MPC-684 Programing tutorials

ACCEL
Last update May 25,2005

(MPC-684F)

このチュートリアルは MPC-684 プログラム開発の概要です。各製品・コマンドについての詳細は 「MPC-684 ユーザーズマニュアル」「製品別マニュアル」「総合カタログ」をご覧下さい。最新情報はホームページ http://www.accelmpc.co.jp をご覧下さい。

目次

MPC-684ファミリ

◆　MPC-684 シリーズの製品概要です。

MPC-684	メイン CPU	RS-232 3CH(プログラム 1、ユーザ 2)
MOP-096	出力ボード	96 点トランジスタ出力。4 枚まで。
MOP-048	出力ボード	48 点トランジスタ出力。8 枚まで。
MIP-096	入力ボード	96 点フォトカプラ入力。4 枚まで。
MIP-048	入力ボード	48 点フォトカプラ入力。8 枚まで。
IOP-048	入出力ボード	24 点ﾌｫﾄｶﾌﾟﾗ入力、24 点ﾄﾗﾝｼﾞｽﾀ出力。8 枚まで。
MPG-314	パルスボード	4 軸、最高 4Mpps、S 字加減速。10 枚まで。
MBK-SH	タッチパネル I/F	デジタル社 GP シリーズダイレクトアクセス
MRS-402	RS-232 拡張ボード	RS-232,485 2CH。2 枚まで。
MPS-324	電源ボード	DC5V 3A システム電源供給
MPC-SLNK	入出力ボード	ｻﾝｸｽ SLINK ﾓｼﾞｭｰﾙ搭載 I/O512 点。2 枚まで。
RACK-68K3	ラック	3 スロット
RACK-N6	ラック	6 スロット
RACK-N13	ラック	13 スロット

