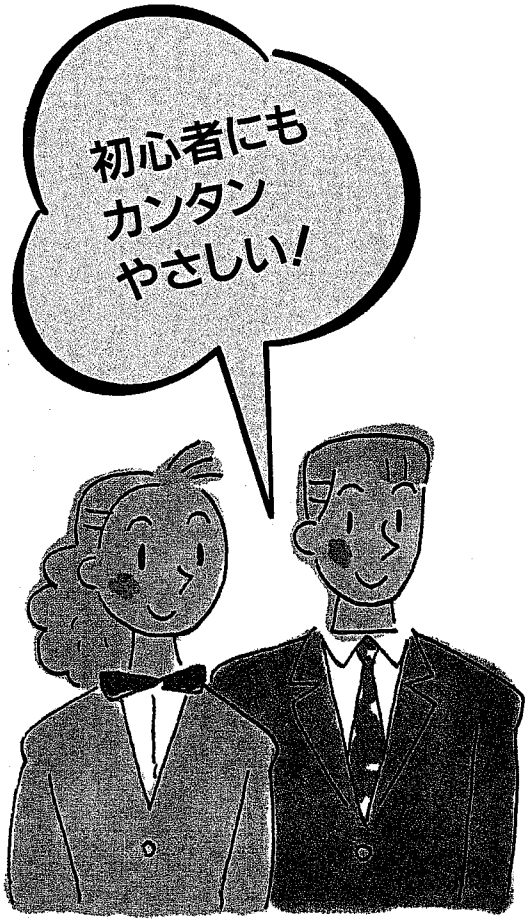
NAiS

制御専用
マルチタスクBASIC言語

FP-BASIC
BASIC LANGUAGE FOR PROGRAMMABLE CONTROLLER

速習ガイドブック



松下電工の制御機器は
グローバルブランド NAiS に統一します。

A&i 快適を科学します

FP-BASIC 速習ガイドブック No.FAF-151① '93・7月

目　次

入門コース

実用コース

テクニック・コース

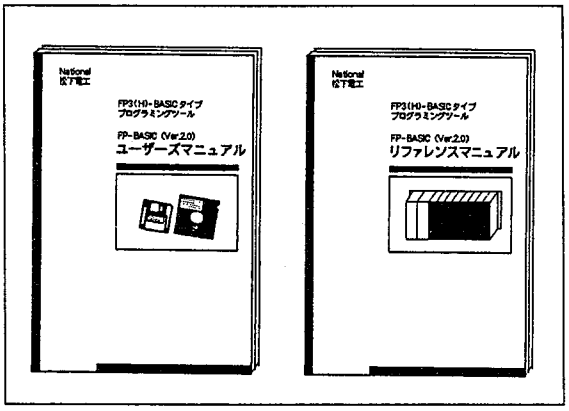
付　録

はじめに

●ＦＰ－ＢＡＳＩＣは、複数の機器制御を同時に行うことができる制御専用のマルチタスクＢＡＳＩＣです。
●入出力Ｉ／Ｏを直接制御できるＯＮ・ＯＦＦ命令、ＳＷ（）関数を持ちますので、どなたにも簡単に制御プログラムを作成していただくことができます。
●このガイドブックは、制御専用プログラム言語ＦＰ－ＢＡＳＩＣの基本的な使い方を説明しています。

・このガイドブックは、FP-BASIC編集ソフト（AFP366108）をNECノートパソコンPC-9801NS/T上で使用することを前提として説明を進めています。
・以下の構成のＰＣをご用意いただき、この研修テキストの記述に沿って操作していただけば、ＦＰ－ＢＡＳＩＣを確実に理解できます。
　BASICタイプＣＰＵユニット
　16点入力ユニット×１台
　16点出力ユニット×１台
＊「７－６ タイマの外部設定」では、追加１台分の16点入力ユニットを使用しています。
＊「９ 高機能ユニットによる割り込み」では、他に割り込みユニット（１台）を使用しています。

《参考マニュアル》
●FP-BASIC編集ソフト ユーザーマニュアル
●FP-BASIC編集ソフト リファレンスマニュアル



入門コースでは、FP-BASIC編集ソフトの基本的な使い方をマスターします。読みながらＰＣとパソコンを操作していただくことにより、ＦＰ－ＢＡＳＩＣの基本的な使い方が理解できるようになっています。

入門コース

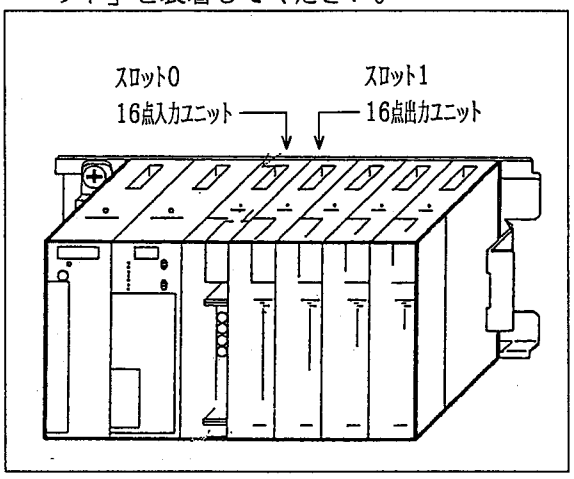
1 ＦＰ－ＢＡＳＩＣの起動

1-1 ＰＣとパソコンの準備

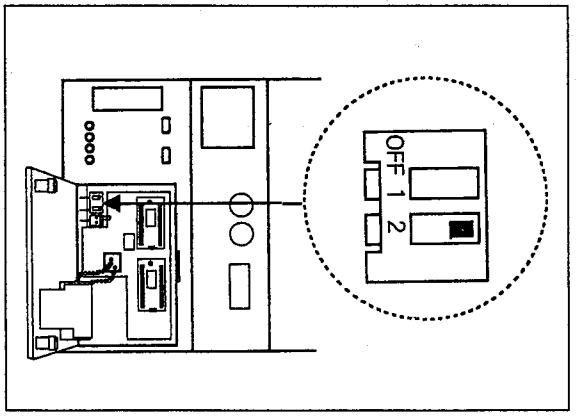
●ＰＣおよびパソコンのセットアップと接続を行います。
●BASICタイプＰＣと、NECノートパソコンPC-9801NS/Tを用意してください。

1 ＰＣの準備

ＰＣには、「電源ユニット」「BASICタイプＣＰＵ」「16点入力ユニット」「16点出力ユニット」を装着してください。



ＣＰＵのスイッチ２（下図参照）をＯＮにして、通信ボーレートを9600bpsに設定します。



2 パソコンの通信条件の設定

パソコンは、「付録Ａ」以降の記述の通りに設定してください。

ポイント

●ボーレートは必ず9600bps(SW2=ＯＮ)に設定してください。

ポイント

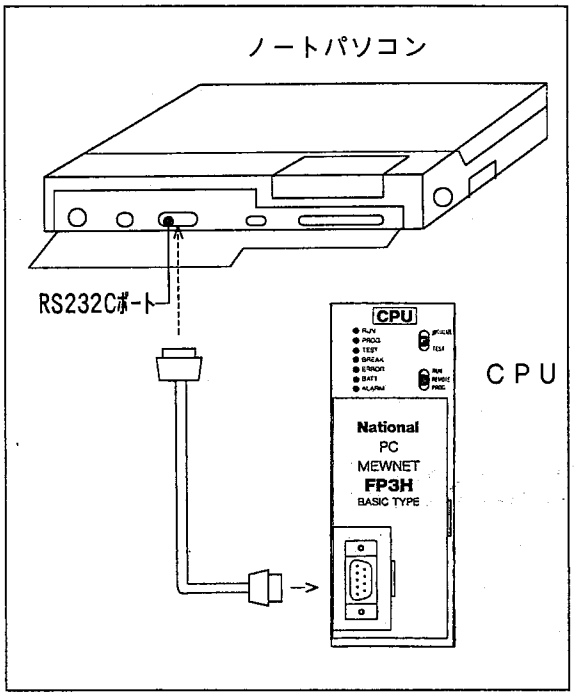
●ＦＰ１ではボーレートスイッチを下にすると9600bpsですが、上にし電源を一度切って入れ直すと、プログラミングのポートがコンピュータリンクMEWTOCOL-COMの通信ポートとなります。

ポイント

●バックアップメモリスイッチＯＮ。
●RAMドライブ使用可能。
●RS232Cインターフェイス使用可能。

3 ＣＰＵとパソコンの接続

パソコンとBASICタイプＣＰＵをRS232C通信ケーブルで接続します。



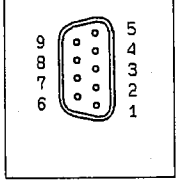
ポイント

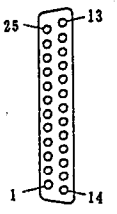
●専用ケーブルを使用します（弊社品番AFB85813）。

●自作する場合は、下記の「■接続ケーブルの結線ついて」を参考にしてください。

■接続ケーブルの結線について

ＣＰＵ側		PC-9801側	
信号名	ピン	ピン	信号名
FG	1	1	FG
SD(TXD)	2	2	SD(TXD)
RD(RXD)	3	3	RD(RXD)
RS(RTS)	4	4	RS(RTS)
CS(STS)	5	5	CS(CTS)
	6	6	DSR(DR)
SG	7	7	SG
CD	8	8	CD(DCD)
ER	9	20	ER





注意

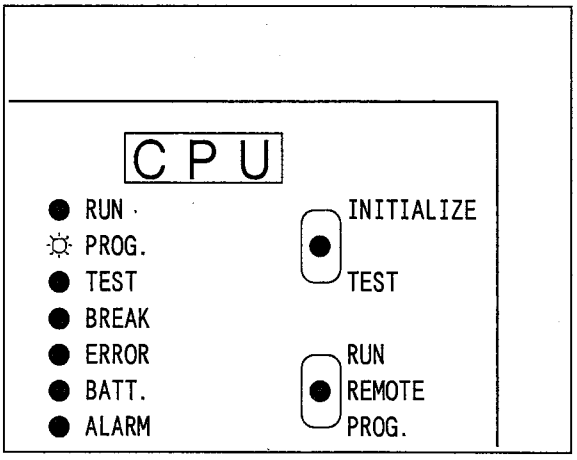
●RS-CS制御信号（フロー制御）は使用しません。

1-2 ＦＰ－ＢＡＳＩＣ編集ソフトの起動

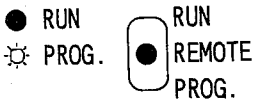
●ＰＣおよびノートパソコンの電源を投入し、ノートパソコンでFP-BASIC編集ソフトを起動します。

1 ＰＣの電源投入

ＣＰＵのモード設定スイッチを「ＲＥＭＯＴＥ」に設定して、ＰＣの電源を投入します。

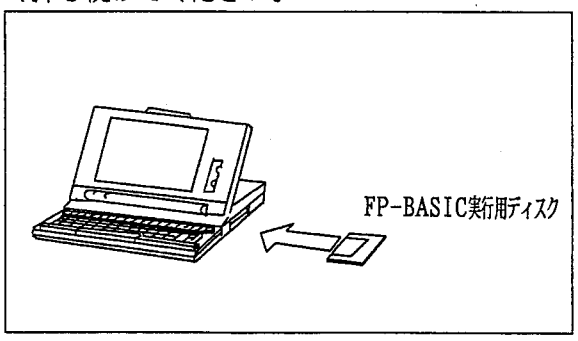


ＦＰ１の場合



2 パソコンでＦＰ－ＢＡＳＩＣを起動

ノートパソコンにFP-BASIC編集ソフトの実行用ディスクをセットしてから、COPYキーを押しながら電源スイッチをＯＮします（電源ＯＮ状態の場合はCOPYキーを押しながらリセットボタンを押します）。こ のときCOPYキーはしばらく押し続けてください。



ポイント

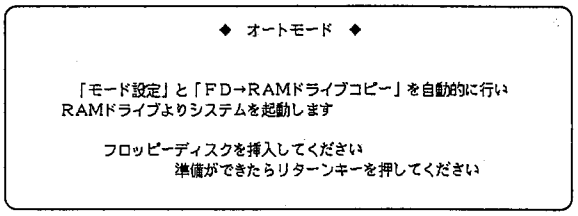
●実行用ディスクは、FP3H-BASICと、FP3-BASIC（64kバイトタイプ）およびFP1-BASICでは作成方法が異なります。

●実行用ディスクの作り方については「付録Ａ」をお読みください。

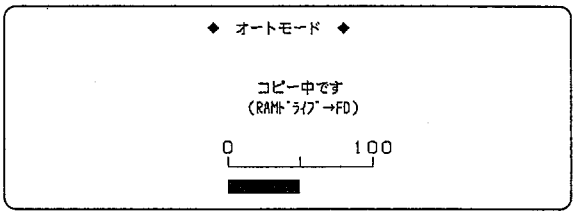
注意

●FP-BASIC編集ソフトを起動するとＲＡＭドライブ の内容が消失します。重要なデータやアプリケーションは、９８ノートメニューの「3.ＲＡＭドライブ→ＦＤコピー」を実行してフロッピーディスクにバックアップしておきます。操作手順については、「付録Ａ」をお読みください。

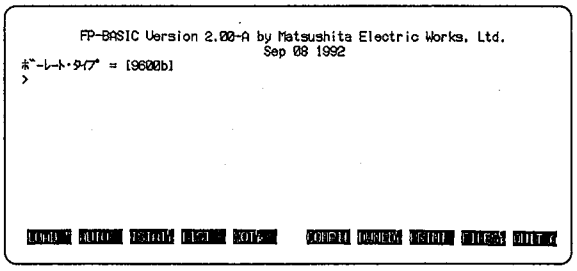
以下の画面が表示されますので↵キーを押します。



コピー中は次の画面が表示されます。



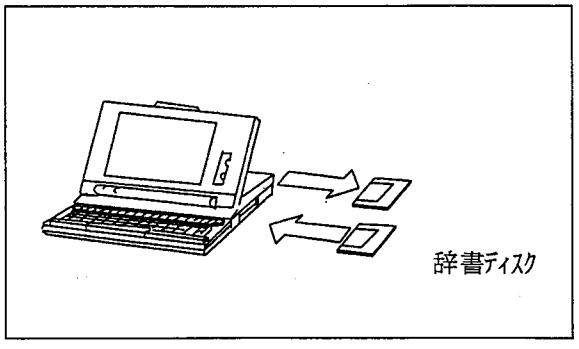
ＦＰ－ＢＡＳＩＣが起動され、以下のような画面が 表示されます。



ＦＰ－ＢＡＳＩＣが起動したら、フロッピードライブから実行用ディスクを取り出してください。

■かな漢字変換を使用する場合

かな漢字変換を使用する場合は、辞書ディスクをセットします。



ポイント

●実行用ディスクの内容がＲＡＭドライブにコピーされ、ＲＡＭドライブからFP-BASIC編集ソフトが起動します。

注意

●辞書ディスクについては、「付録A」をお読みください。

1-3 初期化とＯＮ・ＯＦＦ命令

●ＯＮ
出力をオンします。
例：ON Y_&M10
└──オンする出力番号

●ＯＦＦ
出力をオフします。
例：OFF Y_&M10
└──オフする出力番号

●ＶＥＲＩＮＩＴ
ＣＰＵのメモリをクリアし、ユニットの接続状態を読み込みます。

1 ＣＰＵの初期化

プログラムを始める前にＰＣのＣＰＵのすべてのメモリをクリアし、マザーボードに装着されているユニットの種類をＣＰＵに読み込ませなければなりません。これには、ＶＥＲＩＮＩＴコマンドを実行します。

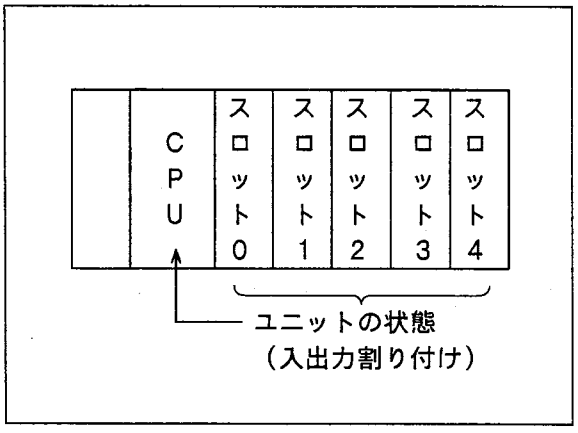
>VERINIT↵ ……入力
命令の入力後必ず
↵キーを押します

└──>はプロンプトと呼びます。
キー入力受け付け可能な状態を表わします。
この後に命令を入力します。

↵キーを入力すると、その行に入力された命令が実行されます。

ＶＥＲＩＮＩＴの説明

●VERINITは、ＣＰＵのすべてのメモリをクリアし、現在装着されているユニットの状態に基づいて、入出力割り付けを初期化します。



●装着ユニットは次の通りです。
スロット0･･･16点入力ユニット
スロット1･･･16点出力ユニット
（スロット番号は、CPUの隣から0、1、2･･）

●データメモリ等のクリアを行わずに、入出力割り付けだけを行う場合は、SLOTCLRを使用します。

2 ＯＮ・ＯＦＦ命令の実行

①まず、ＯＮ命令を入力してみます。

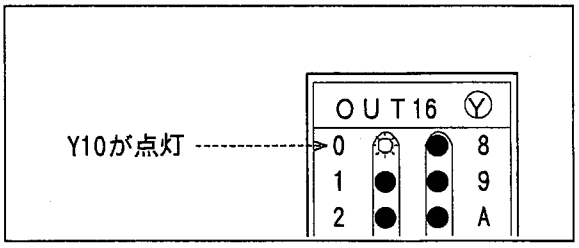
```
>ON Y_&M10↵
```

入力するキーは次の通りです。入力時には、カナのロックを解除してください。大文字の入力には CAPS キーを押します。

0 N スﾍﾟｰｽ Y SHIFT + ロ
（同時に押す）

SHIFT + & M 1 0 ↵
（同時に押す）

ＯＮ命令を実行すると、出力ユニットのY10がＯＮします（表示ＬＥＤが点灯します）。



ＯＮの説明

●ONは、出力を１点単位でオンします。この場合、OFFが実行されるまで、オン状態が保持されます。

●オンする出力の入出力番号については、P.12をご覧ください。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

注意

●Y_&M10の「_」はアンダースコアと読みます。キーボードからの入力は、SHIFT キーを押しながらロキーを押します。ロキーはカタカナの「ロ」のキーです。Ｐ.15参照。

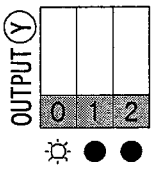
●ＣＰＵのモード設定スイッチが「ＲＥＭＯＴＥ」になっていて、「ＰＲＯＧ.」ＬＥＤが点灯していることを確認してください。

●&Mは、松下電工のＰＣの入出力番号に用いる数値表記の方法で、１の位は16進、10の位以上が10進になります。

例：&M120…16進
　　…10進

10進数で「Y_16」と表記することもできますが、「Y_&M10」と表記したほうが入出力ユニットとの対応が理解しやすくなります。

ＦＰ１の場合

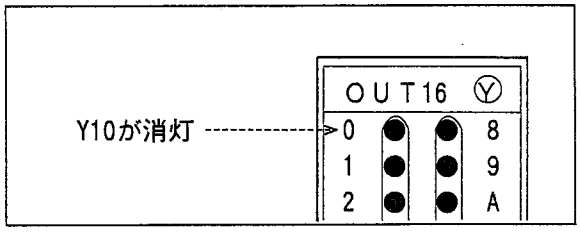


②つぎに、ＯＦＦ命令を入力します。

```
>OFF  Y_&M10↵
```

[O][F][F] [ｽﾍﾟｰｽ] [Y] [SHIFT]＋[ロ]
（同時に押す）
[SHIFT]＋[&][M][1][0] ↵
（同時に押す）

ＯＦＦ命令を実行すると、ＰＣの出力ユニットのY10がＯＦＦします（表示ＬＥＤが消灯します）。



カーソルを再び「ON Y_&M10」の行に移動させて、↵ キーを押すと、ＰＣの出力ユニットのY10が点灯します。

```
>ON Y_&M10↵ ……この行にカーソルを移
>OFF Y_&M10          動させて↵ を押す
```

ＯＮの説明

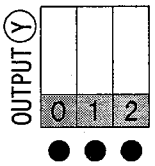
●OFFは、出力を１点単位でオフします。この場合、ONが実行されるまで、オフ状態が保持されます。

●オフする出力の入出力番号については、Ｐ.12をご覧ください。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

ＦＰ１の場合



参考 直接実行・間接実行

●以上のように、FP-BASIC編集ソフトが起動されているパソコンでＯＮ・ＯＦＦ命令などを実行することにより、ＰＣを直接制御することができます。このような命令の使用方法を、直接実行と呼びます。直接実行は、ＦＰ－ＢＡＳＩＣのプロンプト（「>」）の後に命令を入力して↵キーを押します。

●これに対して、ＯＮ・ＯＦＦ命令などをプログラム中に記述することによって実行することを、間接実行と呼びます。

3 ＦＰ－ＢＡＳＩＣを終了し、ＭＳ－ＤＩＳに戻りたい時は（通常不要）

ＥＸＩＴコマンドを入力することにより、ＦＰ－ＢＡＳＩＣは終了します。

```
>EXIT↵……入力
Sure (Y/N) : Y……プログラム編集中の
                 場合に表示されるの
                 で、「Y」を入力
```

＊EXITは[SHIFT]+[f･10]の押下で入力できます。

ＦＰ－ＢＡＳＩＣが終了すると、パソコンの画面にＭＳ－ＤＯＳのプロンプト「A>」が表示されます。

```
A>
```

ＭＳ－ＤＯＳのプロンプト「A>」が表示されてから、フロッピーディスクを取り出して、パソコンの電源を切ってください。最後に、ＰＣの電源を切ります。

注意

●FP-BASIC編集ソフトはＲＡＭドライブから起動されますのでプログラムもＲＡＭドライブに保存されます。

●ＲＡＭドライブに作成したドライブのプログラムを実行用デスクにコピーするには、９８ノートメニューを起動して、「３．ＲＭドライブ→ＦＤコピー」を実行してください。詳しくは「付録Ａ」をお読みください。

●再度ＦＰ－ＢＡＳＩＣを起動する場合、次の操作をします。

(1)FP3H-BASICの場合、ＭＳ－ＤＯＳのプロンプトの後に、「FPB /9↵」と入力します。

```
A>FPB /9↵
```

(2)FP3-BASIC（64Kバイトタイプ）およびFP1-BASICの場合、ＭＳ－ＤＯＳのプロンプトの後に、「FPB /9 /H↵」と入力します。

```
A>FPB /9 /H↵
```

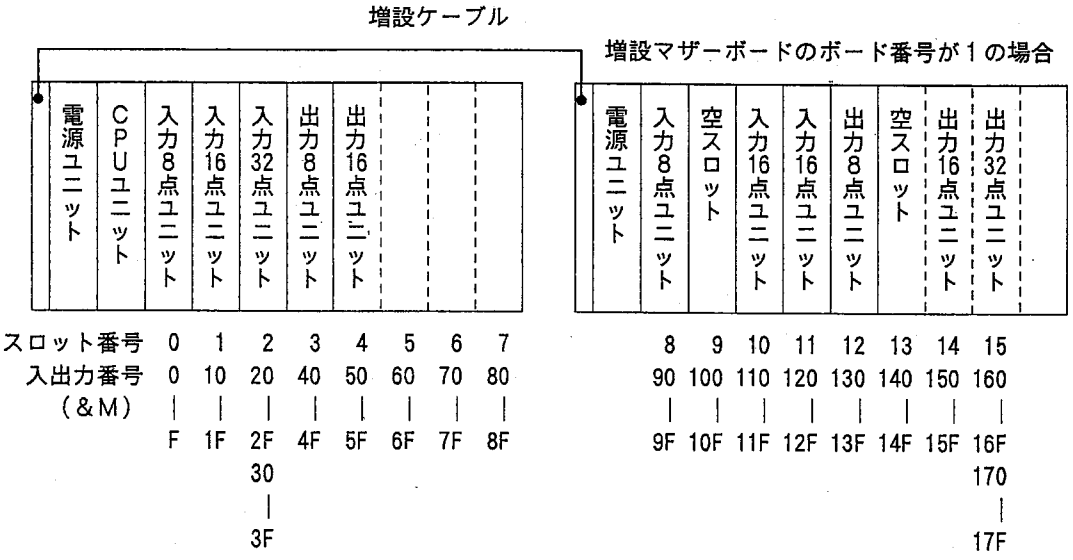
参考１

入出力番号の割り付け方

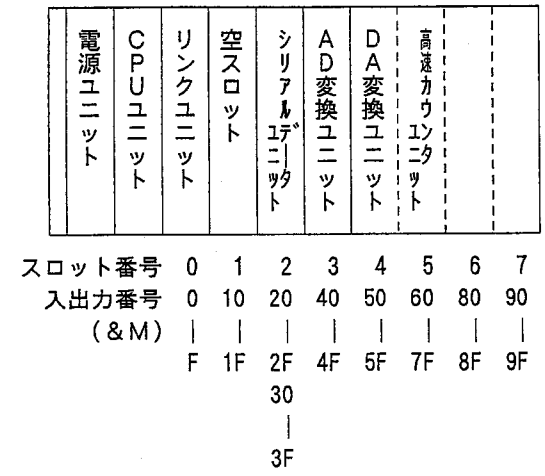
ＶＥＲＩＮＩＴコマンドまたはＳＬＯＴＣＬＲコマンドを実行すると、ＣＰＵはマザーボードのスロットに実装 されているユニットの状態を読み取り、入出力番号（I/O番号）を割り付けます。

①８点ユニットには16点分の入出力番号が割り付けられます。
②空きスロットには16点分の入出力番号が割り付けられます。
③リンクユニットなど入出力に関係のないユニットには、16点分の入出力番号が割り付けられます。
④内部的に入出力番号を持つユニットには、そのユニットが占有する入出力番号が割り付けられます。
＊ユニットが占有する入出力番号については「付録-49」をお読みください。

■通常ユニットの場合



■高機能ユニットの場合



注意

・ＣＰＵのモード設定スイッチが「ＲＥＭＯＴＥ」になっていて、「ＰＲＯＧ.」ＬＥＤが点灯していることを確認してください。
・&Mは松下電工のＰＣの入出力番号に用いる数値表記方法で、１の位は16進数、10の位以上は10進数で表記します。

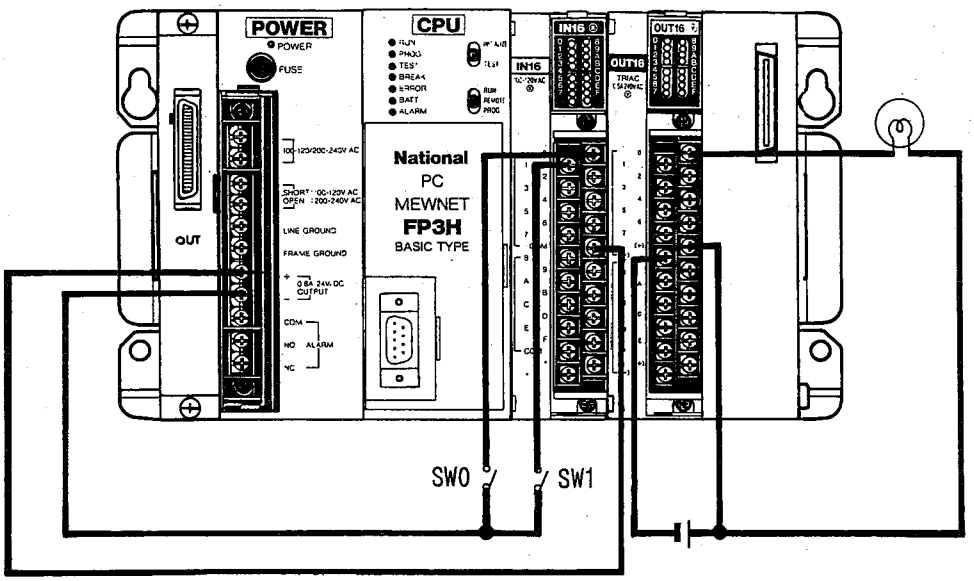
例：&M120
10進数　16進数表記

参考２

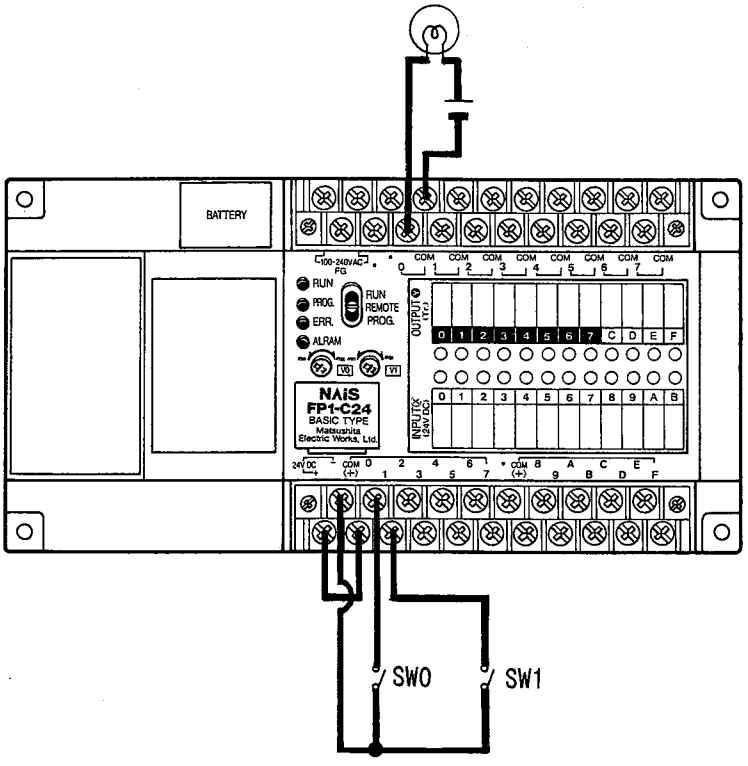
入出力の結線方法

点灯スイッチ（X0）と消灯スイッチ（X1）、およびランプ（Y0）の接続は、以下のとおりです。

ＦＰ３　ＢＡＳＩＣ



ＦＰ１　ＢＡＳＩＣ

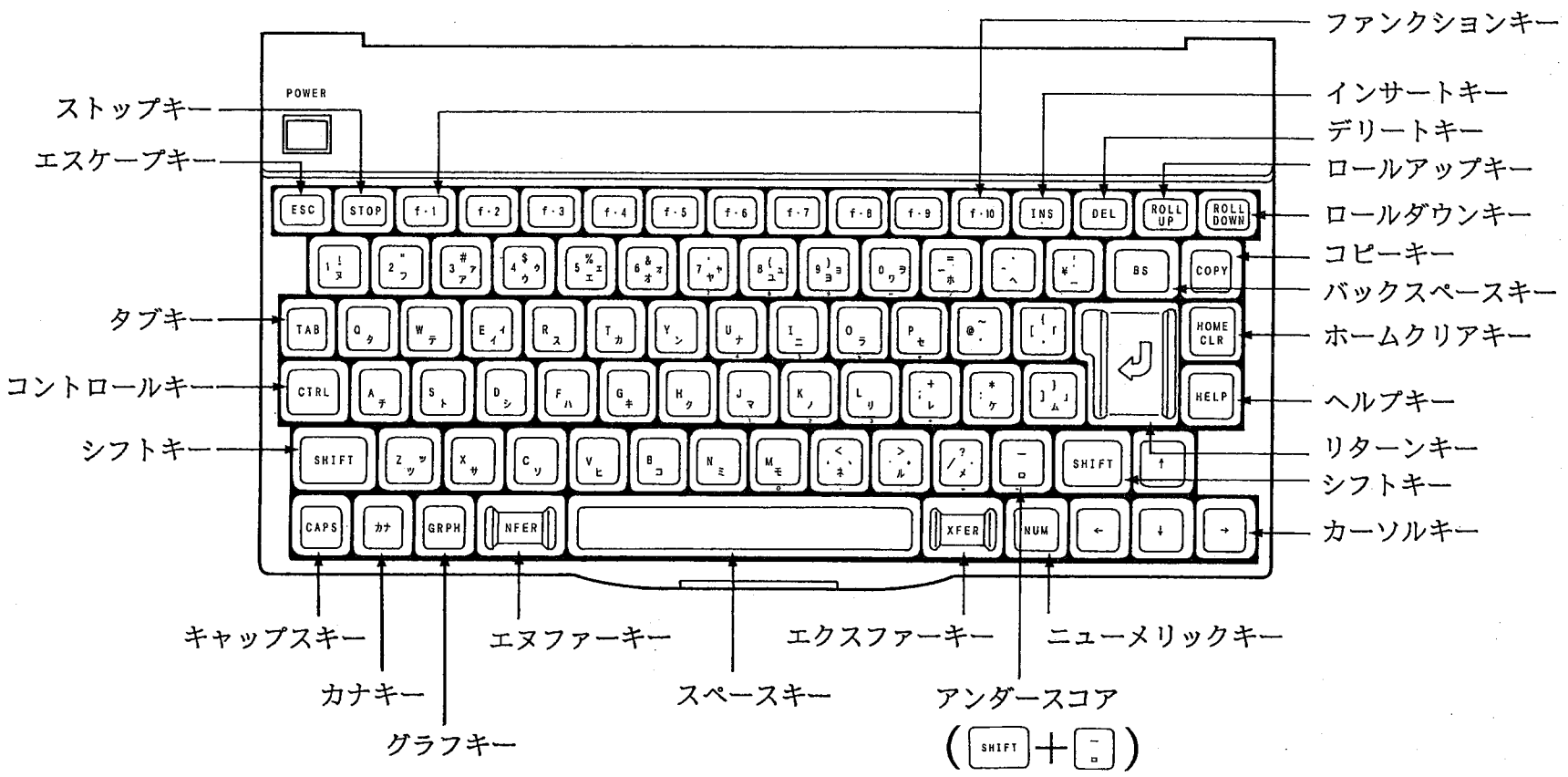


参考３

キー操作の概要

FP-BASIC編集ソフトで使用する基本的なキー操作についてまとめてあります。

キー	キーの名称	機能
[↑] [↓] [←] [→]	カーソルキー	カーソルが上下左右に１文字分移動します。 文字が表示されていない位置には移動しない（行頭の一桁を除く）。 最上行・最下行で [↑] [↓] を押すと画面がスクロールする。
[ROLL UP] [ROLL DWN]	スクロールキー	カーソルが最上行・最下行に移動し、画面がスクロールする。
[HOME CLR]	ホームクリアキー	表示画面を消去する（プログラムは消去されません）
[TAB]	タブキー	カーソルを８桁右に移動する
[INS]	インサートキー	カーソル位置に空白（スペース）を１文字挿入する
[DEL]	デリートキー	カーソルの左隣の１文字を消去し、その位置にカーソルを移動する
[BS]	バックスペースキー	カーソルを左に１文字分移動する（そこにあった文字は消去しない）
[STOP]	ストップキー	実行中のプログラムを停止しＰＣを初期化する（データも初期化される） 停止中の出力の保持・非保持等の状態はＣＰＵの設定により異なる


ファンクションキー
インサートキー
デリートキー
ロールアップキー
ロールダウンキー
コピーキー
バックスペースキー
ホームクリアキー
ヘルプキー
リターンキー
シフトキー
カーソルキー
ストップキー
エスケープキー
タブキー
コントロールキー
シフトキー
キャップスキー
カナキー
グラフキー
エヌファーキー
スペースキー
エクスファーキー
アンダースコア
(SHIFT+-)
ニューメリックキー
POWER
ESC
STOP
f・1
f・2
f・3
f・4
f・5
f・6
f・7
f・8
f・9
f・10
INS
DEL
ROLL UP
ROLL DOWN
BS
COPY
TAB
HOME CLR
CTRL
HELP
SHIFT
SHIFT
CAPS
カナ
GRPH
NFER
XFER
NUM

1-4 ＯＮ・ＯＦＦ・ＷＡＩＴ　ＳＷ（）を使った簡単な制御

●ＦＵＮＣＴＩＯＮ　～　ＦＥＮＤ

FUNCTION <ファイル名>

FEND

……を１本のプログラムとして宣言します。

●ＷＡＩＴ　ＳＷ（）

入力がオンまたはオフされるのを待ちます。

例：WAIT SW(X_&M0)=1

……入力X0がオンされるのを待ちます。

例：WAIT SW(X_&M0)=0

……入力X0がオフされるのを待ちます。

●ＯＮ

出力をオンします。

例：ON Y_&M10

……出力Y10をオンします。

●ＯＦＦ

出力をオフします。

例：OFF Y_&M10

……出力Y10をオフします。

●ＮＥＷ

BASIC編集ソフトのプログラムを消去します。

●ＬＩＳＴ

入力したプログラムを確認します。

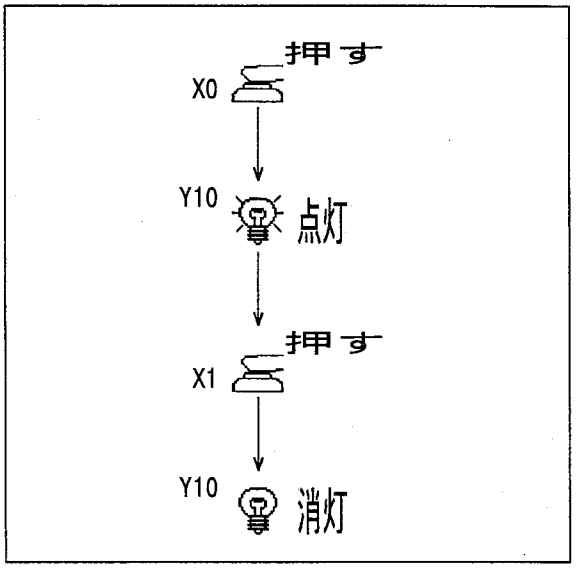
●ＳＡＶＥ

メモリに記憶されているプログラムをディスク上に保存します。

スイッチ２個による簡単なランプの点灯制御をＦＰ－ＢＡＳＩＣでプログラムしてみます。

1 制御内容を決める

点灯スイッチ(X0)を押すとランプが点灯し、次に消灯スイッチ(X1)を押すとランプが消灯するという、簡単なランプ制御を考えてみましょう。

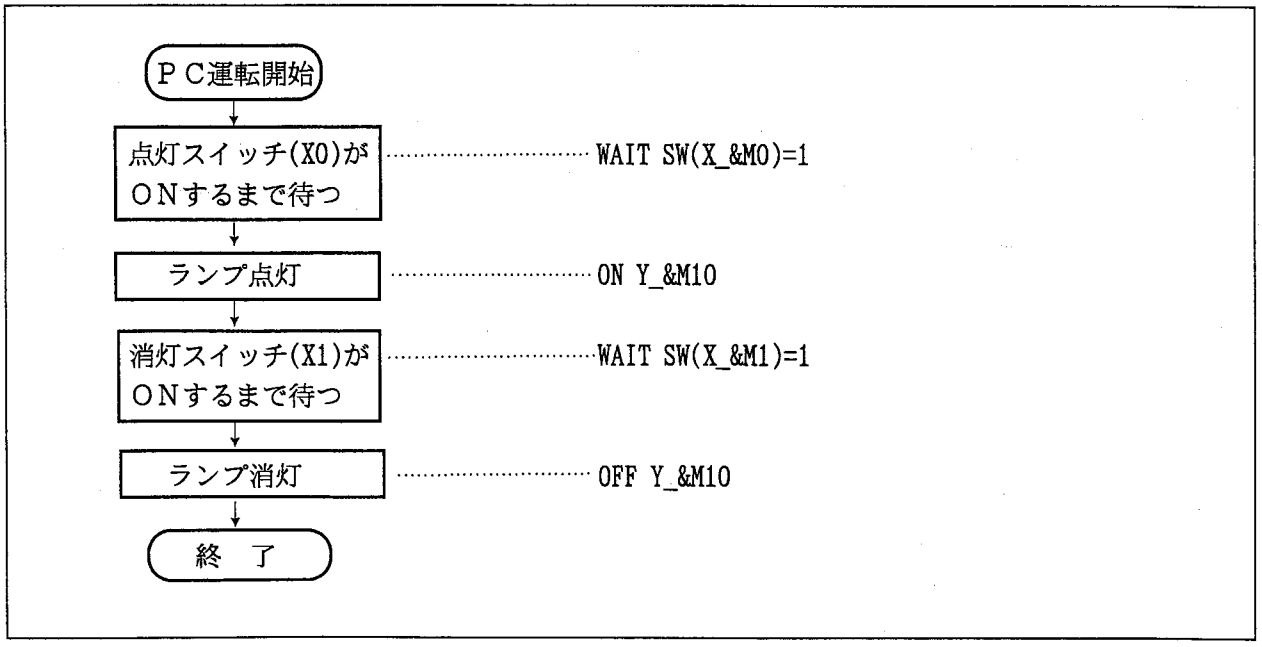


ＷＡＩＴの説明

●WAIT SW(X_&M0)=1は、入力X0がオンされるのを待ちます。
●WAIT SW(X_&M0)=0は、入力X0がオフされるのを待ちます。
●WAIT AW(X_&M0)=1の「=1」は省略することができます。
●ON Y_&M10は、出力Y10をオンします。Y10のオン状態は、次にOFF命令が実行されるまで保持されます。
●OFF Y_&M10は、出力Y10をオフします。Y10のオフ状態は、次にON命令が実行されるまで保持されます。

