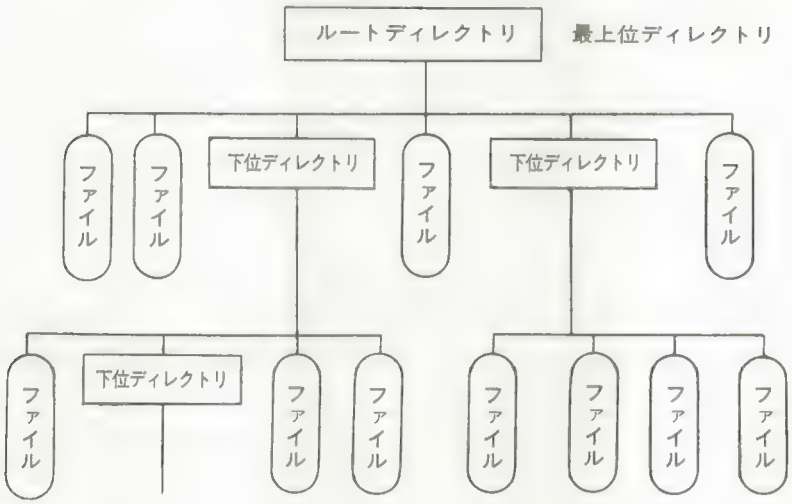
ディスク / 項目		ルートディレクトリ領域 レコード数	ファイル数
3インチまたは5インチFD	2D	16	78
	2DD	16	158
	2HD	16	247
8インチFD	2D	16	247
ハードディスク		16	2504

ディレクトリのファイルに対する構成を、次に示します。

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　エクステンションエリア　（3文字）
18バイト目	パスワードのパック（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：　'86年12月06日土曜日、16時36分 先頭から、年年　月　曜　日日　時時　分分 86　C　6　06　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

ディレクトリの構造は、下図のようなツリー構造になります。



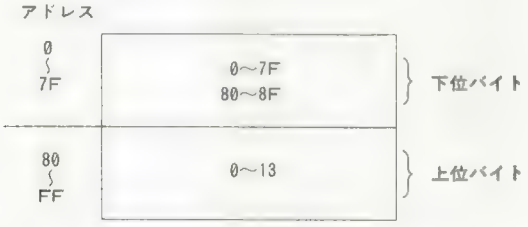
2.7.2 FAT

ＦＡＴはディスクで管理されるファイルのチェイン状態を表し、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の構成は、次のようになっています。



上図のように、ＦＡＴ領域の１レコードを１２８バイトずつに分割し、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の先頭から次表のように必要です。

3インチまたは5インチFD 2D,2DD	2バイト
5インチFD　2HD 8インチFD　2D	3バイト
ハードディスク	4バイト

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インチまたは5インチFD			8インチ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

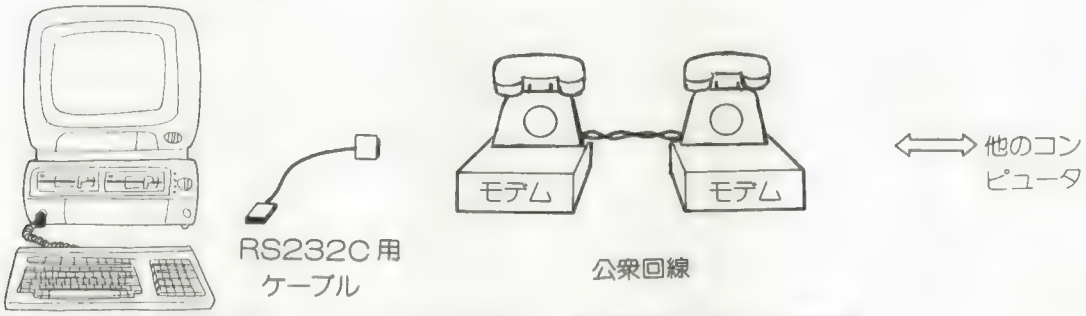
第3章 RS-232Cインターフェイスの使い方

本機は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他の機器との間でシリアルデータの送受信ができます。

3.1 接続方法

3.1.1 接続例

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。



（形名：CZ-8LM1）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

3.1.2 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コネクタのメス型を使用しています。ピン配置と信号端子は次のとおりです。



RS−232Cコネクタ端子表

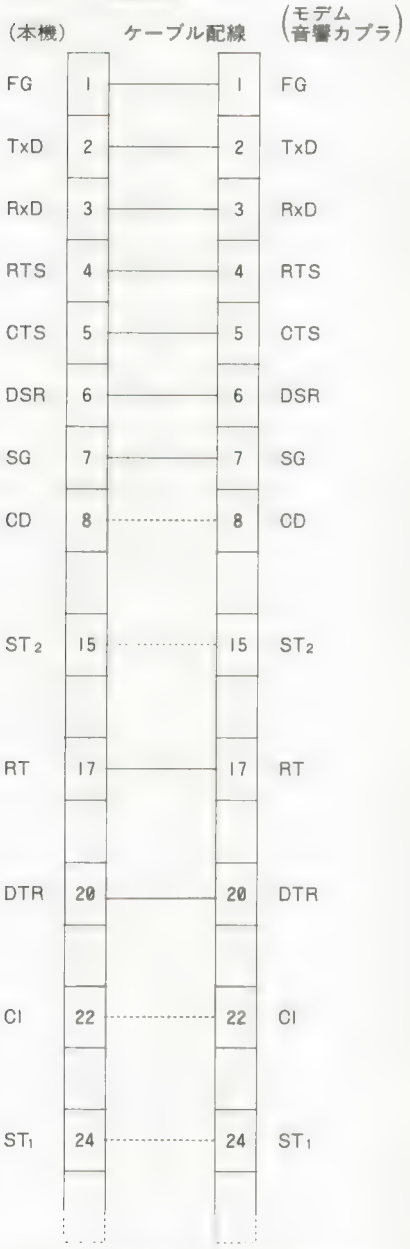
端子番号	信　号　名	信号方向	機　　能
1	FG（保安用アース）	←→	保安用アース
2	T_xD（送信データ）	→	送信データ
3	R_xD（受信データ）	←	受信データ
4	RTS（送信要求）	→	ONにより相手を送信可能とする
5	CTS（送信可）	←	ONの時送信可能と判断する
6	DSR（データセットレディ）	←	ONの時相手が送・受信の準備ができていると判断する
7	SG（信号用アース）	←→	信号用アース
8	CD（キャリア検出）	←	ONの時受信データ有効と判断する
9〜14	NC		
15	ST2（送信信号エレメントタイミング）	←	同期式通信時の送信信号エレメントタイミング信号を入力する
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

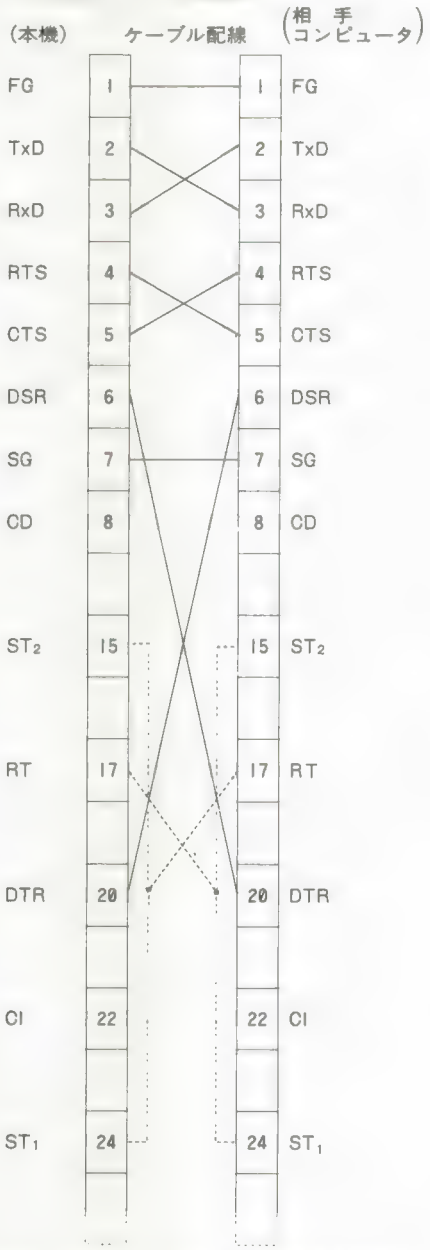
3.1.3 接続方法

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用ケーブルを必要とします。相手機器がモデム（または音響カプラ）かコンピュータかによってコネクタ信

モデムや音響カプラと接続する場合のケーブル配線図



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



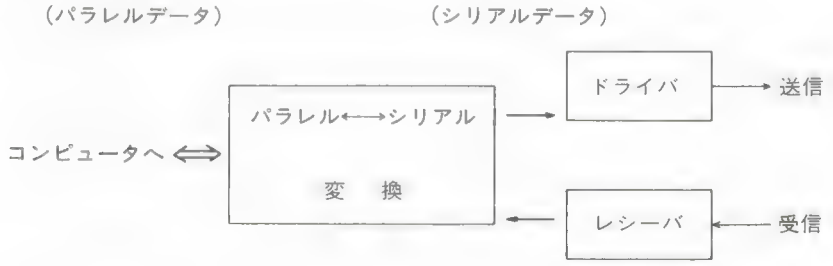
…で示す信号線はBASICではサポートしておらず特に接続する必要はありません。外部同期を使用する場合など、ユーザがマシン語レベルでソフトを作成して使用するときに必要に応じて接続します。

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りますので、相手機器の説明書もよく読んだ上で接続してください。

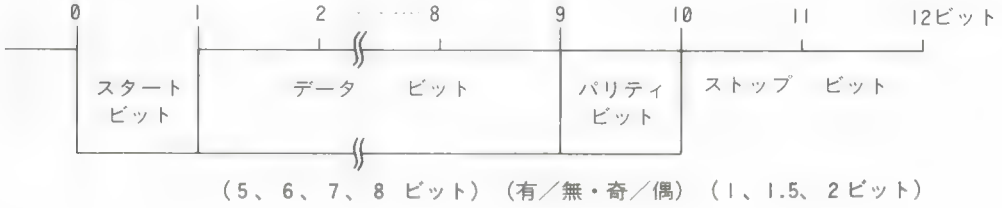
前ページに一般的な接続例を示します。

3.2 RS-232C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で規定された電圧レベル（±１２V）に変換して送信したり、また、逆に受信したシリアルデータをパラレルデータに変換したりするものです。



シリアルデータの形式は、同期通信方式、非同期通信方式によって異なります。本機のＢＡＳＩＣでは非同期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う場合には、このデータ信号フォーマットおよびボーレート（信号の伝達速度）を一致させる必要があります。



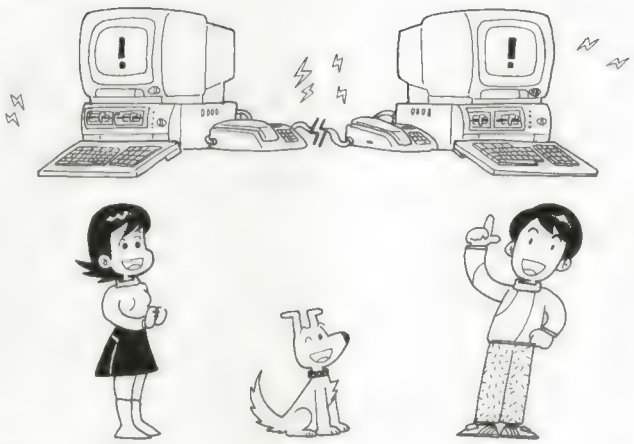
3.3 BASICでRS-232Cインターフェイスを扱う方法

RS－２３２Cインターフェイスでは、データはファイルという概念で取り扱われますので、ＢＡＳＩＣの入出力命令を使って一般の入出力装置と同様に使用できます。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されます。

" COM：通信パラメータ "

通信パラメータには次のものがあります。

・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



次に、各パラメータについて説明します。

●ボーレート

0～6の数値によりボーレートを指定します。

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

通信制御方法を指定します。

X：XON／XOFFコードによる制御を行います。

R：RTS制御信号のON／OFFによる制御をします。

Nまたは省略：通信制御を行いません。

この通信制御とは、データ送受信時に受信側において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方法のことをいいます。

"X"を指定すると、データとしてXOFFコード（&H13）を送信側へ送ることによって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。

"R"を指定すると、RTS信号線をOFFにすることによって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：データビット長7ビットでカナの送受信ができます。

Nまたは省略：データビット長7ビットでカナの送受信ができません。

データビット長7ビットでのカナの送受信はシフトインコードSI（&H0F）があると、それ以降のデータをカナとして処理し、シフトアウトコードSO（&H0E）があると、それ以降のデータを英数字として処理します。

●CR，LFコードの送信処理

C：復帰改行コードとしてCRコード（&H0D）を送信します。

Lまたは省略：復帰改行コードとしてCRコード（&H0D)+LFコード（&H0A）を送信します。

●CR，LFコードの受信処理

C：CRコード（&H0D）の受信によって復帰+改行処理をします。

Lまたは省略：CRコード（&H0D）のみを受信するとそれはデータとして処理されます。CRコード（&H0D)+LFコード（&H0A）を連続して受信すると、復帰+改行処理をします。

●日本語文字列の表現方法指定

J：JIS漢字コードを使用します。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または省略：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。エンドコードを指定すると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からＳＡＶＥ動作を終了します。ＬＯＡＤ命令を実行した場合は、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命令によってオープンした場合は、ＣＬＯＳＥ命令の実行によってエンドコードが送信されます。また、インプットファイルとしてオープンした場合は、エンドコードの受信によってＥＯＦ関数の値が真になります。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。実際に指定するパラメータは、キャラクタコード&Ｈ４０～&Ｈ５Ｆに対応する文字を使用します（&Ｈ６０～&Ｈ７Ｆも可）。たとえば、Ｄ（またはｄ）と指定するとエンドコードとしてコントロール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われません。

〔注〕

・通信パラメータのうちボーレート、パリティ、データ長、ストップビット長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ビットの時に可能です。ただし、データ長７ビットのときでも、ＳＩ／ＳＯコードおよび、ＫＩ／ＫＯコード制御を使用することによって可能です。この場合のデータ形式は、次のようになります。

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、あるいはシフトＪＩＳ漢字コードなどを使用して日本語文字を送受信するには、漢字表示モードをＫＭＯＤＥ１に設定してください。

3.4　入出力命令と割り込み処理

3.4.1　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、アスキー形式でＲＳ－２３２Ｃインターフェイスを介して相手機器に送信します。

```
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プログラムファイルを、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〕

メインメモリ上の機械語プログラムを、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。



LOADM "COM：通信パラメータ"

相手機器より送られてきた機械語プログラムを、ＲＳ－２３２Ｃインターフェイスを介してメインメモリに読み込みます。

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ファイルをオープンし（通信パラメータの設定を行うこと）、ＰＲＩＮＴ＃，ＩＮＰＵＴ＃命令など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すると同一ファイル番号でデータの送受信ができます。

PRINT＃ファイル番号，〔X_1，X_2，……〕

X_1，X_2，……：出力する式あるいは文字列

ＲＳ－２３２Ｃファイルにデータを出力します。データの最後に通信パラメータで指定した送信改行コードを送ります。

INPUT＃ファイル番号，X_1，〔X_2，……〕

X_1，X_2，……：入力した値を格納する変数

相手機器からRS－２３２Cを介して受信したデータを変数に読み込みます。

```
LINPUT#ファイル番号，x$
                    x$：入力した文字列を格納する文字変数
```

相手機器からRS－２３２Cを介して受信したデータ（２５５文字まで）を文字変数に読み込みます。通信パラメータの受信改行コードを受信すると動作を完了します。

```
WRITE#  ファイル番号，〔x1，x2，……〕
                    x1，x2，……：入力した値を格納する変数
```

PRINT#命令とほぼ同じですが、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オーテーション（"）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
```

相手機器から送られてきたn個の文字データを返す関数です。

```
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を返します。

```
EOF（ファイル番号）
```

受信動作開始時は偽（0）を返します。相手機器からエンドコードが送られてくると真（－１）の値を返します。

```
CLOSE#ファイル番号１，〔#ファイル番号２，……〕
```

OPEN文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など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。

〔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信します。

3.4.2 割り込み処理

INPUT#, INPUT$命令を実行すると受信データ待ちの状態となり、データを受信するまでプログラムは停止します。このとき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割り込み処理ルーチンに処理を移すことができます。この処理ルーチンからの復帰はRETURN命令によって行います。

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、停止を指定できます。

```
ON COM GOSUB { 行番号 }
             { "ラベル" }
```

```
RETURN { [行番号] }
       { ["ラベル"] }
```

※行番号、ラベルを省略すると、中断した所から処理を開始します。

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 PRINT D$
80 PRINT #2,D$
90 GOTO 50
100 CLOSE #1,#2
110 END
```

サンプル
プログラム

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 OPTION SCREEN 4
40 INIT "MEM0:"
50 OPEN "O",#1,"MEM0:"+F$
60 OPEN "I",#2,"COM:6N83XNCCND"
70 IF EOF(2)=-1 THEN 120
80 LINPUT #2,D$
90 PRINT D$
100 PRINT #1,D$
110 GOTO 70
120 CLOSE #1,#2
130 END
```

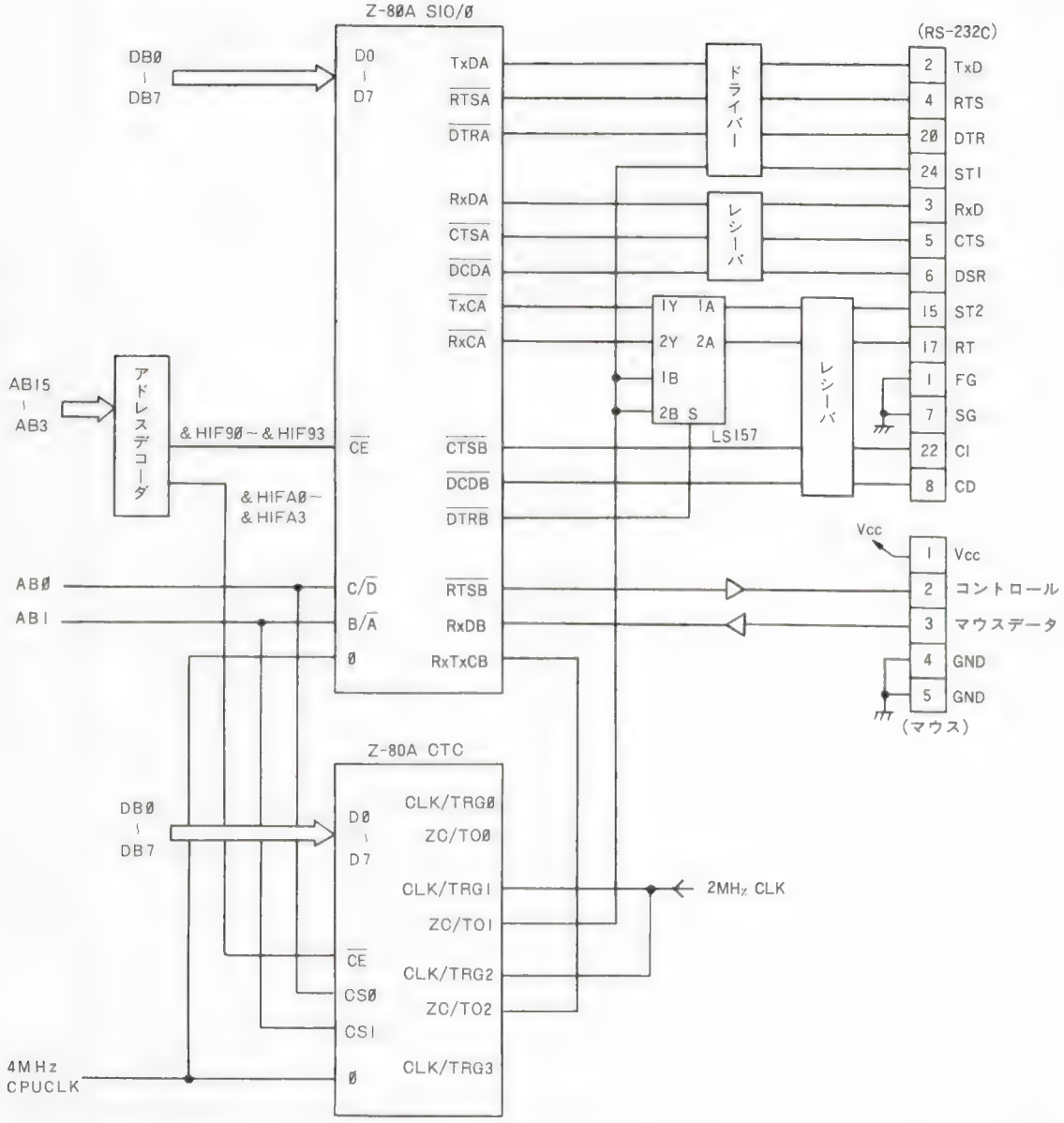
ライン命令実行中にRS－２３２Cインターフェイスよりデータを受信すると割り込みがかかり、そのデータを画面に表示して前の処理を再開します。

```
10 INIT:CLS 4
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
```

3.5　ユーザが機械語でRS-232C用ソフトを作成する場合の使用法

RS－２３２C周辺のシステム図を示します。

データ転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、Ｚ８０Ａ－ＣＴＣ（カウンタ／タイマ用ＬＳＩ）を用い、シリアルデータ通信用にはＺ８０Ａ－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う場合には、このＣＴＣとＳＩＯを通信形態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。



3.5.1 CTC（Z80 A-CTC）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に入力して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ますので、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……&H1FA0
チャンネル1……&H1FA1
チャンネル2……&H1FA2
チャンネル3……&H1FA3

CTCはカウンタまたはタイマとして使用でき、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などを決定する制御語（ここでは&H45）を設定し、続いて時間定数レジスタに分周比として0～&HFF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に出力します。ボーレートと時間定数レジスタの値は一義的には定まらず、SIOの設定によって変わります。これについてはSIOの項で説明します。

3.5.2 SIO（Z80 A-SIO）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、チャンネルBをマウス制御用に使用しています。SIOを動作させるためには、まずコマンド書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データ……&H1F90
チャンネルAコマンド……&H1F91
チャンネルBデータ……&H1F92
チャンネルBコマンド……&H1F93

SIOは、コマンドの書込みレジスタを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になりコマンドを書き込むことができます。たと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うことにより、ＳＩＯは初期化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は、次のようにします。

（送信）