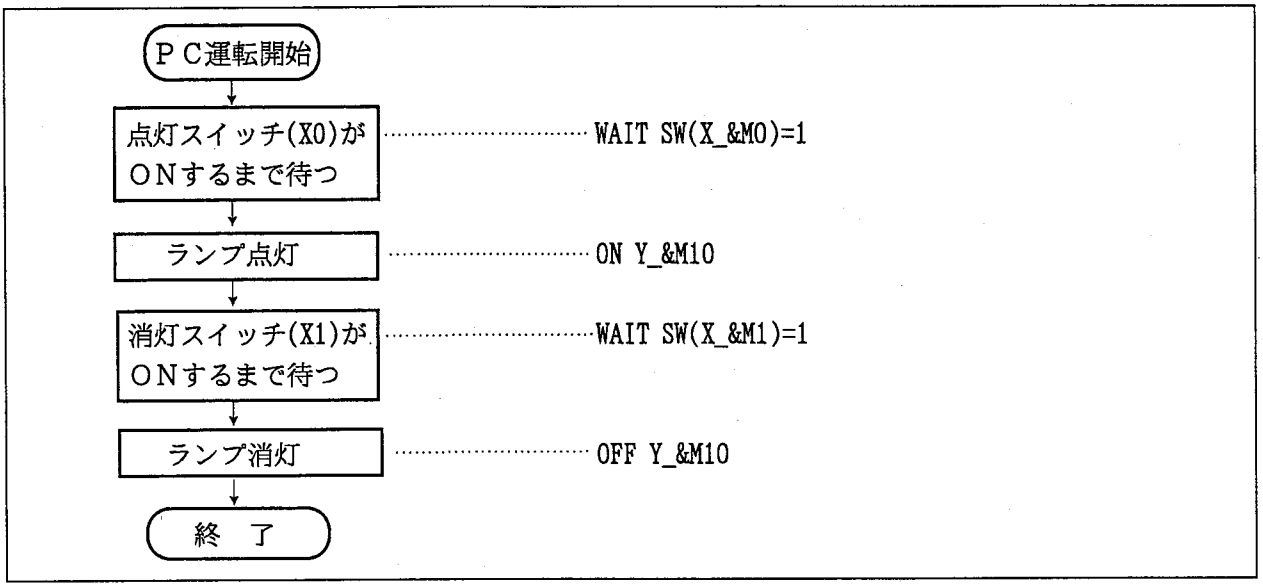
2 制御の内容をフローチャートにする

制御の流れをフローチャートに表す。



3 フローチャートからプログラムを作成

フローチャートに従い、機械の動きを命令に置き換えプログラムを作成します。



4 プログラムの入力手順

ステップ１ メモリのクリア

①プログラムを入力する前に、ＰＣのＣＰＵのメモリとFP-BASIC編集ソフトのプログラムメモリをクリアします。

```
>VERINIT↵ ..... 入力
>NEW↵ .....
```

ステップ２ プログラムの始まりは ＦＵＮＣＴＩＯＮから

①プログラムの最初の一行は「10 FUNCTION MAIN」と入力し、↵キーを押します。プログラムは必ず「ＦＵＮＣＴＩＯＮ」で始めます。「FUNCTION」はプログラムの 始まりを示すものです。「10」は行番号といいます。ＦＰ－ＢＡＳＩＣのプログラムは、行番号の小さい順に実 行されます。

```
>VERINIT
>NEW
>10 FUNCTION MAIN↵ ..... 入力
```

ＶＥＲＩＮＩＴの説明

●VERINITは、ＣＰＵのメモリをクリアし、現在のユニット状態に合わせて入出力割り付けを行います。

ＮＥＷの説明

●NEWは、BASIC編集ソフトのプログラムを消去します。プログラムを初めて作成する時は必ず実行してください。

ＦＵＮＣＴＩＯＮの説明

●「FUNCTION」の後の「MAIN」はプログラムの名前です。「SAMPLE」や「TEST1」など8文字以内ならＯＫですが、「P」で始まる名前は使用できません。

②フローチャートの命令に置き替えてプログラムを作成していきます。行の先頭に行番号を付けるのを忘れないようにしてください。ここでは、行番号を「10」ずつ増していきます。また、一行入力したら、必ず↵キーを押します。

```
>VERINIT
>NEW
>10 FUNCTION MAIN
>20 WAIT SW(X_&M0)↵ ……
>30 ON Y_&M10↵ ……入力
>40 WAIT SW(X_&M1)↵
>50 OFF Y_&M10↵ ……
```

③プログラムの最後の行に、「FEND」を入力し、↵キーを押します。「FEND」はプログラムをしめくくる役目を持ち、これにより「FUNCTION」から「FEND」までを１本のプログラムとして宣言します。

```
>VERINIT
>NEW
>10 FUNCTION MAIN
>20 WAIT SW(X_&M0)
>30 ON Y_&M10
>40 WAIT SW(X_&M1)
>50 OFF Y_&M10
>60 FEND↵ ……入力
```

■入力したプログラムの確認には ＬＩＳＴを入力

入力したプログラムを確認するにはＬＩＳＴコマンドを実行します。またＦＰ－ＢＡＳＩＣでは、↑キーを押すことにより、画面の外に消えてしまった 表示をさかのぼって見ることができます。もちろん、↓キーを押すことによってもとの入力行に戻ることができます。

```
>LIST↵（またはF4）……入力
10 FUNCTION MAIN ……
20 WAIT SW(X_&M0)
30 ON Y_&M10 ……入力されたプログ
40 WAIT SW(X_&M1)     ラムが表示される
50 OFF Y_&M10
60 FEND ……
```

＊LISTは f･4 キーの押下で入力できます。

注意　必ず↵

●１行入力したら必ず↵キーを押してください。↵キーを押さないと、画面に表示されていても、プログラムメモリに記憶されません。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

ＬＩＳＴの説明

●LIST↵とすれば、プログラムを全て表示することができます。

●LIST 30-50↵とすれば、30行目から50行目まで表示できます。

注意

●「LIST」の行には、これら以外の文字がないことを確認して実行してください。「LIST」の後に何か文字があると、１行分の命令全てを実行しますので変な動きをします。HOME CLR キーで表示を消してから、「LIST↵」を実行するのが安全です。

●画面表示を消去するには、HOME CLR キーを押すか、「CLS↵」を入力します。画面は消去されますが、記憶されているプログラムは消去されません。

■プログラムの修正には
上書きと↵キー入力

たとえば、「10 FUNCTION MAIN」とするところを「10 FANCTION MAIN」としてしまったとします。 この場合、[↑][↓][←][→]キーで、間違い個所の「A」の上にカーソルを移動させ、「U」と入力して上書き訂正した後、↵キーを押します。 これで、プログラムは正しく修正されます。

①

```
>10 FANCTION MAIN·····修正個所に
>20 WAIT SW(X_&M0)      カーソルを移動
```

②

```
>10 FUNCTION MAIN ·····修正入力を行い、
>20 WAIT SW(X_&M0)      ↵キーを押す
>30 ON Y_&M10
```

③

```
>10 FUNCTION MAIN
>20 WAIT SW(X_&M0) ·····カーソルは次の
>30 ON Y_&M10           行の先頭に移動
                        して、修正終了
```

■プログラムの保存

FP-BASIC編集ソフトでは、プログラムをディスクに保存することができます。プログラムを保存するには、ＳＡＶＥコマンドを使います。たとえば、「SAVE "A:TEST1"」を実行すると、ディスク上にTEST1.PRGが保存されます。

```
>SAVE "A:TEST1"↵·····入力
Over Write(Y/N) :Y·····同名ファイルがある
                       場合に表示されるの
                       で「Y」を入力する
```

＊SAVE "は[SHIFT]＋[f･1]キーの押下で入力できます。

注意 必ず↵

●「A」を「U」に訂正しても、↵キーを押さないと、表示は変わっても、メモリに記憶されているプログラムは書き変わりません。

ＳＡＶＥの説明

●SAVEは、メモリに記憶されているプログラムをディスク上に保存します。

●SAVE "A:TEST1"は、プログラム保存先を指定します。TEST1はファイル名といい、プログラムをTEST1というファイル名で保存します。

●保存先のドライブは、ここでは９８ＮＯＴＥのＲＡＭドライブになります（ＲＡＭドライブが起動ドライブ＝ドライブＡになります）。フロッピーディスクドライブなど任意のドライブに保存する場合は、ファイル名の前に、「B:」のようにドライブ名を付けます。ドライブAの場合に限り、「A:」が省略できます。

1-5 プログラムのＰＣへの転送

●ＣＯＭＰＩＬＥ
　入力したプログラムを実行形式のプログラムに変換します。
●ＤＷＮＬＤ
　実行形式のプログラムをＰＣへ転送します。

1 実行プログラムへの変換

入力したプログラムをＣＯＭＰＩＬＥコマンドで、実行プログラムに変換します。

```
>COMPILE↵ (またはF6)……入力する
COMPILE END ……コンパイル正常終了の表示
```

＊COMPILEは f·6 キーの押下で入力できます。

2 プログラムの転送

コンパイルした実行プログラムをＤＷＮＬＤコマンドでＰＣに転送します。

```
>DWNLD↵ (またはF7)…… 入力する
```

＊DWNLDは f·7 キーの押下で入力できます。

ＣＯＭＰＩＬＥの説明

●COMPILEは、表示されているプログラムをＰＣが実行できる形式に変換します。この作業をしないと、ＰＣに転送しても実行できません。

ＤＷＮＬＤの説明

●DWNLDは、実行形式のプログラムをＰＣに転送します。

注意

●ＰＣがＰＲＯＧ．状態でないとＤＷＮＬＤはできません。
●プログラム転送時には、ＰＣのＣＰＵのモード選択スイッチが「ＲＥＭＯＴＥ」に設定されていて、「ＰＲＯＧ．」ＬＥＤが点灯していることを確認してください。
●ＰＣがＲＵＮ中の場合、f·10 (QUIT A) を押してください。f·10 (QUIT A) → f·6 (COMPILE) → f·7 (DWNLD) の順で実行すると間違いがありません。
●DWNLD、QUIT等がうまく実行されない時は、STOP キーを押してリセットしてみてください。

参考 ＵＰＬＤ

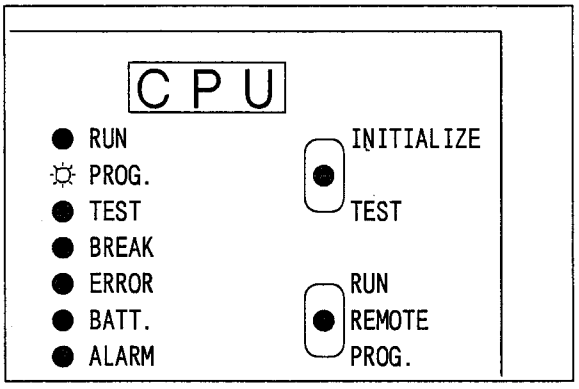
●ＰＣのプログラムをFP-BASIC編集ソフトに読み込むには、「UPLD」を実行します。このとき、「&M」で指定した入出力番号は全て10進数に変換され、ラベル、コメントがなくなったり、変数の宣言命令がなくなり、文法エラーになることがあります。

1-6 プログラムの実行

・ＲＥＭＯＴＥモード実行
　ＣＰＵを「ＲＥＭＯＴＥ」モードの場合、パソコンからＸＱＴコマンドで実行を指示します。
・ＸＱＴ
　プログラムの実行を指示します。
・ＲＵＮモード実行
　ＣＰＵの動作モードスイッチを「ＲＵＮ」に設定します。

(1) ＲＥＭＯＴＥモード実行

ＣＰＵの動作モード設定スイッチを「ＲＥＭＯＴＥ」モードに設定し、ＸＱＴコマンドを実行します。

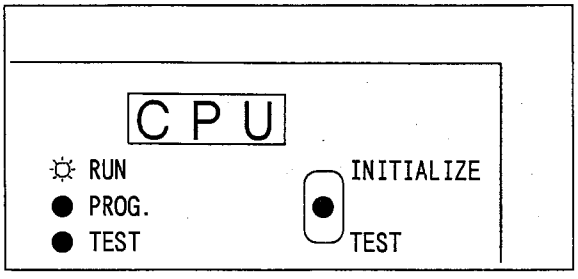


①ＸＱＴコマンドを入力します。

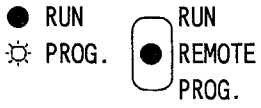
>XQT↵（またはF5）…… 入力

＊XQTは f･5 キーの押下で入力できます。

②ＣＰＵが運転を開始します。



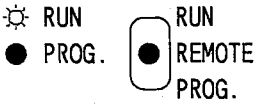
ＦＰ１の場合



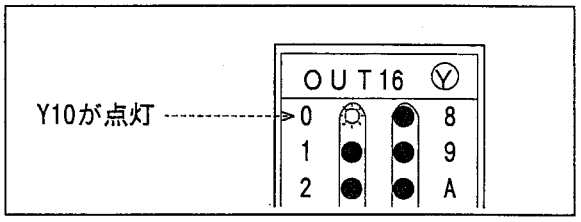
ＸＱＴの説明

●XQTはプログラムを実行させる命令です。
●ＲＥＭＯＴＥモードでXQTとすれば、プログラムが実行されます。
●プログラムの中にも使えます。ＸＱＴ!1,MAIN,30-とすれば、MAINのプログラムの30目から実行させられます。

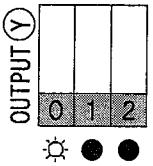
ＦＰ１の場合



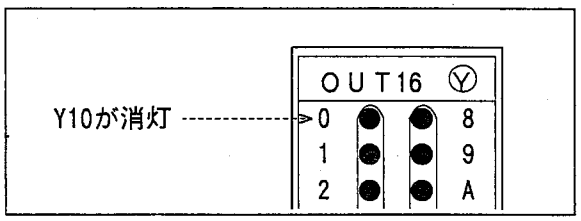
③スイッチ(X0)をオンすると、ＰＣの出力ユニットのY10がオンします（表示ＬＥＤが点灯します）。



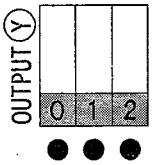
ＦＰ１の場合



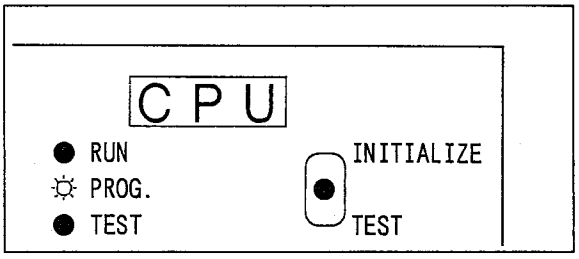
④スイッチ(X1)をオンすると、ＰＣの出力ユニットのY10がオフします（表示ＬＥＤが消灯します）。



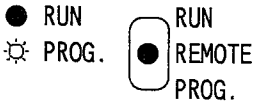
ＦＰ１の場合



⑤ＣＰＵの停止を確認します。ＣＰＵの「ＰＲＯＧ．」ＬＥＤが点灯します。

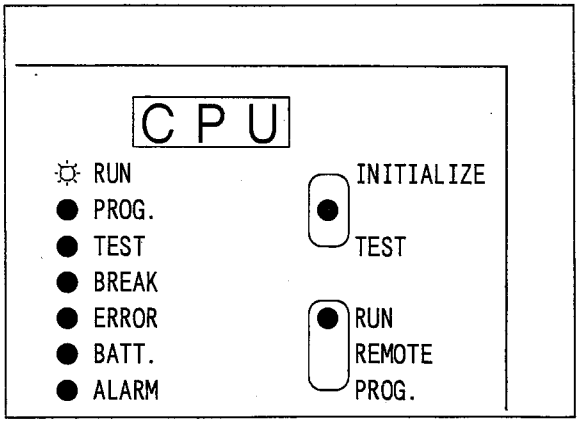


ＦＰ１の場合



(2) ＲＵＮモード実行

ＣＰＵの動作モードスイッチを「ＲＵＮ」に設定して、プログラムを実行します。



注意

●ＲＵＮモード実行では、ＣＰＵとパソコンとの接続は必要ありません。通常の稼働時は、ＲＵＮモードとしておき、電源オンと同時に実行開始します。

●ＲＵＮモード実行の場合は、プログラム終了後ＣＰＵの動作モードスイッチを「ＰＲＯＧ．」に倒しＰＣを停止させてください。

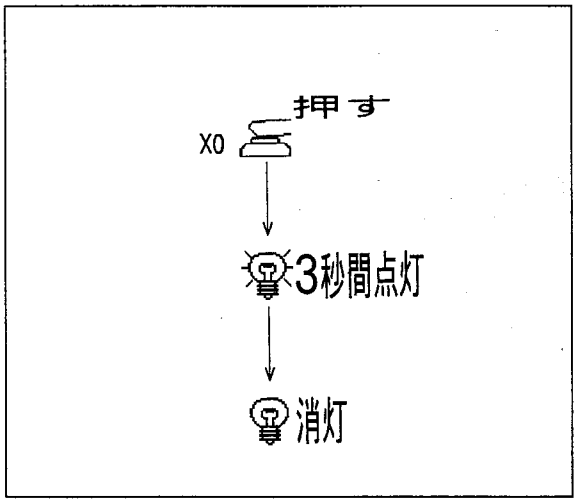
ＦＰ１の場合

1-7 ＷＡＩＴを使ったタイマ制御

●ＷＡＩＴ　＜秒数＞
　指定秒数だけ待ちます。
　例：WAIT 3.00
　　└──３秒待つ

1 タイマを使った点滅制御

前回「1-4」で作成したプログラムを変更して、点灯スイッチ(X0)を押すとランプが点灯し、３秒たって自動的に消灯するプログラムを、ＷＡＩＴ命令を使い作成してみましょう。

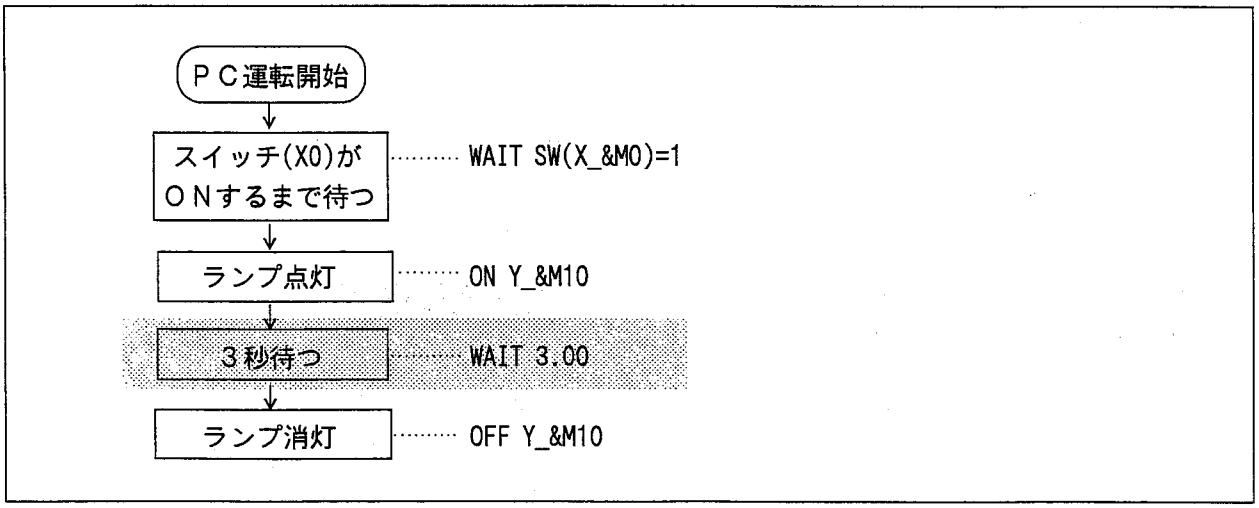


ＷＡＩＴの説明

●「WAIT」は、WAITの後に数値を書けば、タイマ命令になります。「WAIT 3.0」だと、３秒待ってから次の行を実行します。

●WAITの後の〈秒数〉には、0.01～32765の数値を指定できます。

2 制御の流れ（フローチャート）



3 プログラムの入力

「2－1」で作成したプログラムをＬＩＳＴコマンドで表示させ、40行目を修正して、次のようなプログラムを作成します。

```
>LIST↵
10 FUNCTION MAIN
20 WAIT SW(X_&M0)
30 ON Y_&M10
40 WAIT 3.0↵ ……この行を修正し、
50 OFF Y_&M10 ↵キーを押す
60 FEND
```

4 プログラムの転送

入力したプログラム（ソースプログラム）をＣＯＭＰＩＬＥコマンドで実行プログラムに変換し、ＤＷＮＬＤコマンドでＰＣに転送します。

```
>COMPILE↵ ……入力
COMPILE END
>DWNLD↵ ……入力
```

＊COMPILE・DWNLDは、 f·6 キー・ f·7 キーの押下で各々入力できます。

ポイント

●ＬＩＳＴコマンドで表示させたプログラムの40行目を修正入力して、↵キーを押します。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

注意

●プログラム転送時には、ＣＰＵのモードスイッチが「ＲＥＭＯＴＥ」に設定されていて、「ＰＲＯＧ.」ＬＥＤが点灯していることを確認してください。

5 プログラムの実行

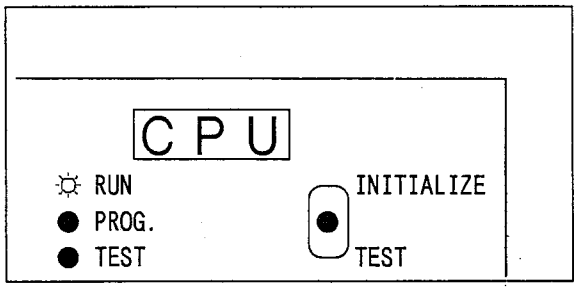
ここでは、ＣＰＵを「ＲＥＭＯＴＥ」モードに設定し、FP-BASIC編集ソフト（パソコン）からＸＱＴコマンドを実行します（リモート実行）。

①ＸＱＴコマンドの実行

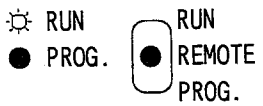
>XQT↵（またはF5）…… 入力する

＊XQTは、f·5キーの押下で入力できます。

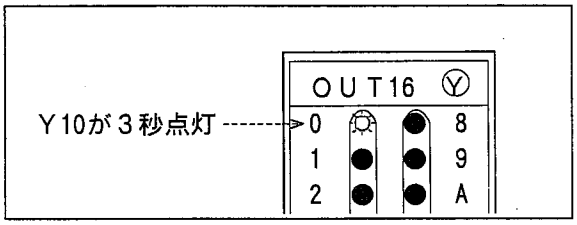
②ＰＣの運転（ＲＵＮ）を確認します。ＣＰＵの「ＲＵＮ」ＬＥＤが点灯します。



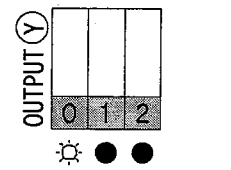
ＦＰ１の場合



③スイッチ(X0)をＯＮすると、ＰＣの出力ユニットのY10が３秒間ＯＮします（表示ＬＥＤが３秒間点灯します）。



ＦＰ１の場合



1-8 ＧＯＴＯを使った繰り返し制御

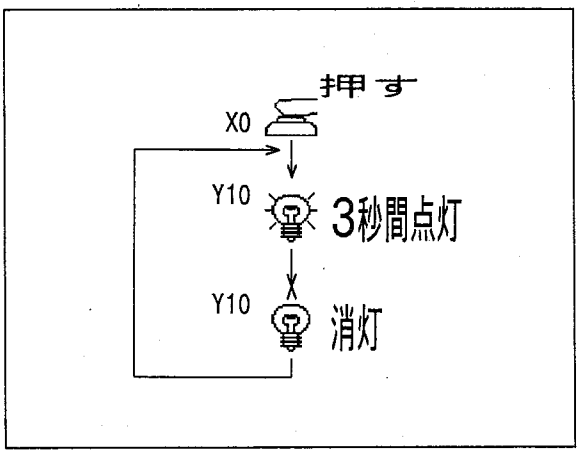
●ＧＯＴＯ
　行番号を指定することにより次に実行する行を指定します。
　例：GOTO 10
　　└──プログラムの実行を１０行目に移す

●ＱＵＩＴ
　プログラムの停止を指示します。

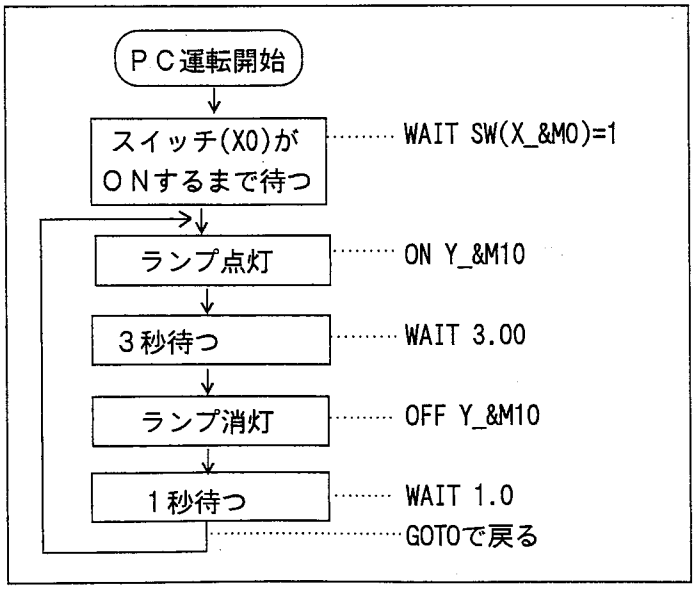
●ＲＥＮＵＭ
　行番号を並びかえます。

1 繰り返し実行される点滅制御

前回「1-7」のプログラムにＧＯＴＯ命令を追加し、点灯スイッチ(X0)を押すとランプが３秒間点灯し、１秒間消灯し、再度３秒間点灯を繰り返すプログラムを作ってみましょう。



2 制御の流れ（フローチャート）



ＧＯＴＯの説明

●GOTO命令は、次に実行するプログラムの行番号を指定することができます。
●GOTO 10と書けば、次に10行目を実行します。
●同じ動きを繰り返す時などに使われます。

ポイント

●ＧＯＴＯで次に実行する行番号を指定します。この場合、プログラムは最初に戻って繰り返しを続けます。

3 プログラムの入力

「1-7」で作成したプログラムをＬＩＳＴコマンドで表示させ、追加・修正を加えることにより、次のようなプログラムを作成します。

```
>LIST
10 FUNCTION MAIN
20 WAIT SW(X_&M0)
30 ON Y_&M10
40 WAIT 3.0
50 OFF Y_&M10
60 FEND
>52 WAIT 1.0↵ ..... この行を追加
>54 GOTO 20↵ .....
```

4 プログラムの転送

入力したプログラム（ソースプログラム）をＣＯＭＰＩＬＥコマンドで実行プログラムに変換し、ＤＷＮＬＤコマンドでＰＣに転送します。

```
>COMPILE↵ .....入力
COMPILE END
>DWNLD↵ .....入力
```

＊COMPILE・DWNLDは、 f·6 キー・ f·7 キーの押下で各々入力できます。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

ポイント

●ＬＩＳＴコマンドで表示させたプログラムに追加入力します。

●追加・修正後のプログラムをＬＩＳＴコマンドで表示すると、次のようになります。

```
>LIST↵
10 FUNCTION MAIN
20 WAIT SW(X_&M0)
30 ON Y_&M10
40 WAIT 3.0
50 OFF Y_&M10
52 WAIT 1.0
54 GOTO 20
60 FEND
```

ＲＥＮＵＭの説明

●RENUM命令は行番号をきれいに並び変える命令です。

●RENUMと入力すると、10行目以降を10とびにきれいに行番号を並びかえます。

●同じ動きを繰り返す時などに使われます。

```
10 FUNCTION MAIN
20 WAIT SW(X_&M0)
30 ON Y_&M10
40 WAIT 3.0
50 OFF Y_&M10
60 WAIT 1.0
70 GOTO 20
80 FEND
```

注意

●プログラム転送時には、ＣＰＵのモードスイッチが「ＲＥＭＯＴＥ」に設定されていて、「ＰＲＯＧ.」ＬＥＤが点灯していることを確認してください。

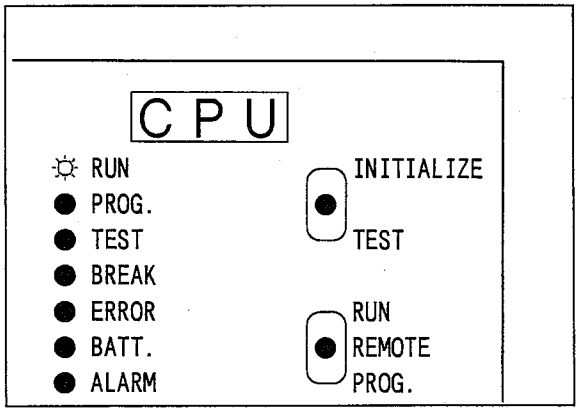
5 プログラムの実行

①ＣＰＵを「ＲＥＭＯＴＥ」モードに設定し、FP-BASIC編集ソフト（パソコン）からＸＱＴコマンドを実行します（リモート実行）。

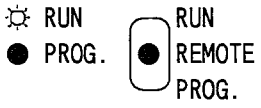
>XQT↵（またはF5）…… 入力する

＊XQTは f·5 キーの押下で入力できます。

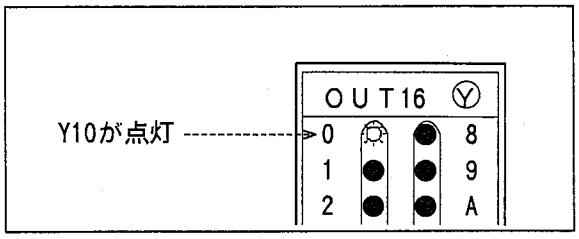
②ＰＣの運転（ＲＵＮ）を確認します。ＣＰＵの「ＲＵＮ」ＬＥＤが点灯します。



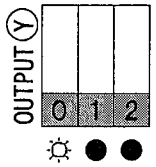
ＦＰ１の場合



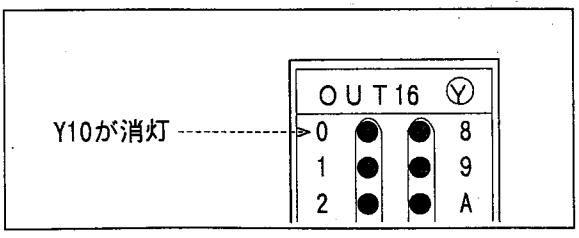
③点灯スイッチ(X0)をオンすると、ＰＣの出力ユニットのY10がオンします（表示ＬＥＤが点灯します）。



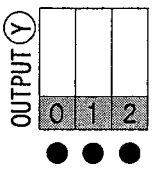
ＦＰ１の場合



④３秒後に自動的にY10が消灯します。



ＦＰ１の場合



③１秒後再びY10が点灯し、点滅を繰り返します。

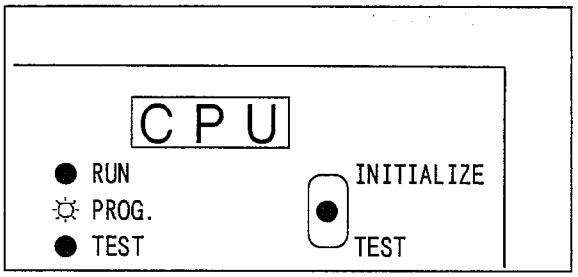
6 ＰＣの停止
（パソコンを使った停止）

①FP-BASIC編集ソフト（パソコン）から、ＱＵＩＴコマンドを実行してＣＰＵを停止します。

```
>QUIT ALL↵ (またはF10)……入力
```

＊QUIT ALLは f·10 キーの押下で入力できます。

②ＰＣの停止を確認します。ＣＰＵの「ＰＲＯＧ．」ＬＥＤが点灯します。

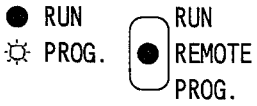


注意

●ＱＵＩＴコマンドを実行してＰＣを停止したとき、出力がオン状態に保持されることがあります(「付録Ｄ ３」参照)。

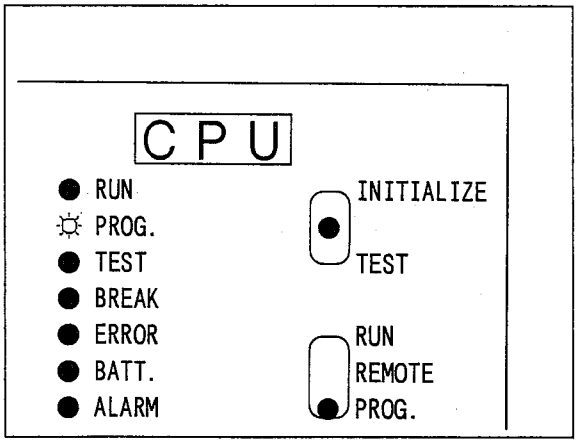
●ＰＣ停止状態で出力がオン状態の場合、STOP キーを押してオフにします。STOP キーを押すと、ＱＵＩＴコマンドが実行され、出力がオフします。

ＦＰ１の場合



■ＲＵＮモード実行時の運転停止
（パソコンを使わない停止）

「ＲＵＮ」モード運転時にＰＣを停止させるには、モード切り替えスイッチを「ＰＲＯＧ．」に切り替 えます。これにより、パソコンを使用せずにＰＣを 停止することができます。

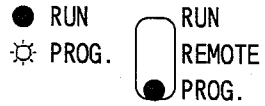


注意

●モード切り替えスイッチを「ＰＲＯＧ．」に切り替えてＰＣを停止したとき、出力がオン状態に保持されることがあります（「付録Ｄ ３」参照)。

●ＰＣ停止状態で出力がオン状態の場合、ＣＰＵのITIALIZE/TESTスイッチを出力ユニットのＬＥＤが消灯するまでINITIALIZE側に倒してください。

ＦＰ１の場合



課題(1)

●点灯スイッチ(X0)をＯＮすると、ランプが３秒点灯→１秒消灯→２秒点灯→１秒消灯してから、点灯スイッチ(X0)の再スタートを待つプログラムを作ってください。 (解答は37ページです)

1-9 ＩＦを使った条件分岐

●ＩＦ　<条件式>　ＴＨＥＮ　<命令文①>　ＥＬＳＥ　<命令文②>

IFの後に条件式を記述し、条件が一致すればTHEN直後の命令①を、
そうでなければELSE直後の命令②を、各々実行します。ＥＬＳＥ以降を
省略することができます。

例(1)：IF SW(X_&M0)=1 THEN ON Y_&M10 ELSE OFF Y_&M10

X_&M0がオンならY_&M10をオンし、それ以外の場合はY_&M10をオフする

例(2)：IF A=5 THEN GOTO 100 ELSE GOTO 200

A(変数)の値が5なら実行を100行目に移し、それ以外なら200行目に移す

注意：IF SW(X_&M0) AND A=1 THEN ～ は可能ですが、
変数

WAIT SW(X_&M0) AND A=1は使用できません。
ＷＡＩＴの場合は、WAIT SW(X_&M0) AND SW(X_&M1)のように
接点だけが許可されます。