```
LD    A，data……　（送信データ）
LD    BC，1F90H
OUT   (C)，A
```

（受信）

```
LD    BC，1F90H
IN    A，(C) ……　（受信データがアキュムレータに入る）
```

ＳＩＯの書き込みレジスタ４の上位２ビット（D_6，D_7）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受信クロック（TxC，$\overline{RxC}$）の比率を示します。その関係は次式のとおりです。

送信・受信クロック＝ボーレート×クロックレート（1，16，32，64）

ＣＴＣおよびＳＩＯの設定と通信クロックの関係を表２に示します。

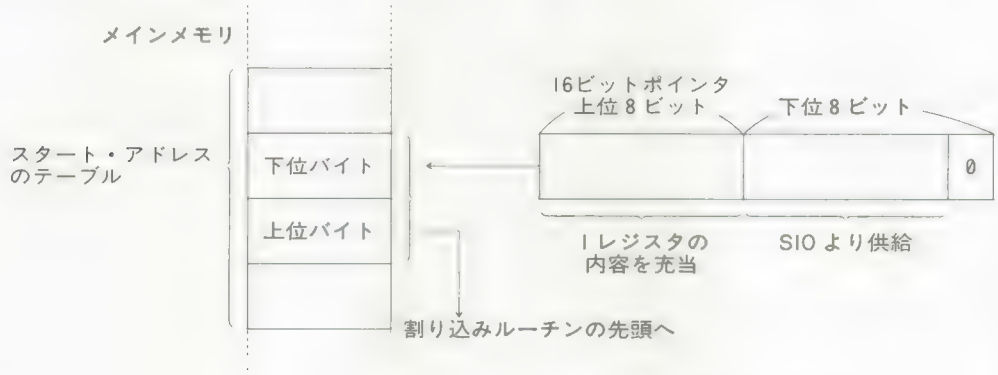
ＣＴＣの分周比（時間定数レジスタ値）	ＳＩＯのクロック･レートに対するボーレート(ボー)		
	×16	×32	×64
13	9600	4800	2400
26	4800	2400	1200
52	2400	1200	600
104	1200	600	300
208	600	300	150

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（TxD，RxD）、コントロール信号線（RTS，CTS，DTR，DSR）とGNG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ほかに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によって行います。ＳＩＯのチャンネルBはマウス用に使用しているため、$\overline{\text{DTRB}}$出力ポートの切り換えの際には、$\overline{\text{RTS}}$$\overline{\text{B}}$出力ポートの状態を変化させないよう注意してください。

3.6　割り込み処理の使用

ＣＰＵとＳＩＯの間で割り込み処理（モード２）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、前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は割り込みを受けつけるとＩレジスタの内容を上位８ビット、ＳＩＯからの割り込みベクトルを下位８ビットとする、１６ビットのポインタの指し示すメモリから２バイトを読み出し、その内容をアドレスとする割り込み処理ルーチンにジャンプします。



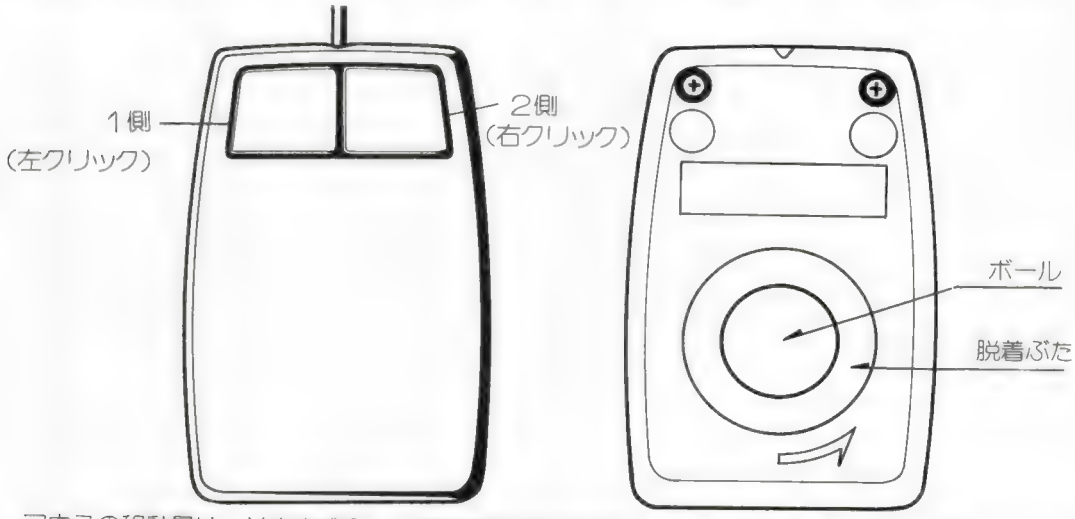
本機のＢＡＳＩＣ（ＣＺ－８ＦＢ０２）を起動するとＩレジスタには&ＨＦ８が設定され、&ＨＦ８３０～&ＨＦ８３Ｂ番地を割り込み処理ルーチンのスタートアドレステーブルとして使用できます。

第4章　マウスの使い方

本機にはマウスが標準装備されています。

4.1　マウスとは

本機のマウスは２ボタン式の機械式マウスです。マウスを移動させるとマウス裏面（ボタンのついた面の反対面）の中央で金属のボールが回転します。マウスは回転に応じた信号を発信します。



マウスの移動量は、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すことができますので、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おり、アプリケーションプログラムでは機能の選択や指示などのセレクト用としてよく使われます。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。

（注）　マウスを使用しているうちに、裏面のふたの中（ボールが入っている部分）にごみが入ってしまうことがありますので、時々ふたをあけ（矢印の方向に回す）、中を清掃してください。

4.2　マウスを使う

本機では、ディスクBASIC（CZ－8FB02）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。

4.2.1　マウス関数の書式と機能について

機能の設定

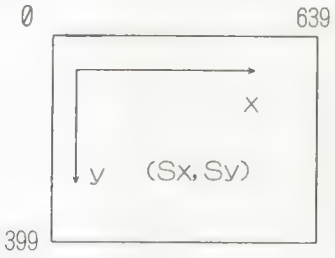
```
MOUSE  0
```

マウス機能をOFF状態にします。マウスを使用しないのにマウス機能をONにしておくと、割り込み(注)の関係で処理スピードが遅くなります。ですから、マウスを使わないときはこの命令を実行してください。

（注）ここでいう割り込みとは、ある仕事をしている途中に別の仕事をし、また元の仕事に戻る、といったことを指します。マウス機能がON状態だと、マウスを使うかどうかにかかわらずマウスからの割り込みが発生します。そのため、他のプログラム実行中にマウスからの割り込み処理を受け付けるため、全体の実行速度が遅くなります。

```
MOUSE  1，Sx，Sy
                    Sx，  Sy：マウス初期座標
```

マウス機能をONにし、初期座標として画面座標（Sx，Sy）を設定します。Sx，Sy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を考えた場合、WIDTH命令によって、設定された解像度と一致するよう設定するのがよいでしょう。ただし、WIDTH命令の設定はOPTION　SCREEN命令の設定状態によってはエラーとなります。



Sxで設定された値がマウス関数のMOUSE(0)の初期値、Syで設定された値がMOUSE(1)の初期値としてセットされます。

Sx，Syを省略するとマウス機能をON状態にするだけとなります。このときSx、SyはBASIC起動時には0が入ります。

また、Sx，Syの値がマウスカーソルの移動範囲の設定の値を超えていた場合は、移動範囲の最大値、または最小値に設定されます。

```
MOUSE  2，d，r
                    d：移動方向（d＝0：X方向，d＝1：Y方向）
                    r：移動比率（1～32）
```

X（横）方向、Y（縦）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さし、d＝0ではX方向、d＝1ではY方向の移動比率が設定されます。移動比率rは

1～32の値をとり、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。これは、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って、座標の変化が小さくなります。

BASIC起動時では、X，Yの移動比率rはそれぞれ10に設定されています。

```
MOUSE  3，Sx1，  Sy1，  Sx2，  Sy2
                          Sx1，  Sy1，  Sx2，  Sy2：座標
```

マウスカーソルの移動範囲を設定します。座標（Sx_1,Sy_1）と（Sx_2,Sy_2）を結ぶ線を対角線とする長方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命令によって、設定した画面の解像度と一致させるのがよいでしょう。

また、MOUSE1，Sx，Syで設定された座標（Sx,Sy）が、この範囲を超えている場合には、移動範囲の最大値または最小値となります。

座標（Sx_1,Sy_1）と（Sx_2,Sy_2）を設定するときは、$Sx_1<Sx_2$，$Sy_1<Sy_2$となるようにしてください。

状態の読み出し関数

```
MOUSE（0）
```

マウスカーソルの現在のX（横）座標を値として返します。

```
MOUSE（1）
```

マウスカーソルの現在のY（縦）座標をこの値として返します。

```
MOUSE（2，b）
                          b：ボタン番号
```

指定されたボタン番号bのボタンが押されているとき真（－1）の値を返し、そうでないときは偽（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

b＝2で右側のボタン

の状態を返します。

MOUSE（3，b）

b：ボタン番号

ボタン番号bのボタンが押されたときのマウスカーソルのX（横）座標を値として返します。

MOUSE（4，b）

b：ボタン番号

ボタン番号bのボタンが押されたときのマウスカーソルのY（縦）座標を値として返します。

MOUSE（5，b）

b：ボタン番号

ボタン番号bのボタンが離されたときのマウスカーソルのX（横）座標を値として返します。

MOUSE（6，b）

b：ボタン番号

ボタン番号bのボタンが離されたときの、マウスカーソルのY（縦）座標をこの関数の値として返します。

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とき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ときのX座標の差が、この関数の値となります。

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（縦）方向の移動距離を値として返します。つまり、最後にこの関数が呼び出されたとき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ときのY座標の差が、この関数の値となります。

4.2.2 マウス関数を動かす

次のプログラムを入力してください。

```
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↵とすると、次のように表示されます。

X＝0　　Y＝0　　SW1＝0　　SW2＝0

ここで、マウスを動かしてみてください。X＝　とY＝の後ろの数字が変わります。

では、ボタンを押したり離したりしてください。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の表示が現れ、離すとそれぞれが0になることがわかります。

では、SHIFT ＋ BREAK でプログラムを止めてLIST↵を入力してください。

プログラムの説明をします。

行番号10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をお使いの方は、本機の標準　高解像度切換えスイッチで合わせてください。プログラムを変更する必要はありません。

行番号20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。

行番号30は、X（横）方向の移動比率を5に設定します。

行番号40は、Y（縦）方向の移動比率を12に設定します。

行番号30と40の移動比率をいろいろ変えて違いをみてください。

行番号50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ときマウスを動かしてください。移動の範囲の値を超えて大きくなったり小さくなったりはしません。

行番号60は、変数Xにマウスカーソルの X座標を、変数Yに Y座標を代入します。

行番号70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ー1が、押されていないと0が代入されます。

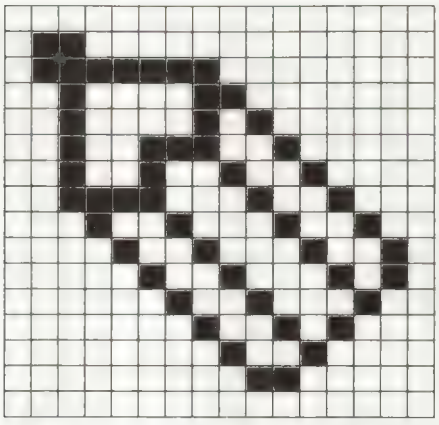
行番号80は、それぞれ変数の値を表示します。

行番号90は、行番号60にジャンプします。

4.2.3 マウスカーソルを表示させる

本機でマウスカーソルを表示させるには、そのためのプログラムが必要です。次に示すのはそのサンプルです。図のよう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、右のボタンでは画面をクリアするというものです。

```
10 INIT:PALET:OPTIONSCREEN0
20 WIDTH 80,25,1,2
30 LINE(0, 0)-( 7, 7),XOR,BF,HEXCHR$("000000 606060 7F7F7F 212121 212121 272727 242424 3C3C3C")
40 LINE(8, 0)-(15, 7),XOR,BF,HEXCHR$("000000 000000 000000 808080 404040 202020 909090 484848")
50 LINE(0, 8)-( 7,15),XOR,BF,HEXCHR$("121212 090909 040404 020202 010101 000000 000000 000000")
60 LINE(8, 8)-(15,15),XOR,BF,HEXCHR$("242424 121212 8A8A8A 444444 282828 909090 606060 000000")
70 DIM A(50)
80 GET@(0,0)-(15,15),A,7
90 X=0:Y=0:XX=0:YY=0
100 MOUSE 1,0,0
110 MOUSE 2,0,5
120 MOUSE 2,1,4
130 MOUSE 3,0,0,624,384
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 CLS0: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。

行番号１０では、画面の初期化をします。

行番号２０では、グラフィック画面の解像度を６４０×４００ドットに設定しています。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としてください。ただし、マウスカーソルの形状は画面の縦横比の関係で多少ゆがみます。

行番号３０～６０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）を対角線とする長方形の中に表示します（実際には見えません）。

行番号７０では、配列Ａを宣言しています。

行番号８０では、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。

行番号９０は、変数の初期値を０にします。

行番号１００では、マウス機能をＯＮにし初期座標を（０，０）に設定します。

行番号１１０～１２０では、Ｘ、Ｙ方向の移動比率をそれぞれ５，４に設定します。

行番号１３０は、マウスカーソルの移動範囲（０，０）ー（６２４，３８４）に設定します。ここで注意していただきたいのは、グラフィック画面の座標が（０，０）ー（６３９，３９９）と一致しない点です。これは行番号１４０のＰＵＴ@命令がＸ＋１５、Ｙ＋１５になっているためです。ＰＵＴ@命令の有効範囲はこの場合Ｘ方向６３９、Ｙ方向３９９ですから、マウスの座標が（６２４，３８４）を超えるとエラーとなります。

行番号20でＷＩＤＴＨ　８０，２５，０，０とした場合は、ＭＯＵＳＥ　３，０，０，６２４，１８４としてください。

行番号１４０・１７０では、行番号８０で読み込んだマウスカーソルを表示します。

行番号１５０は、変数Ｘ，ＹにマウスのＸ，Ｙ座標を代入します。

行番号１６０は、マウスの左ボタンが押されているときに線を引く処理をします。

行番号１８０は、変数ＸＸに変数Ｘの値を、変数ＹＹに変数Ｙの値を代入します。

行番号１９０は、マウスの右ボタンが押されているときに画面をクリアする処理をします。

行番号２００は、行番号１４０にジャンプします。

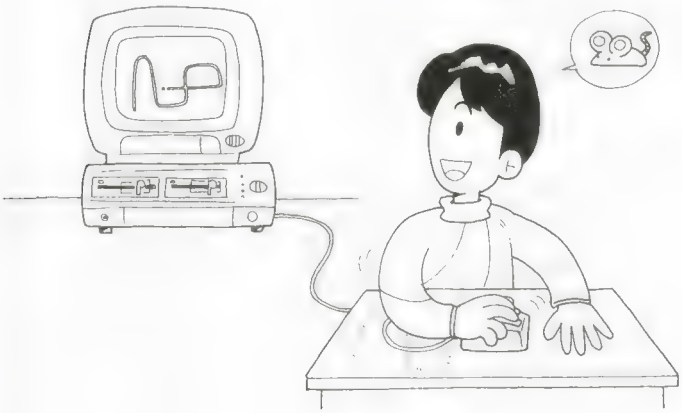
4.3　注意事項

本機には前面と後面に１つずつ、計２つのマウス用コネクタがありますが、この２つのマウス用コネクタには、同時にマウスを取り付けないようにしてください。正常な動作をしません。

・マウスは平らなかたいテーブルの上で使用してください。

・また、水やほこりのないところで使用してください。水やほこりが入ると、マウスをこわすおそれがあります。

・マウスの抜き差しは、電源がＯＦＦの状態の時に行ってください。



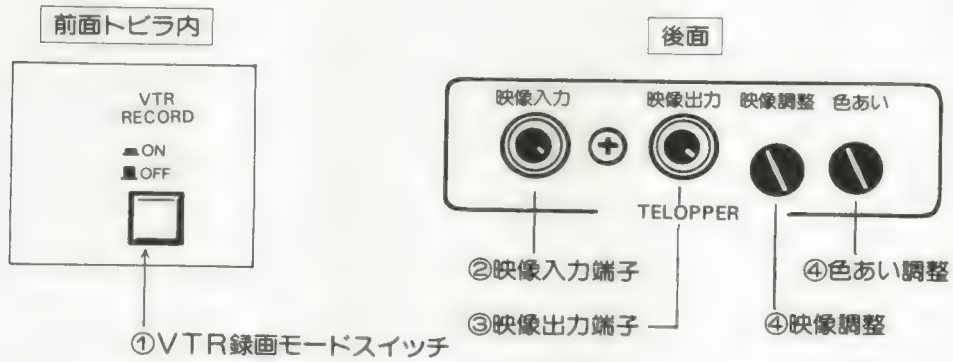
第5章 デジタルテロッパーの使い方

本機では、専用ディスプレイテレビとの組み合せ、あるいは家庭用テレビとの組み合せにより、コンピュータ画像とテレビやビデオ画像との重ね合せ（スーパーインポーズ）画像が表示できます。さらに内蔵のデジタルテロッパ機能を利用すれば、コンピュータ画像やスーパーインポーズ画像などビデオ画像の録画が可能になります。

5.1 特徴

・スーパーインポーズ処理部にSSS（セパレート・サブキャリア・スーパーインポーズ）方式を採用しました。
・コンピュータ映像部のエッヂノイズ（輝度、色、フリッカー）が極めて少なく、安定高画質なスーパーインポーズを実現しました。
・入力映像ソースの画像劣化がほとんどありません。
・VTRなど不安定な入出力ソースにも、安定した同期で、かつコンピュータ映像レベルの色合いを保持するダブルクランプ方式（ピーク&ペデスタルクランプ）を採用しました。

5.2 各部名称



①VTR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、スイッチをON（インターレースモード）にします。
②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です。
③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う端子です。

④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。

5.3 VTR録画モードスイッチ

	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 に録画するときおよび画像取込みを行うときは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） • ノンインターレース走査	• 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） • インターレース走査
映像入力	不　要	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 標準複合信号ですから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）を行なっても同期の乱れがありません。
注意事項	• 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。

本機前面のトビラ内には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。以下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、録画しない場合はＯＦＦにします。

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す。

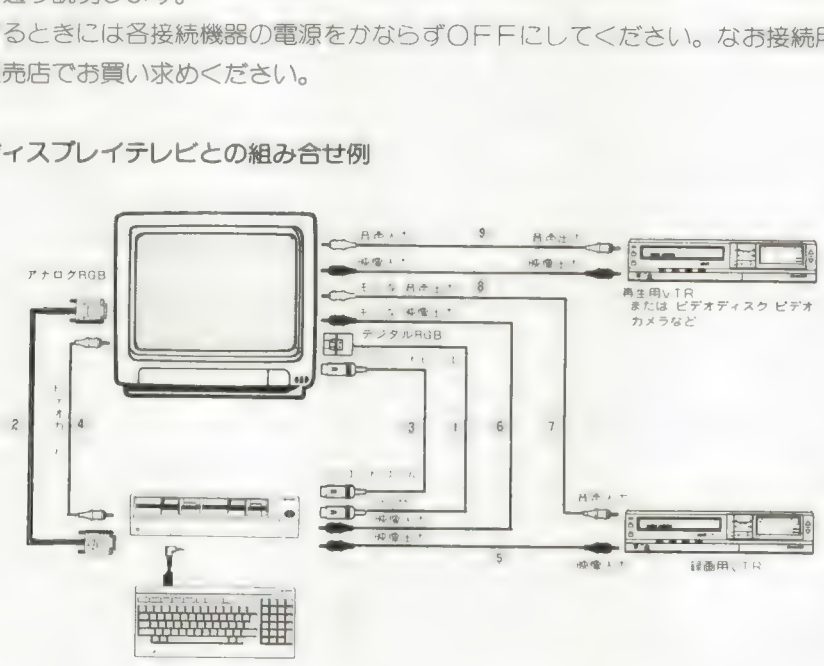
5.4　使用方法

本機内蔵のデジタルテロッパにより、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）。
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、１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、ビデオ入出力端子付テレビなど）の組み合せによってさまざまな接続方法ありますが、ここでは標準的な接続方法を2通り説明します。

接続するときには各接続機器の電源をかならずＯＦＦ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。

●専用ディスプレイテレビとの組み合せ例



接続方法

①～④本機とディスプレイテレビとをデジタルＲＧＢ信号用ケーブル、アナログＲＧＢ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。（②はCZ-６００Ｄのみ）
⑤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。
⑥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。
⑦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。
⑧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。

⑨ディスプレイテレビの音声入力端子と再生用ＶＴＲの音声用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ＯＮにしてください。なお、各接続機器の取り扱い方や操作については、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のトビラ内の、ＶＴＲ録画モードスイッチをＯＮ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－６００Ｄ，８７０Ｄ，８５５Ｄ，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、ビデオ画面（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は、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）。

コンピュータ画像をモニタ録画する場合

本機のキーボードで SHIFT を押しながらテンキーの [.] 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

本機のキーボードで SHIFT を押しながらテンキーの [+] 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。

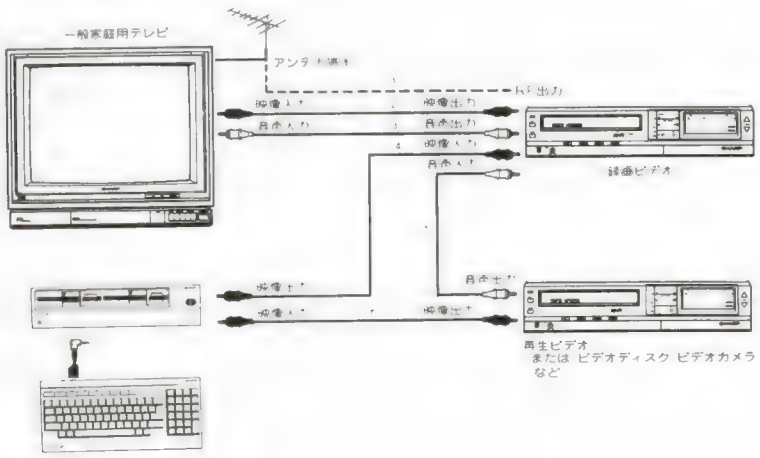
ビデオ画像のモニタ録画をする場合

本機のキーボードで SHIFT を押しながらテンキーの [=] 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出され、録画できる状態になります。

プログラムを組む場合

前面トビラ内のＶＴＲ録画モードスイッチをＯＦＦにし、本機のキーボードで SHIFT を押しながらテンキーの [.] を押します。

●一般家庭用テレビとの組み合せ例



※映像入力端子付テレビの場合は②，③の接続を行い、端子がない場合には①の接続を行います。

接続方法

①テレビに映像入力端子がない場合、この接続を行います。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、接続ができましたら各接続機器の電源をＯＮ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従ってください。

操作方法

専用ディスプレイテレビとの組み合せ例の操作方法を参照してください。
ただし、②項については次の点に注意してください。
②の接続の手順で①を行った場合は、チャンネルを１ ch または２ ch に合わせます。
②③を行った場合はビデオ画面にします。

5.5　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では、グラフィック画面、テレビ画面に対する黒抜き表示を行うことができ、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えます。