(1)ＩＦを使用した動作モード切り替えプログラムの例

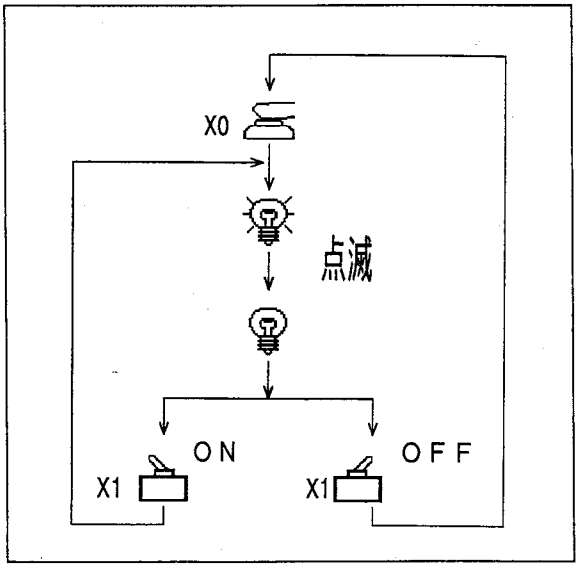
1 動作モード切り替えスイッチを使った制御

前回の「1-8」のプログラムに命令を追加し、動作モード切り替えスイッチを操作することにより、次のように動作するランプ点灯制御プログラムを考えてみましょう。

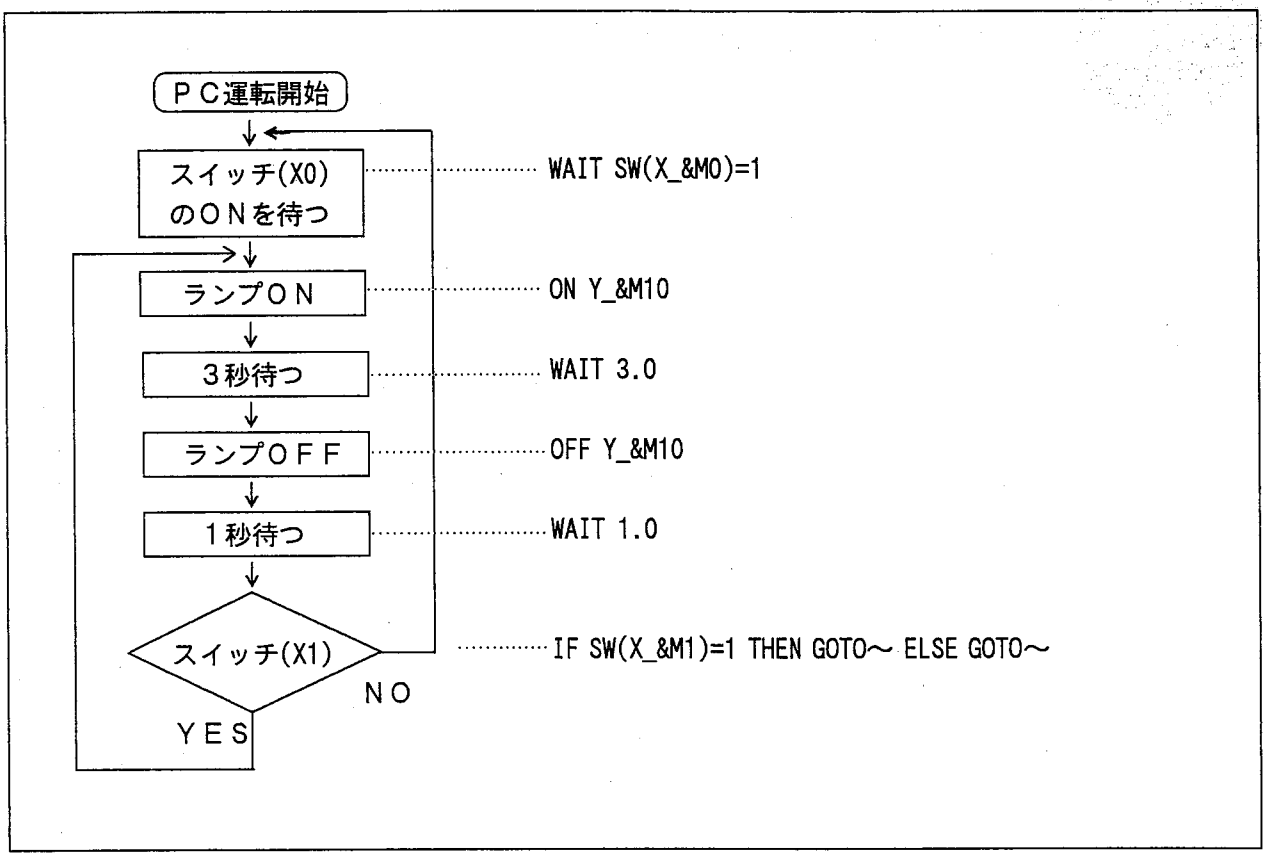
①〔連続運転〕スイッチ(X1)がＯＮ状態でランプは点滅を繰り返す。

②〔１サイクル運転〕スイッチ(X1)がＯＦＦ状態でランプは１回だけ点滅する。

＊動作モード切り替えスイッチ(X1)はオルタネイト（トグル）スイッチとします。



2 制御の流れ（フローチャート）



3 プログラムの入力

次のように70行目を変更します。

```
>LIST↵
10 FUNCTION MAIN
20 WAIT SW(X_&M0)=1
30 ON Y_&M10
40 WAIT 3.0
50 OFF Y_&M10
60 WAIT 1.0
70 IF SW(X_&M1)=1 THEN GOTO 30 ELSE GOTO 20↵
80 FEND
```

ＩＦの説明

●IFの後に条件式を記述し、条件が真ならTHEN直後の命令を、偽ならELSE直後の命令を、各々実行します。

●左の例では、SW(X_&M1)=1が成立すると30行目に実行を移し、それ以外の場合は20行目に実行を移します。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

4 プログラムの転送

入力したプログラム（ソースプログラム）をＣＯＭＰＩＬＥコマンドで実行プログラムに変換し、ＤＷＮＬＤコマンドでＰＣに転送します。

```
>COMPILE↵ (F6)…… 入力する
COMPILE END
>DWNLD↵  (F7)…… 入力する
```

＊COMPILE･DWNLDは、f･6キー・f･7キーの押下で各々入力できます。

5 プログラムの実行

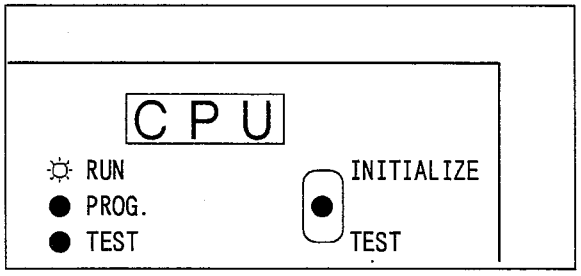
ここでは、ＣＰＵを「ＲＥＭＯＴＥ」モードに設定 し、FP-BASIC編集ソフト（パソコン）からＸＱＴコマンドを実行します(リモート実行)。

①ＸＱＴコマンドの実行

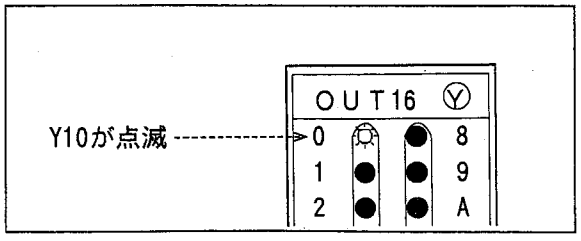
```
>XQT↵……入力する
```

＊XQTは、f･5キーの押下で入力できます。

②ＰＣの運転（ＲＵＮ）を確認します。ＣＰＵの「ＲＵＮ」ＬＥＤが点灯します。



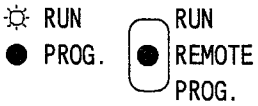
③スイッチ(X0)をＯＮすると、スイッチ(X1)がオンの場合ＰＣの出力ユニットのY10は点滅を繰り返し、スイッチ(X1)がオフの場合出力ユニットのY10が1回だけ点滅します。



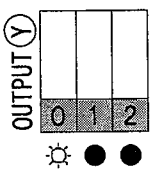
注意

●プログラム転送時には、ＣＰＵのモードスイッチが「ＲＥＭＯＴＥ」に設定されていて、「ＰＲＯＧ.」ＬＥＤが点灯していることを確認してください。

ＦＰ１の場合



ＦＰ１の場合



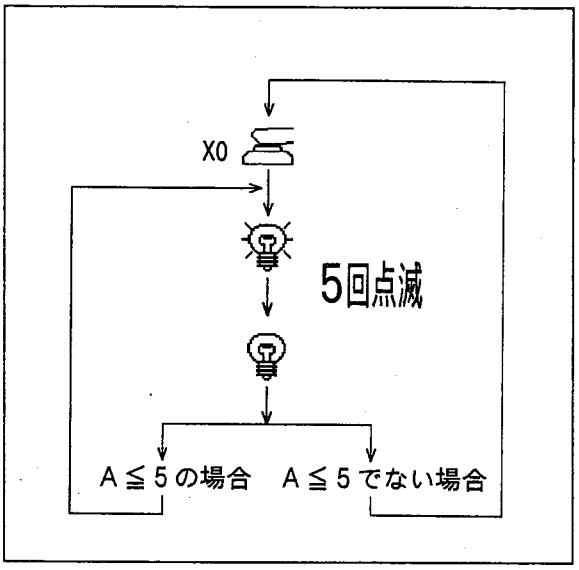
(2)ＩＦを使用したカウンタプログラムの例

1 変数の値による制御

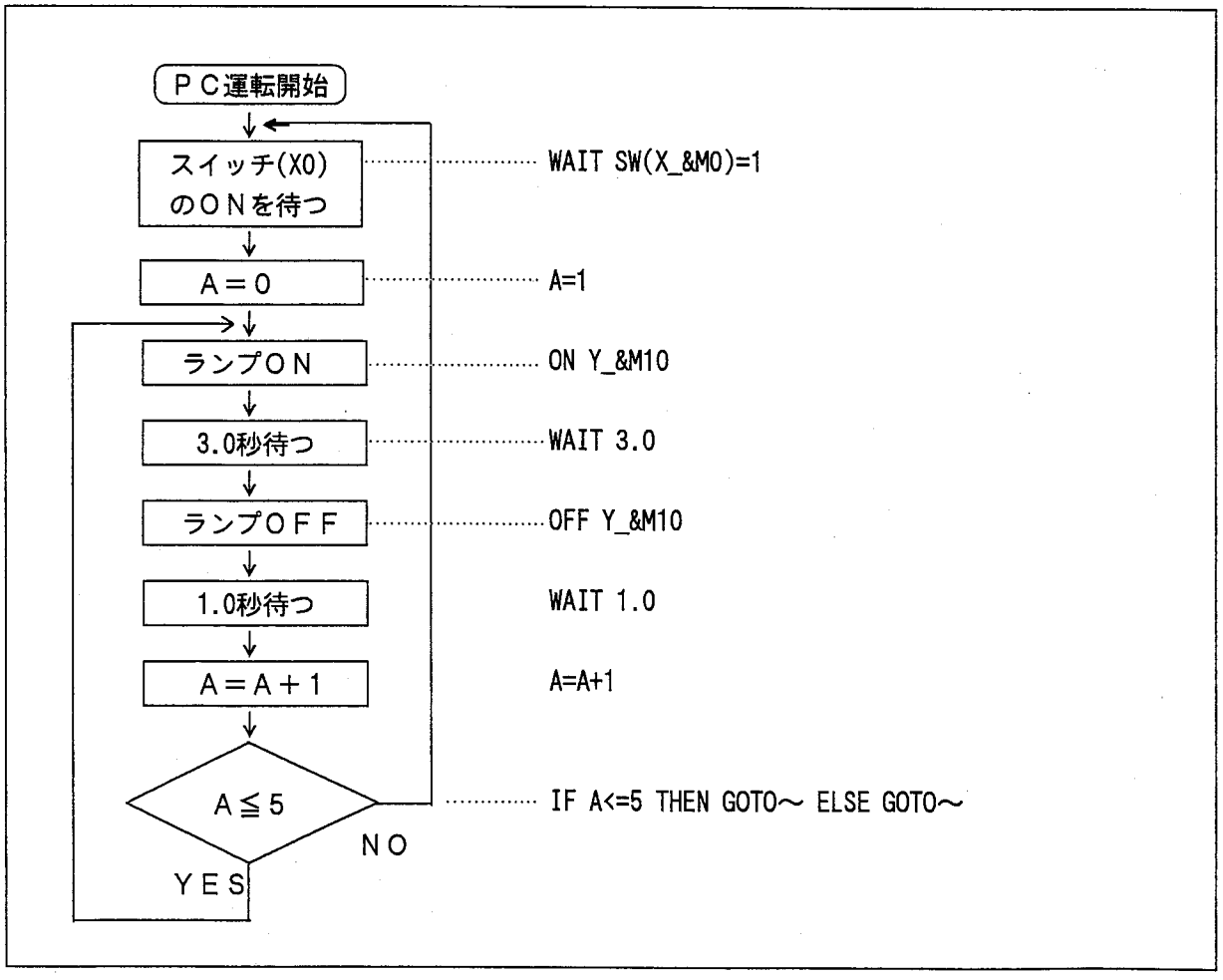
前回の(1)のプログラムに変数（数値の制御）を追加し、変数の値により、次のように動作するランプ点灯制御プログラムを考えます。

①〔運転継続〕変数の値が５未満の状態でランプは点滅を繰り返す。

②〔運転停止〕変数の値が５以上の状態で点滅を終了し、再スタートを待つ。



2 制御の流れ（フローチャート）



3 プログラムの入力

次のようにプログラムをキー入力します。

```
>LIST↵
10 FUNCTION MAIN
20 WAIT SW(X_&M0)=1
30 ON Y_&M10
40 WAIT 3.0
50 OFF Y_&M10
60 WAIT 1.0
70 IF SW(X_&M1)=1 THEN GOTO 30 ELSE GOTO 20
80 FEND
>25 A=0↵
>65 A=A+1↵
>70 IF A<=5 THEN GOTO 30 ELSE GOTO 20↵
>LIST↵
10 FUNCTION MAIN
20 WAIT SW(X_&M0)=1
25 A=0
30 ON Y_&M10
40 WAIT 0.5
50 OFF Y_&M10
60 WAIT 0.5
65 A=A+1
70 IF A<=5 THEN GOTO 30 ELSE GOTO 20
80 FEND
```

命令の説明

●IFの後に条件式を記述し、条件が真ならTHEN直後の命令を、偽ならELSE直後の命令を、各々実行します。

●左の例では、A<5が成立すると30行目に、それ以外の場合は20行目に実行を移します。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

ポイント

●変数Ａが5未満なら点滅を繰り返します。

●変数Aの値を判断し、５未満なら30行目に、そうでなければ20行目に戻ります。

4 プログラムの転送

入力したプログラム（ソースプログラム）をＣＯＭＰＩＬＥコマンドで実行プログラムに変換し、ＤＷＮＬＤコマンドでＰＣに転送します。

```
>COMPILE↵ ……入力する
COMPILE END
>DWNLD↵ ……入力する
```

＊COMPILE･DWNLDは、f･6キー・f･7キーの押下で各々入力できます。

注意

●プログラム転送時には、ＣＰＵのモードスイッチが「ＲＥＭＯＴＥ」に設定されていて、「ＰＲＯＧ.」ＬＥＤが点灯していることを確認してください。

5 プログラムの実行

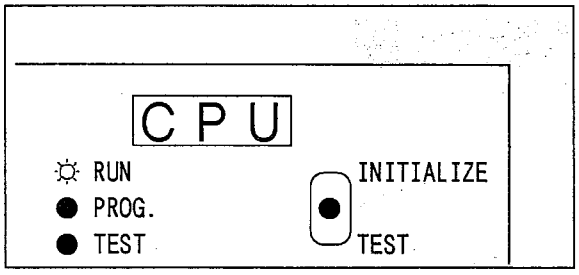
ここでは、ＣＰＵを「ＲＥＭＯＴＥ」モードに設定し、FP-BASIC編集ソフト（パソコン）からＸＱＴコマンドを実行します(リモート実行)。

①ＸＱＴコマンドの実行

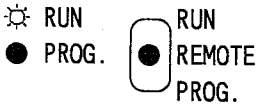
>XQT↵……入力する

＊XQTは、f･5キーの押下で入力できます。

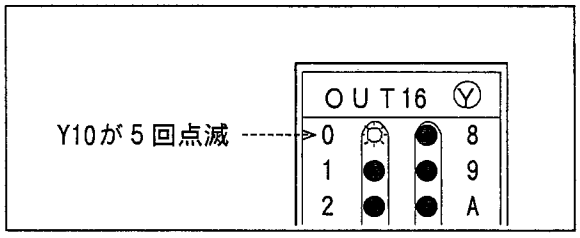
②ＰＣの運転（ＲＵＮ）を確認します。ＣＰＵの「ＲＵＮ」ＬＥＤが点灯します。



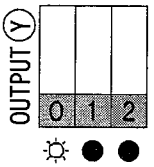
ＦＰ１の場合



③スイッチ(X0)をオンすると、ＰＣの出力ユニットのY10は５回点滅を繰り返します。



ＦＰ１の場合



課題(2)

●条件スイッチ(X1)がオン状態でスタートスイッチ(X0)をオンすれば点滅を５回繰り返してからスタートスイッチ(X0) の入力を待ち、条件スイッチ(X1)がオフの状態ではスタートスイッチ(x0)のオンにより点滅を２回して繰り返してスタートスイッチ（X0）の入力を待つプログラムを作ってください。

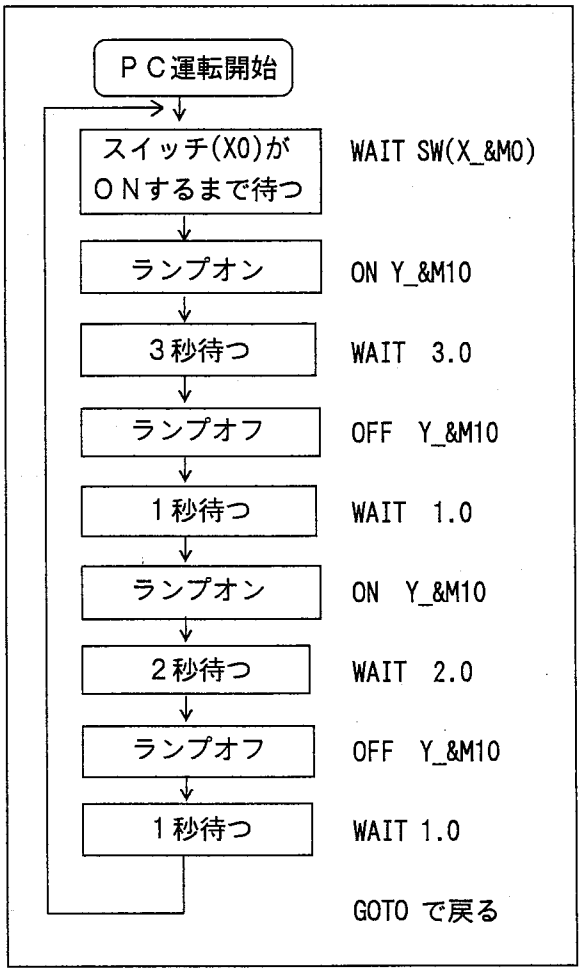
ヒント：もうひとつ変数を増やし（B）、A<=Bと比較する。

(解答は38ページです)

■解答

解答(1)

●フローチャート



●プログラム

```
>LIST
10  FUNCTION  MAIN
20  WAIT  SW(X_&M0)
30  ON  Y_&M10
40  WAIT  3.0
50  OFF  Y_&M10
60  WAIT  1.0
70  ON  Y_&M10
80  WAIT  2.0
90  OFF  Y_&M10
100  WAIT  1.0
110  GOTO  20
120  FEND
```

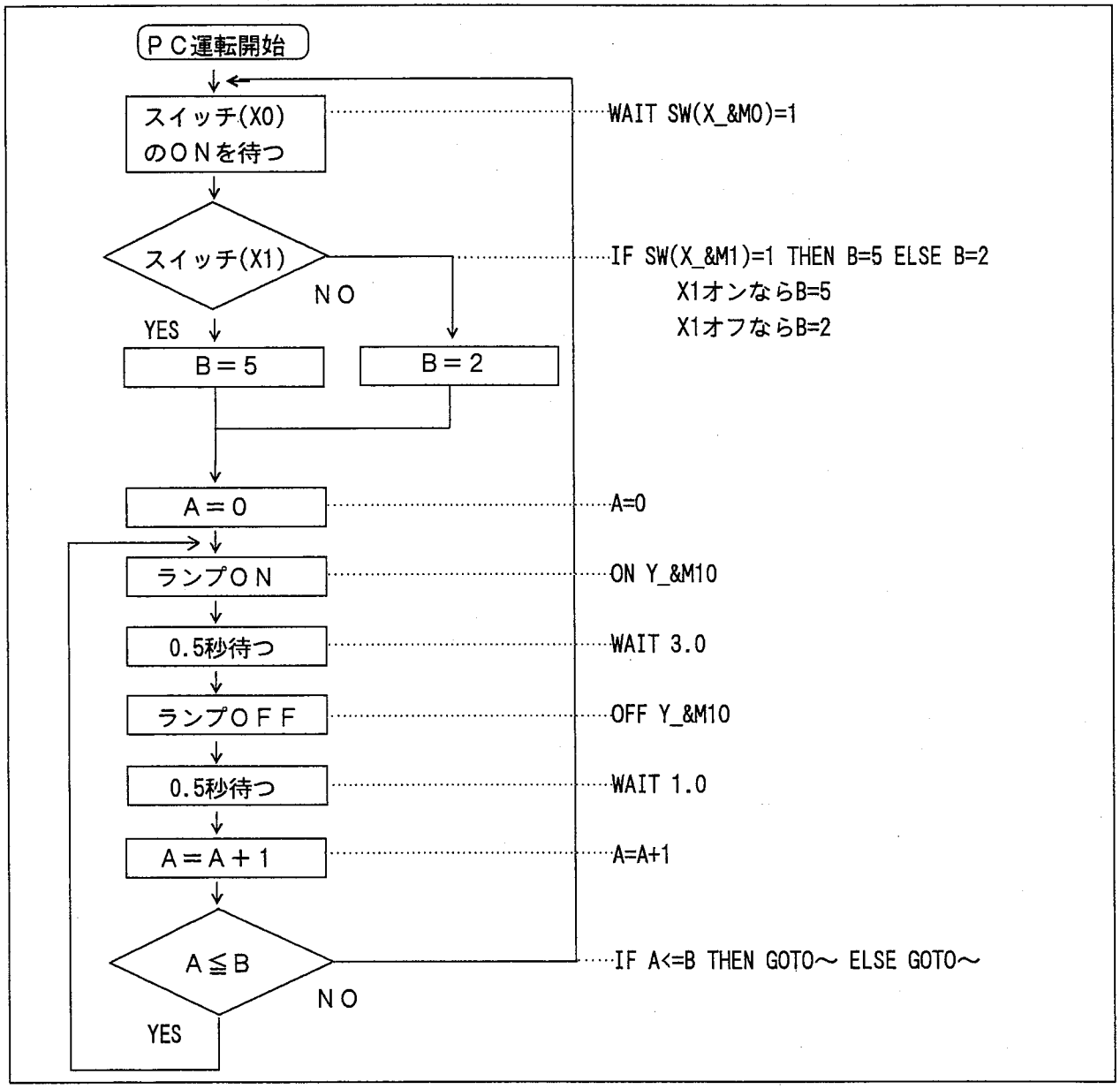
ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

■解答

解答(2)

●フローチャート



●プログラム

```
>LIST
10 FUNCTION MAIN
20 WAIT SW(X_&M0)=1
30 IF SW(X_&M1)=1 THEN B=5 ELSE B=2
40 A=0
50 ON Y_&M10
60 WAIT 3.0
70 OFF Y_&M10
80 WAIT 1.0
90 A=A+1
90 IF A<=B THEN GOTO 50 ELSE GOTO 20
100 FEND
```

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

実用コースでは、実際に制御プログラムを作成するために知っておかなければならないことをまとめました。ＦＰ－ＢＡＳＩＣの特長であるマルチタスクを理解すればほとんどの実用的なプログラムが作成できます。なお、プログラムにはすべてコメントを付けてありますので、コメントの内容もよく読んでください。

実用コース

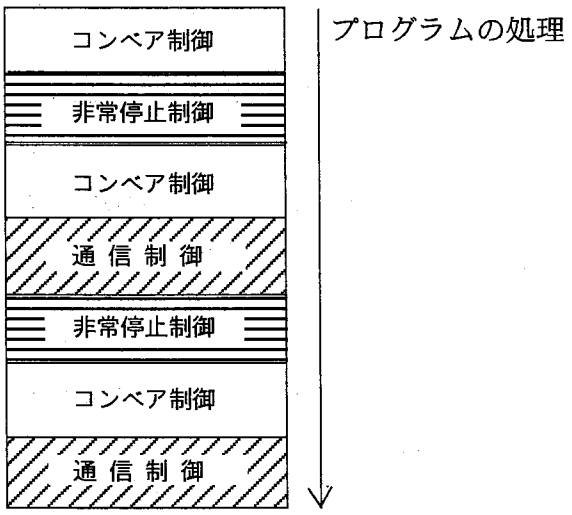
2 マルチタスクプログラミング

2-1 なぜマルチタスクが必要なのか？

今までタスク（プログラム）が１つの場合について説明してきました。タスクが一つだけの場合、「WAIT 0.5」等で待ち状態の時に外部の非常停止信号が入っても、それを読み取り直ちに停止することはできません。
従来のパソコン等のＢＡＳＩＣでは、タスクが１つしかありません（シングルタスク）ので、次の図に示すように１つのタスクでいろいろなプログラムを組み込み、あたかもマルチタスクであるかのごとく複数の仕事を同時にさせるため、非常に複雑なプログラムを組む必要があり、一般的に制御には不向きでした。
マルチタスクプログラムは、これらの問題を解決し、プログラムが非常に簡単に作成できる素晴らしい機能です。マルチタスクプログラムでは、それぞれのプログラムを独立して作成、動作させることができますので、次の図のように非常停止のプログラムとコンベア制御、通信制御等のプログラムが簡単にできます。ここでは、このマルチタスクプログラムの作り方について説明します。

シングルタスクプログラムの場合

●シングルタスクだと、１本のプログラムでごちゃまぜの制御が必要。
●ある１部のプログラム実行に時間がかかると全体の実行時間が遅くなる。



マルチタスクプログラムの場合

●各制御プログラムが全く別々に作れる。
●他の制御プログラムに関係なく実行できる。 } →プログラム作成が非常に簡単になる。

2-2 マルチタスクプログラムの作り方

・ＸＱＴ

マルチタスクはＸＱＴ命令で他のタスクを起動することで、簡単に作れます。

例：XQT !2,JOB2

タスクNO.　タスク名

ＪＯＢ２をタスク２番で起動します。

タスク１

```
10 FUNCTION JOB1
20 XQT !2,JOB2
30 ON Y_&M10
40 WAIT 1.0
50 OFF Y_&M10
60 WAIT 0.5
70 GOTO 30
80 FEND
```

自動的に起動

タスク２を起動

タスク２

```
100 FUNCTION JOB2
110 ON Y_&M11
120 WAIT 2.0
130 OFF Y_&M11
140 GOTO 110
150 FEND
```

●タスクが複数個あるとき、行番号が１番小さいタスク（プログラム）が自動的に起動します。

●必要なタスク（プログラム）は、ＸＱＴ命令で起動します。

●同時に実行できるタスク（プログラム）は、ＦＰ３タイプで16タスク（ＦＰ１タイプでは６タスク）ですが、プログラムは50個まで作れます。条件により起動（ＸＱＴ）、停止（ＱＵＩＴ）させ、たくさんのプログラムを切り替えて実行させられます。

注意　ＸＱＴ

●ＱＵＩＴ命令は同じタスクに対して何度でも実行することができますが、ＸＱＴ命令を起動中のタスクに対して実行するとエラーになります。何度も起動、停止を繰り返す際は注意してください。

2-3 マルチタスクプログラムの例

●ＸＱＴ

プログラムを実行させます。

例：XQT !2,STAMP

……「STAMP」という名称のプログラムをタスク２として起動します。

●ＱＵＩＴ

実行中のプログラムを停止させます。

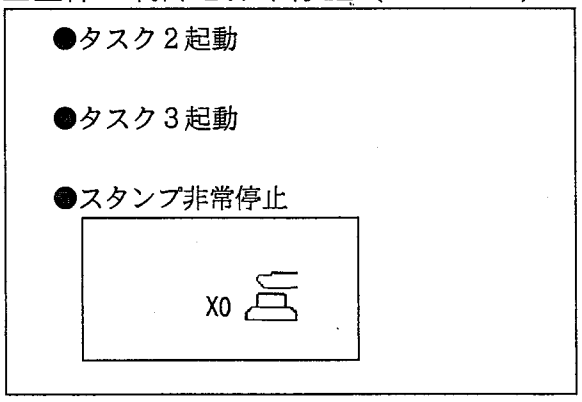
例：QUIT !2

……タスク２のプログラムを停止します。

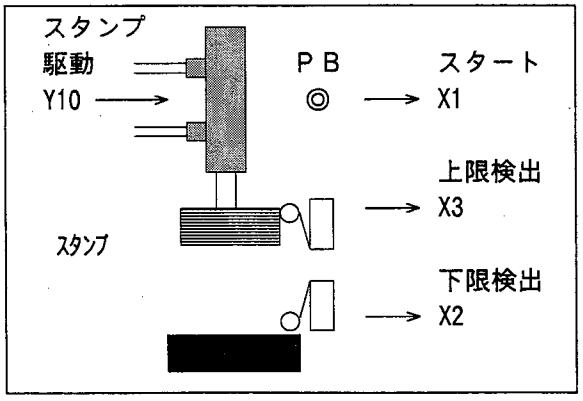
1 工程の説明

マルチタスク機能を使用して、３つの異なるプログラムを使ってみましょう。

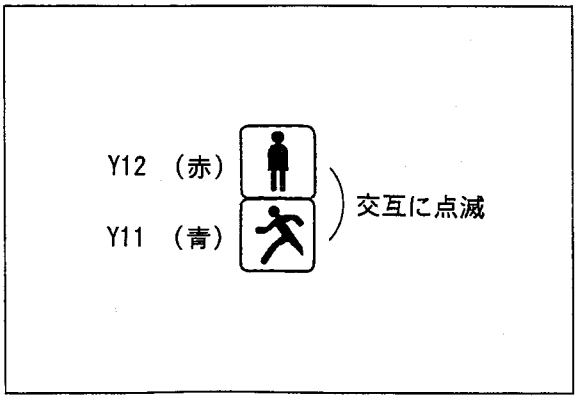
■全体の制御と非常停止（タスク１）



■スタンプ制御（タスク２）

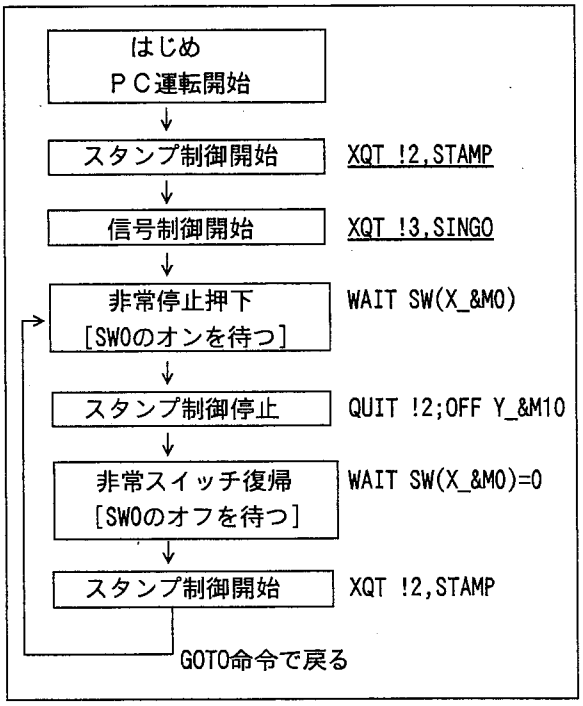


■信号制御（タスク３）



2 各工程の制御の流れ（フローチャート）

■全体の制御と非常停止（タスク１）



（１番目のタスクとして実行する）

この例では、３つのタスクを使った制御プログラムになっており、それぞれのタスク内容は次のようになります。

●タスク１は、他のタスクを起動し、その後非常停止入力を監視します。

●タスク２はスタンプ制御。

●タスク３は信号制御。

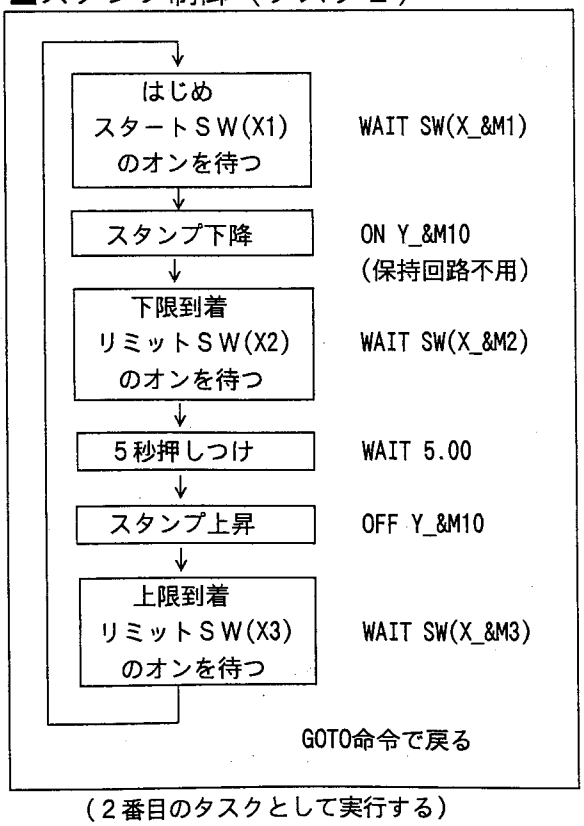
参考

・複数個の非常停止については「5-1」を参照してください。

注意

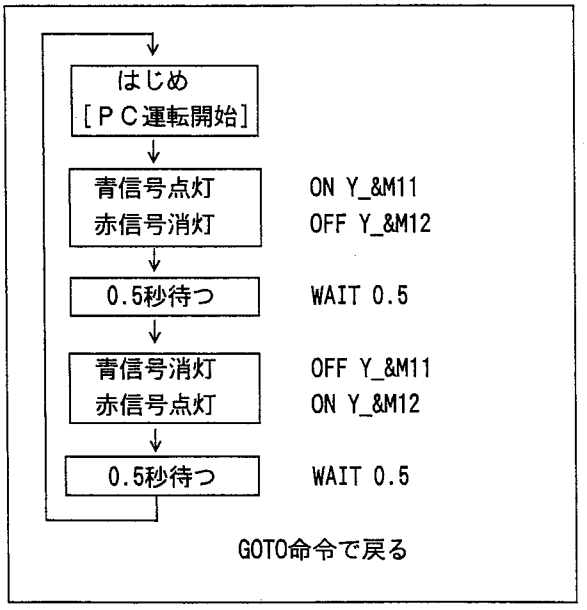
●ＱＵＩＴ命令は同じタスクに対して何度でも実行することができますが、ＸＱＴ命令を起動中のタスクに対して実行するとエラーになります。何度も起動、停止を繰り返す際は注意してください。

■スタンプ制御（タスク２）



（２番目のタスクとして実行する）

■信号制御（タスク３）



（３番目のタスクとして実行する）

2-4 マルチタスクのプログラミング

●ＦＵＮＣＴＩＯＮ　＜プログラム名＞
　プログラムの開始を宣言します。
●ＦＥＮＤ
　プログラムの終了を宣言します。
●ＸＱＴ
　他のプログラムを起動します。
　例：XQT !2,STAMP
　　……ＳＴＡＭＰという名称のプログラムをタスク２として起動します。
●ＱＵＩＴ
　プログラムを停止します。
　例：QUIT !2
　　……タスク２を停止します
●ＡＵＴＯ
　行番号を自動的に発生させます。
　例：AUTO 1000
　　……プログラムの入力を行番号1000からスタートします。

1 プログラム入力の開始

ＶＥＲＩＮＩＴ、ＮＥＷ、ＡＵＴＯの各コマンドを入力します。

```
>VERINI↵ ……入力
>NEW↵
>AUTO 1000↵ ……
1000 ■
```

＊AUTOは f･2 キーの押下で入力できます。

ＡＵＴＯの説明

●AUTOは、行番号を自動的に発生させます。
●AUTO 1000↵とすることにより、行番号1000からスタートします。行番号を1000からはじめると、10000まで行番号の桁数が変らないので、プログラムが見やすくなります。
●ＡＵＴＯモードでのプログラムの入力が終了したら、ＣＯＭＰＩＬＥ等の作業の前に STOP キーを押して（または CTRL ＋ C キーを押して）ＡＵＴＯモードから抜けてください。正しい作業ができなくなります。

2 １番目のプログラムの入力

１番目のプログラムの名称はＭＡＩＮにします（他の名称でも良いわけですが、「スタンプ制御」と「信号制御」の両方を制御するのでＭＡＩＮとしておきます）。スタンプ制御のプログラム名はＳＴＡＭＰに、信号制御のプログラム名はＳＩＮＧＯとします。

```
>AUTO 1000
1000 FUNCTION MAIN
1010 XQT !2,STAMP                    'ｽﾀﾝﾌﾟ工程をﾀｽｸ2で実行する
1020 XQT !3,SINGO                    '信号工程をﾀｽｸ3で実行する
1040 WAIT SW(X_&M0)=0;XQT !2,STAMP   'ﾀｽｸ2停止解除
1050 GOTO 1030                       '1030行目に戻る
1060 FEND
1070 ……ここで STOP キーを押すとAUTOコマンドの
          自動行番号が終了する
```

ＸＱＴの説明

●XQT !2,STAMPはＳＴＡＭＰという名称のプログラムを２番目のプログラム（タスク２）として起動します。

……プログラム名

……タスク番号

ＸＱＴの説明

●'はシングルクォーテーションといい、'以降の記述はコメントとなり実行されません。

ＸＱＴの説明

●;はセミコロンといい、;で区切れば、複数の命令を１行に書けます。WAIT SW(X_&M0);QUIT !2;OFF Y_&M10は、複数の命令を１行に記述しています。

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

3 2番目のプログラムの入力

2番目のプログラムの名称は、ＳＴＡＭＰにします。2番目のプログラムは、行番号2000から始めます。

```
>AUTO 2000
2000 FUNCTION STAMP
2010 WAIT SW(X_&M1)    'ｽﾀｰﾄｽｲｯﾁ(X1)のONを待つ
2020 ON Y_&M10         'Y10をONする
2030 WAIT SW(X_&M2)    '下限ﾘﾐｯﾄｽｲｯﾁ(X2)のONを待つ
2040 WAIT 5.00         '5秒待つ
2050 OFF Y_&M10        'Y10をOFFする
2060 WAIT SW(X_&M3)    '上限ﾘﾐｯﾄｽｲｯﾁ(X3)のONを待つ
2070 GOTO 2010          '2010行に戻る
2070 FEND
2080……ここで STOP キーを押す
```

ＦＰ１の場合

●Y_&M10の代わりにY_&M0を入力します。

参考

●「4-4」を参照して SHIFT ＋ f･1 ～ f･10 のキー登録を変更すれば、キー入力が早くなります。

4 ３番目のプログラムの入力

３番目のプログラムの名称は、ＳＩＮＧＯにします。３番目のプログラムは、行番号3000から始めます。

```
>AUTO 3000
3000 FUNCTION SINGO
3010 ON Y_&M11;OFF Y_&M12;WAIT 0.50  ' 青点灯・赤消灯・0.5秒待
3020 OFF Y_&M11;ON Y_&M12;WAIT 0.50  ' 青消灯・赤点灯・0.5秒待
3030 GOTO 3010                       ' 3010行目に戻る
3040 FEND
3050……ここで STOP キーを押す
```

ＦＰ１の場合

●Y_&M11･12の代わりにY_&M1･2を入力します。

5 プログラムの確認

ＬＩＳＴコマンドで、入力したプログラムを表示します。

```
>LIST↵ ..... 入力
1000 FUNCTION MAIN
1010 XQT !2,STAMP                          'ｽﾀﾝﾌﾟ工程をﾀｽｸ2で実行
1020 XQT !3,SINGO                          '信号工程をﾀｽｸ3で実行
1030 WAIT SW(X_&M0);QUIT !2;OFF Y_&M10     'ﾀｽｸ2停止
1040 WAIT SW(X_&M0)=0;XQT !2,STAMP         'ﾀｽｸ2停止解除
1050 GOTO 1030                             '1030行目に戻る
1060 FEND
2000 FUNCTION STAMP
2010 WAIT SW(X_&M1)   'ｽﾀｰﾄｽｲｯﾁ(X1)のONを待つ
2020 ON Y_&M10        'Y10をONする
2030 WAIT SW(X_&M2)   '下限ﾘﾐｯﾄｽｲｯﾁ(X2)のONを待つ
2040 WAIT 5.00        '5秒待つ
2050 OFF Y_&M10       'Y10をOFFする
2060 WAIT SW(X_&M3)   '上限ﾘﾐｯﾄｽｲｯﾁ(X3)のONを待つ
2070 GOTO 2010        '2010行に戻る
2080 FEND
3000 FUNCTION SINGO
3010 ON Y_&M11;OFF Y_&M12;WAIT 0.50  '青点灯・赤消灯・0.5秒待つ
3020 OFF Y_&M11;ON Y_&M12;WAIT 0.50  '青消灯・赤点灯・0.5秒待つ
3030 GOTO 3010                       '3010行目に戻る
3040 FEND
```