黒抜き表示は、専用ディスプレイテレビ（ＣＺ－６００Ｄ，８７０Ｄ，８５５Ｄ，８５０Ｄ）との組み合わせでは、双方のビデオカット端子を接続することによりＲＧＢ出力で直接行うことができます。（CZ-６００Ｄにて、アナログＲＧＢ出力で行う場合は、アナログＲＧＢケーブルのみの接続でかまいません。） それ以外のモニタでは、本機の映像出力端子から出力されるＮＴＳＣ信号によって、黒抜き表示を行うことができます。

5.6　注意事項

●このテロッパ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。

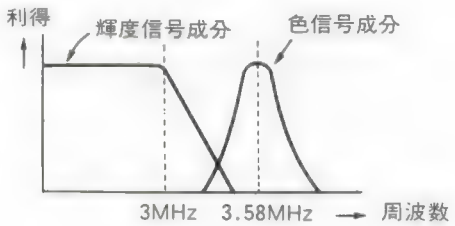
●コンピュータ画像またはスーパーインポーズ画像をＶＴＲに録画する際は、ＮＴＳＣという帯域制限を受けるため、コンピュータ画面を下のいずれかに設定してください。

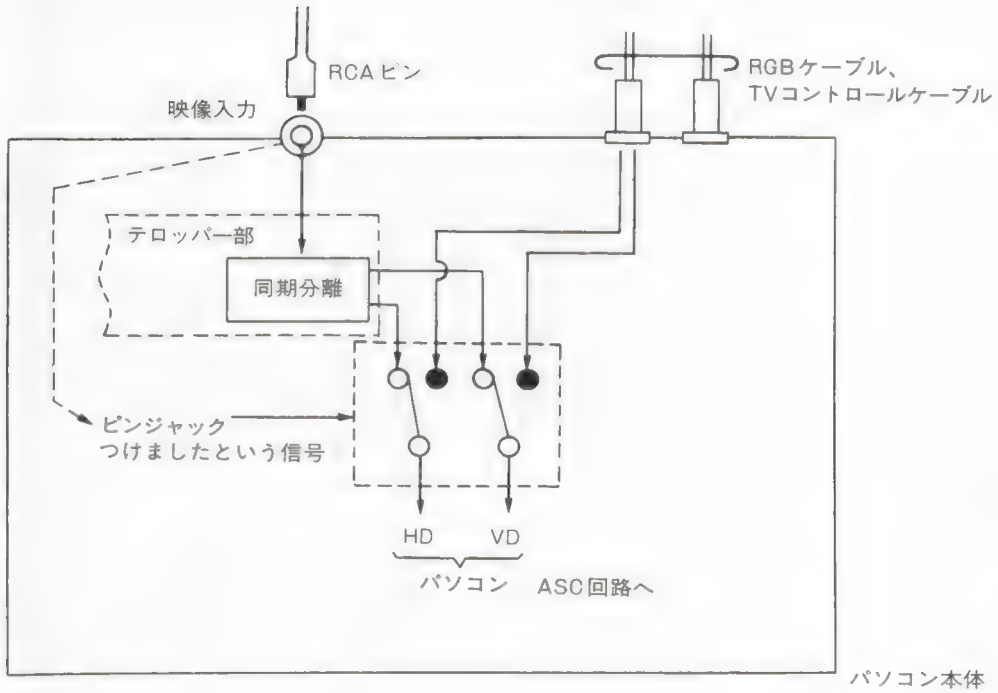
４０字×１０行，４０字×１２行，４０字×２０行，４０字×２５行

また、タイリングペイントのような細かいドット構成の画面は、ちらつきや色変化が生ずることがあります。さらにＶＴＲの性能により、録画画像に色ずれや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ろにある色信号と交錯しないようにするため、3～3.2MHz程度の帯域幅となっています。このため、横８０字（水平６４０ドット）のように細かい指定をしても、見にくかったり色つきが悪かったりします。





●本機の映像入力端子にRCAピンケーブルを接続し、そのケーブルが他の映像機器（VTR、ビデオカメラ、ビデオディスクなど）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ない場合には、本機の映像出力信号はもとよりモニタの画像も乱れます。したがってデジタルテロッパ使用の際は、必ず映像入力端子に他の機器から映像信号を入力してください。もしデジタルテロッパを用いない場合には、VTR録画モードスイッチをOFF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せでは、ディスプレイテレビの水平同期（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う場合には、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、このとき本機では、RCAピンジャックを指し込んだときから、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ピンジャックを指し込んで、映像機器に接続しないまま放置するとHD、VD信号が本機に入らなくなり、画面が乱れて何も見えなくなります。

●映像入力端子にモニタの映像信号と異な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画面が乱れます。モニタ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と同一の映像信号を入れるか、または何も接続しないようにしてください。

●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時またはコンピュータモード時（インターレースモード）のときのコンピュータ画像は色が付かなかったり、部分的に着色されたりすることがあります。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。

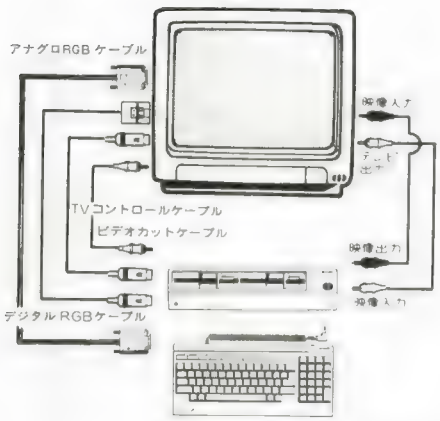
●たとえば、テレビのアンテナ入力信号が弱い場合のテレビ出力や、何度もダビングされたVTRテープの再生信号など、本機に入力される映像信号の同期部分が劣化している場合には、コンピュータ画像が乱れることがあります。

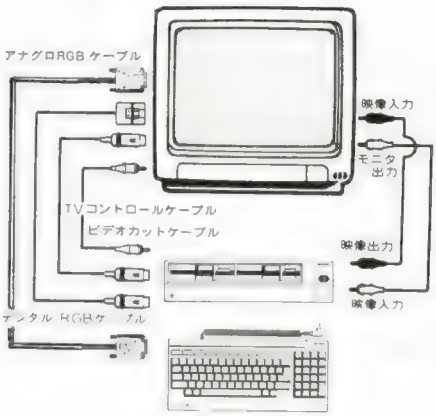
●次のような接続は、異常な画像になるため使用しないでください。これは二重スーパーインポーズとなります。

●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、CRT3命令（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。プログラムで映像出力を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、Y_M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。

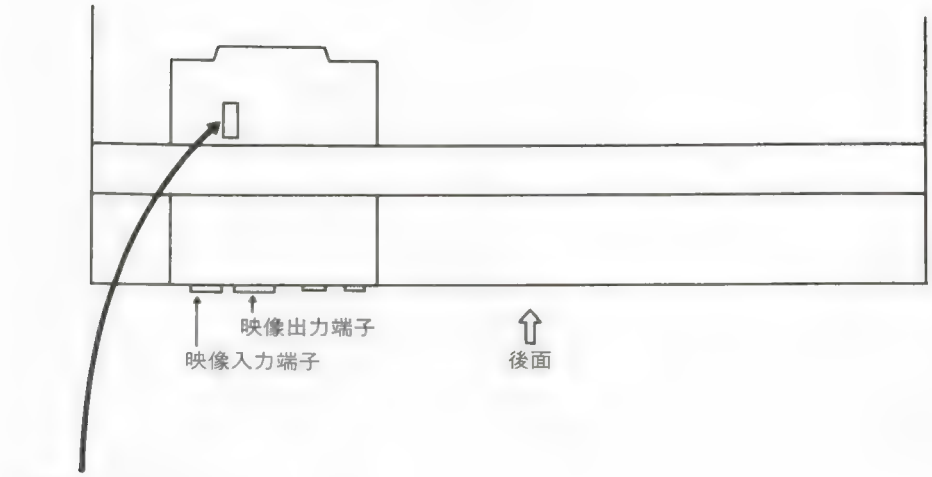
（ｉ）二重スーパーインポーズとなります。

（ii）専用ディスプレイテレビ（ＣＺ-６００Ｄ、ＣＺ－８７０Ｄ、ＣＺ－８５５Ｄ、またはＣＺ－８５０Ｄ）をビデオモードにしたとき、同期流れがおこります。





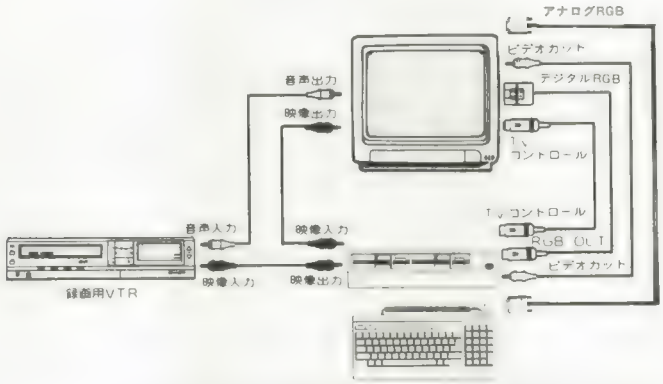
（アナログRGBケーブルはCZ-６００Ｄのみ）



参考

接続応用例①

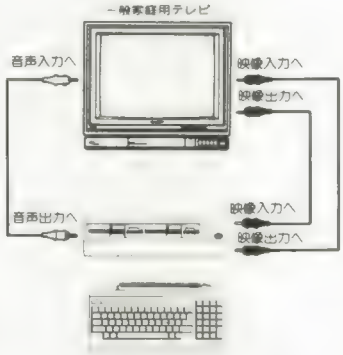
専用ディスプレイテレビ（CZ600D，870D，855D，850D）を使用して〝コンピュータ画像〟や〝テレビとのスーパーインポーズ画像〟、〝テレビ画像〟をVTR　に録画する場合。



（アナログRGBケーブルはCZ-600Dのみ）

接続応用例②

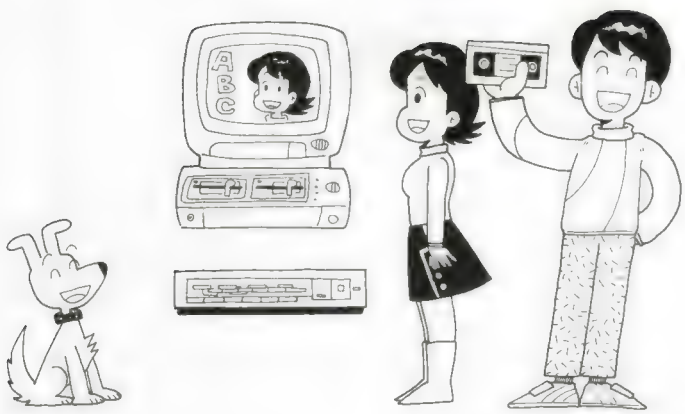
家庭用テレビ（ビデオ入出力端子付）に接続し、テレビ画像を重ね合せスーパーインポーズ画像を楽しむ場合。



ビデオマルチプロセッサ　CZ－8VP1との接続例

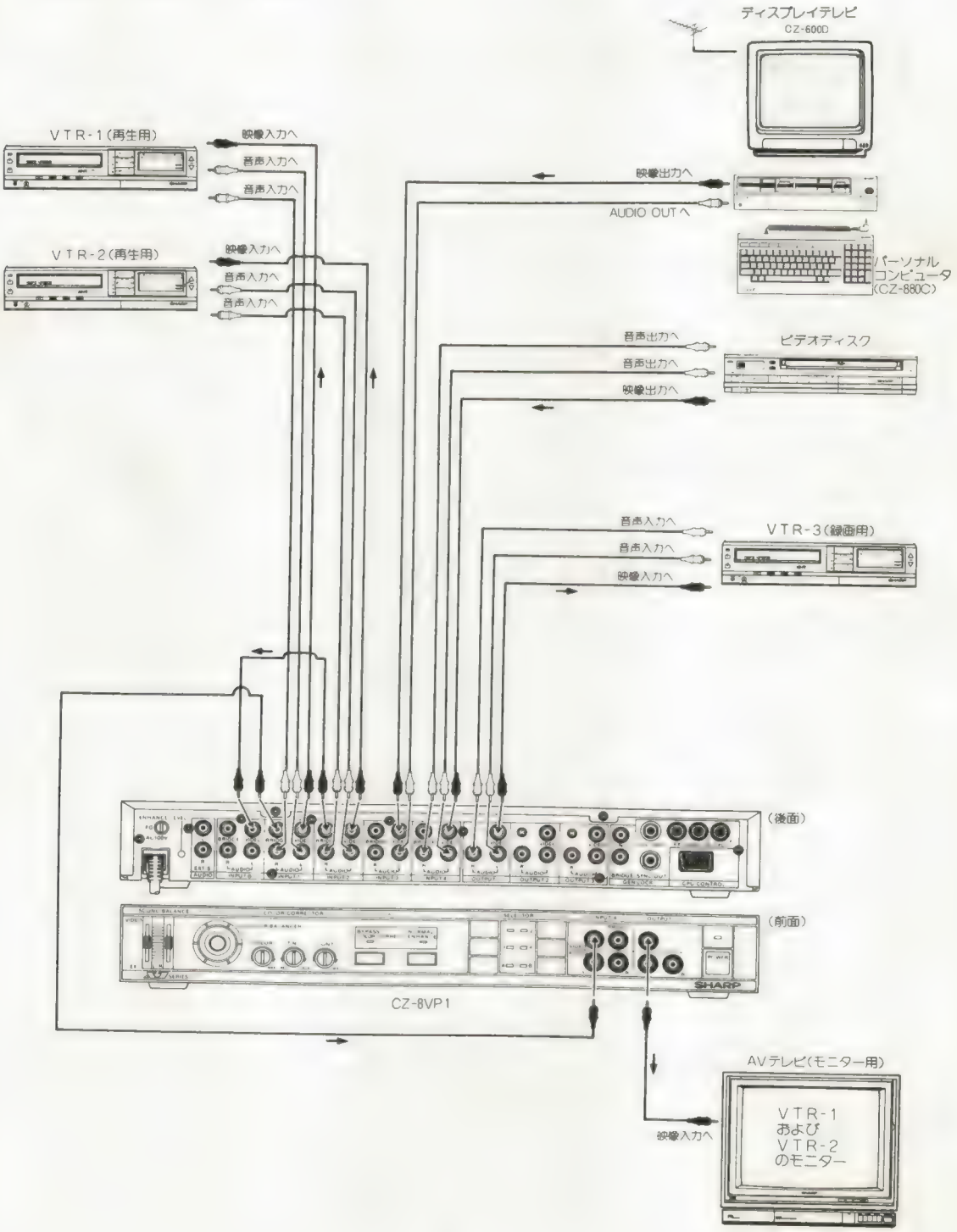
オプションのビデオマルチプロセッサCZ-8VP1は、A／Vスイッチャ、カラーコレクタ、ビデオエンハンサ、GENLOCK機能などを装備し、本機との組み合せによりコンピュータコントロールが可能な映像機器です。特徴は次のとおりです。

- 4入力3出力系と2入力1出力系のA／Vスイッチャを独立に2基搭載しています。また6系統の入力にはすべてブリッジ（共用接続）端子を設けました。
- オーディオ系は、すべてHi－Fiステレオ対応。また、左右独立のサウンドバランスつまみを装備しました。
- 映像のカラー信号を安定再生するカラーコレクタ機能を装備しました。
- ディレイライン方式のビデオエンハンサ機能を装備しました。
- ABロール編集（複数のビデオを再生して編集）ができるGENLOCK端子を装備しました。
- 入力信号切り換えやカラーコレクタ、エンハンサのON／OFFをコンピュータでコントロールすることができます。



接続例

パソコン、ビデオ、ビデオディスクなど、4つのソースを切り換えて録画し、またこれとは独立して2台のビデオをモニタする。



付録

A1　テキスト画面（VRAM）へのアクセス

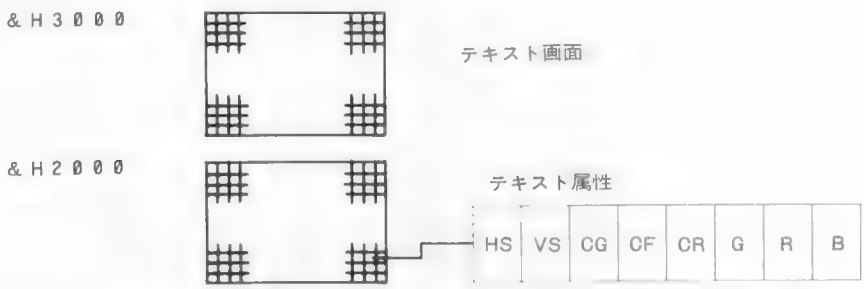
テキスト画面とその属性エリアはともにI　Oポート上にあり、アドレスは以下のとおりです。

テキスト画面：３０００H～３７FFH
属性：２０００H～２７FFH
漢字属性：３８００H～３FFFH

それぞれ２Kバイトの容量を持っています。
テキスト画面とその属性エリアは、ちょうど一対一に対応していて、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。

入力用：POKE@、OUT
出力用：PEEK@、INP

テキスト画面の属性１バイトのビット構成は、次の図のとおりです。



以下、下位ビットから順に説明します。

（1）B，R，G……色の指定（→COLOR）
テキスト画面の文字は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によって行います。

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……文字の点滅モードの指定（→CFLASH）

テキスト画面の文字の点滅モードの指定はビット4で行います。

ビット4が0のとき標準モード、1のとき点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文字のフォントを持っています。

テキスト画面にROMCGの文字フォントで表示させるか、RAMCGの文字フォントで表示させるかの指定は、ビット5で行います。

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指定します。（→CSIZE）

テキスト画面の文字サイズの指定はビット6、ビット7で行います。

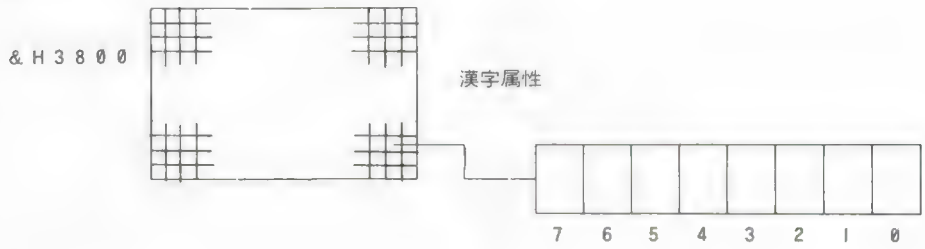
ただし、ビット6（VS）を1に設定するときは、その横1行の属性をすべて1にする必要があります。

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

漢字属性1バイトのビット構成は、次のようになっています。



テキスト属性ビット5	漢字属性		
	ビット4	ビット7	
1	0	0	RAMCG（キャラクタモード）
	0 1 1	1 0 1	RAMCG（外字モード）
0	0 1	0	ROMCG
	0	1	第1水準漢字ROM
	1	1	第2水準漢字ROM

以下、テキスト属性同様、下位ビットから順に説明します。

（1）**下位4ビット**（ビット3，2，1，0）……漢字データ部の上位4ビットコードを示します。
全角文字（漢字ROM）が表示されているとき、漢字ROMのアドレスの上位4ビットが入ります。漢字ROMが選択されていない場合は無視されます。

（2）**ビット4**
漢字ROMが選択されているとき、漢字ROMの第1水準または第2水準の指定を行います。漢字ROM以外が選択されているとき、RAMCGのキャラクタモードか外字モードかの指定を行います。

（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で行います。

このビットが0のときアンダーラインを消去し、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

全角文字の場合ここが0なら文字の左側半分、1なら文字の右側半分であることを示しています。

（5）**ビット7**……ROMCGと漢字ROMの選択の指定

漢字属性のビット4とビット7、およびテキスト属性のビット5（CG）の値によってROMCG、RAMCG、漢字ROMのうちどれが選択されるかが決まります。

A2　組み込み関数以外の数学的関数

組み込み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BASICの関数を使って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してください。

関数名	BASICの形式による公式	関数
secant	A(X)=1/COS(X)	secx
cosecant	B(X)=1/SIN(X)	cosecx
cotangent	C(X)=1/TAN(X)	cotanx
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	$\mathrm{sen}^{-1}x$
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	$\mathrm{cotan}^{-1}x$
hyperbolic sine	I(X)=(EXP(X)-EXP(-X))/2	sinhx
hyperbolic cosine	J(X)=(EXP(X)+EXP(-X))/2	coshx
hyperbolic tangent	K(X)=-EXP(-X)/(EXP(X)+EXP(-X))*2+1	tanhx
hyperbolic secant	L(X)=2/(EXP(X)+EXP(-X))	sechx
hyperbolic cosecant	M(X)=2/(EXP(X)-EXP(-X))	cosechx
hyperbolic cotangent	N(X)=EXP(-X)/(EXP(X)-EXP(-X))*2+1	cotanhx
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	$\mathrm{cotan}^{-1}x$

以上の関数を使うときは、「DEF　FN」命令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
……
```

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桁、倍精度型のとき有効数字１６桁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、次の規則によって行われます。

（１）数値が別の精度の変数に代入されると、それは代入先の変数の精度に変換されます。

〔例〕A％＝3.14……A％は整数型の変数なので、A％の値は３となります。

（２）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!＝3.14159265359# ……Ａ！は単精度型変数なので、Ａ！の値は3.1415927に丸められます。

（３）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、より正確になるということはありません。

〔例〕A#＝3.14…… A#は倍精度型変数なので、A#の値は3.139999999664724になります。

（４）次のような計算を行なうときは、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#は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になります。

（５）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桁まで求めることができます。

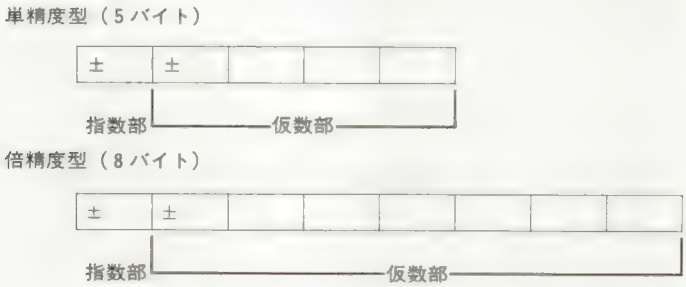
〔例〕A＝SQR（３）……Aの値は1.7320508
A#＝SQR（３#）……A#の値は1.732050807568877

（６）数式の値の型変換を行うには、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われます。



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、このためにおこる演算結果、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、数学的に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およぼす影響について、簡単な例をあげて説明しましょう。

〔例1〕

```
10  X=0
20  FOR  I=0  TO  1500
30  PRINT  X;
40  X=X+. 1
50  NEXT
```

注）　. 1=0.1

このプログラムは変数Xの初期値を0として、0.1ずつ加え、その値を表示するプログラムです。このプログラムを実行すると、はじめは.1，.2，……と増えていきますが、46を過ぎると46，46.1，46.1999999，46.2999999……というように誤差が表れます。

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には0.1を加えるのではなく、整数の1を加え、それを10で割るようにしてください。

```
10  X=0
20  FOR  I=0  TO  1500
```

```
30 PRINT X/10;
40 X=X+1
50 NEXT I
```

〔例2〕

```
10 FOR I=0 TO 1 STEP 0.1
20 PRINT I;
30 NEXT I
```

〔例2〕は、FOR～NEXTループにおいてループ変数の初期値を0、増分を0．1として終了値1まで実行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0、0．1、0．2、……0．9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てください。

```
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#"
```

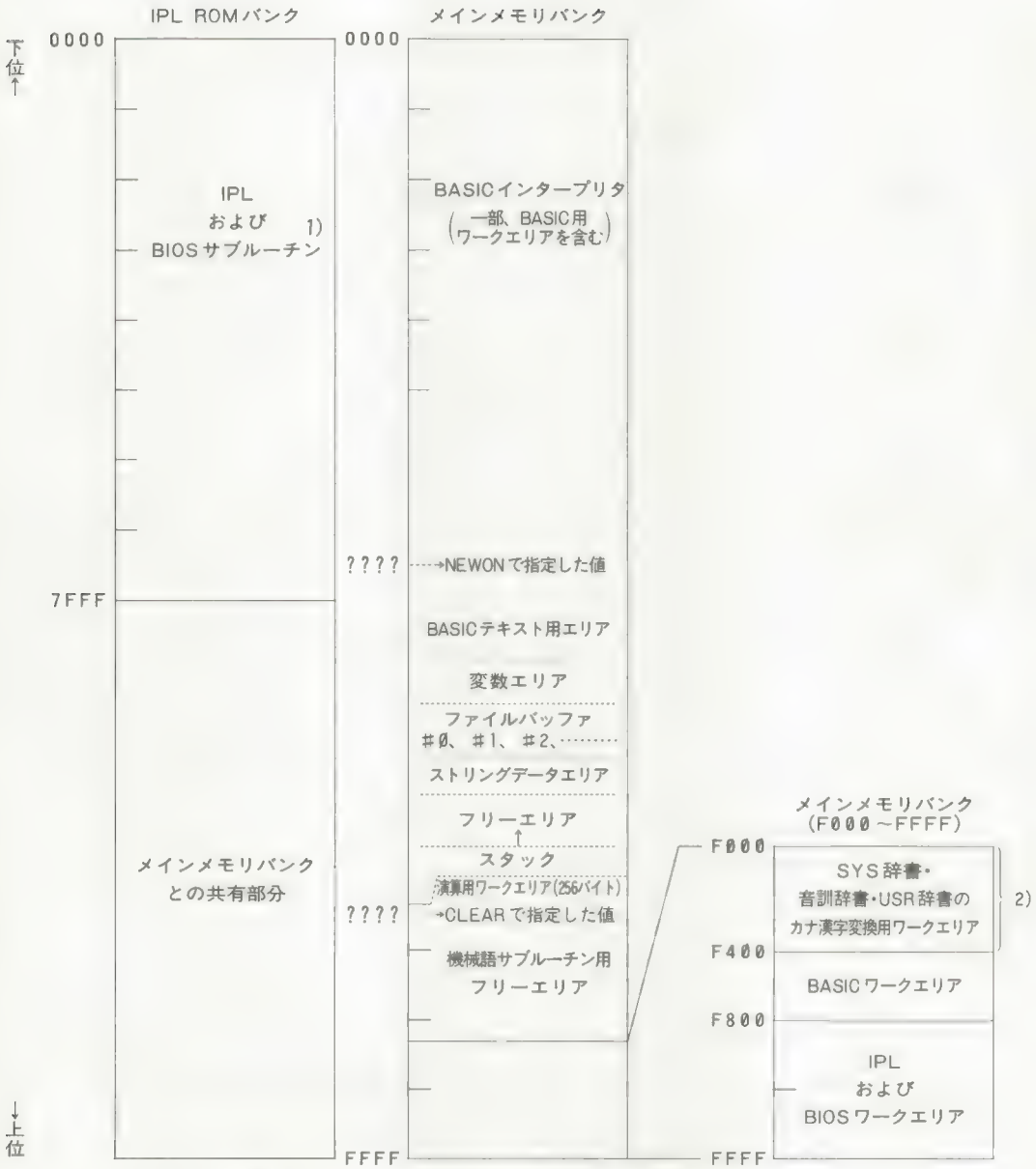
〔例3〕は0．1を3で割って3倍した値を持つX#と、0．1の値を持つY#と比較して等しければ"X#=Y#"と表示するプログラムです。

しかし、このプログラムでは"X#=Y#"と表示しません。これもまたX#とY#の内部表現の誤差によって生じる現象です。

これを解消するには、次のようにIF文の論理式の部分を変えて、10^{-8}の精度で比較するようにしてください。

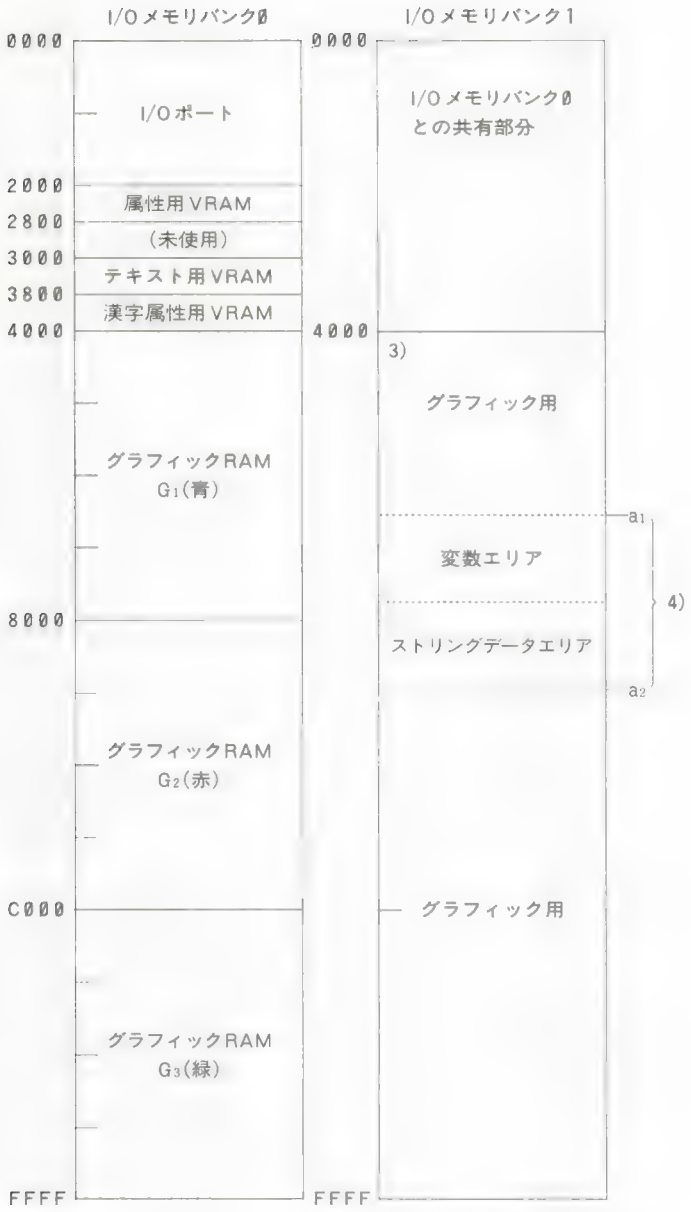
```
10 X#=.1/3*3
20 Y#=.1
30 PRINTUSING"###.##########";X#;Y#
40 IF ABS(X#-Y#)<.00000001 THEN PRINT "X#=
   Y#"
```