タスク１

タスク２

タスク３

ＦＰ１の場合

●Y_&M10・11・12の代わりにY_&M0・1・2が表示されます。

6 コンパイルとＰＣへの転送

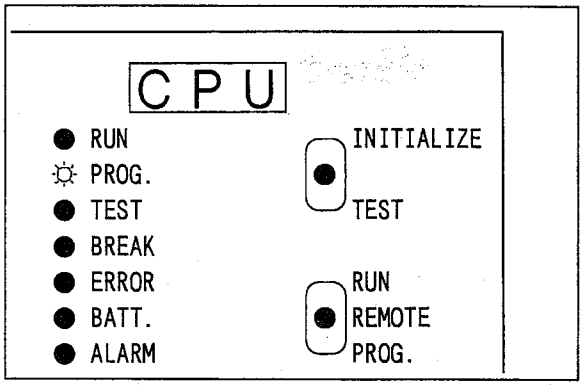
入力したプログをＣＯＭＰＩＬＥコマンドで、コンパイル（実行プログラムに変換）し、ＤＷＮＬＤコマンドでＰＣに転送します。

ＤＷＮＬＤコマンド実行時、ＰＣがＰＲＯＧ．モードになっていることを確認してください。

2-5 マルチタスクプログラムの実行

①ＸＱＴコマンドの実行

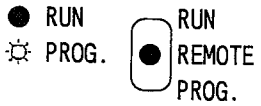
>XQT↵ ..... 入力



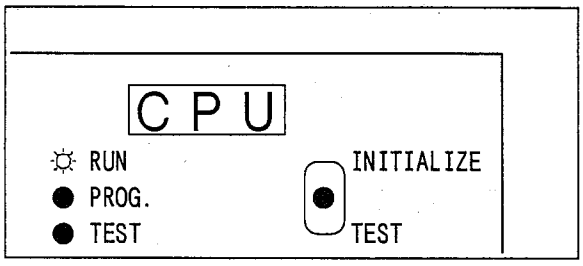
注意

●PCの停止は、f·10 キーまたは STOP キーを押してQUITコマンドを実行します。

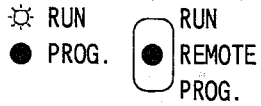
ＦＰ１の場合



②ＰＣの運転（ＲＵＮ）を確認します。ＣＰＵの「ＲＵＮ」ＬＥＤが点灯します。

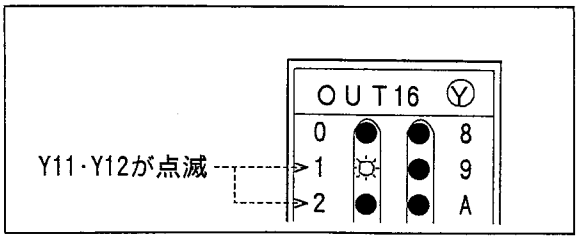


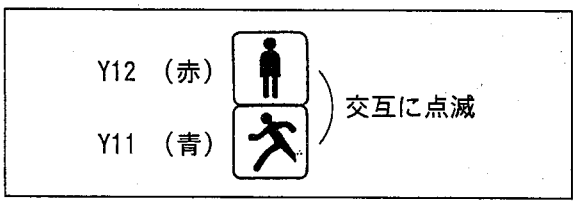
ＦＰ１の場合



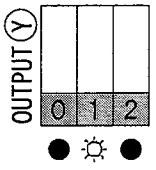
■信号制御

ＣＰＵが運転を開始すると、ＰＣの出力ユニットのY11とY12がオン／オフを繰り返しますします（表示ＬＥＤが点灯・点滅・消灯を繰り返す）。



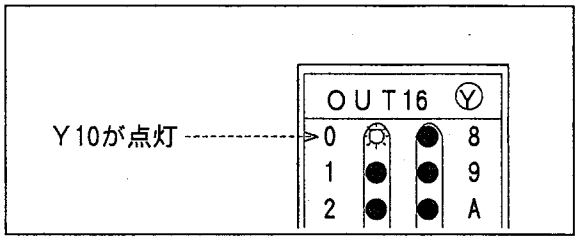


ＦＰ１の場合

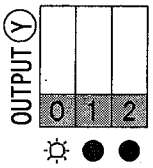


■スタンプ制御

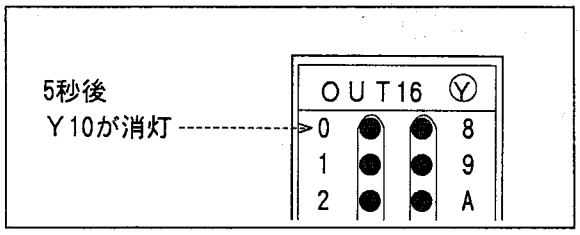
①スイッチ(X1)をオンすると、ＰＣの出力ユニットのY10がオンします（表示ＬＥＤが点灯します）。



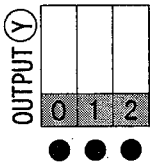
ＦＰ１の場合



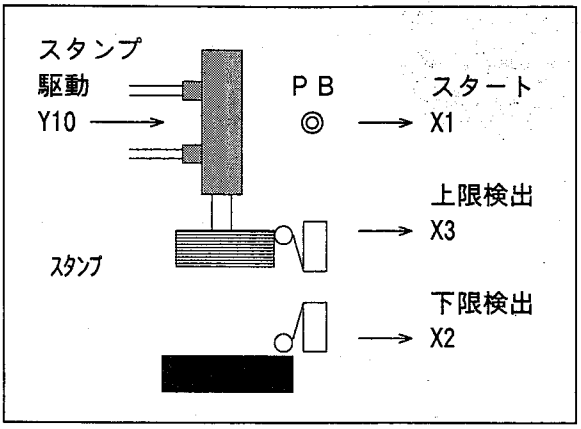
②スイッチ(X2)をオンすると、ＰＣの出力ユニットのY10は5秒間だけオンを保持してから、オフします（表示ＬＥＤは5秒間点灯保持→消灯）。



ＦＰ１の場合

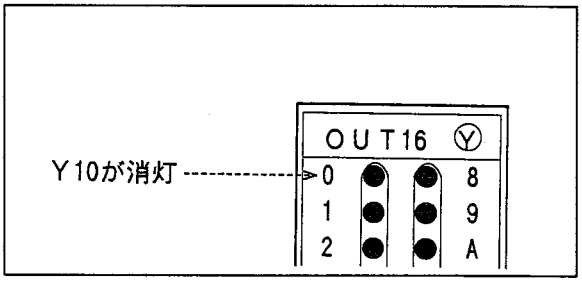


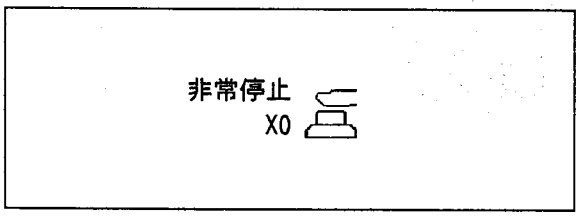
③スイッチ(X3)をオンすると、プログラムは①に戻ります。



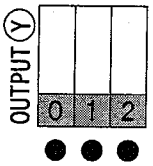
■非常停止

スイッチ(X0)をオンすると、スタンプ制御工程のプログラムを停止し、同時にY10をオフします。スイッチ(X0)がオフになると、スタンプ制御工程のプログラムが再度実行され、スイッチ(X1)の入力を受け入れます。





ＦＰ１の場合



3 プログラムの動作チェック

3-1 プログラムを部分的に実行する方法

・ＸＱＴ
　他のプログラムを起動します。
　例：XQT !1,SINGO
　　……ＳＩＮＧＯプログラムをタスク１として実行します。
　例：XQT !1,SINGO,3010-3020
　　……ＳＩＮＧＯプログラムをタスク１として、
　　行番号3010～3020を実行します。3020は省略できます。

1 実行するプログラムの説明

実行するプログラムは、「2-4」で作成したものと同じです。

```
>LIST↵……入力
1000 FUNCTION MAIN
 :
 :
1060 FEND
2000 FUNCTION STAMP
 :
 :
2080 FEND
3000 FUNCTION SINGO
3010 ON Y_&M11;OFF Y_&M12;WAIT 0.50   '青点灯・赤消灯・0.5秒待つ
3020 OFF Y_&M11;ON Y_&M12;WAIT 0.50   '青消灯・赤点灯・0.5秒待つ
3030 GOTO 3010                        '3010行目に戻る
3040 FEND
>COMPILE↵……入力
COMPILE END
>DWNLD↵……入力
```

ＦＰ１の場合

●Y_&M10・11・12の代わりにY_&M0・1・2を入力します。

注意

●CPUのモード設定スイッチが「ＲＥＭＯＴＥ」になっていて「PROG.」LEDが点灯していることを確認してください。

2 ３番目のプログラムだけを実行する

３番目のプログラム（信号制御工程）だけを実行してみます。「XQT !1,SINGO」と入力して実行します。プログラム名ＳＩＮＧＯプログラムをタスク１として実行するという意味です。

```
>XQT !1,SINGO↵……入力
```

3 行番号を指定して実行する

行番号を指定してプログラムを実行することもできます。以下の例では、プログラム名ＳＩＮＧＯプログラムをタスク１として、行番号3010～3020を実行します。

```
>XQT !1,SINGO,3010-3020↵……入力
```

3-2 タスクの動作状態をチェックする方法

●ＭＯＮＴＳ
実行中の複数のタスクの現在の動作状態を確認します。

1 実行するプログラムの説明

実行するプログラムは、「2-4」で作成したものと同じです。

2 プログラムの実行

FP-BASIC編集ソフトからＸＱＴコマンドを実行します。

```
>XQT↵ ……入力
```

3 ＭＯＮＴＳコマンドの実行

```
>MONTS↵ …… 入力
```

4 ステータスの表示画面

タスク番号　動作状態　実行行番号

```
task01    run  01030   →タスク１は1030行目を実行中
task02   wait  02030   →タスク２はＷＡＩＴ中
task03   wait  03010   →タスク３はＷＡＩＴ中
task04   quit  00000   →タスク４は停止中
task05   quit  00000   →タスク５は停止中
task06   quit  00000   →タスク６は停止中
task07   quit  00000   →タスク７は停止中
task08   quit  00000   →タスク８は停止中
task09   quit  00000   →タスク９は停止中
task10   quit  00000   →タスク１０は停止中
task11   quit  00000   →タスク１１は停止中
task12   quit  00000   →タスク１２は停止中
task13   quit  00000   →タスク１３は停止中
task14   quit  00000   →タスク１４は停止中
task15   quit  00000   →タスク１５は停止中
task16   quit  00000   →タスク１６は停止中
```

ＭＯＮＴＳの説明

●ＭＯＮＴＳとすると、4 で示すように全タスクの動作状態がリアルタイムで連続モニタすることができます。プログラムのチェックに大変役立ちます。ぜひ覚えてください。

注意　ＴＳＴＡＴ

●Ver.2.0未満のFP-BASIC編集ソフトおよびFP1-BASICタイプでは、ＭＯＮＴＳコマンドの代わりにＴＳＴＡＴコマンドを使用してください。ただし、ＴＳＴＡＴコマンドはコマンド実行時の各タスクの動作状態を表示するものであり、リアルタイムな表示はできません。

動作状態 ＞ ｒｕｎ …… 実行中
　　ｑｕｉｔ …… 停止中
　　ｗａｉｔ …… ＷＡＩＴ中
　　ｈａｌｔ …… 一時停止

ESC キーを押すと元の画面に戻ります。

3-3 トレースモード

●ＴＯＮ
実行中のプログラムの行番号を表示するトレースモードに入ります。
●ＴＯＦＦ
トレースモードを解除します。

1 実行するプログラムの説明

実行するプログラムは、「2-4」で作成したプログラムと同じです。

2 トレースモードでの実行

「TON !3」と入力して、３番目のプログラムである信号制御プログラムをトレースします。

```
>TON !3↵ ……入力
>XQT↵ …… 入力
```

3 トレースモードの画面表示

FP-BASIC編集ソフト（パソコン）の画面には、３番目のプログラム（タスク３）の実行中の行番号が表示されます。

```
>TON !3
>XQT
[3:03000]
[3:03010]
[3:03020]
[3:03030]
[3:03010]
[3:03020]
 :
```

4 トレースモードの解除

トレースモードを解除するには、ＴＯＦＦコマンドを実行します。

```
[3:03020]
[3:03030]
>TOFF↵ …… 入力
```

注意

●TOFFは、 SIFT + f･2 キーの押下で入力できます。また、トレースモードは、 STOP キーの押下でも解除できます。
●トレースモードで表示が頻繁に変化するときは、 STOP キーでプログラムを終了させてください。

3-4 入出力のモニタ方法

●MONENT
　モニタする複数の入出力を登録します。
●MONDISP
　MONENTコマンドで登録した複数の入出力を同時にモニタします。

①MONENTコマンドを実行すると、画面上に登録ウインドウが開きます。

```
>MONENT↵ ……入力
```

```
REAL TIME MONITOR I/O ENTRY
1.X_&M0          5.               9.               13.
2.X_&M1          6.               10.              14.
3.Y_&M10         7.               11.              15.
4.               8.               12.              16.
*ﾒｯｾｰｼﾞ :
< Key操作 > TAB:前進  BS:後退
MX_  MY_  MR_  ML_  DT_  LD_  FL_  R_  Z_  B_
```

ポイント

●モニタ対象の入出力の登録は同時に16までできます。登録する入出力（およびメモリ）の種類の切り替えには、[f･1]～[f･10]キーを使用します。
●たとえばX_&M0登録のキー操作は以下の通りです。

[f･8] [SHIFT]+[&] [M] [0]

●[ESC]キーを押すと登録ウィンドウが閉じます。

②MONDISPコマンドを実行すると、画面上にモニタ・ウインドウが開きます。

```
>MONDISP↵ ……入力
```

```
REAL TIME MONITOR                                    (UNIT:HEX)
1.X_&M0                0000 10.
2.X_&M1                0000 11.
3.Y_&M10               0001 12.
4.                          13.
5.                          14.
6.                          15.
7.                          16.
8.
9.
< 進法変換 > H:HEX  D:DEC  O:OCT  B:BIN
```

ポイント

●画面には、MONENTコマンドで登録した入出力の状態が表示されます。
●10進・16進・2進・8進の表記の切り替えには、「D」「H」「B」「O」を入力します。
●[ESC]キーを押すとモニタ・ウインドウが閉じます。

3-5 ダンプによる入出力、データの状態チェック

●ＤＵＭＰＰ

動作中のＣＰＵの入出力、データメモリ、その他の内容を表示し、さらに内容を書き換えることができます。

1 実行するプログラムの説明

実行するプログラムは「2-4」で作成したものと同じです。

2 プログラムの実行

FP-BASIC編集ソフトからＸＱＴコマンドを実行します。

```
>XQT↵ ……入力
```

3 ＤＵＭＰＰコマンドの実行

```
>DUMPP↵ ……入力
```

4 選択画面

モニタする入出力、データメモリ、その他の内容を選びます。処理番号を入力して↵を押してください。

```
1.入力                    10.強制出力
2.出力                    11.強制ﾒﾓﾘI/O
3.ﾒﾓﾘI/O                  12.強制ﾘﾝｸ･ﾒﾓﾘI/O
4.ﾘﾝｸ･ﾒﾓﾘI/O              13.ﾊﾟﾗﾒｰﾀ･ﾒﾓﾘ
5.ﾃﾞｰﾀ･ﾒﾓﾘ                14.共有RAM
6.ﾘﾝｸ･ﾃﾞｰﾀ･ﾒﾓﾘ            15.ｽﾛｯﾄ設定
7.特殊ﾒﾓﾘ･I/O             16.割り込み設定
8.特殊ﾃﾞｰﾀ･ﾒﾓﾘ
9.強制入力
    処理番号を入力して下さい ==>
ﾒｯｾｰｼﾞ :
```

5 表示画面

ＣＰＵの各種動作状態をウインドウ形式で表示し、さらに直接内容を書き換えることができます。表示内容は自由にスクロールできます。

```
        入力I/O  状態表示
          HEX          DEC                        BIN        ASC
0000      0000         00000    0000 0000 0000 0000          . .
0001      0000         00000    0000 0000 0000 0000          . .
0002      0000         00000    0000 0000 0000 0000          . .
0003      0000         00000    0000 0000 0000 0000          . .
```

注意

●Ver.2.0未満のFP-BASIC編集ソフトおよびFP1-BASICタイプでは、ＤＵＭＰＰコマンドの代わりにＤＵＭＰコマンドを使用してください。ただし、ウインドウ形式の表示ではありませんので、表示内容はスクロールしません。

3-6 入出力の強制オン・オフ

●ＦＯＲＣＥ
　入出力を強制的にオン・オフします。
●ＲＦＯＲＣＥ
　強制入出力状態を解除します。

1 実行するプログラムの説明

実行するプログラムは、「2-4」のプログラムと同じものです。

```
>LOAD "TEST4↵ ..... 入力
>LIST↵ ..... 入力
1000 FUNCTION MAIN
1010 XQT !2,STAMP              'ｽﾀﾝﾌﾟ工程をﾀｽｸ2で実行
 :
 :
3030 GOTO 3010
3040 FEND
>COMPILE↵ ..... 入力
COMPILE END
>DWNLD↵ ..... 入力
```

2 プログラムの実行

FP-BASIC編集ソフトからＸＱＴコマンドを実行します。

```
>XQT↵ ..... 入力
```

3 強制入出力の実行

①スイッチ(X1)の代わりに強制入出力でX_&M1をオンします。これにより、X_&M1は外部入力の状態に関係なくオン状態を保持しますので、続けてすぐに強制入出力を解除します。

```
>FORCE X_&M1=1↵ ..... 強制入力
>RFORCE ALL↵ ..... 解除
```

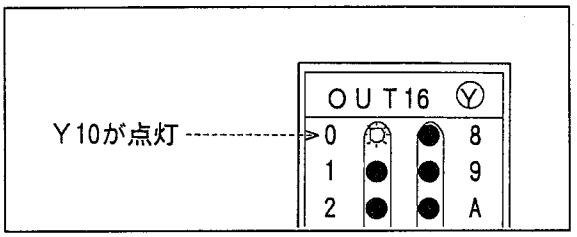
注意

●FP1-BASICタイプでは、強制入出力機能は使用できません。

注意　RFORCEを忘れないでください.

●FORCEは SHIFT + f･3 で入力可。
●RFORCEは SHIFT + f･4 で入力可。
●RFORCEを忘れると、正しい動作ができません。

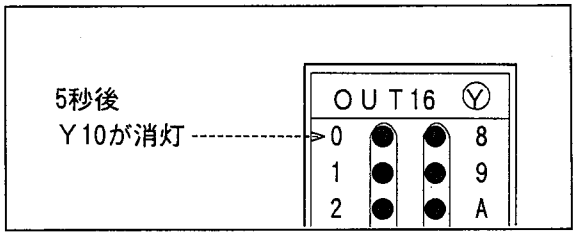
以上の操作により、ＰＣの出力ユニットのY10がオンします（表示ＬＥＤが点灯します）。



②スイッチ(X2)の代わりに強制入出力でX_&M2をオンします。これにより、X_&M2は外部入力の状態に関係なくオン状態を保持しますので、続けてすぐに強制入出力を解除します。

```
>FORCE X_&M2=1↵ ……強制入力
>RFORCE ALL↵ ……解除
```

以上の操作により、ＰＣの出力ユニットのY10は5秒間だけオンを保持してから、オフします（表示ＬＥＤは5秒間点灯保持→消灯）。



③スイッチ(X3)の代わりに強制入出力でX_&M3をオンします。これにより、X_&M3は外部入力の状態に関係なくオン状態を保持しますので、続けてすぐに強制入出力を解除します。

```
>FORCE X_&M3=1↵ ……強制入力
>RFORCE ALL↵ ……解除
```

プログラムは①に戻り、スイッチ(X1)の入力を待ちます。

■強制入出力の補足

●強制入出力の機能は、入力の場合も、出力の場合も等しく利用できます。強制入出力でオン状態またはオフ状態に設定された入出力は、外部機器やプログラム実行の状態にかかわらず、オン状態またはオフ状態を保ちます。

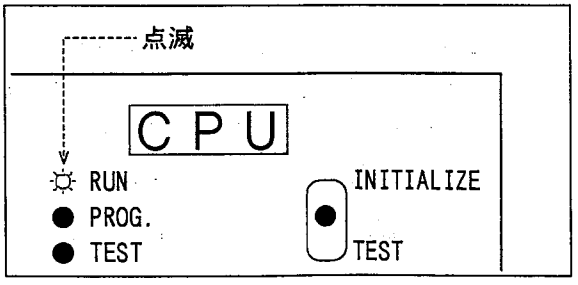
●ＰＣが強制入出力状態で運転をしているとき、ＣＰＵの「ＲＵＮ」ＬＥＤが点滅します。

注意

●FORCEは SHIFT ＋ f･3 で入力可。
●RFORCEは SHIFT ＋ f･4 で入力可。
●RFORCEを忘れると、正しい動作ができません。

注意

●FORCEは SHIFT ＋ f･3 で入力可。
●RFORCEは SHIFT ＋ f･4 で入力可。
●RFORCEを忘れると、正しい動作ができません。



3-7 ＰＲＩＮＴ命令を使った動作状態のチェック

●ＰＲＩＮＴ
　パソコン（FP-BASIC編集ソフト起動中）の画面に任意の文字、数値を表示します。
●ＣＬＳは
　パソコンの画面表示をクリアします。

実行するプログラムは、「3-7」のプログラムを修正したものでます。

```
>LIST↵ ..... 入力
1000 FUNCTION MAIN
1010 XQT !2,STAMP        'ｽﾀﾝﾌﾟ工程をﾀｽｸ2で実行
 :
 :
2000 FUNCTION STAMP
2010 WAIT SW(X_&M1)      'ｽﾀｰﾄｽｲｯﾁ(X1)のONを待つ
2020 ON Y_&M10           'Y10をONする
2030 WAIT SW(X_&M2)      '下限ﾘﾐｯﾄｽｲｯﾁ(X2)のONを待つ
2035 PRINT "2040 スタンプ押し付け中"  '画面表示 <-- ＰＲＩＮＴ命令
2040 WAIT 5.00           '5秒待つ
2045 CLS                 '画面表示クリア
2050 OFF Y_&M10          'Y10をOFFする
2060 WAIT SW(X_&M3)      '上限ﾘﾐｯﾄｽｲｯﾁ(X3)のONを待つ
2070 GOTO 2010           '2010行目に戻る
2080 FEND
 :
 :
3030 OFF GOTO 3010
3040 FEND
>COMPILE↵ ..... 入力
COMPILE END
>XQT↵ ..... 入力
```

プログラムを実行するとパソコン（FP-BASIC編集ソフト）の画面には次のような表示が５秒間表示されます。

```
>XQT↵ ..... 入力
2040 スタンプ押し付け中
      ..... 2040行目実行中（スタンプ押し
            付け中）のみ表示される
```

注意

●ＰＲＩＮＴ命令を記述したプログラムを実行する場合は、必ずパソコン（FP-BASIC編集ソフト起動中）を接続してください。
●プログラム動作確認後、ＰＲＩＮＴ命令をとるか、前に'をつけ'PRINT ○○とし、プリント命令を無効にして設備の制御を行ってください。
●FP-BASIC編集ソフトが起動されているパソコンが接続されていない状態で、ＰＲＩＮＴ命令を実行するとエラーになります。
●ＰＲＩＮＴを頻繁に繰り返すと、ＣＰＵをQUIT ALLで止めることができません。その際は STOP キーで停止させてください。

3-8 テスト実行モード

●ＰＡＵＳＥ
　プログラム中に記述することにより、プログラム実行を一時停止させます。
●ＴＥＳＴ
　ＣＰＵのテストモード実行の実行状態を選択します。
●ＣＯＮＴ
　ＰＡＵＳＥ命令により一時停止状態にあるプログラムの実行を再開させます。
●ＳＴＥＰ
　ＰＡＵＳＥ命令により一時停止状態にあるプログラムを、
　次の１行だけ実行させます（ステップモード実行）。

(1) 途中実行の方法

1 実行するプログラムの説明

「2-4」で作成したプログラムにＰＡＵＳＥ命令を記述して、実行します。プログラムの2065行に注目してください。

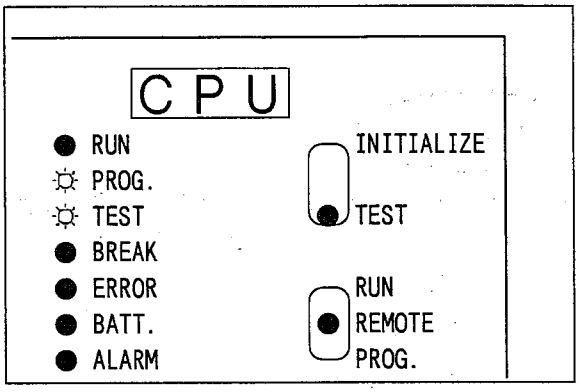
```
>LIST↵ ..... 入力
1000 FUNCTION MAIN
1010     XQT !2,STAMP     'ｽﾀﾝﾌﾟ工程をﾀｽｸ2で実行
 :
 :
2000 FUNCTION STAMP
2010 WAIT SW(X_&M1)     'ｽﾀｰﾄｽｲｯﾁ(X1)のONを待つ
2020 ON Y_&M10          'Y10をONする
2030 WAIT SW(X_&M2)     '下限ﾘﾐｯﾄｽｲｯﾁ(X2)のONを待つ
2040 WAIT 5.00          '5秒待つ
2045 CLS                '画面表示クリア
2050 OFF Y_&M10         'Y10をOFFする
2060 WAIT SW(X_&M3)     '上限ﾘﾐｯﾄｽｲｯﾁ(X3)のONを待つ
2065 PAUSE              'プログラム一時停止
2070 GOTO 2010          '2010行目に戻る
2080 FEND
 :
 :
3040 FEND
>COMPILE↵ .....入力
COMPILE END
>XQT↵ .....入力
```

ＰＡＵＳＥ
プログラムを任意の実行行で一時停止する

注意
●FP1-BASICタイプには、テスト実行モードの機能はありません。

2 テストモードの設定

①ＣＰＵのスイッチ設定
ここでは、ＣＰＵを「ＴＥＳＴ」モードに設定します。また、運転はリモート実行で行います。ＣＰＵをテストモードに設定すると、「ＴＥＳＴ」ＬＥＤが点灯します。



②ＴＥＳＴコマンドによるモード設定
テストモードには３つの機能があり、ＴＥＳＴコマンドにより使用する機能を設定します。今回はポーズモードを使用しますので、「TEST 1」を実行します。

```
>TEST 1↵ ..... 入力
```

命令の説明

●TESTは、テストモードの３つの機能を選択します。（モード選択は、ＰＲＯＧ．モードで実行することもできます。）
●ＣＯＮＴは、動作中のＣＰＵに対して実行することにより、ＰＡＰＡＵＳＥ命令で一時停止しているプログラムを再開させます。
●ＳＴＥＰは、ＰＡＵＳＥ命令により一時停止しているプログラムを再開させる点ではＣＯＮＴと同じですが、つぎの１行を実行した時点で再度プログラムは一時停止します。（ＳＴＥＰモード実行）。

参考

●現在のテストモードをＴＥＳＴコマンドを使って確認することができます。

```
>TEST↵ ..... 入力
1
```

モード	モード名	機能
0	シュミレーションモード	Ｉ／Ｏへの出力をせずにプログラムを実行
1	ポーズモード	ＰＡＵＳＥ命令を有効にする
2	シュミレーション／ポーズモード	上記両方の機能を合わせもつ

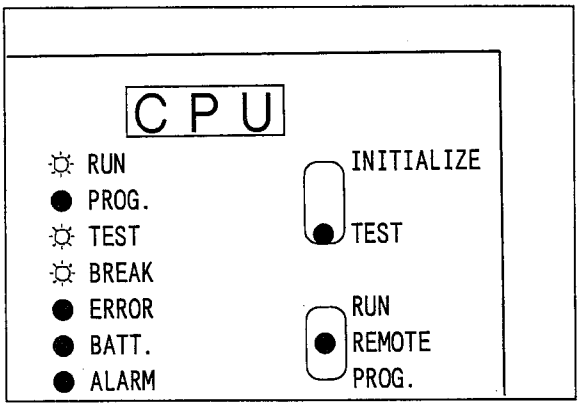
3 プログラムの実行

FP-BASIC編集ソフトからＸＱＴコマンドを実行します。

```
>XQT↵ ..... 入力
```

4 プログラムの中断

スタンプ動作が一巡しても、スタートスイッチ(X1)が受け付けられず、このとき、ＣＰＵの「ＢＲＥＡＫ」ＬＥＤが点灯します。



5 プログラムの再開

FP-BASIC編集ソフトからＣＯＮＴコマンドを実行すると、プログラムが再開されます。このとき、ＣＰＵの「ＢＲＥＡＫ」ＬＥＤは消灯し、プログラムは次のＰＡＵＳＥ命令が実行されるまで続行されます。

```
>CONT↵ ..... 入力
```

ポイント

・プログラムを中断させ、ＰＲＩＮＴ文等で変数の内容や入出力の状態をモニタすればゆっくりチェックできます。

例：PRINT A↵

PRINT INW(WX_0)↵

注意

●CONTは、SHIFT ＋ f･8 キーを押下で入力できます。

(2) １行実行の方法

1 ＰＡＵＳＥ命令の記述

ステップ実行させたいプログラムの始めにＰＡＵＳＥ命令を記述します。

```
1000 FUNCTION MAIN
1050 PAUSE
1010      XQT !2,STAMP      'ｽﾀﾝﾌﾟ工程をﾀｽｸ2で実行
 :
 :
```

2 ＣＰＵの設定

INITIALIZE/TESTモードスイッチをＴＥＳＴ側に倒します。

3 テストモードの設定

「TEST 1」と入力し↵を押します。

```
>TEST 1↵ ..... 入力
```

4 プログラムの実行

FP-BASIC編集ソフトからＸＱＴコマンドを実行します。

```
>XQT↵ ..... 入力
```

5 実行結果

以上の例では、１００５行でＰＡＵＳＥ状態となり、プログラムは１００５行で停止します。
ここで、

```
>STEP↵ ..... 入力
```

とキー入力すれば、プログラムを１行ずつ実行できます。繰返し１行ずつ実行させたい時は、カーソルを「STEP」の行に移動させて↵を押せば繰返し１行ずつ実行します。

注意

●INITIALIZE/TESTモードスイッチがＴＥＳＴ側に倒されていない時は、ＰＡＵＳＥ命令は無効となります。

このコースでは、プログラム作成のテクニックをまとめました。これをマスターすればほとんどのプログラムが楽に作成できます。

テクニック・コース

4 FP−BASICの基本テクニック

4-1 プログラムの基本的な作り方

(1)フローチャートを書く。
機械の動きをフローチャートに表わし、プログラムの実行順序を設計します。（プログラムの作成を容易にし、機械の引き渡し、後日の修正作業時に役立ちます。）

■フローチャートの作成手順
・機械の動きをおおまかに表す概略フローチャートを書く。プログラムの大きな流れを明確にしたり、他の人に機械やプログラムの動きを説明したり確認したりするのに必要です。
・概略フローチャートの各ブロックを詳細フローチャートに書く。機械のおおまかな動きの各部分を詳しく分解したフローチャートで、プログラム作成はこれを基にして行います。

(2)タスク１は各タスクのスタート、停止、非常停止等の入力信号を常に監視し、各タスクの起動（ＸＱＴ）、停止（ＱＵＩＴ）の制御をします。タスク２からは、コンベア、穴開け、ネジ締め、通信等の制御工程ごとにタスクを割り当てます。（最大16タスクまで同時に起動できます。）

(3)初期設定
出力の保持・非保持等パラメータメモリの設定や、使用する変数（数値データ）の宣言などを行います。

■変数宣言
・変数（整数型）はその値の大きさに応じて、
BYTE A,B,C,…… A,B,C = -128～+127
INTEGER A,B,C … A,B,C = -32,768～+32,766
LONG A,B,C …… A,B,C = -2,147,483,548～+2,147,483,647
のように指定します。

(4)プログラム（タスク）の起動方法
タスク起動時は、起動中のタスクを２重に起動するとエラーになりストップしますので、タスク起動中信号を作り、タスクが起動中かどうかチェックしてから起動します。（停止命令ＱＵＩＴは何回でもＯＫ。）

(5)非常停止の方法
非常停止は、タスク１で常時チェックしておき、信号が入った場合は、ＱＵＩＴ命令でタスクを停止させた後、出力をＯＦＦします。タスク起動中信号もオフし再始動可能にします。

参考
市販エディタも使えます
ＦＰ−ＢＡＳＩＣのプログラム（拡張子=.PRG）はテキストファイル（MS-DOSのASCIIファイル）ですので、プログラムの入力編集にはＲＥＤ・Ｍｉｆｅｓ・Ｖｚエディターなどの市販のテキストエディタが使用できます。また、一太郎などのワープロソフトを使用してプログラムを入力して、テキストセーブすることにより作成することもできます。いずれの場合も、拡張子を「.PRG」にしてセーブすれば、FP-BASIC編集ソフトからＬＯＡＤコマンド（f·1キー押下）で読み出すことができます。

4-2 プログラム作成テクニック

(1)プログラムの行番号は１０００から始めよう。
１０００からだと、９９９０まで桁数が変わらずプログラムの書き出し位置がずれないので、見やすく、きれいなプログラムになります。

(2)プログラムにコメントを入れよう。
ＲＥＭ命令、または ' 記号をキーインすれば、これ以後の文字はプログラムではなく、コメントとなります。コメントを入れておれば、プログラムが見やすく、管理が容易になります。自分のためにも、メンテナンスの人のためにも、必ず入れてください。

■コメントの例

```
 :
1080 WAIT 10.00
1090 ON Y_&M20
2000 '************ STAMP ****************
2010 '
2020 WAIT SW(X_&M0)       'WAIT START SW
2030 ON Y_&M10            'STAMP DOWN
 :
```

(3)戸籍を作ろう
プログラムの最初の部分に　で説明したコメント文で、
・プログラム名
・目的、注釈
・作成者
・作成日
等を書いておけば、後日たいへん役立ちます。

```
1000 ' JOB_1  ｺﾝﾍﾞｱﾓｰﾀ制御
1010 ' 93-05-30              作成者：山本一郎
1020 FUNCTION JOB_1
 :
 :
```

(4)プログラムの保存を忘れないよう。
作成したプログラムは必ず、フロッピーディスクに保存しよう。せっかく開発したプログラムが消えてしまいます。誤って電源が切れることがありますので、完成していなくても定期的に保存しておいたほうが安全です。

■プログラムの保存と読み出し
・SAVE "A:ファイル名"・・・・・保存
・LOAD "A:ファイル名"・・・・・読み出し

(5)入出力は名前に置き換えよう。
・入力、出力は変数として名前を登録しておけば非常に便利です。
・こうしておけば、入出力番号の変更があっても、登録部分を替えるだけで済みます。そうでない場合にはプログラム全体を変更しなければなりません。
・番号でなく、名前でプログラムできるためプログラムの理解が容易です。

■入力名・出力名の置き換え例

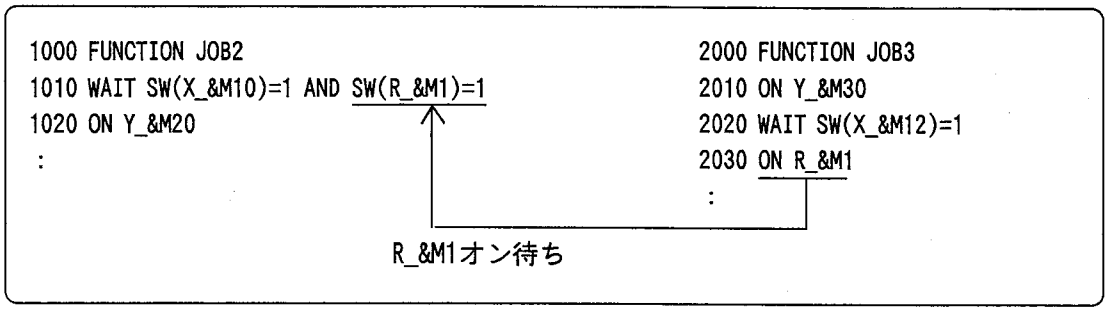
```
 :
1000 INTEGER LS1,MOTOR1
1010 LS1=&M10 ;MOTOR1=&M20
 :
1200 WAIT SW(X_LS1)
1210 ON Y_MOTOR1
 :
```

注意 入出力置き換え

●入出力は必ず変数に置き換えてプログラムしないと、入出力の変更が大変です。

(6)ＡＮＤ条件を使ってタイミングを合わせよう。
タスク間の同期やインターロックは、ＡＮＤ条件を使って、条件が揃うまで実行を待機させられます。また、インターロックとして使うこともできます。

■タイミング合わせ（タスク間同期）の例



(7)スピードアップテクニック
・１行に複数の命令を書こう。
；（セミコロン）記号を入れれば、１行に命令がたくさん書けます(ﾏﾙﾁｽﾃｲﾄﾒﾝﾄ)。
・タスク数を少なくしよう。
不要なタスクを停止させ、実行タスク数をできるだけ減らしましょう。
・ＷＡＩＴで待とう。
ＩＦ命令でも待てますが、ＷＡＩＴ命令で待った場合ＷＡＩＴ中はそのタスクは実行されませんのでタスクの回転が早くなります。

■マルチステイトメントの例

```
ON Y_&M10
WAIT 1.00
OFF Y_&M10
```

⇩

```
ON Y_&M10;WAIT 1.00;OFF Y_&M10  '1SEC PULSE
```

(8)プログラムの流れはシンプルにしよう。
ＩＦ命令はできるだけ使わずに、ＷＡＩＴ命令で流れをすっきりさせる。ＩＦ命令を多用するとラダー命令と同じプログラムとなり、プログラムの作成、チェックがたいへんです。（シリンダーが往復運動をして止まらなくなります。）もし使う時は判定条件をうまく考え流れをスッキリさせてください。

注意

●IF SW(X_&M10) AND A=1 THEN ～は可能だが、
　　……変数
WAIT SW(X_&MO) AND A=1は不可。WAITの場合は、WAIT SW(X_&MO) AND SW(X_&M1)のように、接点だけが許可されます。

(9)ＷＡＩＴですっきり並べよう。

プログラム作成時、ＷＡＩＴ命令を先頭にし、１行に複数命令を並べると、見やすく、プログラムチェックも楽にできます。

```
1000 WAIT SW(X_^M0);ON Y_&M10                ' スタートスイッチで部品の右移動
1010 WAIT SW(X_&M1);OFF Y_&M10;ON Y_&M11     ' 右移動完了信号で右移動停止、上昇開始
1020 WAIT SW(X_&M2);OFF Y_&M11;ON Y_&M12     ' 上昇完了信号で上昇停止、ヒータ加熱
1030 WAIT 3.5       ;ON Y_&M12               ' 3.5秒加熱後、部品の排出モータ始動
1040 WAIT SW(X_&M3);OFF Y_&M12               ' 排出完了信号で排出モータ停止
```

以上のような方法でプログラムすることにより、

・プログラムの長さが、短く、すっきりして、１頁にたくさんのプログラムが書ける。
・機械の動きを１行単位でステップ化して表わせるため、理解とチェックが簡単になる。
・トレース時またはＴＳＴＡＴ、ＭＯＮＴＳ命令でモニタ時に、実行番号がちらつきにくく見やすい。
・プログラム実行時間が速くなり、プログラムメモリも節約できる。