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。



3) OPTION SCREEN 1 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 SCREEN 0 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。

4) 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。
a_1 は変数エリアの先頭アドレス、a_2 はエンドアドレス。
VDIM CLEAR を実行していなければ 4000h ～ FFFF が変数エリアとして使用できる。

A6　ユーザ定義文字の作り方

英数字、カナ、セミグラフィック文字、特殊文字などの図形文字は、小さなドットが集まって構成されています。1つの文字を構成するドットは、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（点の集まり）がROMCG（Read Only Memory Character Generator）、および漢字ROMに記憶されています。

しかし、自分の使いたい図形文字がなかった場合はどうしたらよいでしょう。そのためにRAM CG（Random Access Memory Character Generator）が用意されています。これはユーザが好きなドットパターンをデザインして記憶させておく場所です。

ユーザがデザインできる図形文字（以下、ユーザ定義文字といいます）は、そのサイズによって

（1）縦　8×横　8ドット
（2）縦16×横　8ドット
（3）縦16×横16ドット

の3つのタイプがあります。

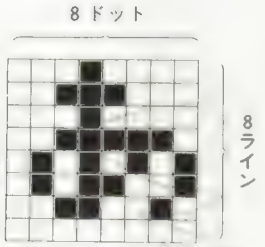
それでは、一つひとつ作り方と表示の仕方について説明していきます。

（1）縦8×横8ドットの場合

縦8×横8ドットのユーザ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すビットパターンが、1個のユーザ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桁が、横8×縦1ドットパターンを表し、8バイトの文字列で縦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見てみましょう。



■はドットがセットされていることを示す。

ユーザ定義文字は8×8ドットパターンで表され、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し、さらに16進表現に変換すると、次のようになります。

（2進数表現）		（16進表現）	
00100000	→	10	……①
00111000	→	38	……②
00010000	→	10	……③
00111110	→	3E	……④
01010101	→	55	……⑤
01011001	→	59	……⑥
00110010	→	32	……②
00000000	→	00	……⑧

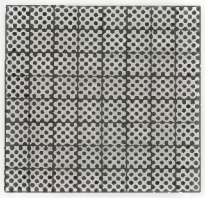
したがって、このドットパターンは、

HEXCHR$("1038103E55593200")
①②③④⑤⑥⑦⑧

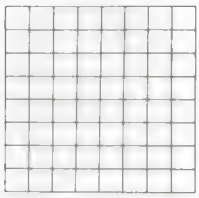
となります。

シアン色は、青と緑を合成することによって表示されます。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，緑の画面に対してパターンを、それぞれ次のように定義しなければなりません。

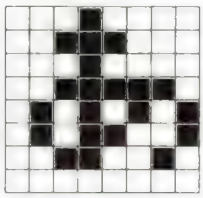
（青の画面）



（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	00111000	⇨38
11111111	⇨FF	00000000	⇨00	00010000	⇨10
11111111	⇨FF	00000000	⇨00	01111110	⇨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$("FFFFFFFFFFFFFFFF0000000000000000
                                00103810 3E55593200")
```

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文字分です。たとえば、これをコード32に定義するには、

```
DEFCHR$(32)= HEXCHR$("FFFFFFFFFFFFFFFF000
              0000000000000103810 3E55593200")
```

と書きます。

こうして定義した文字は、CGEN1とPRINT#0命令を使って、画面に表示することができます。

（例）

```
100  KMODE  0
110  CGEN1
120  A$="FFFFFFFFFFFFFFFF0000000000000000
     0103810 3E55593200"
130  DEFCHR$(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バイトの文字列で済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$(32)=HEXCHR$("1038103E55593200103
              8103E55593200103810 3E55593200")
```

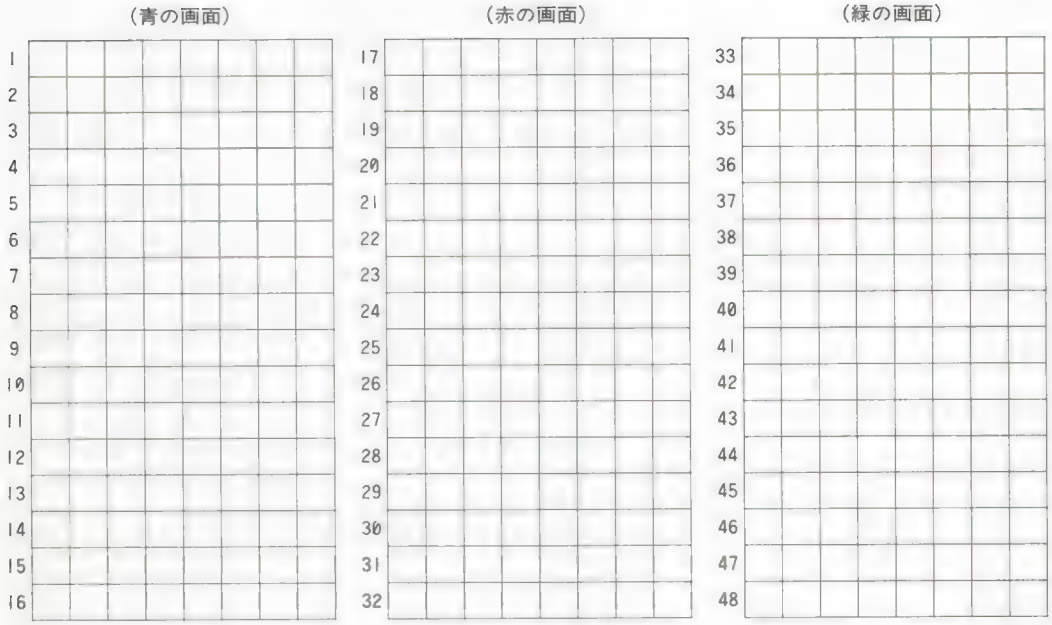
としないで、

```
DEFCHR$(32)=HEXCHR$("1038103E55593200")
```

とするだけで済みます。

（2）縦16×横8ドットの場合

縦16×横8ドットのユーザ定義文字を定義するには、8×8ドットの文字が縦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ドットのユーザ定義文字は、１６×８のドットパターンで表されます。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し、さらに１６進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桁（48バイト）
```

と書きます。

こうして定義した文字は、CGENとＰＲＩＮＴ#0の命令を使って画面に表示することができます。

(例)

```
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$(0)
150  CGEN0
  ※  &H100のコードは  PRINT#0, CHR$(&H00)
      &H102のコードは  PRINT#0, CHR$(&H02)
      &H104のコードは  PRINT#0, CHR$(&H04)
      ……
```

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枚共に同じデータの場合は、１６バイトの文字列で済ませることができます。

（３）縦16×横16ドットの場合

縦１６×横１６ドットのユーザ定義文字の定義には、９６バイトの文字列が必要です。

１６×１６ドットのユーザ定義文字は、１６×１６のドットパターンで表されます。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し、さらに１６進表現に変換してコードの文字列にします。