4-3 プログラム作成の注意点

(1)必ず FUNCTION から始まり FEND で終わる。

(2)命令の綴りを間違えないこと。ファジー制御されていませんので１字間違えても受け付けられません。

(3)変数、ラベルでは使えない綴りがありエラーが発生します。命令と同じ綴りや、ＰＵＬＳＥ等Ｐで始まる名前はエラーになります。文法エラーの原因が分からない時は一度名前を変えてみてください。

(4)０とＯを間違えないよう注意してください。ゼロ（０）とオー（Ｏ）は非常に似ていますが間違えるとエラーになります。

(5)プログラム保存時ファイル名に注意
同じファイル名があるとプログラムが壊されます。

4-4 一般ＢＡＳＩＣ命令（Ｎ８８ＢＡＳＩＣ）との違い

ＦＰ－ＢＡＳＩＣの命令のほとんどは、一般のＮ８８ディスクＢＡＳＩＣと同じですが、追加した特殊命令以外で異なるものがあります。

内容	Ｎ８８ディスクＢＡＳＩＣ	ＦＰ－ＢＡＳＩＣ
マルチステートメント記号	：	；
ラベル記号、表記法	GOTO *LOOP ～ *LOOP	GOTO LOOP ～ LOOP:
実行命令	ＲＵＮ	ＸＱＴ
プリント	ＰＲＩＮＴ　または　？	ＰＲＩＮＴ　だけ
変数名　文字数	４０文字以内	８文字以内　１字目Ｐは禁止
フロッピードライブ指定	LOAD "1:○○"	LOAD "A:○○"

4-5 キー入力のスピードアップ

・ＫＥＹ
　ファンクションキーに入力を割り付けます。
　例：KEY 1,ON Y_&M
　　f・1 キーに"ON Y_&M"を登録する。

・ＫＥＹ　ＬＩＳＴ
　ファンクションキーへの割り付け内容を表示します。
　例：KEY LIST

・KEY 1,ON Y_&M↵と入力するだけで、次からは f・1 キーを押すだけでON Y_&Mまでキーインしたのと同じ働きをします。

```
>KEY 1,ON Y_&M↵
        ..... f・1 キーに"ON Y_&M"を登録する
```

・RETURNキーまで含んで登録するときは、～ ¥rと入力すればRETURNキーと同じ働きをします。

```
>KEY 1,・・・・・¥r↵
          ..... 登録する入力の内容
```

・KEY 1～10は f・1 ～ f・10 に、KEY 11～20は SHIFT を押した状態の f・1 ～ f・10 に対応します。

・キー登録は、KEY LIST↵と入力することにより確認できます。

```
>KEY 1,・・・・・¥r↵
      1,"LOAD ¥"
      2,"AUTO"
      3,"TSTAT¥r"
      :
      :
```

注意

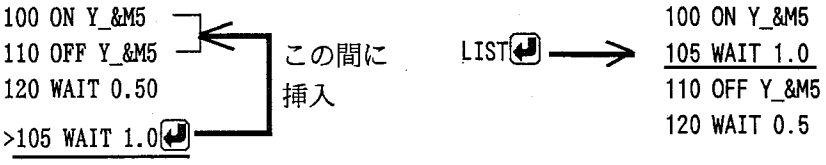
●以下の登録は変更しないでください。

ＴＳＴＡＴ …… f・3 キー
ＬＩＳＴ …… f・4 キー
ＸＱＴ …… f・5 キー
ＣＯＭＰＩＬＥ …… f・6 キー
ＤＷＮＬＤ …… f・7 キー
ＱＵＩＴ　ＡＬＬ …… f・10 キー

4-6 プログラム編集テクニック

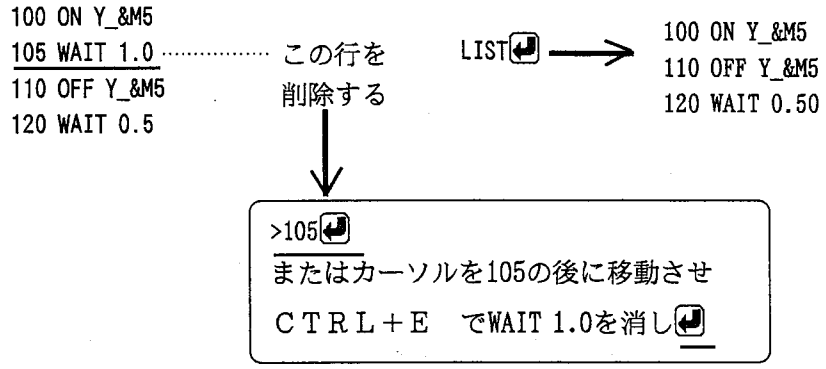
(1)行の挿入

行間にプログラムを挿入するには、挿入したい行間の行番号で入力します。



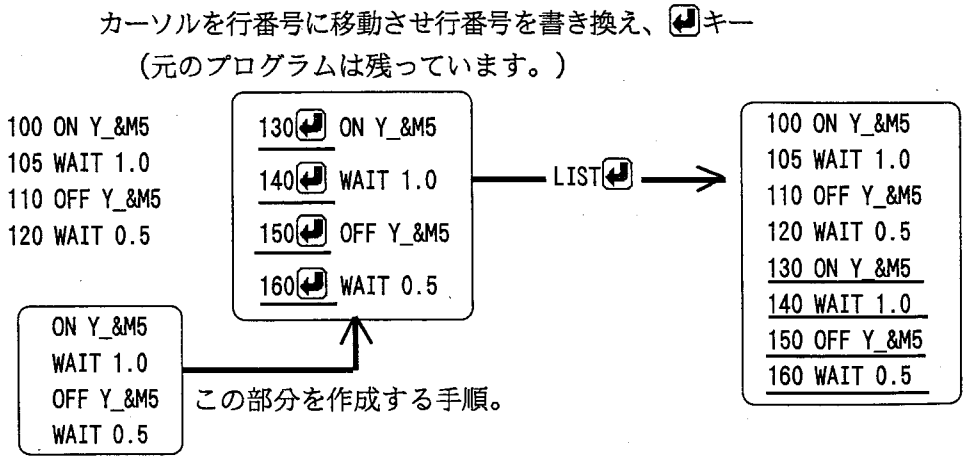
(3)行の削除

行間の削除は行番号だけにし、↵キーでできます。



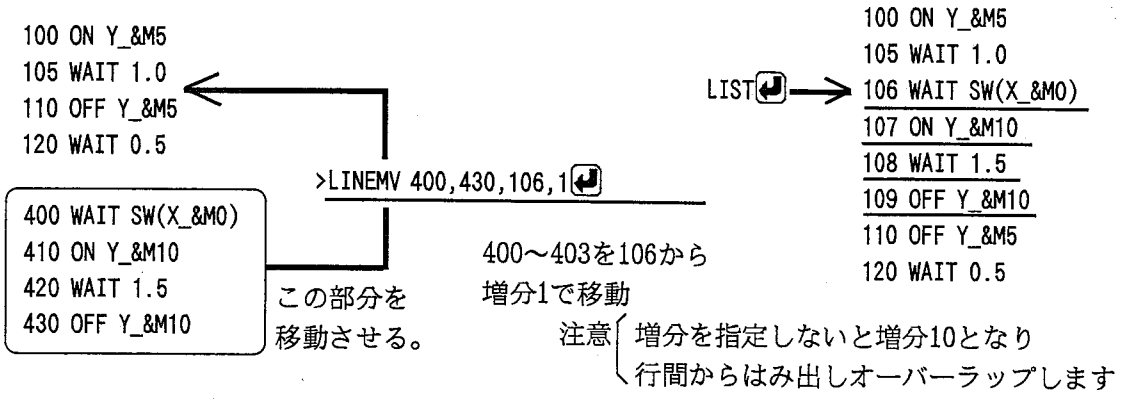
(3)繰り返しは行番号を書き換えて

同様なプログラムを入力する必要がある場合、すでに入力済の行番号を書き換えれば、プログラム全体を打ち込まなくても済みます。



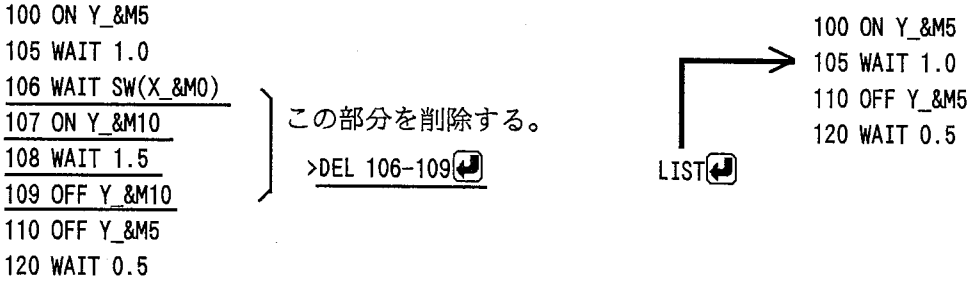
(4)ＬＩＮＥＭＶでブロック移動。

ＬＩＮＥＭＶ命令を使えば、プログラムをブロック単位で移動できます。



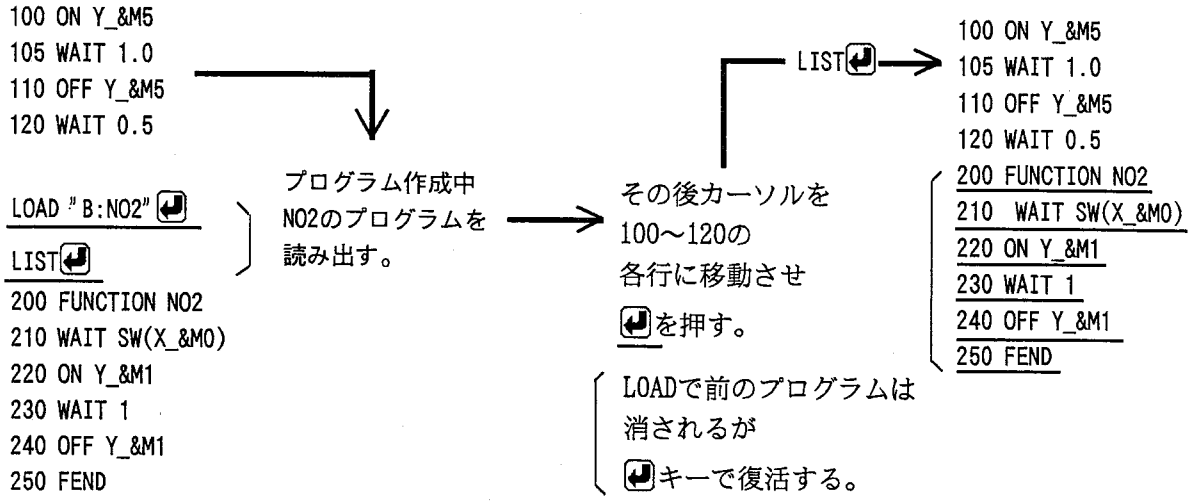
(5)ＤＥＬでプログラムをブロック削除。

ＤＥＬ命令でプログラムをブロックで削除できます。



(6)プログラムの結合。

作成中のプログラムとフロッピーディスクの中のプログラムをつなぎ合わせ１本のプログラムをつくる。



5　FP−BASICの応用テクニック

5-1　複数個の非常停止プログラムの作り方

●非常停止が１つの場合
　ＷＡＩＴとタスク起動中フラグを使用して非常停止プログラムを作成します（「２−３」　「２−４」参照）。
●非常停止が複数必要な場合
　ＩＦとタスク起動中フラグを使用して非常停止プログラムを作成します。
　(1) IF命令を使い同時に何点でもチェックできるようにする。
　(2) タスク起動中フラグを使い２重起動を防ぐ。

1　複数個非常停止の基本パターン

一般に１つの工程制御プログラムの非常停止は以下の例示プログラムの＿＿（下線）ように２行で記述できます。

ポイント

●起動中フラグR1を使って、<u>ＸＱＴ命令の２重起動を防いでいます。</u>

X_&ME ：非常停止スイッチ
X_&MF ：非常停止復帰スイッチ
SUB_1 ：工程制御プログラム名
Y_&M10：工程制御プログラムの出力
変数R1：工程制御プログラムのタスク起動中フラグ

```
1000 FUNCTION MAIN
1010 INTEGER R1                                      ・変数宣言（R1が整数使用できるようになる）
1020 XQT !2,JOB_1;R1=1                               ・タスク２を起動しフラグR1=1に設定
1030 IF SW(X_&M0)=1 AND R1=1 THEN QUIT !2;OFF Y_&M10;R1=0
1040 IF SW(X_&M0)=0 AND R1=0 THEN XQT !2,JOB_1;R1=1
1050 GOTO 1030
1060 FEND
```

（1040行へ）
非常停止スイッチ(X0)
ＯＮ→ＯＦＦ

R1=1のときだけ
・タスク２を起動する
・フラグR1=1に設定する

（1030行へ）
非常停止スイッチ(X0)
ＯＦＦ→ＯＮ

R1=0のときだけ
・タスク２を停止する
・Y10をＯＦＦする
・フラグR1=0に設定する

一般に１つの工程制御プログラムの非常停止はこの例のように２行で記述できます。

2 非常停止が２つのプログラム例

非常停止が２つある場合は、先の基本パターンを２組使います。ＭＡＩＮプログラムの中で常にタスク起動中フラグR1･R2を監視していることに注目してください。

ポイント

●起動中フラグR1･R2を使って、ＸＱＴ命令の２重起動を防いでいます。

```
>LIST
1000 FUNCTION MAIN
1010 INTEGER R1,R2
1020 XQT !2,JOB_1;R1=1
1030 XQT !3,JOB_2;R2=1
1040 IF SW(X_&M3)=1 AND R1=1 THEN QUIT !2;OFF Y_&M10;R1=0
1050 IF SW(X_&M3)=0 AND R1=0 THEN XQT !2,JOB_1;R1=1
1060 IF SW(X_&M4)=1 AND R2=1 THEN QUIT !3;OFF Y_&M11;R2=0
1070 IF SW(X_&M4)=0 AND R2=0 THEN XQT !3,JOB_2;R2=1
1080 GOTO 1040
1090 FEND
1100 FUNCTION JOB_1
1110 WAIT SW(X_&M1)
1120 ON Y_&M10
1130 WAIT 0.50
1140 OFF Y_&M10
1150 WAIT 0.50
1160 GOTO 1120
1170 FEND
1180 FUNCTION JOB_2
1190 WAIT SW(X_&M2)
1200 ON Y_&M11
1210 WAIT 0.50
1220 OFF Y_&M11
1230 WAIT 0.50
1240 GOTO 1200
1250 FEND
```

タスク１の非常停止（R1を監視）

タスク２の非常停止（R2を監視）

工程制御プログラム１

工程制御プログラム２

5-2 インターロックプログラムの作り方

●インターロック

同時にＯＮしてはいけない出力、または同時に実行してはいけないプログラム等には、IF命令やＷＡＩＴ命令を使って、インタロックをとります。

```
1020 IF SW(X_&M0)=1 AND SW(Y_&M11)=0 THEN ON Y_&M10
1030 IF SW(X_&M1)=1 AND SW(Y_&M10)=0 THEN ON Y_&M11
```

ポイント

●Y10、Y11が、同時にオンしないようにＡＮＤ命令でインタロックをとっています。

5-3 手動プログラムの作り方

●手動プログラム

IF THEN ELSEを入力の数だけ並べてください。

例: 3010 IF SW(X_&M1)=1 THEN ON Y_&M10 ELSE OFF Y_&M10
3020 IF SW(X_&M2)=1 THEN ON Y_&M11 ELSE OFF Y_&M11

ＦＰ１では特別に次の命令が準備されております（こちらの方が実行スピードが約３倍早くなります）。

例： 3010 OUT &M10,&M1……X1オンでY10オン、X1オフでY10オフ
3020 OUT &M11,&M2……X2オンでY11オン、X2オフでY11オフ

X_&M0……手動／自動切り替えスイッチ
X_&M1･X_&M2･･･手動スイッチ

変数M……タスク起動中フラグ
M=1：自動プログラム起動中
M=0：手動プログラム起動中

```
>LIST
1000 FUNCTION JOB1
1010 INTEGER JX,SX;JX=0,SX=0
1020 IF SW(X_&M0)=0 AND JX=0 THEN QUIT !3;SX=0;XQT !2,JIDOU;JX=1
1030 IF SW(X_&M0)=1 AND SX=0 THEN QUIT !2;JX=0;XQT !3,SHUDOU;SX=1
1040 GOTO 1020
1050 FEND
```

自動／手動切り替え（Mを監視）

手動運転プログラム

手動スイッチごとに
ＩＦ文で出力をオン、オフします。

```
3000 FUNCTION SHUDOU
3010 IF SW(X_&M1)=1 THEN ON Y_&M10 ELSE OFF Y_&M10
3020 IF SW(X_&M2)=1 THEN ON Y_&M11 ELSE OFF Y_&M11
 :
 :
3040 FEND
```

自動運転プログラム

```
2000 FUNCTION JIDOU
2010 ON Y_&M10
2020 WAIT 1.00
2030 OFF Y_&M10
2040 WAIT 1.00
2050 GOTO 2010
2060 FEND
```

5-4 機械の動作時間チェックプログラムの作り方

●ＯＮＥＲＲ

エラー発生時に実行を移すサブルーチンを指定します。

例：ONERR ER1

……エラー時にサブルーチンＥＲ１に実行を移す。

●ＴＭＯＵＴ

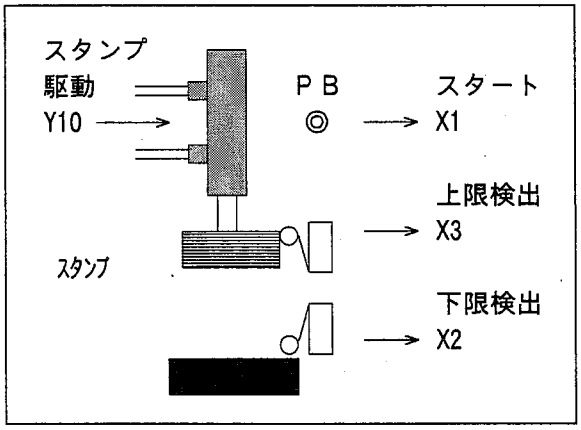
次に実行するＷＡＩＴ命令のタイムアウト時間を指定します。

例：TMOUT 5

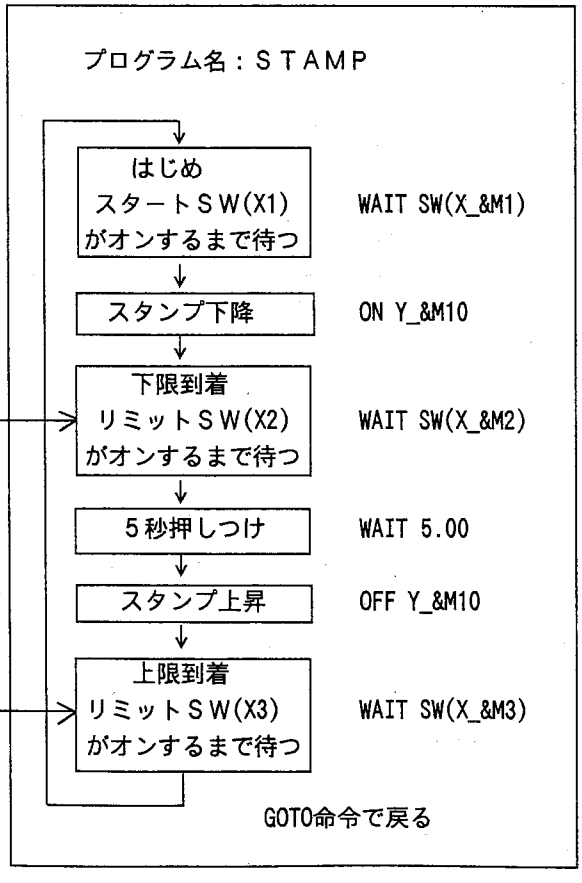
……次に実行するＷＡＩＴ命令のタイムアウトを５秒に設定する。

1 タイムアウトエラー処理の例

■スタンプ制御



●フローチャート



TMOUT 5
タイムアウト５秒を設定
５秒以内にX_&M2がオンしないと
タイムアウトエラーになる。

TMOUT 3
タイムアウト３秒を設定
３秒以内にX_&M3がオンしないと
タイムアウトエラーになる。

2 プログラム例

```
>LIST
1000 FUNCTION STAMP
1010 INTEGER ERNO
1020 ONERR ER1 ·····エラー時にサブルーチンER1を実行する設定
1030 WAIT SW(X_&M1)
1040 ON Y_&M10
1050 TMOUT 5;ERNO=2 ·····タイムアウトを５秒に設定し、ERNO=2をセット
1060 WAIT SW(X_&M2)=1 ·····下限リミットスイッチ入力のＷＡＩＴ命令
1070 WAIT 3.00
1080 TMOUT 3;ERNO=4 ·····タイムアウトを３秒に設定し、ERNO=4をセット
1090 WAIT SW(X_&M3)=1 ·····下限リミットスイッチ入力のＷＡＩＴ命令
1100 TMOUT 0 ·····タイムアウトを無効に設定
1110 GOTO 1030
1120 ER1:
1130 IF ERR(0)<>1004 THEN END ·····タイムアウトエラー以外なら終了
1140 SELECT ERNO
1150  CASE 2 ·····下限リミットスイッチ入力タイムアウトで
1160   ON Y_&M14 ···  Y_&M14をＯＮする
1170  CASE 4 ·····上限リミットスイッチ入力タイムアウトで
1180   ON Y_&M15 ···  Y_&M15をＯＮする
1190  SELEND
1200 ECLR ·····エラー割り込み解除
1210 RETURN
1220 FEND
```

3 エラー処理サブルーチンの動作

エラーサブルーチン動作は次の通りです。

・エラー内容がタイムアウトエラーかどうかを確かめ、もしそうでなければプログラムの実行を終了（ＥＮＤ命令）します。

・エラー内容がタイムアウトエラーなら、変数ERNOの値に応じて、Y_&M14またはY_&M15をオンすることにより、タイムアウトが発生したのが下限リミットスイッチか上限リミットスイッチかを知らせます。

命令の説明

●TMOUT 0は、タイムアウトの設定を無効にします。また、TMOUT命令で変数を使用することもできます（例：TMOUT A）。

●変数ERNOは、タイムアウトエラー発生時にプログラムのどの個所を実行中かが分かるようにするのに使用されています。

●ERR(0)は、発生しているエラーのエラーナンバを知るためのシステム関数です。()の中には常に「0」を記述します。

●ECLRは、エラー割り込みを解除します。

●SELECTは、指定された値に応じたCASEブロックを実行します。ここでは、ERNOの値により、CASE 2、CASE 4と実行内容が変わります。SELECT～CASEは、SELENDで終わります。Ver.2.0未満のFP-BASIC編集ソフトおよびＦＰ１ ＢＡＳＩＣ ＣＰＵでは使用できません。

●RETURNは、サブルーチンＥＲ１の終了を宣言します。

●ENDは、プログラムを終了させます。

5-5 出力の保持／非保の設定方法

●ＰＲＭ

ＢＡＳＩＣタイプＣＰＵのパラメータメモリの内容を変更します。

例：>PRM 15,0↵

……パラメータ15の値を「0」に設定する

●ＰＲＭＣＬＲ

パラメータメモリを初期化します。

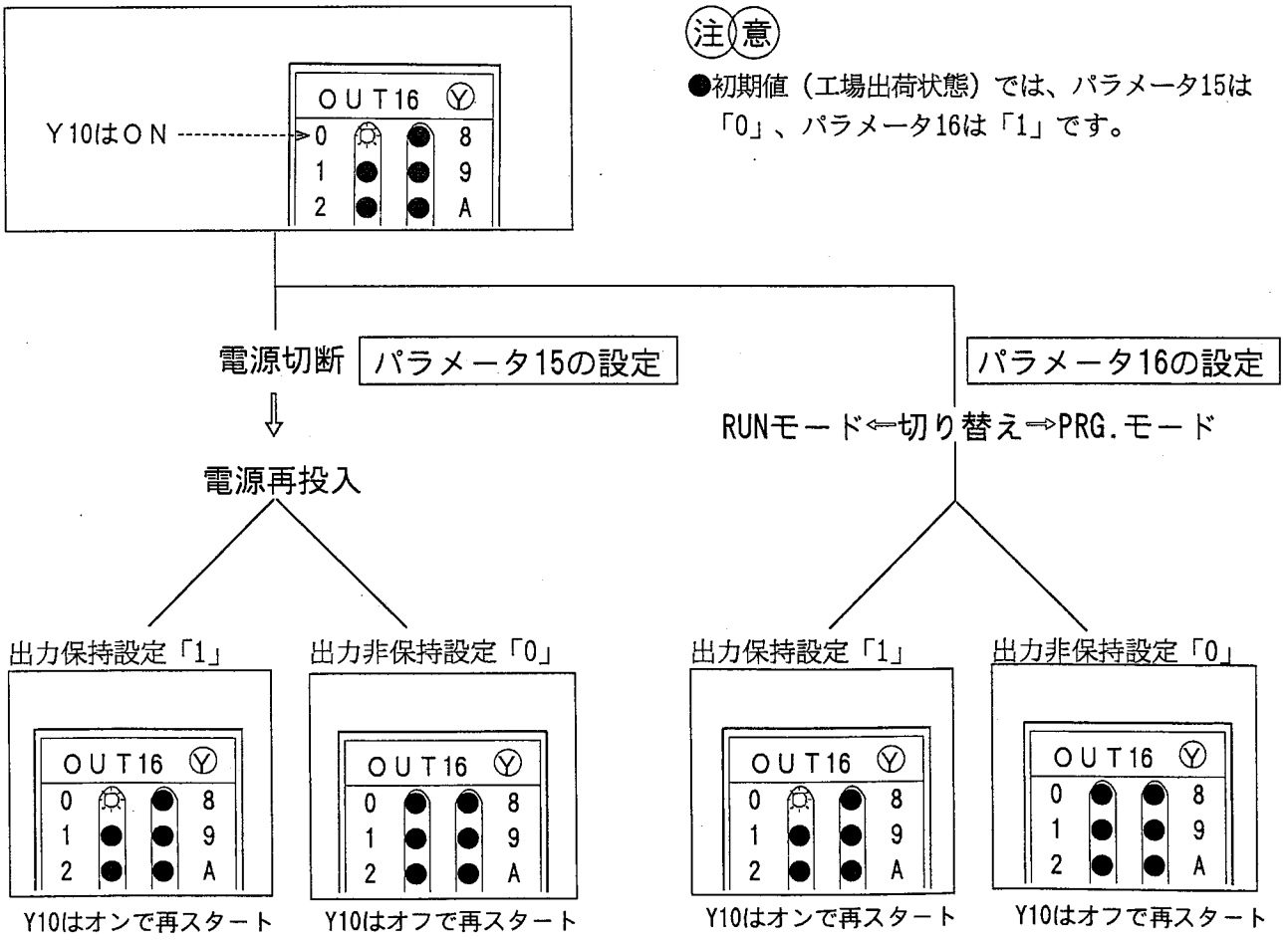
●ＰＲＭ ()

パラメータのメモリを読み出します。

例：PRINT PRM(15)

……パラメータ15の値を表示する。

1 出力の保持と非保持



注意

●初期値（工場出荷状態）では、パラメータ15は「0」、パラメータ16は「1」です。

2 パラメータメモリの読み出し方法

出力の保持・非保持設定は、パラメータメモリ15と16に割り当てられています。パラメータメモリの読み出しは次のようにして行います。

```
>PRINT PRM(16)↵ ……パラメータ16の値を表示
1 ……値は「1」なのでプログラムモードで出力保持
```

3 パラメータメモリの設定方法

電源切断時またはプログラムモードの出力を保持から非保持に変更する方法を紹介します。電源切断時の出力の保持・非保持はパラメータメモリ15で設定されていて、初期設定ではその値は「0」（非保持）になっています。プログラムモード時の出力の保持・非保持はパラメータメモリ16で設定されていて、初期設定ではその値は「1」（保持）になっています。保持から非保持に設定を変更するには、パラメータメモリの値を「1」から「0」に変更すればよいわけです。パラメータメモリの値は、次のようにして変更します。

```
>PRM 15,0↵ ……パラメータ15、16の値を
>PRM 16,0 「0」に設定する
```

4 全パラメータの表示プログラム

パラメータメモリの設定状態をまとめてみたい場合は、次のようなプログラムが便利です。実行すると、パラメータ0～55までを画面表示します（↑ ↓キーで画面をスクロール可能）。

```
>LIST↵
1000 FUNCTION DISPRM
1010 INTEGER I;I=0
1020 FOR I=0 TO 55
1030 PRINT I,"=",PRM(I)
1040 NEXT I
1050 FEND
```

■プログラム実行例

```
0=128
1=0
2=0
3=0
4=0
 :
 :
54=128
55=0
```

ポイント

- ●パラメータメモリ15は電源切断時の出力の保持・非保持を設定し、パラメータメモリ16はプログラムモード時の出力の保持・非保持を設定します。
- ●パラメータメモリ15・16の初期設定（出荷時設定）は、15：0（非保持）、16：1（保持）に設定されています。

注意

- ●プログラムの始めの方に次のような行を追加すれば、プログラム起動により、自動的に出力を非保持に設定します。

```
 :
1100 PRM 15,0
1110 PRM 16,0
 :
```

注意

- ●ＦＰ１の場合は、PRM16だけです。PRM15はありません。

注意

- ●ＦＰ１の場合は、パラメータ0～35を読み出します。
- ●パラメータはＤＵＭＰＰコマンドを実行することによっても状態表示と内容変更ができます。

5-6 タイマの外部設定

(1) 切り替えスイッチによるタイマ設定

1 設定値が電源オフで消えてもよい場合

タイマ設定値が電源を切断した時点で消えてしまってもよい場合は、設定値の格納場所に変数を使用します。変数は、電源の切断により初期化されますので、タイマ設定値は電源投入により毎回初期化されます。

```
>LIST↵
1000 FUNCTION SET_TM1
1010 INTEGER A
1020 WAIT SW(X_&M0)=1
1030 IF SW(X_&M1)=1 THEN A=2
1040 IF SW(X_&M2)=1 THEN A=4
1050 IF SW(X_&M3)=1 THEN A=6
 :
 :
1500 ON Y_&M10
1510 WAIT A
 :
 :
```

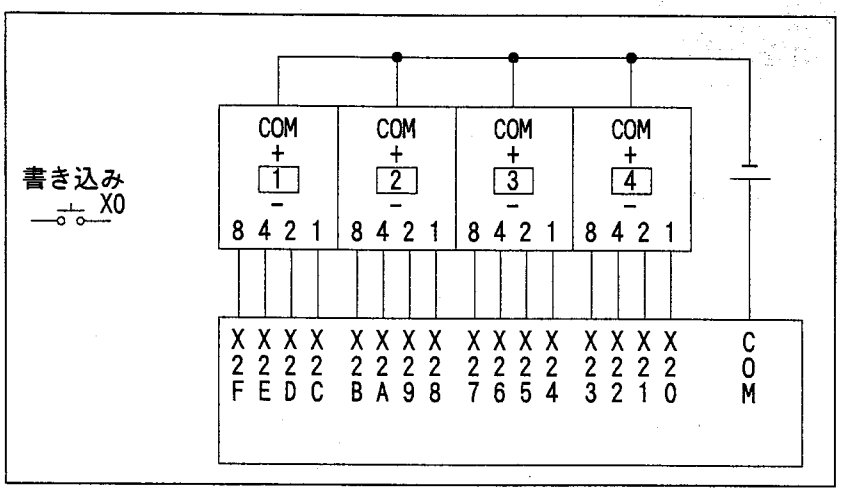
2 設定値が電源オフでも保持される必要がある場合

タイマ設定値が電源を切断した時点で消えてしまうと困る場合は、設定値の格納場所にデータメモリを使用します。データメモリは、パラメータ8で保持に設定されていて、電源を切断しても保持されますので、タイマ設定値は電源切断により初期化されることはありません。

```
>LIST↵
1000 FUNCTION SET_TM2
1010 INTEGER A
1020 WAIT SW(X_&M0)=1
1030 IF SW(X_&M1)=1 THEN OUTW DT_0,2
1040 IF SW(X_&M2)=1 THEN OUTW DT_0,4
1050 IF SW(X_&M3)=1 THEN OUTW DT_0,6
 :
 :
1500 A=INW(DT_0)
1510 ON Y_&M10
1520 WAIT A
 :
 :
```

(1) デジスイッチによるタイマ設定

4桁のデジスイッチが入力X20～X2Fに接続されていて、書き込みスイッチＸ０オンでデジタルスイッチの値をタイマの値として書き込みます。



1 設定値が電源オフで消えてもよい場合（毎回設定値を書き換える）

タイマ設定値が電源を切断した時点で消えてしまってもよい場合は、設定値の格納場所に変数を使用します。変数は、電源の切断により初期化されますので、タイマ設定値は電源投入により毎回初期化されます。

```
>LIST↵
1000 FUNCTION SET_TM3
1010 INTEGER A
1020 WAIT SW(X_&M0)
1030 A=BIN(INW(WX_2))
:
:

1500 ON Y_&M10
1510 WAIT A
:
:
```

2 設定値が電源オフでも保持される必要がある場合

タイマ設定値が電源を切断した時点で消えてしまうと困る場合は、設定値の格納場所にデータメモリを使用します。データメモリは、パラメータ8で保持に設定されていて、電源を切断しても保持されますので、タイマ設定値は電源切断により初期化されることはありません。

```
>LIST↵
1000 FUNCTION SET_TM4
1010 INTEGER A
1020 WAIT SW(X_&M0)=1
1030 OUTW DT_0,BIN(INW(WX_2))
:
:
1500 A=INW(DT_0)
1510 ON Y_&M10
1520 WAIT A
:
:
```

注意

●タイマの設定は、16進数(Binary)で設定しなければなりません。

●書き込み入力X0でAに、書き込み入力X1でBに、各々デジスイッチの値を格納すれば複数の設定が可能です。

5-7 自己保持回路を１６個一度に作る方法

●ＯＵＴＷ

ワード（16点）単位で出力をオン、オフします。

例：OUTW WY_2,0

……Y_&M20～2Fをオフします。

●ＩＮＷ ()

ワード（16点）単位で出力のオン、オフ状態を読み出します。

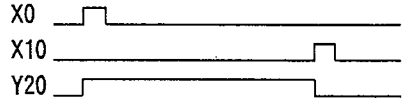
例：INW(WX_0)

……X_&0～Fの状態を読み出します。

１６個の入力信号それぞれがオンしたことを記憶、保存しておくためのプログラム方法。

```
Ｘ０   ＯＮ で Ｙ２０ ＯＮを保持
Ｘ１０ ＯＮ で Ｙ２０ ＯＦＦ

Ｘ１   ＯＮ で Ｙ２１ ＯＮを保持
Ｘ１１ ＯＮ で Ｙ２１ ＯＦＦ
Ｘ２   ＯＮ で Ｙ２２ ＯＮを保持
Ｘ１２ ＯＮ で Ｙ２２ ＯＦＦ
       ：
       ：
ＸＦ   ＯＮ で Ｙ２Ｆ ＯＮを保持
Ｘ１Ｆ ＯＮ で Ｙ２Ｆ ＯＦＦ
```




```
1000 FUNCTION JOB1
1010 OUTW WY_2,0                          'Y20-2Fクリア
1020 OUTW WY_2,INW(WY_2) OR INW(WX_0)     'X0-FでY20-2Fをオン
1030 OUTW WY_2,INW(WY_2) AND INV(INW(WX_1))  'X10-1FでF20-2Fをオフ
1040 GOTO 1020
1050 FEND
```

＊Y20-2Fをオフさせるのが他のプログラムの時は１０３０行目は不要です。

5-8 サブルーチンの作り方と長所

●ＧＯＳＵＢ

サブルーチンに実行を移します。

例：GOSUB <行番号>

<行番号>から始まるサブルーチンに実行を移します。

GOSUB <ラベル>

<ラベル>から始まるサブルーチンに実行を移します。

●サブルーチンの効用

(1)同じ処理を何度も書かなければならない時、別のところにプログラムを書き、「ＧＯＳＵＢ　～」で実行させれば、簡単に何度でも実行させられます。

(2)サブルーチンを使用して、実際の実行部分を別のところに書き、その名前で実行順を書けば、プログラムの動きを大まかに表現でき、、プログラムがすっきりし、見やすくなります。

(1) 何度も実行させる場合

1 サブルーチンの呼び出し

```
:
1010 WAIT WS(X_&M0)
1020 GOSUB TENTOU
1030 WAIT SW(X_&M1)
1040 GOSUB TENTOU
:
```

2 サブルーチンプログラム

```
2000 TENTOU:
2010 ON Y_&M10
2020 WAIT 2.00
2030 OFF Y_&M10
2040 RETURN
```

ポイント

●サブルーチンの最後には必ずRETURNを記述します。RETURNを忘れると、もとの実行部分に戻れません。

(2) 大まかな動きを表現する場合

1 サブルーチンの呼び出し

```
:
:
1010 GOSUB SHOKIKA        'initialize
1020 GOSUB FUNON          'fan start
1030 GOSUB HEATERON       'heater start
:
:
```

2 サブルーチンプログラム

```
2000 SHOKIKA:     'initialize
2010 PRM 15,1
:
:
2090 RETURN
3000 FANON:       'fan motor ON
3010 ON Y_&M11
:
:
3090 RETURN
4000 HEATERON:    'heater ON
4010 ON Y_&M12
:
:
4090 RETURN

:
:
```

ポイント

- GOSUB、GOTOで実行行を指定するとき、ラベルを指定する方法のほかに、行番号を指定することができます。
- ラベルには８文字までの英数字を使用することが できますが、先頭文字は必ず「英字（アルファベ ット）」でなければなりません。なお、飛び先のラベルには、末尾に「:（コロン）」を付けます。
- 実際のプログラムでは行番号の方が簡単です。

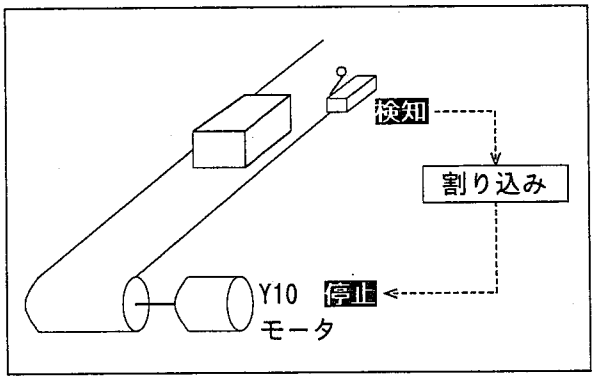
5-9 割り込み処理による高速応答

ＦＰ－ＢＡＳＩＣには割り込みユニットを使用した高速な割り込み処理が用意されています。この割り込み処理は、ＷＡＩＴ命令が実行中であっても即時に実行され、リアルタイム（１ｍｓ程度）な応答が求められる制御に威力を発揮します。

1 割り込み処理の説明

コンベアモータ上のワークを検出して、割り込みですぐ停止させる場合です。

●制御動作



検知スイッチ(X_&M20)オンによる
割り込み(INT0)発生により実行

⇒

●フローチャート

●通常プログラム　MAIN

起　動
↓
割り込み定義　　ON INT(0)
↓
コンベア
モータＯＮ　　ON Y_&M10

●割り込み処理①プログラム　TASK_INT

モータ停止　　OFF Y_&M10
↓
割り込み解除　　INT(0) CLR

注意

●割り込みユニットはスロット２に挿入します（ＣＰＵ→16点入力ユニット→16点出力ユニット→割り込みユニットの順）。

●検出スイッチは、割り込みユニットのINT0(X_&M20)に接続します。

2 プログラム例

ＯＮ　ＩＮＴ（）命令による入力割り込みを定義すれば、定義した入力があると、割り込み処理タスクが実行されます。しかも、ＷＡＩＴ命令が実行中の割り込みであっても、即時に割り込み処理タスクが実行されます。

```
1000 FUNCTION MAIN
:
1020 ON INT(0)  TASK_INT              '割り込みの定義
1030 INT(0) ON                        '割り込み許可
:
1100 ON Y_&M10                        'コンベアモータ始動
1110 WAIT SW(Y_&M10)=0
:
:
1190 FEND
:
:
2000 FUNCTION TASK_INT·····················割り込み処理プログラム
2010 OFF Y_&M10                       'コンベアモータ停止
2030 INT(0) CLR                       '割り込み信号をクリヤする
2040 FEND
```

注意

- ●割り込みを禁止させたいときは、INT(0) OFFの命令を実行させてください。
- ●割り込み処理は別のタスクに記述し、一般にプログラムの中で割り込みを解除しておきます。
- ●割り込み処理ルーチンではＷＡＩＴ命令は使用できません。
- ●割り込み処理ルーチンプログラムの最後にRETURN命令は必要ありません。FENDを実行すると、自動的にもとのプログラム実行にもどります。

5-10 分割コンパイル

長いプログラムの動作確認、プログラム修正を行う際、１本のプログラム（ファイル名がひとつ）で作業すると、たった１行の変更でもプログラム全体をＣＯＭＰＩＬＥするときは全プログラムをＣＯＭＰＩＬＥする必要があり変換時間が長くなり、作業効率が悪くなります。
こんな時、プログラムをいくつかの小さなプログラムに分割して作成し、それぞれにプログラム名を付け、フロッピーディスクに保存しておけば、修正のあったプログラムだけをＣＯＭＰＩＬＥするだけで済みますので、作業効率がよくなります。
汎用性のあるプログラムなども、登録しておき、分割コンパイル機能を使い、必要に応じて呼び出し使うこともできます。

1 使用方法

分割コンパイルのために、次の準備をしてください。

(1)FUNCTION～FENDの各プログラムをフロッピーディスクにＳＡＶＥ（保存）しておきます。

(2)最初に起動させるプログラムの中で、グローバル変数（全プログラムで共通に使用する変数）をＥＮＴＲＹ命令で登録する。（各プログラムの中だけでしか使わないローカル変数は何もしなくて良い。）

例：1000 FUNCTION JOB1
　　1010 ENTRY REAL ONTIME
　　　　……ONTIMEをREAL（実数）のグローバル変数として定義

また、この変数を使用するプログラムの側では、ＥＸＴＥＲＮ命令でグローバル変数を使用することを宣言します。

例：2000 FUNCTION JOB2
　　2010 EXTERN REAL ONTIME
　　　　……グローバル変数のONTIMEを使用しますという宣言

(3)他のプログラムを起動する時は、そのプログラム名の中であらかじめＥＸＴＥＲＮ命令を使用して宣言強い起きます。

例：1000 FUNCTION JOB1
1010 ENTRY REAL ONTIME
1020 EXTERN FUNCTION JOB2
1030 EXTERN FUNCTION JOB3
1040 XQT !2,JOB2;XQT !3,JOB3

FUNCTION JOB2とFUNCTION JOB3の為の起動の宣言
（実際の起動はＸＱＴ命令で行います。）

(4)これらのプログラムをＣＯＭＰＩＬＥしておき、ＬＩＮＫ命令で結合して１つのプログラムとしてＤＷＮＬＤ命令でＰＣに転送して実行します。

2 分割コンパイルの例

ＪＯＢ１・ＪＯＢ２・ＪＯＢ３のプログラムを分割コンパイルする例を記載します。この例では、ＯＮＴＩＭＥ（出力オン時間）をグローバル変数として全プログラムで使用しています。また、変数A、Bはローカル変数として使用しています。

次のプログラムをファイル名ＪＯＢ１の名称でＳＡＶＥします。

```
1000 FUNCTION JOB1
1010 ENTRY REAL ONTIME          'ONTIMEをREALでグローバル変数宣言
1020 EXTERN FUNCTION JOB2       'JOB2を起動するための宣言
1030 EXTERN FUNCTION JOB3       'JOB3を起動するための宣言
1040 XQT !2,JOB2;XQT !3,JOB3    'JOB1･JOB3を起動
1050 ONTIME=1                   'ONTIMEを１にセット
1060 ON Y_&M10;WAIT ONTIME;OFF Y_&M10        '１秒点灯
1070 FEND
```

次のプログラムをファイル名ＪＯＢ２の名称でＳＡＶＥします。

```
2000 FUNCTION JOB2
2010 EXTERN REAL ONTIME  'グローバル変数ONTIMEを使用する宣言
2020 REAL A              'ローカル変数
2030 A=ONTIME*2          'ONTIMEの２倍
2040 ON Y_&M11;WAIT A;OFF Y_&M11       '２秒点灯
2050 FEND
```

次のプログラムをファイル名ＪＯＢ３の名称で
ＳＡＶＥします。

```
3000 FUNCTION JOB3
3010 EXTERN REAL ONTIME  'グローバル変数ONTIMEを使用する宣言
3020 REAL B              'ローカル変数
3030 B=ONTIME*3          'ONTIMEの3倍
3040 ON Y_&M12;WAIT B;OFF Y_&M12      '３秒点灯
3050 FEND
```

④各プログラムをＣＯＭＰＩＬＥし、オブジェクトファイル（～.OBJ）をフロッピーディスクＢドライブに作成します。

```
>COMPILE "B:JOB1"↵ ……… 入力
COMPILE END
>COMPILE "B:JOB2"↵ ……
COMPILE END
>COMPILE "B:JOB3"↵ ……
COMPILE END
```

⑤フロッピーディスクＢドライブ各プログラムのオブジェクトファイル（実行形式ファイル）をＬＩＮＫ命令で結合し、ファイル名ＨＬＩＮＫを作成します。

```
>LINK B:JOB1+B:JOB2+B:JOB3,A:HLINK↵
```

並び順でプログラムが実行されます。

⑥ハードディスクのオブジェクトファイルＨＬＩＮＫをＰＣに転送します。

```
>DWNLD,"A:HLINK"↵
```

⑦ＸＱＴ命令でプログラムを実行します。

```
>XQT↵
```

この例では、Y10～12が同時に点灯し、
Y10は１秒後に消灯
Y11は2秒後に消灯
Y12は3秒後に消灯
します。

注意

- ●各プログラムの行番号は何番でもかまいません。
- ●各プログラムの１行は、８０文字以内にしてください。
- ●各プログラムを結合したテキストファイルは作れません。
- ●ＰＣ上で動作しているプログラムをパソコンにＵＰＬＤし、ＬＩＳＴで表示させた場合、エラーが発生することがあります。分割コンパイルした場合、プログラムの確認は結合する前の各プログラムを参照してください。
- ●グローバル変数宣言していない、ローカル変数は各プログラムで同じものを使用できますが、ＵＰＬＤすると、２重使用のエラーが発生します。（ただし実行は正常に行われます。）

付　録

付録A　ＦＰ－ＢＡＳＩＣ編集ソフトのインストール

FP-BASIC編集ソフトをNEC 9801NS/Tにインストールします。以下のものを用意してください。

用意するもの

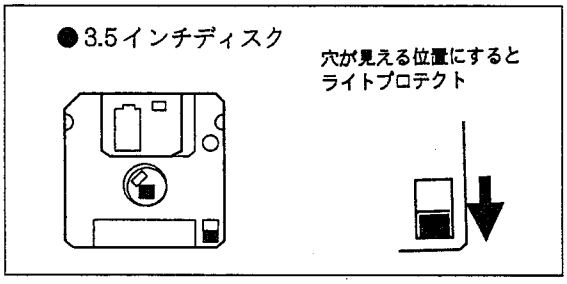
①NEX 9801NS/T
②NEC 日本語MS-DOS Ver.3.30D
③FP-BASIC編集ソフト Ver.2.10
④未使用フロッピーディスク（2HDタイプ）

重要

NEC日本語MS-DOSおよびFP-BASIC編集ソフトのフロッピーディスクにはプロテクトを施してください。

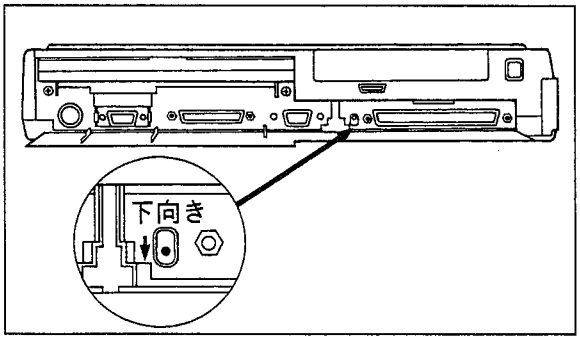
注意

・FP-BASIC編集ソフトのインストールを実行すると、パソコンのとRAMドライブの内容が消失します。RAMドライブ中に重要なデータやアプリケーションがある場合は、事前に98ノートメニューの「3.RAMドライブ→FDコピー」を実行することによりフロッピーディスクにバックアップをとります。操作手順については81ページをお読みください。



1 バックアップメモリスイッチの設定

電源をＯＮする前に、PC9801NS/TのバックアップメモリスイッチがＯＮになっていることを確認します。



2 ９８ＮＯＴＥセットアップディスクによる初期化

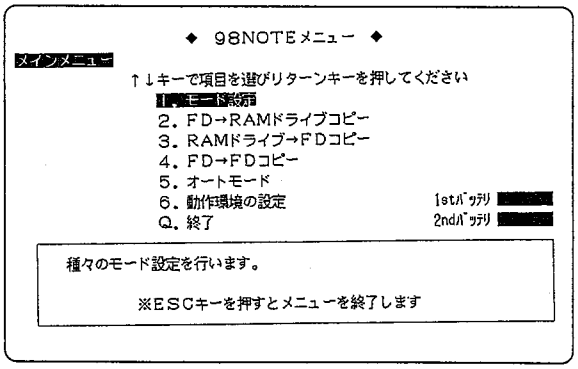
はじめて電源をＯＮする場合およびバッテリーが完全に消耗している場合は、PC9801NS/T付属の９８ＮＯＴＥセットアップディスクを使用することによって、9801NS/Tを初期化します。操作の手順は、9801NS/T付属の『ガイドブック』をお読みください。

3 ９８ＮＯＴＥメニューによる各種設定

９８ＮＯＴＥメニューを起動して各種設定を行います。この設定は、FP-BASIC編集ソフトを使用する上で非常に重要ですので、必ず以下の手順に従って操作してください。なお、以下の手順で設定を行うと、ＲＡＭドライブの内容が消去されるなどして、従来使用中のソフトが使用できなくなる場合がありますので、必要に応じてあらかじめＲＡＭドライブの内容をバックアップしてください。

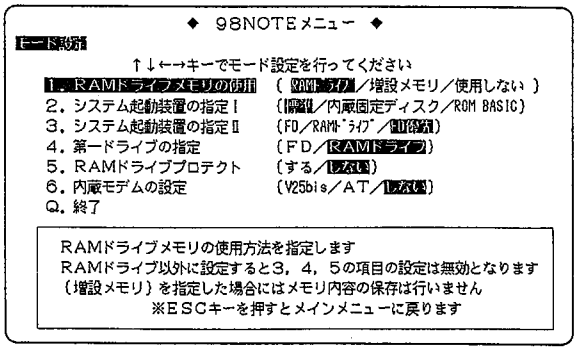
①９８ＮＯＴＥメニューの起動

HELPキーを押しながら本体の電源をＯＮします（電源ＯＮ状態の場合はHELPキーを押しながらリセットスイッチを押します）。HELPキーはしばらく押し続けてください。このとき、フロッピーディスクドライブにはないも入れないでください。画面には、９８ＮＯＴＥメニューのメインメニューが表示されます。



②モード設定メニューの表示

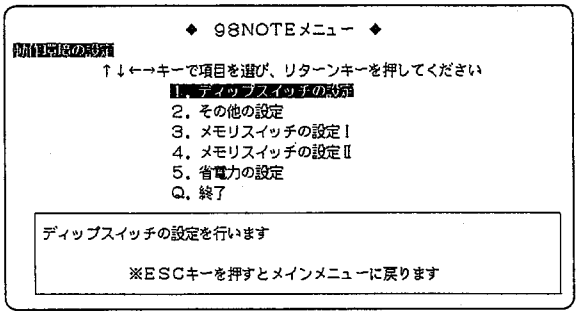
メインメニューの「１．モード設定」を選ぶと、モード設定の画面が表示されます。



モード設定が以上のようになっていることを確認したら、「Q．終了」を選択して、メインメニューに戻ります。

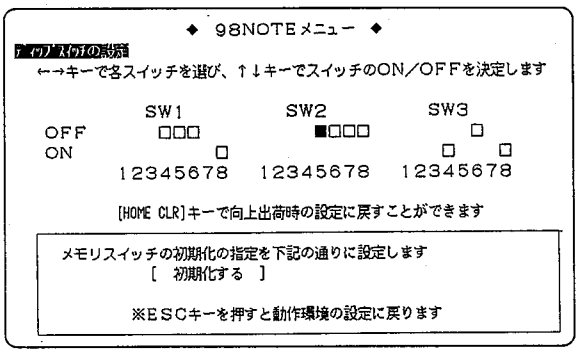
③動作環境の設定

メインメニューの「6．動作環境の設定」を選択すると、以下の動作環境の設定メニューが表示されます。



④ディップスイッチの設定

動作環境の設定メニューの「1．ディップスイッチの設定」を選択すると、ディップスイッチの設定画面が表示されます。設定は工場出荷時の状態にします。HOME CLRキーを押すと、工場出荷時の状態になります。



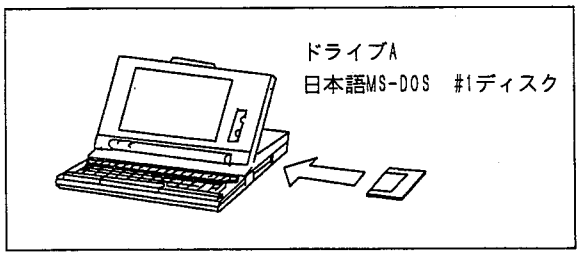
ディップスイッチの設定が以上のようになっていることを確認したら、ESCキーを押して動作環境の設定メニューに戻ります。

⑤９８ＮＯＴＥメニューの終了

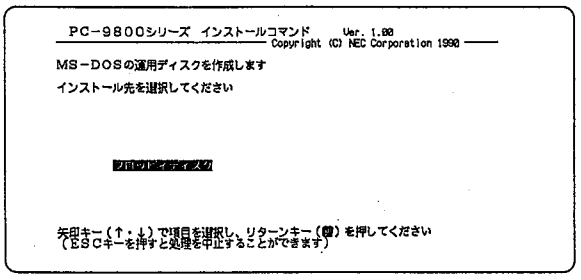
動作環境の設定メニューで「Q．終了」を選択し、９８ＮＯＴＥメニューのメインメニューに戻ります。さらに、メインメニューで「Q．終了」を選択して９８ＮＯＴＥメニューを終了させます。

4 ＭＳ－ＤＯＳの起動

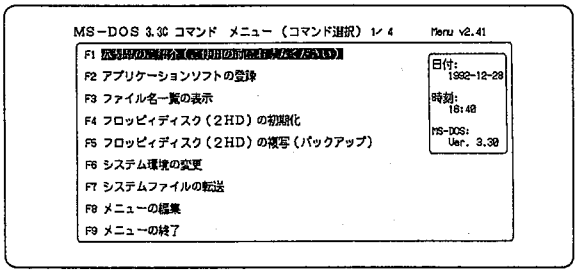
日本語MS-DOSの＃１ディスクをフロッピードライブに入れて、電源スイッチをＯＮします（電源ＯＮ状態の場合はリセットボタンを押します）。以上の操作により、日本語MS-DOSのシステムが起動します。



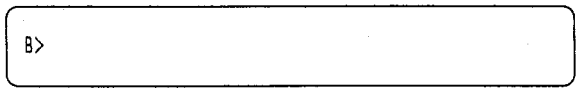
起動後の画面に、インストールコマンドの画面（下図参照）が表示されたらESCキーを押してから、画面の指示に従ってインストールコマンドを終了させてください。



メニュー画面が表示されたらSTOPキーを押すことにより、メニュー画面を終了させてください。

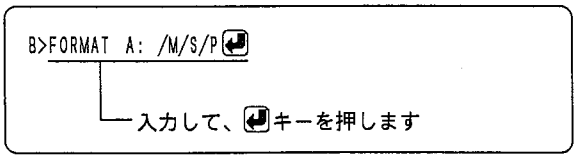


以下のMS-DOSの入力待ち状態（プロンプト「B>」）の画面が表示されます。

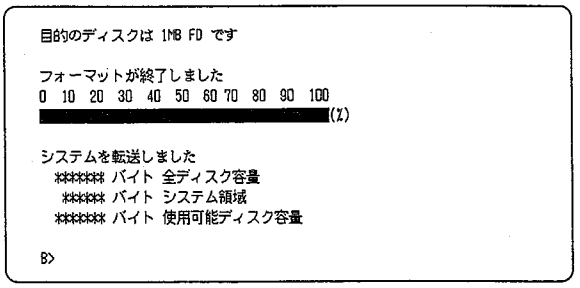


5 ＲＡＭドライブの初期化

フロッピードライブ（ドライブB）からMS-DOSを起動した場合、ＲＡＭドライブはドライブAになります。フォーマットコマンド「FORMAT A:/S」を以下のように入力して、ＲＡＭドライブを初期化します。



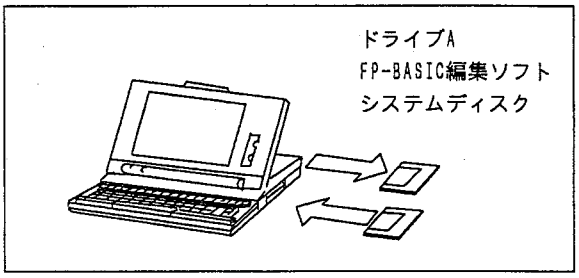
以下が表示されます。



以上の操作で、ＲＡＭドライブは初期化されました。

6 ＦＰ－ＢＡＳＩＣ編集ソフトのコピー

フロッピードライブから日本語MS-DOSの＃１ディスクを取り出し、代わりにFP-BASIC編集ソフトのシステムディスクを入れてください。

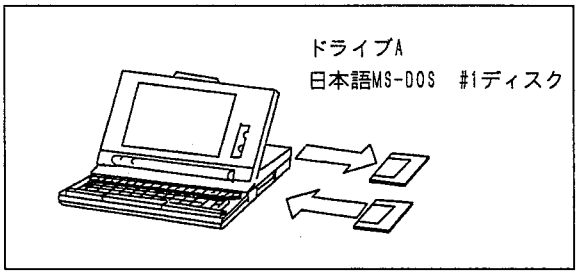


コピーコマンド「COPY *.* A:」を以下のように実行します。

```
B>COPY *.* A:↵
          └─ 入力して、↵を押します
```

以上により、FP-BASIC編集ソフトシステムディスクの内容がＲＡＭディスクにコピーされます。

コピーが終了したら、フロッピードライブからFP-BASIC編集ソフトのシステムディスクを取り出して、代わりに日本語MS-DOSの＃１ディスクを入れてください。



7 起動環境の設定

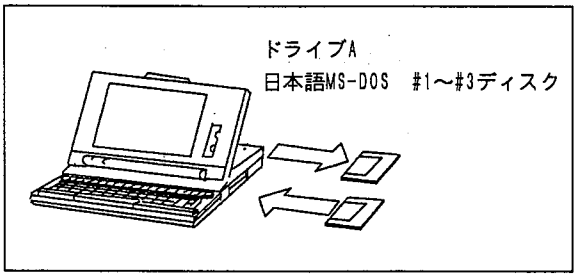
画面のプロンプト「B>」を「A>」に変更します。次のように「B>」の後に、「A:」と入力します。

```
B>A:↵
     └─ 入力して、↵キーを押します
```

つぎに、以下のように「SETUP B:」と入力して、起動環境設定プログラムを実行します。

```
A>SETUP B:↵
          └─ 入力して、↵キーを押します
```

起動環境設定プログラムが実行されますので、画面の指示に従って操作してください。MS-DOSのバージョンによっては途中で、画面の指示に従って日本語MS-DOSの＃１ディスクを＃２または＃３ディスクに入れ替えてください。



途中で「かな漢字変換にＮＥＣＤＩＣ．ＤＲＶ（単漢字変換）を組み込みますか？」と表示されますので、Ｙキーまたは Ｎ キーを入力してから↵キーを押してください。

起動環境設定プログラムが終了したら、以下のように入力します。この操作を行わないと、PC-9801NS/Tでは FP-BASIC編集ソフトは正常に動作しませんので、必ず以下の手順で操作してください。

●FP3H-BASICの場合

```
A>COPY CON AUTOEXEC.BAT↵ ──────────┐
FPB /9↵ ──────────┐                  │
^Z ──────┐        │                  │
         │        └─ 2行目入力       └─ 1行目入力
         └─ 3行目 CTRL + Z を入力して
            ↵キーを押す
```

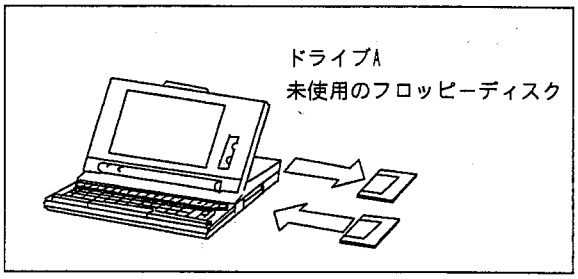
●FP3-BASIC（64kバイトタイプ）
またはFP1-BASICの場合

```
A>COPY CON AUTOEXEC.BAT↵ ──────────┐
FPB /9 /H↵ ──────┐                 │
^Z ──────┐       │                 │
         │       └─ 2行目入力      └─ 1行目入力
         └─ 3行目 CTRL + Z を入力して
            ↵キーを押す
```

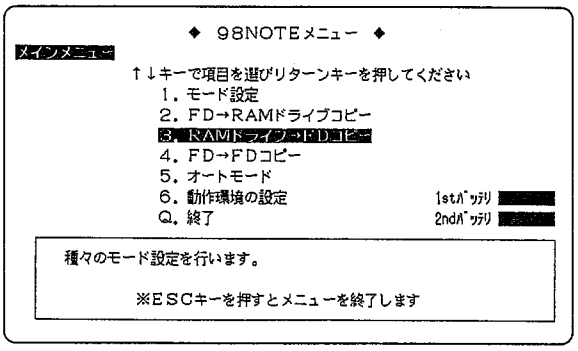
以上で、ＲＡＭディスクにFP-BASIC編集ソフトの実行システムが作成されました。

8 実行用ディスクの作成

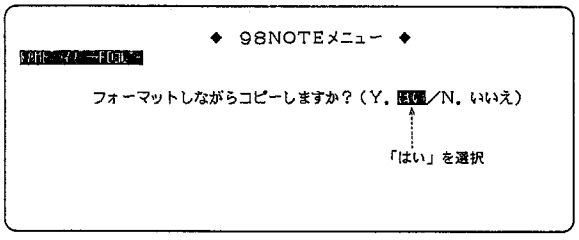
フロッピードライブから日本語MS-DOSのディスクを取り出して、代わりに未使用のフロッピーディスクを入れてください。なお、未使用フロッピーディスクにはプロテクトを施さないでください。



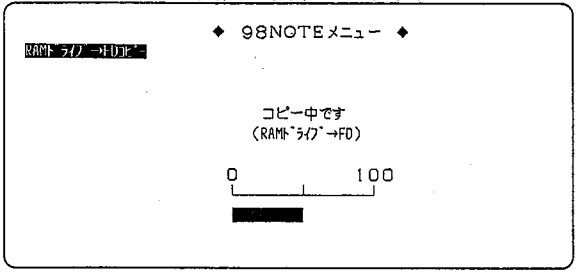
HELPキーを押しながらリセットボタン（パソコン本体左側面）を押して、９８ＮＯＴＥメニューを起動します（HELPキーはしばらく押し続けてください）。



９８ＮＯＴＥメニューで「３．ＲＡＭドライブ→ＦＤコピー」を選択し、画面の表示に従ってＲＡＭディスクの内容を未使用フロッピーディスクにコピーします。以下の画面では、フォーマットしながらコピーするので、← →キーではいを選択して↵キーを押します。

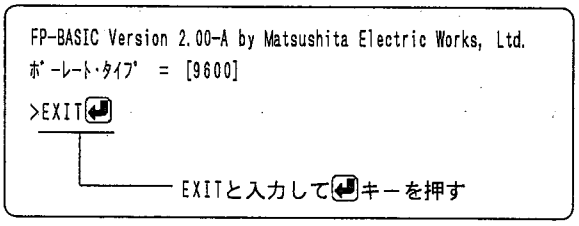


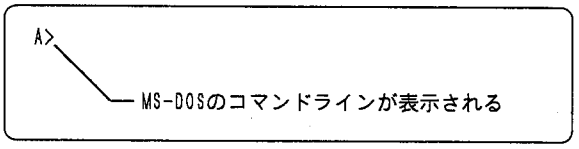
コピー中は以下の画面が表示されます。



コピーが終了したら、 キーを押して９８ＮＯＴＥメニューを表示させ、フロッピーディスクを取り出します。取り出したフロッピーディスクには「FP-BASIC実行用ディスク」と記入したラベルを貼っておきます。

９８ノートメニューで、「Ｑ．終了」を選択すると、FP-BASIC編集ソフトが起動しますので、「EXIT」と入力して↵キーを押すとMS-DOSのコマンドラインが表示されます。





MS-DOSのコマンドラインを確認したら、パソコンの電源をＯＦＦします。

なお、「かな漢字変換にＮＥＣＤＩＣ．ＤＲＶ（単漢字変換）を組み込みますか？」に対し、Ｙキーを入力してた場合、かな漢字変換の辞書ディスクとして日本語MS-DOS（Ver.3.30D）の＃３ディスクを使用します。

注　意

- FP-BASIC編集ソフトはＲＡＭドライブから起動されますので、プログラムもＲＡＭドライブに保存されます。
- ＲＡＭドライブに作成したドライブのプログラムを実行用ディスクにコピーするには、９８ノートメニューを起動して、を起「３．ＲＡＭドライブ→ＦＤコピー」を実行してください。

付録B　ＲＯＭの作成方法

ＲＯＭライタ用のＨＥＸファイルの作成方法およびＲＯＭライタへの転送方法について説明します。

1 ＲＯＭライタ用ＨＥＸファイルの作成方法

FP-BASIC編集ソフトでＲＯＭライタ用のＨＥＸファイルを作成することができます。ＣＯＭＰＩＬＥコマンド実行時に「，Ｈ」オプションを付けることにより、インテルＨＥＸ形式のＲＯＭライタ用ファイルが作成されます。

●システムメモリからコンパイルする場合

```
>COMPILE,H↵
COMPILE END
        └── SYSTMP.HEXがディスクに作成される
```

●ソースファイルからコンパイルする場合

```
>COMPILE "TEST",H↵
COMPILE END
        └── TEST.HEXがディスクに作成される
```

2 プリンタポートを使用したＲＯＭライタへの転送

FP-BASIC編集ソフトで作成したＨＥＸファイルをプリンタポートからＲＯＭライターに転送します。

①FP-BASIC編集ソフトを終了し、MS-DOSのプロンプトを表示させます。

```
A>
```

②パソコンのプリンタポートとＲＯＭライタを接続します。

③ＲＯＭライタを受信可能な状態に設定します。
アバールデータ製の「ＰＥＣＫＥＲ－11」の場合、操作手順は JOB D 1 SET です。

④MS-DOSのコピーコマンドを以下のように入力します。

●システムメモリからコンパイルした場合

```
A>COPY SYSTMP.HEX PRN↵
          └── 入力して、↵キーを押す
```

●ソースファイルからコンパイルした場合

```
A>COPY TEST.HEX PRN↵
          └── 入力して、↵キーを押す
```

注意

- 「付録 A FP-BASIC編集ソフトのインストール」で作成した実行用ディスクにはPRINT.SYSが含まれていますので、プリンタポートはファイル名「PRN」としてMS-DOSから認識されます。
- インテルＨＥＸ形式のファイルはASCII（テキスト）形式ですのでCOPYコマンドが使用できます。

3 シリアルポートを使用したＲＯＭライタへの転送

FP-BASIC編集ソフトで作成したＨＥＸファイルをシリアルポート（ＲＳ２３２Ｃインターフェイス）からＲＯＭライターに転送します。

①FP-BAIC編集ソフトを終了し、MS-DOSのプロンプトを表示させます。

```
A>
```

②ＣＯＮＦＩＧ．ＳＹＳにＲＳＤＲＶ．ＳＹＳを登録します。以下のように操作します。

```
A>COPY  CONFIG.SYS+CON  CONFIG.SYS
CONFIG.SYS
CON
DEVICE=RSDRV.SYS↵          ← 正確に入力して、↵キーを押す
^Z                          ← ここで、CTRLキーとZキーを同時に押し、↵キーを押す
```

③ＲＳＤＲＶ．ＳＹＳをＲＡＭディスクにコピーする。フロッピードライブに日本語MS-DOSの＃１ディスクを入れて、以下のように入力します。

```
A>COPY  B:RSDRV.SYS↵
                  └── 入力して、↵キーを押す
```

注　意

・システムディスクにRSDRV.SYSが存在し、デバイスドライバとしてCONFIG.SYSに登録されていれば、パソコンのRS232Cポートはファイル名「AUX」としてMS-DOSから認識されます。

④ＳＰＥＥＤ．ＥＸＥをＲＡＭディスクにコピーする。フロッピードライブに日本語MS-DOSの＃１ディスクまたは＃２ディスクを入れて、以下のように入力します。

```
A>COPY  B:SPEED.EXE↵
                  └── 入力して、↵キーを押す
```

注　意

・SPEED.EXEは、MS-DOSのバージョンによって入っているディスク異なりますので、注意してください。

・ここで、９８ＮＯＴＥメニューの「3.ＲＡＭディスク→ＦＤコピー」の機能を使用すれば、シリアルポートが使用可能な実行用ディスクが作成できます。

⑤フロッピーディスクを取り出してから、リセットボタンを押して、ＲＡＭドライブからFP-BASIC編集ソフトを再起動する。

⑥FP-BASIC編集ソフトを終了し、MS-DOSのプロンプトを表示させます。

```
A>
```

⑦パソコンのプリンタポートとＲＯＭライタを接続します。

⑧ＲＯＭライタを受信可能な状態に設定します。
アバールデータ製の「ＰＥＣＫＥＲ－11」の場合、操作手順はJOB D SETです。

⑨通信条件を設定します。
アバールデータ製の「ＰＥＣＫＥＲ－11」の場合、以下のように入力します。

```
A>SPEED RO 4800 B8 PN S1 XON↵
                  └── 入力して、↵キーを押す
```

⑩MS-DOSのコピーコマンドを次のように入力します。

●システムメモリからコンパイルした場合

```
A>COPY  SYSTMP.HEX  AUX↵
                  └── 入力して、↵キーを押す
```

●ソースファイルからコンパイルした場合

```
A>COPY  TEST.HEX  AUX↵
                  └── 入力して、↵キーを押す
```

注　意

・インテルＨＥＸ形式のファイルはASCII（テキスト）形式ですのでCOPYコマンドが使用できます。

付録C　ＦＰ－ＢＡＳＩＣソフトウェア仕様

1 ＦＰ－ＢＡＳＩＣ編集ソフトの主な機能

総合機能		ＰＣ用コンパイラ型マルチタスク構造化ＢＡＳＩＣ。 最大タスク数16。 タスク間グローバル変数／ローカル変数装備。
ＰＣ動作環境設定機能		ＣＰＵ初期化。 ＣＰＵパラメータメモリ設定。 スロット割り付け。
編集機能	エディタ	スクロール可能フルスクリーンエディタ。 編集可能サイズ　　最大255バイト/行　　最大64Kバイト/ファイル。 ファイルセーブ／ロード。
	コンパイラ	ＰＣ実行プログラム作成／転送（分割コンパイル可能）。 ＲＯＭライタファイル作成。 実行ファイルサイズ最大128Kバイト（OBJファイルとSYMファイルの合計）
ＰＣ運転/デバッグ機能		運転／停止　トレースモード　テストモード。 運転モード／ステータス表示。 I/O、メモリ、変数のモニタ。 I/O、メモリの状態のダンプ表示。 タイマ経過値表示／設定。 高機能ユニット割り込み設定状態の表示。 I/O、メモリ、その他の状態の一括ダンプ表示と編集。 I/O、メモリ、変数の状態のファイルセーブ・ロード。 強制入出力。
ＰＣターミナル機能		パソコンのディスクドライブアクセス可能。

2 FP-BASIC編集ソフトの動作環境

適応ＰＣ	ＣＰＵＦＰ３Ｈ－ＢＡＳＩＣタイプＣＰＵ ＦＰ－ＢＡＳＩＣのすべてのコマンド、命令、関数が使用できる。 Ｖｅｒ．２．０以降のＦＰ３－ＢＡＳＩＣタイプＣＰＵ ＦＰ－ＢＡＳＩＣのすべてのコマンド、命令、関数が使用できる。 Ｖｅｒ．２．０未満のＦＰ３－ＢＡＳＩＣタイプＣＰＵ 使用できるコマンド、命令、関数が限定される。
Ｉ／Ｏユニット	高機能ユニットを含むすべてのＩ／Ｏユニットが使用できる。
適応パソコン	ＮＥＣ　ＰＣ９８０１シリーズ　メモリ640KB実装　640×400ドット表示モード。 （初代ＰＣ９８０１、ＰＣ９８０１Ｅ／Ｆ／Ｕ２、ＰＣ９８０１ＸＡ、ＰＣ９８０１ＬＴは使用できません。） ＥＰＳＯＮ　ＰＣ286シリーズ／386シリーズ　640×400ドット表示モード。
ＯＳ	NEC日本語MS-DOS Ver.3.10以降。 CONFIG.SYSにPRINT.SYSの組み込みが必要（RSDRV.SYSの組み込みは不用）。
メモリ容量	520Kバイト以上の空きメモリの確保が必要。 ＮＥＣＤＩＣ．ＤＲＶ以外のかな漢字変換システムを使用する場合は、ＥＭＳメモリが必要です。
適応ケーブル	AFB85813 RS232Cケーブル（ＣＰＵ－パソコン接続用）　ケーブル長3m。

3 起動オプションスイッチ　起動書式は「A>FPB 〈ｽｲｯﾁ〉」です。（「A> 」はMS-DOSのコマンドライン）。

スイッチ	起動時の状態
無	通常起動。INSキーでカーソル位置に１文字文のスペースを挿入。
/9	ボーレート9600bpsで起動。
/H	Ver.2.0未満のＢＡＳＩＣタイプＣＰＵ（AFP3251等）に対応。
/i	挿入モード起動。INSキーで上書きモードと挿入モードを切り替え。 挿入モード時のカーソル形状は▉に変化する。
/I	挿入モード起動。INSキーで上書きモードと挿入モードを切り替え。 挿入モード時のカーソル形状は変化しない。
/S	前回終了時の状態で起動。起動時に「SYSTMP.PRG」をメモリに読み出す。

4 ファンクションキーへのコマンド割り付け

■通常の画面表示（出荷時）

■[SHIFT]キーを同時に押したときの画面表示（出荷時）

■ファンクションキーへのコマンド割り付け（出荷時）

ファンクションキー	コマンド割り付け
ｆ・１	「LOAD "」を入力する。
ｆ・２	「AUTO」を入力する。
ｆ・３	「TSTAT」を入力して、リターンキーを押す。
ｆ・４	「LIST」を入力する。
ｆ・５	「XQT」を入力して、リターンキーを押す。
ｆ・６	「COMPILE」を入力して、リターンキーを押す。
ｆ・７	「DWNLD」を入力して、リターンキーを押す。
ｆ・８	「PRINT」を入力する。
ｆ・９	「FILES」を入力して、リターンキーを押す。
ｆ・１０	「QUIT ALL」を入力して、リターンキーを押す。
ＳＨＩＦＴ＋ｆ・１	「SAVE "」を入力する。
ＳＨＩＦＴ＋ｆ・２	「TOFF」を入力して、リターンキーを押す。
ＳＨＩＦＴ＋ｆ・３	「FORCE」を入力する。
ＳＨＩＦＴ＋ｆ・４	「RFORCE」を入力する。
ＳＨＩＦＴ＋ｆ・５	「QUIT」を入力して、リターンキーを押す。
ＳＨＩＦＴ＋ｆ・６	「COMPILE "」を入力する。
ＳＨＩＦＴ＋ｆ・７	「UPLD」を入力して、リターンキーを押す。
ＳＨＩＦＴ＋ｆ・８	「CONT」を入力して、リターンキーを押す。
ＳＨＩＦＴ＋ｆ・９	「STEP」を入力して、リターンキーを押す。
ＳＨＩＦＴ＋ｆ・１０	「EXIT」を入力する。

＊コマンド割り付けはKEYコマンドおよびKEY LISTコマンドで変更・表示可能。

5 ＦＰ－ＢＡＳＩＣ記述方法

直接実行と間接実行	
直接実行	コマンドを直接キーボードから入力して実行すること。
間接実行	行番号を使用して記述された一連のプログラムを実行すること。
コマンドと命令	
コマンド	直接実行で使用する命令群。
命令	間接実行で使用する命令群。
命令語（ステートメント）と命令文	
命令語 （ステートメント）	単独で使用される場合と命令語の後にオペランドを伴う場合がある。 オペランドには、定数、変数、演算子、関数およびそれらの組み合わせを用いる。
命令文	プログラムの中で実行される最小単位の記述。 例：式または関数　例：A=3　　B=SW(X_0) 命令語　　例：END 命令語＋オペランド 例：ON Y_&M2F
コメント（注釈文）	
ＲＥＭ文	プログラム中の注記・注釈のためのコメント（実行を伴わない文）。 ＲＥＭ文の代わりに、「' （シングルクォーテーション）」を使用可能。
行と行番号	
行	プログラムが間接実行される場合に、順次実行される命令文の集まり。 プログラムの１行中には、１つあるいは複数の命令文が記述される。 行番号と文との間には必ず１文字以上のスペースを入れる。 １行の長さは最大255バイトまで。
行番号	1～32767までの整数で記述する。 プログラムは、任意の行番号で開始することができる。 行番号は行ごとに大きな値になり、これを行番号の増分と呼ぶ。
マルチステートメント （複文）	１行に複数の文が記述されること。 命令文を「;（セミコロン）」を使用して区切る。
ラベルとラベル名	
ラベル	プログラムの実行順序を変更するとき、直接行番号を指定する代わりに、 ラベルを指定することができる。
ラベル名	８文字までの英数字が使用でき、先頭の１文字は必ず英字。 予約語および予約語の直後の数字または「_」（アンダースコア）は使用禁止。 英字の大文字と小文字は区別されない。 分岐先のラベル（ラベル行）ではラベル名の後に「:（コロン）」を付ける。