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桁（９６バイト）
```

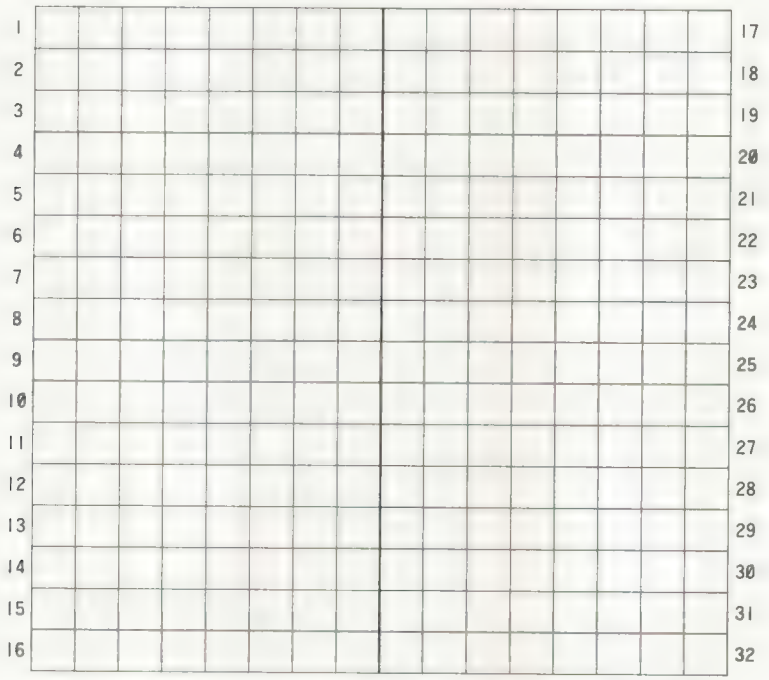
と書きます。

こうして定義した文字は、KMODE１を実行後、普通のＰＲＩＮＴ命令を使って画面に表示することができます。

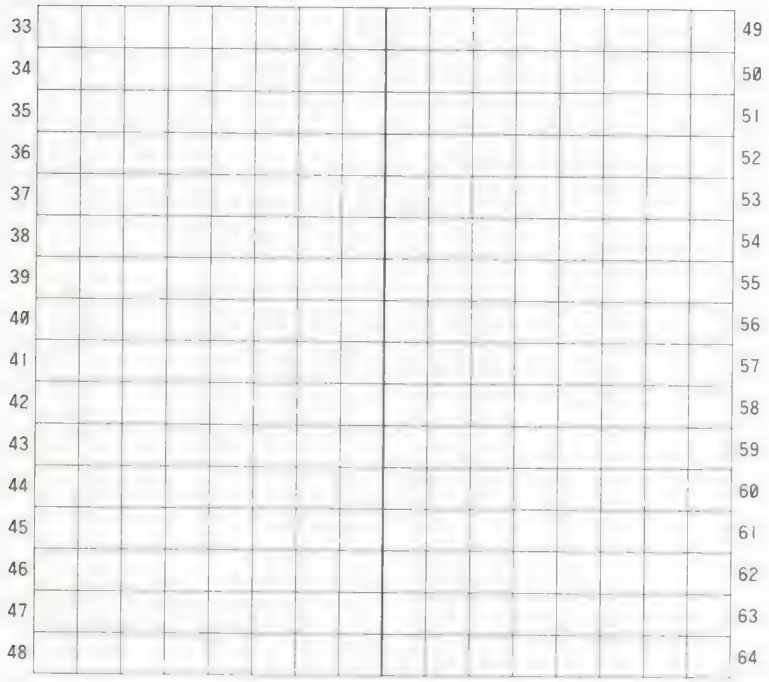
```
KMODE1
PRINT  CHR$(&J7621)
```

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バイトの文字列で済ませることができます。

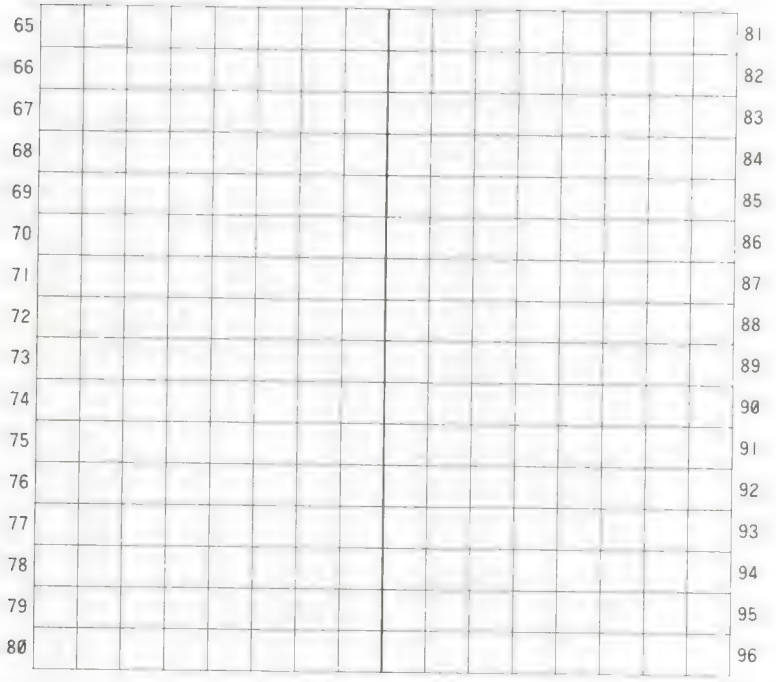
（青の画面）



（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

索引

■ア
アスキー形式……123
アドレス情報……192
アンダーライン……63，64

■イ
1行削除……10
1行挿入……10
1行変更……11
一字変換方式……96，97，99，100，103
一点鎖線……73
色あい調整……222
色指定……60，70，71
インサートモード……36，43
インターフェイス……183
インターレースモード……223，227
引用符（"）……3

■ウ
うるう年……30

■エ
英数字直接入力方式……96，99
映像出力端子……221
映像調整……222
映像入力端子……221
エクステンション……119
エディットモード……38，39
エラー処理ルーチン……167
円……78
エンドコード指定……205
エンベロープ……180，181，182
エンベロープジェネレータ……180

■オ
オクターブ……175
音の長さ……175
終わりかぎ括弧「」」……96
音訓辞書の高速利用……106
音訓変換方式……97，99，105
音響カプラ……201
音程……174
音量……175
音量調整……5，26

■カ
カーソル……2，35
カーソルコントロールキー……14，19，34
カーソルの移動……34
階層ディレクトリ……109
外部記憶装置……20，148
学習機能……104，105
カタカナ入力モード……16
楽器の演奏……170
カナ入力方式……96，99
カナ漢字変換方式……96，99
画面座標系……67，68，70
画面の初期化……65
画面のページ数……51，52
画面分割……58
画面モード……38
画面を消す、クリアする……16，36，65
カラーモード……50，70
カラーコード……52，53，70
カラー画面……50
カレントディレクトリ……112，114
カレンダー付きタイマ機能……26
漢字コード……59
漢字表示モード……58
漢字変換……90，100
漢字BASIC……86
漢字ROM……241
間接出力モード……95，99
間接モード（プログラムモード）…6

■キ
キーワード……3
記憶領域の確保……165
機械語サブルーチン……165，166，167
機械語ファイル……123
機械語モニタ……155，167
記号……105
起動……2
ギリシア文字……105
キャリッジリターンキー……2
キャラクタジェネレータ（CG）…63
キャピタルロックキー……92
休符……175
境界色……79
行の結合……37
行の追加……38
行の分割……36
行番号……6
行番号の整理……39

■ク
句点「。」……96
区点コード……96，100
クラスタ……194，195
グラフィックメモリ……46，65

グラフィックVRAM……46，47，50，100，146
グラフィック画面……44，45，46，47，50，56，67，70
グラフィック座標系……65，67
グラフィック描画……149
クロック……26，28
クロックレート……212
黒抜き表示……225

■ケ
ケーブル配線……201
現在時刻の設定……31
現在時刻を表示する……31

■コ
高解像度ディスプレイ……44
コード入力方式……99，101，107
コピーモード……41
コロン（：）……36
コントロールコード……42
コンピュータ画面……26，31，80，222

■サ
最小単位のレコード……194
座標入力装置（ポインティングデバイス）…214

■シ
シークレット属性……122
シーケンシャルアクセスファイル…110，134，136，137，138
辞書機能……106
システムフォーマット（二次フォーマット）…193
システム辞書、ユーザ辞書変換方式…99
出力ページ……52
消音……26
シリアルデータ……202
信号端子……199

■ス
数学的関数……235
数値精度の変換……236
数値データの誤差……227
スーパーインポーズ……221，227
スーパーインポーズ画面……26，31，32，80，81，213
スキップ……24
スクリーンエディット……34，35
スクロール……32，34
図形文字……44
ストリングディスクリプタ……166
ステートメント……7，17

■セ
正多角形……76
セーブ（プログラムの保存）……20，123
セクタ番号……192，193
セミグラフィック（パターン）……44，58，59
全角文字（日本語文字）……39，40，86，99，101
全角文字の削除……40
全角文字の修正……40
全角文字の挿入……40
全角文字の表示……40
専用ディスプレイテレビ……26，45

■ソ
粗調整レジスタ……179

■タ
代入する……8
タイリングパターン……72，75，82
タイリングペイント……79
タイマ……26，28，29，30
タイマコントロール……31
濁点「゛」……95
縦横2倍文字……62
ダブルクランプ方式……221
単密度……188

■チ
チェイン……130
中間コード……123
中間色コード……54
中間色一覧表……54，55
中間色指定……79
直接モード（ダイレクトモード）…6，99

■ツ
通信制御……204
通信パラメータ……205
通常モード……87
ツリー構造……197

■テ
ディスクタイプ……191，195
ディスクドライブ……189
ディスクBASIC……2
ディスプレイテレビ……44
ディスプレイモード……44，45，47，48，
ディレクトリ……195，196，197
ディレクトリ領域……193
データ信号フォーマット……202
データの送受信……202
データ領域……194，195
テキストエリア……88，91
テキスト画面……44，56，60，67，
テキスト座標系……56，67
デジタルテロッパ……227
デバイス名……21
テレビ画面……26，31
テレビタイマコントロール……26
テレビタイマコントロール画面……27
点線……72，73
テンポ……16
点滅モード……61
点滅（ブリンク）文字……61

■ト
トーンジェネレータ……178
ドット……71
ドットパターン……84，85，241
トラック番号……192

■ニ
日本語処理機能……86
日本語入力……150
日本語入力モード……40，41，87，97，98
入力する……8
入力ページ……52
入力モードの選択……99，121

■ヌ
ヌルコード……64

■ノ
ノイズジェネレータ……177，180
ノンインターレースモード……223

■ハ
ハードディスク……111
ハードコピー……38
倍密度……193
倍文字……61
始めかぎ括弧「「」……96
破線……72，73
パス……112
パスワード……119
パリティ……203
パレットコード……53，70，72，82
半角文字（英数カナ文字）……40，86
番組予約……28
半濁点「゜」……95，99
反転文字……60
反転モード……60

■ヒ
微調整レジスタ……178
ビデオマルチプロセッサ……230
ビデオ画面……224
非同期通信方式……202
標準解像度モード……39
標準／高解像度切換えスイッチ……46
標準ディスプレイ……44

■フ
ファイル……110
ファイルディスクリプタ……117
ファイルバッファ……138
ファイルの属性……121
ファイル番号……62
ファイル名……23
ファンクションキー……23
ファンクションキーの内容表示……58
フォーマッティング……179
フォーマット（一次フォーマット）…21，191
ブザー音……174
フリーエリア……146，147，148，150，
プログラム……7
プログラムテキスト……146
プログラムの実行中断……16，19
文（ステートメント）……7

■ヘ
ヘッド番号……192
ベリファイ……127
変換フィールドエリア……87，90，107
変換フィールドエリアの確保……97
編集……8
変数……7
変数領域……146
偏平率……78

■ホ
ホーム位置……35
ボーレート……203
ボーダーカラー……80，81
補色……60
ボリュームコントロール……180

■マ
マージ……126
マウス……214
マウスカーソル……219
マウス関数……214，215，218
マウス用コネクタ……220
マルチステートメント……17
マルチページモード……52

■ミ
ミキサー……177，180
ミュージック機能……174

■メ
命令（コマンド）……7
メインメモリ……6，146
メモリマップ……231

■モ
文字のコピー……38
文字の削除……18，35，36
文字の挿入……17，35，36
文字の属性……60
モデム……194
モニタ録画……224

■ユ
ユーザ座標系……68，70
ユーザ定義文字……236
ユーザ登録文字（外字）……102

■ヨ
横2倍文字……62

■ラ
ライトプロテクト……21
ラインスタイル……72，73
ランダムアクセスファイル……138，139，140，143

■リ
リードアフターライト属性……1220
リピート機能……34

■ル
ルートディレクトリ（最上位ディレクトリ）…195，196

■レ
レコード……138
レジスタ……169，177

■ロ
ロード（プログラムの再生）……21，127
ローマ字ーカナ変換一覧表……94，
ローマ字ーカナ変換方式……93，96
ロシア文字……105
論理アドレス……194

■ワ
ワークエリア……146
ワード……43
ワード単位でのカーソル移動……36
和音……175
割り込み処理……205，208，213

■A
Aレジスタ（アキュムレータ）……169
ASCIIコード……63
ASK命令……27
AUTO＊……109

■B
BASICインタプリタ……146
BASICファイル……123
BEEP命令……174
BF……75
BIOS ROM……146
BOOT命令……151
BREAK……16
Bレジスタ……170

■C
CALL命令……167
CANVAS命令……51
"CAS0:" ……115
CFLASH……61
CGEN……63
CHDIR命令……114
CHAIN命令……130
CHANNEL n……32
CIRCLE命令……77，78
CLEAR命令……165
CLOSE……133
CLS命令……3，19，65
COLOR……60，70，71
CONSOLE#……109
CONSOLE命令……57，58，97
CREV……60
CRT 0／1／2／3……31
CSIZE……61
CTC……176，209，210，211
CVD……140
CVI……140
CVS……140

■D
D（dump memory）……156
DATE$=……31
DAY$=……31
DEレジスタ……169，170
DEFCHR$……63
DEFUSR命令……168
DELETE……13，19
DEVICE命令……106，120，189

■E
EDIT文……38，39
END……7
EOF関数（END OF FILE） 137
ERASE命令……153

■F
F（find data）……156
FAT（File Allocation Table）…193，194，197，198
FAT領域……183
FIELD文……139
FILES命令……24，111，114
FFキー……24
FRE関数……154

■G
G（gosub）……158
GET……140
GET@……64，83

■H
HLレジスタ……169

■I
INIT命令……65，70，193
IPL（Initial Program Loader）…161
IPLリセット……151

■J
JIS漢字コード……96，101

■K
KEYLIST命令……58
KILL命令……121
KMODE命令……59, 86
KSEN命令……63, 64

■L
L (load) ……156
LFILES……25
LINE命令……64, 72, 74, 75
LIST命令……8, 19, 123
LIST*……108
LLIST*……108
LOAD命令……21, 126
LOAD?……23, 128
LOADM命令……127
LOCATE命令……17, 57, 58, 62
LSET……139

■M
M (set memory) ……156
MERGE命令……123
MKDIR命令……115
MKD$……139
MKI$……139
MKS$……139
MON命令……166
MOUSE……214, 215, 216
MUSIC……174

■N
NAME命令……121
NEW……1
NEWON命令……2, 151
NTSC信号……223, 226

■O
ON ERROR GOTO……170
OPEN命令……132
OPTION SCREEN命令…50, 148, 149

■P
P (printer switch) ……156
PAINT命令……79
PALET命令……53, 80, 81
PALET@命令……81
PATTERN命令……84
PEEK関数……167
PLAY……4, 176
POINT関数……67
POKE命令……166
POLY命令……76, 77
POSITION命令……84
PRESET命令……71, 72
PRINT……3, 4
PRINT#……133
PRW命令……82
PSET命令……71
PSG (プログラムサウンドジェネレータ) …176, 178
PUT……139
PUT@命令……83, 84

■R
R (return) ……156
"R" モード指定……138
RAMCG……63
RCAピンケーブル……227
RENUM……11, 12, 19, 39
REPEAT……34
RGB信号……225, 226
RMDIR命令……117
ROMCG……63, 241
RSET……139
RS-232Cインターフェイス…199, 205
RS-232Cインターフェイス用コネクタ…199
RUN命令……6, 7, 10, 126

■S
S (save) ……156
SAVE命令……123
SAVEM命令……125
SCREEN命令……50, 52
SCROLL n……32
SET命令……122
SIO……209, 211, 212
SOUND……176
SOUND@……276
SSS……221
Syntax error……3, 5
SYMBOL命令……84, 83

■T
T (transfer data) ……159
TAB……36
TEMPO……174, 175
TIMER表示ランプ……29
TIME$=……31
TVPW ON/OFF……31

■U
USR関数……167, 168
USR関数内でのエラー処理……170

■V
V (verify) ……160
VDIM命令……146, 153
VERIFY……128
VOL n……32
VTR録画モードスイッチ……221, 222

■W
W (device write) ……163
WIDTH命令……44, 45, 46, 58
WINDOW命令……67, 68, 70
WRITE#……131

■X
XOR……72, 73
X1フォーマット……191, 192, 193

■Y
Y (device read) ……162, 164
YMキラースイッチ……227

■Z
Z80A……146

■その他
@記号……78
! (change access mode) 161
(set screen mode) ……161
2D……191
2DD……191
2HD……191
3インチフロッピーディスク……189
5インチハードディスク……189
5インチフロッピーディスク……191, 192
8インチフロッピーディスク……191, 192

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	年　月　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-365
T0062-A