使用できる文字と記号			
1バイト文字	英字（大文字・小文字）、数字、カタカナ、特殊記号。 英字と記号にはあらかじめ決められた特別の意味を持つ予約語がある。 「P」および「＿（アンダースコア）」は、変数名、ラベル名の先頭には使用できない。 特殊記号のうち「"（ダブルクォーテーション）」は文字定数として使用できない。 英字の大文字と小文字の区別をしない。 例：英字　　A B C D E … a b c d e … 　　数字　　0 1 2 3 4 5 6 7 8 9 … 　　特殊記号　! " # $ % & ' ( ) , . / ; : [ ] ¥ ^ ` - … 　　カタカナ　ｱ ｲ ｳ ｴ ｵ …		
2バイト文字	英字（大文字・小文字）、数字、カタカナ、特殊記号、ひらがな、漢字。 2バイト文字で命令語や変数を記述することはできない。 2バイト文字の使用は、文字定数とコメントに限らる。 例：英字　　Ａ Ｂ Ｃ Ｄ Ｅ … ａ ｂ ｃ ｄ ｅ … 　　数字　　０ １１ ２ ３ ４ ５ ６ ７ ８ ９ … 　　特殊記号　！ ” ＃ ＄ ％ ＆ ’ （ ） ， ． ／ ； ： … 　　カタカナ　ア イ ウ エ オ … 　　ひらがな　あ い う え お … 　　漢字　　松下電工株式会社		
特殊記号の読み方	記　号	読み方	特別の用途（使用時に注意が必要です）
	-	マイナス	演算、範囲指定
	:	コロン	ラベル行
	;	セミコロン	マルチステートメント（複文）の区切り記号
	'	シングルクォーテーション	ＲＥＭ（コメント）の省略型
	"	ダブルクォーテーション	文字定数
	,	コンマ	パラメータの区切り
		スペース	行番号、命令語、パラメータの区切り
	.	ピリオド	ファイル名と拡張子の区切り
	_	アンダースコア	なし
	*	アスタリスク	ファイル名、拡張子のワイルドカード指定
	?	クエスション	なし
予約語			
予約語	変数名、ラベル名には、予約語を使用できない。 予約語の直後の数字または「__」（アンダースコア）も使用禁止。		

予約語一覧

"	(	)	+	-	/	<	<=
<>	=	=<	=>	>	>=	ABS()	ACCEL
ADALM()	ADASET	ADAVRG	ADDATA	ADDATA()	ATAN()	ATN()	BCD()
BIN$()	BIN()	BLKMOV	BLKOUT	BYTE	CALL	CASE	CHR$()
CLEAR	CLOSE	CLRLIB	CLRVAL	CLRVAR	CLS	COLOR	COMPILE
CONSOLE	CONT	COS()	CVB()	CVL	CVS()	CVW()	CX()
CY()	DAALM()	DAERR()	DALIMIT	DALSET	DARDY()	DASET	DATA
DATA$	DATE$	DEC	DECLR	DECO()	DEFAULT	DIM	DIST()
DLOAD	DO	DREAD	DRV	DT_	DUMP	DUMPC	DUMPINT
DUMPP	DUMPR	DWNL	ELSE	ELSEIF	ENCO()	END	ENDIF
ENTRY	EOF()	ERN	ERR()	EXIT	EXP()	EXT	FEED
FEND	FERROR()	FILES	FLOAT	FL_	FOR	FORCE	FRE
FRE()	FREV	FUNCTION	GO	GOSUB	GOTO	HALT	HCDATA()
HCFSET	HCISET	HCOUTEN	HCPSET	HCRESET	HCSTAT()	HEX$()	HOME
HORDR	HTEST	ID	IDAC	IDBC	IDBRID	IDBWID	IDCAN
IDCLR	IDCRSP()	IDPRR	IDPRW	IDREADY()	IDRECV	IDRI	IDRID
IDRV	IDRVA	IDRVH	IDRVHA	IDRVHAID	IDRXACK	IDRXDT	IDSEND
IDTCHR	IDTXACK()	IDTXDT	IDTXRDY()	IDTYPE	IDW	IDWI	IDWRX()
IDWV	IDWVA	IDWVH	IDWVHA	IDWVHAID	IF	IFBIN()	INCIND()
INH()	INL()	INPUT	INPUT$()	INPUTR	INSTR()	INT	INT()
INTEGER	INV()	INW()	KILL	LDCR	LDIO	LDVAL	LD_
LEFT$()	LEN()	LINE	LIST	LOC()	LOCATE	LOF()	LOG()
LONG	LOOP	LROLL()	LSHIFT()	L_	MAP	MAXDEV	MEW$()
MID$	MID$()	MKB$()	MKL$()	MKS$()	MKW$()	MOD	MODEMON
MONCLR	MONDISP	MONENT	MONTS	MOVE	MYINT	MYTASK()	NAMENEG()
NEXT	NGX	NGY	NOP	NOTOCT$()	OFF	ON	ONERR
ONSW	OPEN	OR	OUT	OUTD	OUTH	OUTL	OUTW
PALET	AUSE	PDEL	PEEK()	PIOSET	PODATA()	POFCHG	POKE
POORG	POPSET	POREOS	PORESET	POSTART	POSTAT()	PRGSIZE	PRINT
PRINTR	PRIVATE	PRM	PRM()	PRMCLR	PRMR	PRMR()	PSADRS()
PSADSET	PSAID()	PSBUSY()	PSCTRL	PSDSET	PSECLR	PSERR()	PSJOBNO
PSLOAD	PSORGH	PSORGS	PSPRM	PSPRM()	PSREADY()	PSSAVE	PSSTART
PSSTAT()	PSSTOP	PSTYPE	PSX	PSY	PULSE	QUIT	RANDMIZE
RANGE	READ	READ	REAL	RECV	RECVB	REFORCE	REM
RENUM	RESET	RESTORE	RESUME	RETRN	RETURN	RIGHT$()	RIOCLR
RIOSET	RND()	ROFLG()	RROLL()	RSHIFT()	RSLOTRUN	RXO	RXQTZE
R_	SDERR()	SDRCDT	SDRDY()	SDRECV	SDRESET	SDRXACK	SDSEND
SDTCHR	SDTERM	SDTXACK()	SDTXDT	SDTXRDY()	SDWRX()	SEGT()	SEL
SELECT	SELEND	SEND	SENDB	SET	SGN()	SIN()	SLOT()
SLOTCLR	SLOTI	SLOTICLR	SLOTR()	SPACE$()	SPEED	SQR()	STEP
STR$()	STRING	SVCR	SVIO	SVVAL	SW()	SWAP	SWITCH
TAN()	TEST	THEN	TIME	TIME$	TIME$()	TIME()	TIMER
TIMER()	TMOUT	TO	TOFF	TON	TSTAT	TW()	TWAIT
UNIT()	UPLDVAL()	VARIABLE	VER	VERINIT	WAIT	WEND	WHILE
WL_	WRITE	WR_	WX_	WY_	X_	Y_	¥
_ACLMAX	_ENCDIV	_HDIR	_HSENSE	_HSPEED	_HTELIM	_MNAME	_PMATCH
_SELAXIS	_SPDMAX	_ZPHASE					

6 使用できる数値

使用できる数値	
1バイト整数型	範囲：-128～+127 &B0～&B01111111 &O0～&O177 &H0～&H7F &M0～&M7F
2バイト整数型	範囲：-32,768～+32,767 &B0～&B01111111 11111111 &O0～&O177777 &H0～&H7FFF &M0～&M2047F
4バイト整数型	範囲：-2,147,483,648～+2,147,483,647 &B0～&B01111111 11111111 11111111 11111111 &O0～&O17777777777 &H0～&H7FFFFFFF &M0～&M1342177272F
4バイト実数型	精度が保証される０に最も遠い値：-1.2E+38 ～ +1.2E+38 精度が保証される０に最も近い値：-1.2E-38 ～ +1.2E-38
整数表記	
10進表記	例：　1　　123　　6210
２進表記	例：　&B1　　&b1011011
８進表記	例：　&O123　　&o722
16進表記	例：　&H123　　&h12F
ミュー表記	例：　&M123　　&m12F
実数表記	
浮動小数点表記	例：　56.128　-96785.34　-1.58769
指数表記	例：　2.368795E+3　-1.587914E+5

＊ミュー表記（＆M表記）は、末尾1桁だけが16進数表記（０～Ｆ）で、その他は１０進数表記（主に入出力番号の指定に使用）。

7 定数

使用できる定数定数	
数値定数	整数または実数の値をもつデータ。 1バイト整数・２バイト整数・４バイト整数・４バイト実数のいずれか。 値によりデータ型が決まる。 10進、2進、8進、16進、ミュー、浮動小数点、指数の任意の表記が可能。 例：0　1　256　3.1415 ・・・
文字定数 （文字列）	必ず「"（ダブルクォーテーション）」で囲む。 最大長80バイト。

8 変数

変数	
変数名	８文字までの英数字が使用でき、先頭の１文字は必ず「P」以外の英字。 予約語と予約語の直後の数字または「_」（アンダースコア）は使用禁止。 変数名の末尾に「＄」を付けなければならない。 英字の大文字と小文字は区別されない。 数値変数の例：A　B1　C20　FP3　FPBASIC ･･･ 文字変数の例：A$　FP3$　FPBASIC$　A$*4　B1$*8　FP$*255 ･･･
数値変数	
値と型	１バイト整数、２バイト整数、４バイト整数、４バイト実数のいずれか。 値が代入される前のは「０」として扱われる。
変数宣言	使用時に変数宣言（型宣言）をする。 変数宣言を省略すると４バイト整数型として定義される。 変数宣言を省略するとタスク間でグローバルな変数として定義される。 変数宣言　型宣言の意味 BYTE A　１ﾊﾞｲﾄ整数型 INTEGER A　２ﾊﾞｲﾄ整数型 LONG A　４ﾊﾞｲﾄ整数型 REAL A　４ﾊﾞｲﾄ実数型 注　意 ・変数宣言はプログラム中の各タスクブロック（FUNCTION ～ FEND）の最初にまとめて記述。 ・タスクブロックの途中で変数宣言をすると正しく解釈されない場合がある。
文字変数	
値とサイズ	通常0～80ﾊﾞｲﾄの文字列。 「$」の後に「*」と0～255数値を記述することにより、変数のサイズ を任意に決めることができる（Ver.2.0未満のCPUでは使用不可）。 値が代入される前は「空の文字列（”” ヌルストリング）」として扱われる。
変数宣言	変数宣言は各タスクブロック（ＦＵＮＣＴＩＯＮ ～ ＦＥＮＤ）の最初にまとめて記述する。 タスクブロックの途中で変数宣言をすると正しく解釈されない場合がある。 例：STRING A$　　0～80ﾊﾞｲﾄの文字列 STRING A$*n　nﾊﾞｲﾄの文字列（1≦n≦255）
グローバル変数とローカル変数	
グローバル変数	タスク間でグローバルな変数。 通常の型宣言をした場合、および型宣言を省略した場合。 例：210 BYTE A
ローカル変数	タスク間でローカルな変数。 ＰＲＩＶＡＴＥ型の型宣言命令を使用した場合。 例：PRIVATE BYTE A
型変換	
代入による型変換	代入される変数の型に変換。
算術演算式による型変換	精度の異なる演算は精度の低い値を精度の高い方に変換。
論理演算式（ビット演算式）による型変換	すべて整数に変換（小数点以下を四捨五入）して演算。

9 配列

配列	
配列	同じような変数をいくつも使用する場合、同じ変数名を付け、 「A(10)」のように配列中の要素の番号を添え字として指示する。
配列名	８文字までの英数字の後に添え字を付ける。 ８文字までの英数字は変数名と同じ規則に従う。
配列の次元	文法的には２５６次元まで可能。 実際にはメモリ容量とエディタが１行に記述できる文字数（２５５文字）に制限される。
添え字	「0」から始まる。 「BYTE A(10)」で宣言した配列変数の添え字は「0」～「10」まで。
型とサイズ	配列は変数と同じ方法で型宣言できる。 配列はタスク間でグローバルにもプライベートにも宣言できる。 文字列配列は変数と同じ方法で各要素のサイズ（バイト長）を指定できる。 例：BYTE A(10)　　PRIVATE BYTE A(10)　　INTEGER A(5,7) 　　STRING A(10)　　PRIVATE STRING A$*8(4,2)

10 式と演算子

式と演算子				
式	定数、変数、関数および式自体を演算子で結合したもの。			
演算子	演算のための特別の記号または文字列。			
演算子の種類				
四則演算	＋	加算	Ａ＝１＋１	値 ２
	－	減算	Ａ＝２－１	値 １
	＊	乗算	Ａ＝３＊２	値 ６
	／	除算	Ａ＝４／２	値 ２
	ＭＯＤ	除算の余り	Ａ＝５ＭＯＤ３	値 ２
論理演算 (ﾋﾞｯﾄ演算)	ＯＲ	論理和	Ａ＝１ＯＲ２	値 ３
	ＡＮＤ	論理積	Ａ＝１ＡＮＤ２	値 ０
	ＸＯＲ	排他的論理和	Ａ＝１ＸＯＲ３	値 ２
比較演算	＝	等しい	Ａ＝（Ｂ＝Ｃ）	
	＜＞	等しくない	Ａ＝（Ｂ＜＞Ｃ）	
	＜	より大きい	Ａ＝（Ｂ＜Ｃ）	値 ・真の時：－１
	＞	より小さい	Ａ＝（Ｂ＞Ｃ）	・偽の時：０
	＜＝	以上	Ａ＝（Ｂ＜＝Ｃ）	
	＞＝	以下	Ａ＝（Ｂ＞＝Ｃ）	

演算子の優先順位			
優先順位	1	()	「() カッコ」で囲まれた式
	2	関数	数値関数・文字関数
	3	－	マイナス記号
	4	＊／	乗算・除算
	5	ＭＯＤ	除算の余り
	6	＋－	加算・減算
	7	＝＜＞	比較演算
	8	ＮＯＴ	否定
	9	ＡＮＤ	論理積
	10	ＯＲ	論理和
	11	ＸＯＲ	排他的論理和

11 ＦＰ－ＢＡＳＩＣ（Ｖｅｒ．２）命令語一覧

■キーコマンド

1.ＦＰ－ＢＡＳＩＣの準備

ＣＰＵの初期化	
EXIT	ＦＰ－ＢＡＳＩＣを終了する
VER	ＢＡＳＩＣタイプＣＰＵのバージョン名を表示する
VERINIT	ＢＡＳＩＣタイプＣＰＵのすべての状態を初期化する
CLRVAR	ＢＡＳＩＣタイプＣＰＵのプログラムと変数を初期化する
＊！CLRVAL	ＢＡＳＩＣタイプＣＰＵのグローバル変数を初期化する
RESET	ＢＡＳＩＣタイプＣＰＵをリセットする
＊！MAP	タスクごとにローカル変数領域を割り付ける
＊！FRE	タスクごとのローカル変数領域の空きエリアサイズを返す
＊！FREV	グローバル変数領域の空きエリアサイズを返す
パラメータメモリの設定	
PRMCLR	パラメータメモリの設定を初期化する
PRM	パラメータメモリを設定する
PRM()	パラメータメモリの現在の設定値を返す
スロット割り付け	
＊ SLOTCLR	スロット割り付けを初期化する（フリーロケーション）
＊ SLOT	スロット割り付けをする
＊ SLOT()	スロット割り付けの設定値を返す
キー定義	
KEY	ファンクションキーにコマンドを割り付ける
KEY LIST	ファンクションキーへのコマンド割り付けの内容を表示する

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

2.プログラム編集

ソースプログラムのエディット	
NEW	パソコンのプログラムメモリを初期化する
AUTO	行番号を自動的に発生する
RENUM	行番号をつけ変える
CLS	パソコンの画面を初期化（表示を消去）する
LIST	編集中のプログラムを画面に表示する
LLIST	編集中のプログラムをプリンタに印字する
！LINEMV	行を移動する
！FIND	編集中のプログラムから文字列を検索する
ファイルのセーブとロード	
SAVE	パソコンのプログラムメモリからディスクにプログラムを書き込む
LOAD	ディスクからパソコンのプログラムメモリにプログラムを読み出す
FILES	ディスクに書き込まれているファイルを一覧表示する
NAME	ディスクに書き込まれているファイルの名前を変更する
KILL	ディスクに書き込まれているファイルを削除する
オブジェクトプログラムの作成と転送	
COMPILE	ソースプログラムをオブジェクトプログラムに変換する
DWNLD	オブジェクトプログラムをパソコンのプログラムメモリからＣＰＵに転送する
UPLD	オブジェクトプログラムをＣＰＵからパソコンのプログラムメモリに転送する
分割コンパイル	
！ENTRY	分割コンパイル時のソースファイル間グローバル変数を宣言する。
！EXTERN	分割コンパイル時のソースファイル間グローバル変数またはファンクション名の参照を宣言する。
！LINK	分割コンパイルのオブジェクトをリンクし、実行オブジェクトを作成する。

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

3.プログラムの実行と停止

XQT	ＢＡＳＩＣタイプＣＰＵのプログラムを実行する
HALT	実行中のタスクを一時停止する
RESUME	一時停止中のタスクの実行を再開する
QUIT	実行中または一時停止中のプログラムを終了する

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

4.デバッグ機能

トレースモード	
TON	実行中の行番号を画面に表示する（トレースモード）
TOFF	TONコマンドで設定したトレースモードを解除する
テストモード	
＊ TEST	テストモードの設定、および表示をする
＊ CONT	PAUSE命令で一時停止しているタスクの実行を再開する
＊ STEP	PAUSE命令により一時停止しているタスクの実行をステップモードで再開する
＊ PAUSE	テストモード時、実行中のタスクを一時停止する
実行中モード、ステータスの表示	
＊ MODE	ＢＡＳＩＣタイプＣＰＵの動作モードを表示する
TSTAT	タスクごのステータスを表示する
＊！MONTS	タスクごとのステータスをリアルタイムに表示する
Ｉ／Ｏ、データメモリ、変数のモニタ	
MON	Ｉ／Ｏ、データメモリおよび変数の値をリアルタイムに表示する
＊！MONENT	リアルタイムでモニタする複数のＩ／Ｏおよびデータメモリ数を登録する
＊！MONDISP	複数のＩ／Ｏおよびデータメモリの値をリアルタイムにモニタする
Ｉ／Ｏ、データメモリのダンプ	
DUMP	Ｉ／Ｏおよびデータメモリの内容を読み出して、表示する
タイマ経過値の表示および設定	
＊ TIME	タイマ経過値を設定する
＊ TIME()	タイマ経過値を返す（秒単位）
高機能ユニットの割り込み設定の表示	
＊！DUMPINT	割り込み設定状況の一覧を表示する
Ｉ／Ｏ、データメモリ、その他の内容の一括表示と編集	
＊！DUMPC	Ｉ／Ｏ、データメモリ、その他の内容を一括して読み出してスクリーン表示する
＊！DUMPP	Ｉ／Ｏ、データメモリ、その他の内容を一括して読み出してスクリーン表示と編集をする
強制入出力	
＊ FORCE	Ｉ／Ｏ、データメモリの強制セットおよびリセットを設定する
＊ RFORCE	Ｉ／Ｏ、データメモリの強制セットおよびリセットを解除する
Ｉ／Ｏ、データメモリ、変数のセーブとロード	
＊！SVIO	Ｉ／Ｏをディスクにセーブする
＊！LDIO	Ｉ／Ｏをディスクからロードする
＊！SVVAL	グローバル変数をディスクにセーブする
＊！LDVAL	グローバル変数をディスクからロードする
＊！SVCR	共有メモリをディスクにセーブしする
＊！LDCR	共有メモリをディスクからロードする

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

■ステートメント

1.制御命令

実行制御構文	
FUNCTION～FEND	プログラムの開始および終了を宣言する
END	プログラムを終了する
GOTO	プログラムの実行を分岐する
GOSUB	サブルーチンに実行を移す
RETURN	サブルーチンから復帰する
＊ ON～GOTO	式の値に応じて分岐する
＊ ON～GOSUB	式の値に応じてサブルーチンを呼出す
＊！SELECT ～ CASE	式の値と一致するＣＡＳＥ文を実行する
IF THEN ELSE	条件式の結果により、プログラムの実行を制御する
＊！IFB THEN ～ ELSE	条件式の結果により、プログラムの実行を制御する
FOR～NEXT	命令文を繰り返し実行する
＊！DO～LOOP	条件式が満たされている間、命令文を繰り返し実行する（DO条件ループ）
＊！WHILE ～ WEND	条件式が満たされている間、命令文を繰り返し実行する（WHILE条件ループ）
＊！EXIT	ループから強制的に脱出する
＊！CALL ～ FUNCTION	他のプログラムをサブルーチンとして呼び出す
割り込み制御	
ONSW()	入力割り込み処理を定義する
ONERR	エラー割り込み処理を定義する
ECLR	エラーステータスを初期化（クリア）する
ERR()	エラー番号を返す
割り込みユニット制御命令	
！ON INT()	高機能ユニットからの割り込み処理を定義する
！INT() ON	割り込みを許可する
！INT() OFF	割り込みを禁止する
！INT() CLR	割り込み処理を解除する
！MYINT()	割り込み処理ルーチンで、実行中の割り込み番号を返す
変数宣言	
BYTE	変数を１バイト整数として使用するよう宣言する
INTEGER	変数を２バイト整数として使用するよう宣言する
LONG	変数を４バイト整数として使用するよう宣言する
REAL	変数を４バイト実数として使用するよう宣言する
STRING	変数を文字列変数として使用するよう宣言する
＊！PRIVATE	変数をローカル変数として宣言する

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

2.一般命令

ＢＡＳＩＣ一般命令	
PRINT	パソコンの画面にデータを表示する
＊ PRINT USING	指定した書式（フォーマット）で、画面にデータを表示する
＊！COLOR	テキスト画面に表示される文字の色を指定する
＊！LOCATE	テキスト画面のカーソル位置を制御する
＊！DATA	DREAD文で読み出されるデータを定義する
＊！DREAD	DATA文で定義したデータを読み出し、変数に代入する
＊！RESTORE	DREAD文で読み出すDATA文を指定する
ディスクファイルアクセス命令	
＊！OPEN	ファイルを開く
＊！CLOSE	ファイルを閉じる
＊！PRINT #	ファイルにデータを出力する
＊！INPUT	ファイルからデータを入力し、変数に代入する
＊！LINE INPUT	ファイルから入力されるデータを区切らず一括して文字変数に代入する
＊！INPUT$()	ファイルから指定された長さの文字列を取り出して返す
＊！LOC()	ファイル中の論理的な現在位置を返す
＊！LOF()	ファイルの大きさを返す
＊！EOF()	ファイルの終了コードを返す
変数操作命令	
＊！INC	変数をインクリメントする（カウントＵＰ）
＊！DEC	変数をディクリメントする（カウントＤＯＷＮ）
＊！SWAP	２つの変数の値を入れ替える
Ｉ／Ｏポートアクセス命令	
ON	Ｉ／Ｏを１ビット単位でＯＮする
OFF	Ｉ／Ｏを１ビット単位でＯＦＦする
OUTL	ワード指定したＩ／Ｏ、データメモリの下位８ビットに、データを出力する
OUTH	ワード指定したＩ／Ｏ、データメモリの上位８ビットに、データを出力する
OUTW	ワード指定したＩ／Ｏ、データメモリに１６ビットデータを出力する
OUTD	ワード指定したＩ／Ｏ、データメモリに３２ビットデータを出力する
SW()	Ｉ／Ｏの状態を１ビット単位で返す
INL()	ワード指定したＩ／Ｏ、データメモリの下位８ビットの内容を返す
INH()	ワード指定したＩ／Ｏ、データメモリの上位８ビットの内容を返す
INW()	ワード指定したＩ／Ｏ、データメモリの内容を１６ビット単位で返す
IND()	ワード指定したＩ／Ｏ、データメモリの内容を３２ビット単位で返す
＊！BLKOUT	Ｉ／Ｏへの一括書き込みをする
＊！BLKMOV	Ｉ／Ｏ間のデータ転送をする
タイマ命令	
WAIT	プログラムを一時的に停止状態にする
TMOUT	ＷＡＩＴ文のタイムアウト時間を設定する
システム変数とシステム関数	
！DATE$	日付を設定する
！DATE$()	日付を返す
！TIME$	時刻を設定する
！TIME$()	時刻を返す
＊！TIMER	システム用タイマの経過値を初期化する
＊！TIMER()	システム用タイマの経過値を返す（１０ｍｓ単位）
MYTASK()	タスク番号を返す

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

3.組み込み関数

数値関数	
ABS()	数値の絶対値を返す
INT()	数値の小数点以下を切り捨てる
LSHIFT()	数値をビット単位で左シフトする
RSHIFT()	数値をビット単位で右シフトする
LROLL()	数値をビット単位で左ローテート（回転）する
RROLL()	数値をビット単位で右ローテート（回転）する
BCD()	バイナリコードを２進化10進コードに変換する
BIN()	２進化10進コードをバイナリコードに変換する
DECO()	数値をデコード変換する
ENCO()	数値をエンコード変換する
INV()	数値をビット反転する
NEG()	数値の２の補数を返す
SEGT()	数値を７セグメント表示用データに変換する
UNIT()	複数の数値の下位４ビットを結合する
DIST()	数値を４ビット単位で分離する
＊！RND()	乱数を返す
＊！RANDMIZE	新しい乱数系列を設定する
三角関数	
＊！ATN()	数値の逆正接（アークタンジェント）を返す
＊！COS()	数値の余弦（コサイン）を返す
＊！SIN()	数値の正弦（サイン）を返す
＊！TAN()	数値の正接（タンジェント）を返す
対数関数	
＊ LOG()	数値の自然対数を返す
＊ EXP()	ｅを底にする指数関数の値を返す
SQR()	数値の平方根（スクェアールート）を返す
文字関数	
ASC()	１バイトの文字に対応するキャラクターコードを返す
CHR$()	キャラクターコードに対応する１バイトの文字を返す
LEN()	文字列の文字数を返す
RIGHT$()	文字列の右端から指定した文字数分の文字列を返す
LEFT$()	文字列の左端から指定した文字数分の文字列を返す
MID$()	文字列の指定した位置から指定した文字数分の文字列を返す
＊！MID$()	文字列の一部分を別の文字列で置き換える
SPACE$()	指定した文字数分の空白（スペース）を返す
＊！INSTR()	文字列中の何文字目に指定した文字列があるかを返す

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

数値文字変換関数	
BIN$()	数値を２進表記の文字列に変換する
OCT$()	数値を８進表記の文字列に変換する
HEX$()	数値を16進表記の文字列に変換する
MEW$()	数値をＭＥＷ表記の文字列に変換する
STR$()	数値を文字列に変換する
VAL()	文字列を数値に変換する
＊！MKB$()	１バイト整数を１バイトの文字に変換する
＊！MKW$()	２バイト整数を２バイトの文字列に変換する
＊！MKL$()	４バイト整数を４バイトの文字列に変換する
＊！MKS$()	４バイト実数を４バイトの文字列に変換する
＊！CVB()	１バイトの文字を１バイト整数に変換する
＊！CVW()	２バイトの文字列を２バイト整数に変換する
＊！CVL()	４バイトの文字列を４バイト整数に変換する
＊！CVS()	４バイトの文字列を４バイト実数に変換する

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

■高機能ユニット専用命令

1.リモートＩ／Ｏシステムのスロット割り付け

＊！PRMR	リモートＩ／Ｏ子局の占有スロット数を設定する
＊！SLOTR	リモートＩ／Ｏ子局のスロット割り付けをする
＊！RIOSET	リモートＩ／Ｏシステムに子局の占有スロット数とスロット割り付けの設定を転送する
＊！RIOCLR	リモートＩ／Ｏシステムのスロット割り付けを初期化する（フリーロケーション）
＊！DUMPR	リモートＩ／Ｏ子局の占有スロット数とスロット割り付けの設定値を表示する
＊！PRMR()	リモートＩ／Ｏ子局の占有スロット数の設定値を返す
＊！SLOTR()	リモートＩ／Ｏ子局のスロット割り付けの設定値を返す

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

2.ＭＥＷＮＥＴデータ転送命令

＊ SEND	ワードデータをＭＥＷＮＥＴに送信する
＊ RECV	ワードデータをＭＥＷＮＥＴから受信する
＊ SENDB	ビットデータをＭＥＷＮＥＴに送信する
＊ RECVB	ビットデータをＭＥＷＮＥＴから受信する

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

3.高機能ユニット制御命令

通常スロットの高機能ユニット入出力命令	
＊ PRINT%	高機能ユニットにデータを出力する
＊ INPUT%	高機能ユニットからデータを入力する
＊ WRITE%	Ｉ／Ｏまたはデータメモリから、高機能ユニットにデータを書き込む
＊ READ%	高機能ユニットから、Ｉ／Ｏまたはデータメモリにデータを読み出す
リモートＩ／Ｏシステムの高機能ユニット入出力命令	
＊！PRINTR%	リモートＩ／Ｏ子局の高機能ユニットにデータを出力する
＊！INPUTR%	リモートＩ／Ｏ子局の高機能ユニットからデータを入力する
＊！WRITER%	Ｉ／Ｏまたはデータメモリから、リモートＩ／Ｏ子局の高機能ユニットにデータを書き込む
＊！READR%	リモートＩ／Ｏ子局の高機能ユニットから、Ｉ／Ｏまたはデータメモリにデータを読み出す

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

D／A変換ユニット専用制御命令	
＊！DALIMIT	出力の制限処理を設定する
＊！DALSET	出力の上限値、下限値を設定する
＊！DASET	入力値を設定する
＊！DARDY()	設定完了ステータスを返す
＊！DAERR()	エラーステータスを返す
＊！DAALM()	アラームステータスを返す
A／D変換ユニット専用制御命令	
＊！ADAVRG	平均処理を設定する
＊！ADASET	サンプリング平均回数を設定する
＊！ADLIMIT	出力の制限処理を設定する
＊！ADLSET	出力の上限値、下限値を設定する
＊！ADSCALE	スケーリング処理をするチャンネルを設定する
＊！ADSSET	スケーリング値を設定する
＊！ADDATA	変換データを読み出す
＊！ADRDY()	変換準備完了ステータスを返す
＊！ADERR()	エラーステータスを返す
＊！ADALM()	アラームステータスを返す
＊！ADDATA()	変換データを返す
シリアルデータユニット専用制御命令	
＊！SDRESET	シリアルデータユニットをリセットする
＊！SDTCHR	終端コードを設定する
＊！SDTERM	終端コードの送信指定を設定する
＊！SDTXDT	送信データを書き込む
＊！SDRXDT	受信データを読み出す
！SDSEND	データを送信する（送信可能チェックから受信完了確認まで）
！SDRECV	データを受信する（受信データの有無チェックから受信完了確認まで）
＊！SDRXACK	受信データの読み出し完了を通知する
＊！SDRDY()	動作可能確認ステータスを返す
＊！SDTXRDY()	データ送信可能ステータスを返す
＊！SDTXACK()	データ送信完了ステータスを返す
＊！SDWRX()	データ受信待ち状態で待機する（受信データがあるまでWAITする）
＊！SDERR()	通信エラーステータスを返す
パルス出力ユニット専用制御命令	
＊！POISET	初期値を書き込む
＊！POPSET	目標値を書き込む
！PORESET	カウンタリセットをする
！POSTART	パルス出力をスタートする
＊！POREOS	パルス出力方向を切り替える
！POFCHG	パルス周波数を切り替える
＊！POORG	原点復帰スタートする
＊！POSTAT	ユニットの動作状態を読み出す
＊！PODATA()	経過値を返す

＊印はFP1-BASICタイプでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

高速カウンタユニット専用制御命令	
！HCISET	初期値を書き込む
！HCPSET	目標値を書き込む
＊！HCFSET	入力フィルタ時定数を設定する
！HCRESET	カウンタリセットする
！HCOUTEN	出力を許可
＊！HCSTAT	動作状態を返す
！HCDATA()	経過値を返す
位置決めユニット専用制御命令	
＊！PSTYPE	ユニットのタイプを宣言する
＊！PSCTRL	稼働モードを設定する
＊！PSSTART	ジョブを始動する
＊！PSJOBNO	ジョブ始動するデータNo.を設定する
＊！PSORGH	機械原点復帰する
＊！PSORGS	ソフト原点復帰する
＊！PSSTOP	ジョブを停止する
＊！PSPRM	位置決めユニットのパラメータを共有メモリに設定する
＊！PSPRM()	共有メモリのパラメータの設定値を返す
＊！PSPCLR	共有メモリのパラメータを共有メモリに転送する
＊！PSPSET	共有メモリのパラメータをシステムメモリに転送する
＊！PSSAVE	位置データをディスクにセーブする
＊！PSLOAD	位置データをディスクからロードする
＊！PSDSET	位置決めユニットの位置データを設定する
＊！PSREADY()	準備完了ステータスを返す
＊！PSSTAT()	動作状態を返す
＊！PSADRS()	現在値を返す
＊！PSAID()	補助出力を読み出す
＊！PSBUSY()	動作状態を返す
＊！PSERR()	エラーコードを返す

＊印はFP1-BASICＣＰＵでは使用できません。！印はVer2.00未満のFP3-BASICタイプＣＰＵでは使用できません。

付録D　ＢＡＳＩＣタイプＣＰＵソフトウェア仕様

1 ＣＰＵのプログラム領域

プログラム領域	FP3H-BASICタイプＣＰＵでは128Kバイト FP-BASICタイプＣＰＵでは64Kバイト プログラム領域にはユーザープログラムが格納される。
ファイルメモリ(FL_)	ユーザープログラムの占有範囲を小さくすることにより、プログラム領域の残りの分をファイルメモリに割り当てることができる（データメモリと同様に使用可）。 ファイルメモリは、パラメータメモリのPRM(0)で設定することができる。
変数領域	FP3H-BASICタイプＣＰＵでは約80Kバイト FP3-BASICタイプＣＰＵでは約20Kバイト 変数領域は、ローカル変数領域とグローバル変数領域にあらかじめ分割されて、割り付けられる。現在の割り付け状態は、MAPコマンドで確認できる。また、ローカル変数領域は、MAPコマンドを使用して、タスクごとに１Ｋバイト単位で任意に割り付けることができ、グローバル変数には残りの領域が割り付けられる。

2 入出力・メモリ構成

名称	ワードアドレス	ビットアドレス	機能
外部入力	WX_0～127	X_&M0～127F	入力ユニットに対応するI/0。
外部出力	WY_0～127	Y_&M0～127F	出力ユニットに対応するI/0。 外部に出力しない部分はメモリI/0として使用可。
メモリI/0	WR_0～97	R_&M0～97F	ＣＰＵ内部でだけ使用できるI/0。
特殊メモリI/0	WR900～910	R_&M9000～910F	あらかじめ用途が決まっているI/0。 読み出し専用です。
リンクメモリI/0	WL_0～127F	L_&M0～127F	ＰＣリンク使用時のデータ通信用のメモリI/0。 リンク用に使用しない部分はメモリI/0に使用可。
データメモリ	DT_0～2047		ＣＰＵ内部で使用できるメモリ。
特殊データメモリ	DT_9000～9255		システムで用途が決まっているデータメモリ。 読み出し専用で、書き込みできない。
リンクデータメモリ	LD_0～255		ＰＣリンク使用時のデータ通信用のメモリ。 リンク用に使用しない部分はデータメモリとして使用可。
ファイルメモリ	FL_0～ 任意に設定可能		ユーザープログラムエリアの未使用領域を利用する拡張メモリ。データメモリとして使用可。

＊電源ＯＦＦ時、リセット時、エラー時などの出力およびメモリの保持／非保持については、次ページを参照のこと。
＊ビットアドレスのミュー表記（&M表記）は、最下位の桁が１６進表記、それ以外の桁が１０進表記。
　ワードアドレスは、ミュー表記の最下位桁を除いた値になる。

3 ＢＡＳＩＣタイプＣＰＵの初期化

ＦＰ３ｈ－ＢＡＳＩＣタイプＣＰＵの出力、メモリ、変数などの内容は、電源ＯＦＦ、イニシャライズスイッチの操作、動作モードスイッチの操作、初期化コマンド実行などにより、次表のように初期化される。なお、保持・非保持の設定はパラメータメモリで設定できる。

モード													
	RUN REMOTE PROG.	RUN	−	○	○		○	○		○	○		○	○
			−	○	○		○	○		○	○		○	○
		PROG.	−	○		○	○	○	○	○	○	○	○	○
			−	○		○	○	○	○	○	○	○	○	○
操作内容／コマンド実行			CPUの電源投入	CPUのイニシャライズスイッチ	PROG.モードへの切り替え	RUNモードへの切り替え	VERINITコマンド	RESETコマンド	CLRVARコマンド	PRMCLRコマンド	SLOTCLRコマンド	DWNLDコマンド	REFORCE ALLコマンド	STOPキー押下
パラメータメモリ							○			○				
I/Oマップ							○				○			
プログラム							○		○			○		
I/O出力(Y_)		保持		○			○	○						
		非保持	○	○			○	○						○
		プログラムモード非保持	↑	○	○	○	○	○						↑
メモリI/O(R_)		保持		○			○	○						
		非保持	○	○			○	○						○
データメモリ(DT_)		保持		○			○	○						
		非保持	○	○			○	○						○
リンクメモリI/O(L_)		保持		○			○	○						
		非保持	○	○			○	○						○
リンクデータメモリ(LD_)		保持		○			○	○						
		非保持	○	○			○	○						○
ファイルメモリ(FL_)		保持		○			○	○						
		非保持	○	○			○	○						○
グローバル変数			*	*		*	○	*	○			○		
強制入出力			○	○			○	○					○	○

＊印は、Ver.2.00未満のFP3-BASICタイプＣＰＵのみ初期化される。

4 パラメータメモリ設定一覧（VERINITコマンド実行時）

No. 説明	初期値（出荷時）	表示設定範囲の説明	
0* ユーザープログラム領域	128	16～128Kバイト	*1
7 メモリI/Oの保持エリア	60	0～98（ワード単位で設定可）	*2
8 データメモリの保持エリア	0	0～2048	*2
9* ファイルメモリの保持エリア	0	0～57344ワード	*2
10 PCリンク0 リンクメモリI/Oの保持エリア	0	0～64（ワード単位で設定可）	*2
11 PCリンク1 リンクメモリI/Oの保持エリア	64	64～128（ワード単位で設定可）	*2
12 PCリンク0 リンクデータメモリの保持エリア	0	0～128	*2
13 PCリンク1 リンクデータメモリの保持エリア	128	128～256	*2
15 出力の保持・非保持の設定	0	0：非保持　1：保持	
16* プログラムモードの出力の保持・非保持の設定	1	0：非保持　1：保持	
19* 保持・非保持エリアの設定方向	1	0：設定値未満を保持 (Ver.2.00未満のCPUと同じ) 1：設定値以降を保持	 *2
21 出力ユニットヒューズ断時	0	0：ＰＣ停止　1：ＰＣ運転継続	
22 特殊ユニット異常時	0	0：ＰＣ停止　1：ＰＣ運転継続	
23 I/O照合の異常時	0	0：ＰＣ停止　1：ＰＣ運転継続	
24 演算エラーの発生時	0	0：ＰＣ停止　1：ＰＣ運転継続	
26* エラー発生時	0	0：ＰＣ停止　1：ＰＣ運転継続	
27* リモートI/O交信異常時	0	0：ＰＣ停止　1：ＰＣ運転継続	
28* リモートI/O子局上のユニット異常時	0	0：ＰＣ停止　1：ＰＣ運転継続	
32 MEWNET SEND-RECV命令タイムアウト時間	800	10ms～81.9sec（2.5×設定値）	
35* リモートI/O子局接続待ちモードの設定 (接続を待って運転を開始するかどうか)	1	0：解除(接続を待たずに運転開始) 1：設定(接続を待って運転開始)	
40 PCリンク0 リンクメモリI/Oのリンク範囲	0	0～64ワード	*3
41 PCリンク0 リンクデータメモリのリンク範囲	0	0～128ワード	*3
42 PCリンク0 リンクメモリI/O送信開始ワードNo.	0	0～63	
43 PCリンク0 リンクメモリI/O送信ワード数	0	0～64	
44 PCリンク0 リンクデータメモリ送信開始ワードNo.	0	0～127	
45 PCリンク0 リンクデータメモリ送信ワード数	0	0～127	
50 PCリンク1 リンクメモリI/Oのリンク範囲	0	0～64ワード	*3
51 PCリンク1 リンクデータメモリのリンク範囲	0	0～128ワード	*3
52 PCリンク1 リンクメモリI/O送信開始ワードNo.	64	64～127	
53 PCリンク1 リンクメモリI/O送信ワード数	0	0～64	
54 PCリンク1 リンクデータメモリ送信開始ワードNo.	128	128～255	
55 PCリンク1 リンクデータメモリ送信ワード数	0	0～127	

*No.0・9・16・19・26・27・28・35はVer.2.00未満のBASICタイプＣＰＵには存在しません。

*1 設定したユーザープログラム領域以外をファイルメモリ（FL_）として使用することができる。

*2 指定値未満（指定値を含まない）を保持にするか、指定値以降（指定値を含む）を保持にするかは、パラメータメモリNo.19で設定できる。デフォルトは設定値以降を保持。

*3 リンクエリアはアドレス0から設定値未満（設定値を含まない）までが利用できる。設定値以降（設定値を含む）のアドレスはメモリI/Oまたはデータメモリとして使用することができる。

ＦＰ１－ＢＡＳＩＣタイプの場合は、上記表の内7・8・26だけが有効です。
ＦＰ１－ＢＡＳＩＣタイプの他のパラメータについては、マニュアルをお読みください。

5 特殊メモリＩ／Ｏ一覧

ＦＰ３Ｈ ＢＡＳＩＣ ＣＰＵ ワードアドレスWR＿９００		
R_&M9000	自己診断エラー	0：正常時 1：検出時　エラーコードはDT_9000に格納。このコードはＥＲＲに反映。
R_&M9001	瞬停検知	0：正常時 1：検出時　DT_9001に検出した瞬停回数を格納。
R_&M9002	ヒューズ断検知	0：正常時 1：検出時　DT_9002,9003に検出したスロット番号を格納。
R_&M9003	特殊ユニット異常	0：正常時 1：検出時　DT_9004,9005に検出したスロット番号を格納。
R_&M9004	I/O照合異常	0：正常時 1：検出時　DT_9010,9011に検出したスロット番号を格納。
R_&M9005	電池異常	0：正常時 1：検出時
R_&M9006	電池異常保持	0：正常時 1：検出時　一度電池異常を検出すると復帰後も保持。 電源ＯＦＦかイニシャライズＳＷ操作で０に復帰。
R_&M9007	演算異常	0：正常時 1：検出時　発生した行番号をDT_9017に格納。
R_&M9008	リモートI/O Ｒ／Ｗエラー	0：正常時 1：検出時　コントロールデータが指定外の時。 マスタユニットがない時。 アドレス修飾エラーが発生している時。 書き込みデータが指定範囲を越えている時。
ＦＰ３Ｈ ワードアドレスWR＿９０１ ＢＡＳＩＣ ＣＰＵ		
R_&M9010	常時ＯＮ	
R_&M9011	常時ＯＦＦ	
R_&M901A	0.1秒クロックパルス	
R_&M901B	0.2秒クロックパルス	
R_&M901C	1秒クロックパルス	
R_&M901D	2秒クロックパルス	
R_&M901E	1分クロックパルス	

FP3H BASIC CPU ワードアドレスWR_902		
R_&M9020	RUNモード	0：PROG. モード時 1：RUNモード時
R_&M9021	テストモード	0：通常モード時 1：テストモード時
R_&M9022	ポーズ	0：プログラム実行中 1：PAUSE命令またはSTEP命令で一時停止中
R_&M9023	ポーズモード	0：テストRUN時のポーズ命令無効 1：テストRUN時のポーズ命令有効
R_&M9024	シミュレーションモード	0：テストRUN時にも通常出力をする 1：テストRUN時に実出力をしない
R_&M9027	リモートモード	0：CPUリモートモード時以外 1：CPUリモートモード時（PROG.⟷RUN切り替え可）
R_&M9028	強制モード	0：強制入出力をしていないとき（解除時） 1：I/O、メモリI/Oの強制入出力をしているとき（解除時）
R_&M902B	割り込み異常フラグ	0：割り込み正常時 1：割り込み異常発生時
FP3H BASIC CPU ワードアドレスWR_903		
R_&M9030	MEWNET送受信命令実行可（SEND/RECV命令）	0：実行不可（実行中） 1：実行可
R_&M9031	MEWNET送受信命令実行完了（SEND/RECV命令）	0：正常終了 1：異常終了　異常コードはDT_9039にセットされる。
R_&M9035	リモートI/Oメモリアクセス命令実行可能	0：実行不可（実行中） 1：実行可
R_&M9036	リモートI/Oメモリアクセス命令完了	0：正常終了 1：異常終了　異常コードはDT_9036にセットされる。
FP3H BASIC CPU ワードアドレスWR 905		
R_&M9050	MEWNETリンク１伝送異常	0：正常時 1：伝送異常または設定異常
R_&M9051	MEWNETリンク２伝送異常	0：正常時 1：伝送異常または設定異常
R_&M9052	MEWNETリンク３伝送異常	0：正常時 1：伝送異常または設定異常

FP3H BASIC CPU ワードアドレスWR_906		
R_&M9060	PCリンク0用 ユニットNo. 1 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9061	PCリンク0用 ユニットNo. 2 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9062	PCリンク0用 ユニットNo. 3 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9063	PCリンク0用 ユニットNo. 4 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9064	PCリンク0用 ユニットNo. 5 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9065	PCリンク0用 ユニットNo. 6 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9066	PCリンク0用 ユニットNo. 7 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9067	PCリンク0用 ユニットNo. 8 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9068	PCリンク0用 ユニットNo. 9 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9069	PCリンク0用 ユニットNo. 10 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M906A	PCリンク0用 ユニットNo. 11 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M906B	PCリンク0用 ユニットNo. 12 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M906C	PCリンク0用 ユニットNo. 13 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M906D	PCリンク0用 ユニットNo. 14 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M906E	PCリンク0用 ユニットNo. 15 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M906F	PCリンク0用 ユニットNo. 16 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時

FP3H BASIC CPU ワードアドレスWR_907		
R_&M9070	PCリンク0用　ユニットNo. 1 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9071	PCリンク0用　ユニットNo. 2 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9072	PCリンク0用　ユニットNo. 3 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9073	PCリンク0用　ユニットNo. 4 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9074	PCリンク0用　ユニットNo. 5 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9075	PCリンク0用　ユニットNo. 6 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9076	PCリンク0用　ユニットNo. 7 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9077	PCリンク0用　ユニットNo. 8 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9078	PCリンク0用　ユニットNo. 9 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9079	PCリンク0用　ユニットNo. 10 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M907A	PCリンク0用　ユニットNo. 11 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M907B	PCリンク0用　ユニットNo. 12 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M907C	PCリンク0用　ユニットNo. 13 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M907D	PCリンク0用　ユニットNo. 14 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M907E	PCリンク0用　ユニットNo. 15 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M907F	PCリンク0用　ユニットNo. 16 動作モードメモリI/O	0:PROG. 時 1:RUN時

FP3H BASIC CPU ワードアドレスWR_908		
R_&M9080	PCリンク1用 ユニットNo. 1 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9081	PCリンク1用 ユニットNo. 2 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9082	PCリンク1用 ユニットNo. 3 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9083	PCリンク1用 ユニットNo. 4 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9084	PCリンク1用 ユニットNo. 5 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9085	PCリンク1用 ユニットNo. 6 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9086	PCリンク1用 ユニットNo. 7 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9087	PCリンク1用 ユニットNo. 8 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9088	PCリンク1用 ユニットNo. 9 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M9089	PCリンク1用 ユニットNo. 10 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M908A	PCリンク1用 ユニットNo. 11 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M908B	PCリンク1用 ユニットNo. 12 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M908C	PCリンク1用 ユニットNo. 13 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M908D	PCリンク1用 ユニットNo. 14 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M908E	PCリンク1用 ユニットNo. 15 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時
R_&M908F	PCリンク1用 ユニットNo. 16 伝送保証メモリI/O	0:伝送異常時または停止時 1:伝送正常時

FP3H BASIC CPU ワードアドレスWR_909		
R_&M9090	PCリンク1用　ユニットNo. 1 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9091	PCリンク1用　ユニットNo. 2 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9092	PCリンク1用　ユニットNo. 3 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9093	PCリンク1用　ユニットNo. 4 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9094	PCリンク1用　ユニットNo. 5 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9095	PCリンク1用　ユニットNo. 6 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9096	PCリンク1用　ユニットNo. 7 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9097	PCリンク1用　ユニットNo. 8 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9098	PCリンク1用　ユニットNo. 9 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M9099	PCリンク1用　ユニットNo. 10 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M909A	PCリンク1用　ユニットNo. 11 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M909B	PCリンク1用　ユニットNo. 12 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M909C	PCリンク1用　ユニットNo. 13 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M909D	PCリンク1用　ユニットNo. 14 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M909E	PCリンク1用　ユニットNo. 15 動作モードメモリI/O	0:PROG. 時 1:RUN時
R_&M909F	PCリンク1用　ユニットNo. 16 動作モードメモリI/O	0:PROG. 時 1:RUN時

FP1 BASIC CPU		
R_&M9000	自己診断エラー	0：正常時 1：検出時　エラーコードはDT_9000に格納。このコードはＥＲＲに反映。
R_&M9005	電池異常	0：正常時 1：検出時
R_&M9007	電池異常保持	0：正常時 1：検出時　一度電池異常を検出すると復帰後も保持。 電源ＯＦＦで０に復帰。
R_&M9008	演算異常	0：正常時 1：検出時。
R_&M901A	0.1秒クロックパルス	
R_&M901B	0.2秒クロックパルス	
R_&M901C	1秒クロックパルス	
R_&M901D	2秒クロックパルス	
R_&M9020	ＲＵＮモード	0：ＰＲＯＧ．モード時 1：ＲＵＮモード時
R_&M9027	リモートモード	0：ＣＰＵリモートモード時以外 1：ＣＰＵリモートモード時（PROG.←→RUN切り替え可）
R_&M902A	外部割り込み許可	0：禁止時 1：許可時
R_&M902B	割り込み異常フラグ	0：正常時 1：異常時
R_&M9032	ＲＳ２３２Ｃポート選択	0：シリアル通信使用時以外 1：シリアル通信使用時
R_&M9036	I/Oリンク異常	0：正常時 1：異常時
R_&M9037	ＲＳ２３２Ｃ伝送エラー	0：正常時 1：異常時
R_&M9038	ＲＳ２３２Ｃ受信完了	0：ターミネータ受信時以外 1：ターミネータ受信時
R_&M9039	ＲＳ２３２Ｃ送信完了	0：送信要求時 1：送信完了時
R_&M903A	高速カウンタ制御中	0：高速カウンタ停止中 1：高速カウンタ実行中

6 特殊データメモリ一覧

FP3H BASIC CPU

アドレス	名称	内容／備考
DT_9000	自己診断コード	自己診断の結果が格納されます。
DT_9002	ヒューズ断ユニット	スロット0 ・・・・・・・・・・・・・・・ スロット15 bit0 ・・・・・・・・・・・・・・・ bit15　　0：正常　1：異常
DT_9003	ヒューズ断ユニット	スロット16 ・・・・・・・・・・・・・・ スロット31 bit0 ・・・・・・・・・・・・・・・ bit15　　0：正常　1：異常
DT_9004	異常高機能ユニット	スロット0 ・・・・・・・・・・・・・・・ スロット15 bit0 ・・・・・・・・・・・・・・・ bit15　　0：正常　1：異常
DT_9005	異常高機能ユニット	スロット16 ・・・・・・・・・・・・・・ スロット31 bit0 ・・・・・・・・・・・・・・・ bit15　　0：正常　1：異常
DT_9010	照合異常ユニット	スロット0 ・・・・・・・・・・・・・・・ スロット15 bit0 ・・・・・・・・・・・・・・・ bit15　　0：正常　1：異常
DT_9011	照合異常ユニット	スロット16 ・・・・・・・・・・・・・・ スロット31 bit0 ・・・・・・・・・・・・・・・ bit15　　0：正常　1：異常
DT_9017	演算エラー行番号	最初の演算エラーが発生した行番号が格納されます。
DT_9020	ユーザープログラム容量	PRM(0)に設定されたユーザープログラムの容量が格納。 単位：ワード。
DT_902	ファイルメモリ最大値	ファイルメモリ（FL_）の最大番号が格納されます。
DT_9022	スキャンタイム	スキャンタイム（マルチタスクの１周期）の現在の値が格納されます。 例：200→20ms以内　225→22.5ms以内
DT_9023	スキャンタイム	スキャンタイム（マルチタスクの１周期）の最小値が格納されます。
DT_9024	スキャンタイム	スキャンタイム（マルチタスクの１周期）の最大値が格納されます。
DT_9029	ポーズ行番号	テストＲＵＮ時のポーズしている行番号が格納されます。
DT_9036	リモートI/Oメモリアクセス命令完了コード	H00：正常時：０ H5B：タイムアウト（相手先不在で送信不可状態） H68：ｱｸｾｽｴﾘｱ無ｴﾗｰ （指定したメモリアクセスエリアがスレーブユニット上に存在しない） H71：送信アンサー待ちタイムアウトエラー H72：送信バッファアンサー空き待ちタイムアウトエラー H73：レスポンス待ちタイムアウトエラー
DT_9039	MEWNET送受信命令完了コード（SEND/RECV）	H00：正常終了 H71：送信アンサー待ちタイムアウトエラー H72：送信バッファアンサー空き待ちタイムアウトエラー H73：レスポンス待ちタイムアウトエラー

FP3H BASIC CPU

アドレス	名称	内容／備考
DT_9140	ＰＣリンクステータス	ＰＣリンク０の受信回数　ＲＩＮＧカウンタ
DT_9141	ＰＣリンクステータス	ＰＣリンク０の受信間隔　現在値（*2.5ms）
DT_9142	ＰＣリンクステータス	ＰＣリンク０の受信間隔　最小値（*2.5ms）
DT_9143	ＰＣリンクステータス	ＰＣリンク０の受信間隔　最大値（*2.5ms）
DT_9144	ＰＣリンクステータス	ＰＣリンク０の送信回数　ＲＩＮＧカウンタ
DT_9145	ＰＣリンクステータス	ＰＣリンク０の送信間隔　現在値（*2.5ms）
DT_9146	ＰＣリンクステータス	ＰＣリンク０の送信間隔　最小値（*2.5ms）
DT_9147	ＰＣリンクステータス	ＰＣリンク０の送信間隔　最大値（*2.5ms）
DT_9148	ＰＣリンクステータス	ＰＣリンク１の受信回数　ＲＩＮＧカウンタ
DT_9149	ＰＣリンクステータス	ＰＣリンク１の受信間隔　現在値（*2.5ms）
DT_9150	ＰＣリンクステータス	ＰＣリンク１の受信間隔　最小値（*2.5ms）
DT_9151	ＰＣリンクステータス	ＰＣリンク１の受信間隔　最大値（*2.5ms）
DT_9152	ＰＣリンクステータス	ＰＣリンク１の送信回数　ＲＩＮＧカウンタ
DT_9153	ＰＣリンクステータス	ＰＣリンク１の送信間隔　現在値（*2.5ms）
DT_9154	ＰＣリンクステータス	ＰＣリンク１の送信間隔　最小値（*2.5ms）
DT_9155	ＰＣリンクステータス	ＰＣリンク１の送信間隔　最大値（*2.5ms）
DT_9160	リンクユニットＮｏ．	ＬＩＮＫ１のユニットＮｏ．が格納されます。
DT_9161	異常フラグ	ＬＩＮＫ１の異常フラグが格納されます。
DT_9162	リンクユニットＮｏ．	ＬＩＮＫ２のユニットＮｏ．が格納されます。
DT_9163	異常フラグ	ＬＩＮＫ２の異常フラグが格納されます。
DT_9164	リンクユニットＮｏ．	ＬＩＮＫ３のユニットＮｏ．が格納されます。
DT_9165	異常フラグ	ＬＩＮＫ３の異常フラグが格納されます。
DT_9170	ＬＩＮＫ１ステータス	ＰＣ－ＬＩＮＫアドレス重複先
DT_9171	ＬＩＮＫ１ステータス	テストモード結果
DT_9172	ＬＩＮＫ１ステータス	トークン紛失回数
DT_9173	ＬＩＮＫ１ステータス	２重トークン回数
DT_9174	ＬＩＮＫ１ステータス	無信号状態回数
DT_9175	ＬＩＮＫ１ステータス	同期異常回数
DT_9176	ＬＩＮＫ１ステータス	送信ＮＡＣＫ
DT_9177	ＬＩＮＫ１ステータス	送信ＮＡＣＫ
DT_9178	ＬＩＮＫ１ステータス	送信ＷＡＣＫ
DT_9179	ＬＩＮＫ１ステータス	送信ＷＡＣＫ
DT_9180	ＬＩＮＫ１ステータス	送信アンサー
DT_9181	ＬＩＮＫ１ステータス	送信アンサー
DT_9182	ＬＩＮＫ１ステータス	未定義コマンド
DT_9183	ＬＩＮＫ１ステータス	パリティエラー回数
DT_9184	ＬＩＮＫ１ステータス	Ｅｎｄｃｏｄｅ受信エラー
DT_9185	ＬＩＮＫ１ステータス	フォーマットエラー
DT_9186	ＬＩＮＫ１ステータス	ＮＯＴサポート
DT_9187	ＬＩＮＫ１ステータス	自己診断結果
DT_9188	ＬＩＮＫ１ステータス	ループ切り替え回数
DT_9189	ＬＩＮＫ１ステータス	リンク不可状態発生回数
DT_9190	ＬＩＮＫ１ステータス	主ルート入力断線回数
DT_9191	ＬＩＮＫ１ステータス	副ルート入力断線回数
DT_9192	ＬＩＮＫ１ステータス	ループ再構成処理中
DT_9193	ＬＩＮＫ１ステータス	ループ運転モード
DT_9194	ＬＩＮＫ１ステータス	ループ入力状態

FP3H BASIC CPU

アドレス	名称	内容／備考
DT_9200	ＬＩＮＫ２ステータス	ＰＣ－ＬＩＮＫアドレス重複先
DT_9201	ＬＩＮＫ２ステータス	テストモード結果
DT_9202	ＬＩＮＫ２ステータス	トークン紛失回数
DT_9203	ＬＩＮＫ２ステータス	２重トークン回数
DT_9204	ＬＩＮＫ２ステータス	無信号状態回数
DT_9205	ＬＩＮＫ２ステータス	同期異常回数
DT_9206	ＬＩＮＫ２ステータス	送信ＮＡＣＫ
DT_9207	ＬＩＮＫ２ステータス	送信ＮＡＣＫ
DT_9208	ＬＩＮＫ２ステータス	送信ＷＡＣＫ
DT_9209	ＬＩＮＫ２ステータス	送信ＷＡＣＫ
DT_9210	ＬＩＮＫ２ステータス	送信アンサー
DT_9211	ＬＩＮＫ２ステータス	送信アンサー
DT_9212	ＬＩＮＫ２ステータス	未定義コマンド
DT_9213	ＬＩＮＫ２ステータス	パリティエラー回数
DT_9214	ＬＩＮＫ２ステータス	Ｅｎｄｃｏｄｅ受信エラー
DT_9215	ＬＩＮＫ２ステータス	フォーマットエラー
DT_9216	ＬＩＮＫ２ステータス	ＮＯＴサポート
DT_9217	ＬＩＮＫ２ステータス	自己診断結果
DT_9218	ＬＩＮＫ２ステータス	ループ切り替え回数
DT_9219	ＬＩＮＫ２ステータス	リンク不可状態発生回数
DT_9220	ＬＩＮＫ２ステータス	主ルート入力断線回数
DT_9221	ＬＩＮＫ２ステータス	副ルート入力断線回数
DT_9222	ＬＩＮＫ２ステータス	ループ再構成処理中
DT_9223	ＬＩＮＫ２ステータス	ループ運転モード
DT_9224	ＬＩＮＫ２ステータス	ループ入力状態
DT_9230	ＬＩＮＫ３ステータス	ＰＣ－ＬＩＮＫアドレス重複先
DT_9231	ＬＩＮＫ３ステータス	テストモード結果
DT_9232	ＬＩＮＫ３ステータス	トークン紛失回数
DT_9233	ＬＩＮＫ３ステータス	２重トークン回数
DT_9234	ＬＩＮＫ３ステータス	無信号状態回数
DT_9235	ＬＩＮＫ３ステータス	同期異常回数
DT_9236	ＬＩＮＫ３ステータス	送信ＮＡＣＫ
DT_9237	ＬＩＮＫ３ステータス	送信ＮＡＣＫ
DT_9238	ＬＩＮＫ３ステータス	送信ＷＡＣＫ
DT_9239	ＬＩＮＫ３ステータス	送信ＷＡＣＫ
DT_9240	ＬＩＮＫ３ステータス	送信アンサー
DT_9241	ＬＩＮＫ３ステータス	送信アンサー
DT_9242	ＬＩＮＫ３ステータス	未定義コマンド
DT_9243	ＬＩＮＫ３ステータス	パリティエラー回数
DT_9244	ＬＩＮＫ３ステータス	Ｅｎｄｃｏｄｅ受信エラー
DT_9245	ＬＩＮＫ３ステータス	フォーマットエラー
DT_9246	ＬＩＮＫ３ステータス	ＮＯＴサポート
DT_9247	ＬＩＮＫ３ステータス	自己診断結果
DT_9248	ＬＩＮＫ３ステータス	ループ切り替え回数
DT_9249	ＬＩＮＫ３ステータス	リンク不可状態発生回数
DT_9250	ＬＩＮＫ３ステータス	主ルート入力断線回数
DT_9251	ＬＩＮＫ３ステータス	副ルート入力断線回数

FP3H BASIC CPU

アドレス	名称	内容／備考
DT_9252	ＬＩＮＫ３ステータス	ループ再構成処理中
DT_9253	ＬＩＮＫ３ステータス	ループ運転モード
DT_9254	ＬＩＮＫ３ステータス	ループ入力状態

FP1 BASIC CPU

アドレス	名称	内容／備考
DT_9000	自己診断コード	自己診断の結果が格納されます。
DT_9019	2.5ms ＲＩＮＧカウンタ	2.5ms毎に自動的に＋1するデータメモリ。 前回値を記憶していて、今回値と減算後絶対値変換することにより、 経過時間を得ます。
DT_9022	スキャンタイム	スキャンタイム（マルチタスクの１周期）の現在の値が格納されます。 例）200→20ms以内　225→22.5ms以内
DT_9023	スキャンタイム	スキャンタイム（マルチタスクの１周期）の最小値が格納されます。
DT_9024	スキャンタイム	スキャンタイム（マルチタスクの１周期）の最大値が格納されます。
DT_9040	ボリューム入力0	ボリューム０の値（0～255）を格納
DT_9041	ボリューム入力1	ボリューム１の値（0～255）を格納
DT_9042	ボリューム入力2	ボリューム２の値（0～255）を格納
DT_9043	ボリューム入力3	ボリューム３の値（0～255）を格納
DT_9044	高速カウンタ経過値	高速カウンタの経過値の下位16ビットを格納
DT_9045	高速カウンタ経過値	高速カウンタの経過値の上位16ビットを格納
DT_9046	高速カウンタ目標値	高速カウンタの目標値の下位16ビットを格納
DT_9047	高速カウンタ目標値	高速カウンタの目標値の上位16ビットを格納
DT_9052	高速カウンタ制御	高速カウンタ制御命令ＭＶでカウンタのリセットカウント許可、 クリアなど指定するときに使用。
DT_9059	シリアル通信異常コード	上位8ビット・・・・RS232Cの異常コードが格納されます。 下位8ビット・・・・RS422の異常コードが格納されます。

7 ユニットの入出力点数と機能番号

ユニット品名		製品番号	入力点数	出力点数	機能番号
空ユニット			－ *	－	302
８・１６点入力ユニット			8/16	－	120
３２点入力ユニット			32	－	130
６４点入力ユニット			64	－	140
１２８点入力ユニット			128	－	150
８・１６点出力ユニット			－	8/16	102
３２点出力ユニット			－	32	103
６４点出力ユニット			－	64	104
１２８点出力ユニット			－	128	105
Ａ／Ｄ変換ユニット		AFP3400	16	－	220
Ｄ／Ａ変換ユニット		AFP3410/3411	16	－	220
シリアルデータユニット		AFP3460	16	16	222
パルス出力ユニット		AFP3480	16	16	222
高速カウンタユニット	２ＣＨタイプ	AFP3621	16	16	222
	４ＣＨタイプ	AFP3622	16	16	222
位置決めユニットＦタイプ	１軸タイプ	AFP3431	16	16	222
	２軸タイプ	AFP3432	32	32	233
割り込みユニット		AFP3452	16	－	220
ID/Xコントロールユニット	１ＣＨタイプ	AFP3470	16	16	222
	２ＣＨタイプ	AFP34701	16	16	222
	1CH + RS232C	AFP34702	16	16	222
マイクロ波Ｉ／Ｄコントロールユニット		AFP3471	16	16	222
バーコードリーダコントロールユニット		AFP34601	－ *	－	402
データプロセスユニット		AFP3461	16	16	222
リモートI/Oマスタユニット		AFP3740	－ *	－	402

＊空ユニット、ＭＥＷＮＥＴ－Ｈ／Ｗ／Ｐユニット、ＣＣＵ、バーコードリーダインターフェイスユニットなど実際に入出力を持たないユニットは、 SLOTCLRコマンドを実行した場合、１ワード分の入出力点数を占有します（フリーロケーション）。ただし、これらのユニットも、SLOTコマンドを使用すれば、入出力占有点数を０として割り付けることができます（任意割り付け）。

付録Ｅ　エラーコード一覧

1 ＰＣ自己診断エラーコード一覧　　表示：<< SELF CHECK ERROR 23 !! >>

エラーコード	エラー内容
ERR 23	ＲＡＭ異常１エラー
ERR 25	ＲＡＭ異常２エラー
ERR 26	ユーザーＲＯＭチェックエラー
ERR 27	特殊ユニット装着制限エラー
ERR 28	パラメータメモリ異常エラー
ERR 29	メモリ管理テーブル異常エラー
ERR 30	割り込み異常１エラー
ERR 31	割り込み異常２エラー
ERR 33	マルチパラメータ照合チェックエラーコード
ERR 35	スレーブ上に禁止ユニットが実装
ERR 36	リモートＩ／Ｏ仕様制限エラー（重複あるいは範囲オーバー）
ERR 38	Ｉ／Ｏターミナルボード登録エラー
ERR 40	出力ユニットヒューズ断エラー
ERR 41	特殊ユニット暴走エラー
ERR 42	Ｉ／Ｏ照合エラー
ERR 45	演算エラー発生
ERR 46	リモートＩ／Ｏ交信エラー
ERR 47	スレーブ上のＩ／Ｏ属性エラー
ERR 50	電池電圧低下エラー
ERR 51	リモートＩ／Ｏ終端局エラー
ERR 53	ラダー用Ｉ／Ｏマップ照合チェック警告コード
ERR 60	ＥＳＢ受信バッファオーバーフローエラー

2 ＦＰ－ＢＡＳＩＣエラーコード一覧　　表示：!! ERROR 31

エラーコード	エラー内容
ERR 1	ＮＥＸＴ文に対するＦＯＲ文がない
ERR 2	文法上のエラーが発生した
ERR 3	ＧＯＳＵＢ文がないのに、ＲＥＴＵＲＮ文を実行しようとした
ERR 4	プログラム中にＧＯＴＯ文、ＧＯＳＵＢ文合わせて４４８個以上ある または、ラベルが４１０個以上ある
ERR 5	関数のパラメータが規定範囲外である
ERR 6	数値や変数の演算オーバーフローが発生した
ERR 7	ＧＯＳＵＢ～ＲＥＴＵＲＮ文のネストが規定値を越えた
ERR 8	ＧＯＴＯ、ＧＯＳＵＢで呼ばれた行番号がない
ERR 9	宣言されていない配列変数を使用しようとした 配列の大きさが定義サイズを越えた（配列領域オーバー）
ERR 10	変数を２重定義しようとした
ERR 11	数値や変数を０で割った
ERR 12	コマンドとして使用すべき命令をステートメントで使用した
ERR 13	左カッコと右カッコの数が合わない
ERR 14	パラメータの数が合わない（配列で配列要素が合わない）
ERR 15	定義されていない関数、変数または配列を使用しようとした
ERR 16	ＦＵＮＣＴＩＯＮ～ＦＥＮＤ間以外にプログラムがある
ERR 17	ＦＵＮＣＴＩＯＮ～ＦＥＮＤの中でＦＵＮＣＴＩＯＮ宣言をした
ERR 18	変数宣言が行の先頭にない
ERR 21	変数の型変換でエラーが発生した（オーバーフロー）
ERR 22	パラメータの設定テーブルが設定範囲を越えた

エラーコード	エラー内容
ERR 23	1行の文字数が多すぎる（255文字まで）
ERR 24	データタグエラー （未定義中間コード命令エラー・未定義コマンド実行・パラメータタイプ不適当）。
ERR 25	数値変数エラー 桁が多い（ＩＮＰＵＴ文によるＡＳＣＩＩ文字が数値に正しく変換できない）。
ERR 26	存在しない機器を使用しようとした
ERR 27	存在しない入出力番号を指定した
ERR 28	ＦＯＲループにＮＥＸＴ文がない
ERR 30	ＩＮＰＵＴ文によるデータの受信数が合わない
ERR 31	通信タイムアウトエラーが発生した
ERR 32	ＳＲＣ－４２が立ち上がっているのにＲＩＯＣの準備ができていない
ERR 33	ＲＳ２３２Ｃバッファオーバーフロー。受信バッファいっぱいでさらに送信された
ERR 34	パリティ・オーバーラン・フレーミングエラーが発生した
ERR 36	受信時のバッファオーバーフロー（入力が８０文字を越えた）
ERR 39	指定した行番号が存在しない。
ERR 40	指定したタスクが存在しない。
ERR 41	指定したタスクが起動できない（実行中のタスク）
ERR 45	タスク実行中に許されないコマンドを実行した
ERR 53	プログラムファイルが存在しない
ERR 57	同じ名前のプログラムファイルが既に存在している
ERR 60	ディスクに空きスペースがない
ERR 62	プログラムファイルの名前が正しくない
ERR 63	ディスクが準備できていない
ERR 66	ディス書き込みエラー
ERR 67	ドライブセレクトエラー
ERR 68	書き込み禁止ディスクに書き込もうとした
ERR 70	ＧＯＳＵＢ文がないのにＲＥＴＵＲＮを実行しようとした
ERR 71	数値オーバーフロー
ERR 73	パラメータ数値が範囲外
ERR 75	中間コードエラー （未定義タグエラー・宣言または定義されていない関数変数を使用しようとした）
ERR 79	ENDIF文に対するIF文がない
ERR 80	ELSE文に対するIF文がない
ERR 81	IF文に対するENDIF文がない
ERR 83	関数や変数が長すぎる（中間コード定義領域が足りない）
ERR 84	オブジェクトオーバーフロー（中間コードの格納領域が足りない）
ERR 85	EXITの出口がない
ERR 86	システムワークのメモリチェックエラー
ERR 87	ファイルメモリのメモリチェックエラー
ERR 88	DOに対するLOOP文がない
ERR 90	IF、SELECT等のネストオーバー
ERR 89	LOOP文に対するDOがない
ERR 93	関数に与えるパラメータ数が合わない
ERR 100	デバイス通信エラー（シリアルＢＵＳ）
ERR 135	タグが異常
ERR 136	与えられた関数が存在しない
ERR 137	規定範囲外のパラメータが指定された
ERR 208	1行の文字数が58を越えた（PRINT文は58文字まで）

エラーコード	エラー内容
ERR 258	オブジェクトがありません
ERR 259	ラベル情報が大きすぎる
ERR 260	ＦＵＮＣＴＩＯＮが大きすぎる
ERR 261	メモリが足りません
ERR 1002	タスク起動中のＸＱＴ
ERR 1003	ＨＡＬＴ中の再起動異常
ERR 1004	タイムアウト
ERR 1005	指定されたシンボルが存在しない
ERR 1006	モードが違っている
ERR 1007	スタックが空である
ERR 1008	コマンドの使用方法が間違っている
ERR 1009	パラメータが間違っている（DT_・LD_のビット操作）
ERR 1010	未定値演算
ERR 1011	割り込み中のサブルーチンコール
ERR 1012	指定スタックがＰＡＵＳＥ状態ではない
ERR 1013	メモリＲＯＭエラー
ERR 1014	メモリプロテクトエラー
ERR 1015	ＭＥＷＮＥＴデータリンクエラー
ERR 1016	ＭＥＷＮＥＴデータリンクがアクティブでない
ERR 1017	指定されたスロットに共有ＲＡＭは存在しない
ERR 1018	ＷＡＩＴで他のデバイスの入力が指定された
ERR 1019	ＷＡＩＴ　Ｉ／ＯのＩ／Ｏ数が多すぎる
ERR 1020	タスクの指定が不適当
ERR 1098	Ｉ／Ｏ種別エラー
ERR 1099	メモリアロケーションエラー
ERR 1101	リモートＩ／Ｏ設定エラー
ERR 1200	数値を表す正しい文字列ではない
ERR 1201	変数のタイプが等しくない
ERR 1202	データを最後まで読み込んだ後、再びＲＥＡＤを実行した
ERR 1203	ＤＡＴＡ文が見つからない
ERR 1250	親局が見つからない
ERR 1251	親局が最大親局数（４）を越えている
ERR 1252	子局が最大子局数（32）を越えている
ERR 1253	スロット数が最大スロット数を越えている
ERR 1254	スロット機能番号が正しくない
ERR 1255	親局の総スロット数が最大親局総スロット数（64）を越えている
ERR 1256	全体の総スロット数が最大（全体）総スロット数（128）を越えている
ERR 1257	リモートＩ／Ｏメモリアクセスエラー
ERR 1300	ユニットＩＤが正しくない
ERR 1301	ユニット種別が正しくない
ERR 1302	チャンネル番号が正しくない
ERR 1303	平均回数が正しくない
ERR 1304	終端コードタイプが正しくない
ERR 1305	入力フィルタ定数が正しくない
ERR 1306	ユニットタイプが正しくない
ERR 1307	軸番号が正しくない
ERR 1308	ＪＯＢ番号が正しくない
ERR 1309	始動データNo.が正しくない
ERR 1310	データ長が正しくない

エラーコード	エラー内容
ERR 1311	ＩＤアドレスが正しくない
ERR 1312	ＩＤ番号が正しくない
ERR 1313	クリアＩＤ番号が正しくない
ERR 1314	エラーレスポンスを受け取った（ＩＤ／Ｘユニットのみ）
ERR 1315	ハードウェアの制限（定義）によりこのステートメントは実行できません。
ERR 1316	高機能ユニットエラー
ERR 1400	ファイル名の指定が間違っている
ERR 1401	オープンされていないファイルを参照しようとした
ERR 1402	既にオープンされているファイルを再度オープンしようとした
ERR 1403	ファイル中のすべてのデータを読み尽くした後、さらに読み込もうとした
ERR 1404	ファイルがオープンできない
ERR 2000	致命的エラー
ERR 2002	Ｉ／Ｏ　２度読みエラー
ERR 2003	Ｉ／Ｏ　点数異常
ERR 2004	ベーシックスタックオーバー
ERR 2005	ＯＮＥＲＲ処理中のエラー
ERR 2006	ＩＦレベルオーバー
ERR 2007	内部タグコードエラー
ERR 3000	通信シーケンスエラー

付録F　ＡＳＣＩＩ／ＪＩＳキャラクタコード表

上位／下位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		DEL	スペース	0	@	P	'	p			スペース	ー	タ	ミ		
1	SOH	DC1	!	1	A	Q	a	q			。	ア	チ	ム		
2	STX	DC2	"	2	B	R	b	r			「	イ	ツ	メ		
3	ETX	DC3	#	3	C	S	c	s			」	ウ	テ	モ		
4	EOT	DC4	$	4	D	T	d	t			、	エ	ト	ヤ		
5	ENT	NAK	%	5	E	U	e	u			・	オ	ナ	ユ		
6	ACK	SYN	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7	BEL	ETB	'	7	G	W	g	w			ァ	キ	ヌ	ラ		
8	BS	CAN	(	8	H	X	h	x			ィ	ク	ネ	リ		
9	HT	EM	)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A	LF	SUB	*	:	J	Z	j	z			ェ	コ	ハ	レ		
B	VT	ESC	+	;	K	[	k	{			ォ	サ	ヒ	ロ		
C	FF	→	,	<	L	¥	l	¦			ャ	シ	フ	ワ		
D	CR	←	−	=	M	]	m	}			ュ	ス	ヘ	ン		
E	SO	↑	.	>	N	^	n	~			ョ	セ	ホ	″		
F	SI	↓	／	?	O	_	o				ッ	ソ	マ	°		

NAiS the Newest in Automation & Intelligent Systems

●お問い合わせは

松下制御機器株式会社

東北営業所	☎022-223-8163	横浜営業所	☎045-321-1235	京滋営業所	☎075-681-0237
栃木営業所	☎0286-34-0161	神奈川西営業所	☎0462-28-9533	松下グループ営業所	☎ 06-350-3246
関東営業所	☎0273-63-2033	長野営業所	☎0263-28-0790	西部広域営業所	☎ 06-350-3244
埼玉営業所	☎048-665-7991	東部FAセンター	☎03-3454-6190	近畿営業所	☎ 06-350-3241
千葉営業所	☎0471-63-6161	静岡営業所	☎054-261-7711	阪南営業課	☎0724-27-3481
茨城営業課	☎0292-43-8868	浜松営業所	☎053-442-0531	兵庫営業所	☎078-735-8601
東部特機営業所	☎03-3454-6342	名古屋西営業所	☎052-581-8861	岡山営業所	☎086-245-3701
東部広域営業所	☎03-3454-6187	名古屋東営業所	☎052-581-8861	四国営業課	☎0878-41-4473
東部車載営業所	☎03-3454-6065	春日井営業課	☎0568-33-5111	中国営業所	☎082-247-9084
首都圏東営業所	☎03-3454-6188	豊田営業所	☎0565-35-2181	九州営業所	☎092-522-5545
首都圏西営業所	☎03-3454-6186	北陸営業所	☎0762-37-3663	西部FAセンター	☎ 06-354-0531
西東京営業所	☎0425-28-2241	中部FAセンター	☎052-581-8861		

National 松下電工

松下電工株式会社
FAシステム機器事業部

〒571 大阪府門真市門真1048 TEL.(06)908-1131〈大代表〉

●商品改良のため、仕様・外観を変更することがありますのでご了承ください。

●本品のうち、戦略物資(又は役務)に該当するものの輸出にあたっては、外為法に基づく輸出(又は役務取引)許可が必要です。詳細につきましては事業部までご相談ください。

●このマニュアルの記載内容は平成5年7月現在のものです。

FAF-151① 199307-2Y Ho