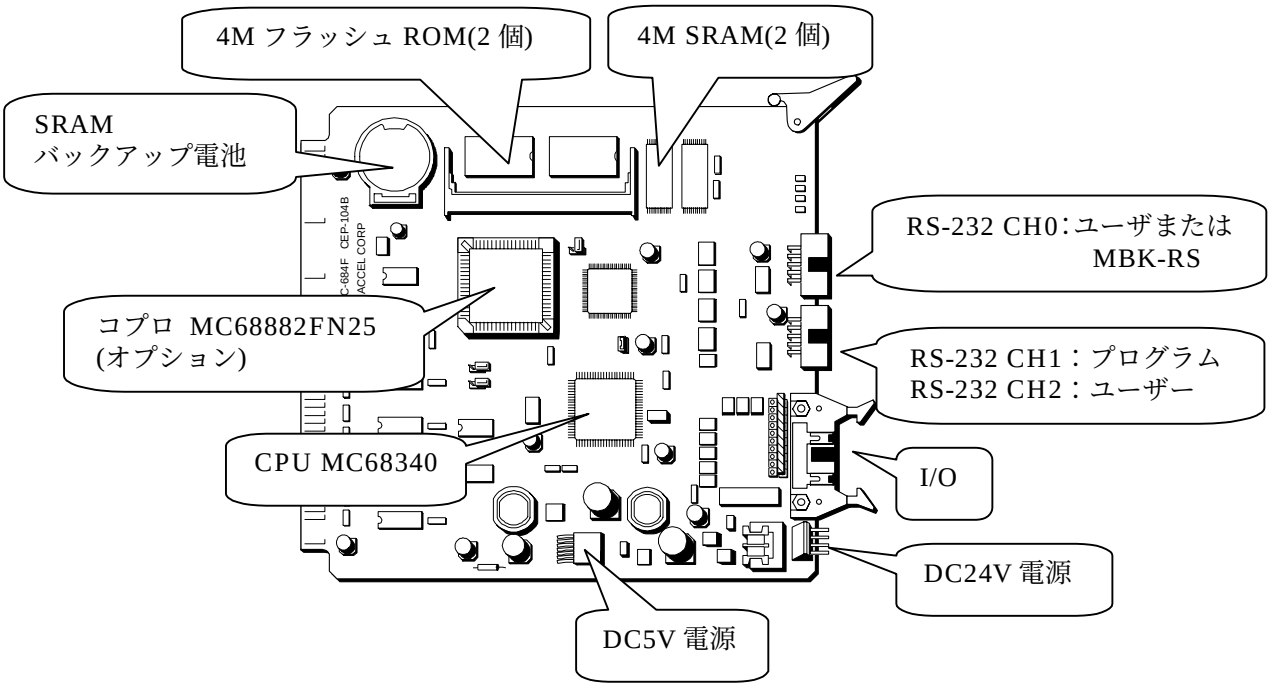
例えば

MPC-684 &　IOP-048 & MPG-314 & RACK-68K3 で

ユーザ RS-232 2CH、入出力 48 点、パルス 4 軸のコントローラとなります。

MPC-684の特徴

- プログラム容量　500Kbyte(約 25000 行)
- ポイントデータ　4 軸×13000 個
- 配列　15 配列 全 10000 個
- 変数　2000 個
- タスク　0(メイン)～31、32(特殊)
- メモリ I/O　-1～-8192 ビット(-1～-1024 バンク)
- RS-232 ポート　プログラム 1ch、ユーザ 2ch(そのうち 1ch はデジタル社タッチパネルとダイレクトアクセス可能)
- RS-232 バッファ　各ユーザチャンネル入出力 256 バイト
- ローカル変数　26 個。
- カレンダ・時計機能　RTC は搭載していませんがタッチパネルを接続するとタッチパネルのデータを利用できます。
- I/O　入力 8 点、出力 4 点(DC24V50mAMax)
- 搭載 DC5V 電源　2A(Max3A その内 300mA を自己消費)
- その他　文字列配列、コプロ搭載(オプション)



プログラム開発環境

ハードウェア

・ パソコン	Windows パソコン。RS-232 が必要 (RS-232 が無い場合は USB-RS をお使いください。)
・ プログラミングケーブル	MPC とパソコンを RS-232 接続 (USB-RS 使用時は不要です。)

アプリケーション

- MPC の開発環境は SetupDisk でインストールされます。
- 最初のインストールは必ず Setupdisk で行ってください。その後のアップデートは実行ファイル(*.EXE)の差し替えで Ok です。
- 標準セットアップフォルダは C:¥Program Files¥ACCEL です。

主なアプリケーション

FTMW　　ターミナルソフト

MPC と接続して編集・デバッグ、パソコンからの読み込み・保存を行います。
MPC の開発には必須のアプリです。
(ファイル名は「FTMW32.EXE」ですが、本文では「FTMW」としています)

SYSLDW　　システムローダ

MPC のバージョンアップ時に用います。フラッシュ ROM 内のシステムデータを書き換えます。

MPCED　　オフラインエディタ

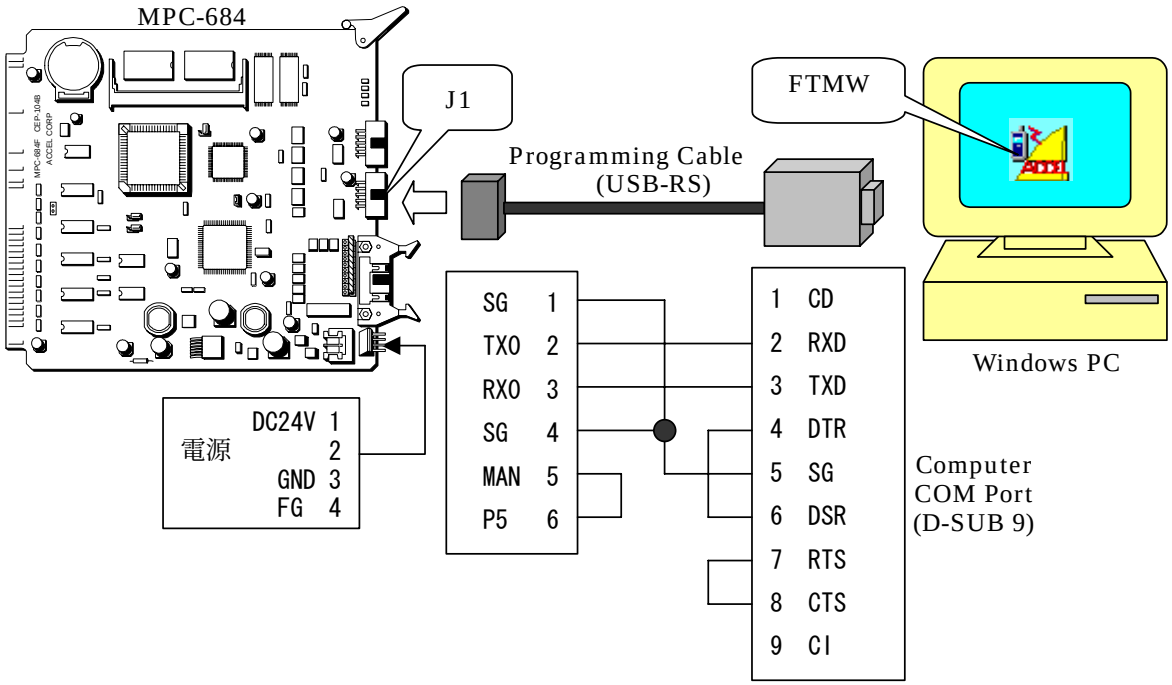
MPC 専用のオフラインエディタです。制御文・ラベル・コメントを色分けします。

MVC　　MPC コマンドビューワ

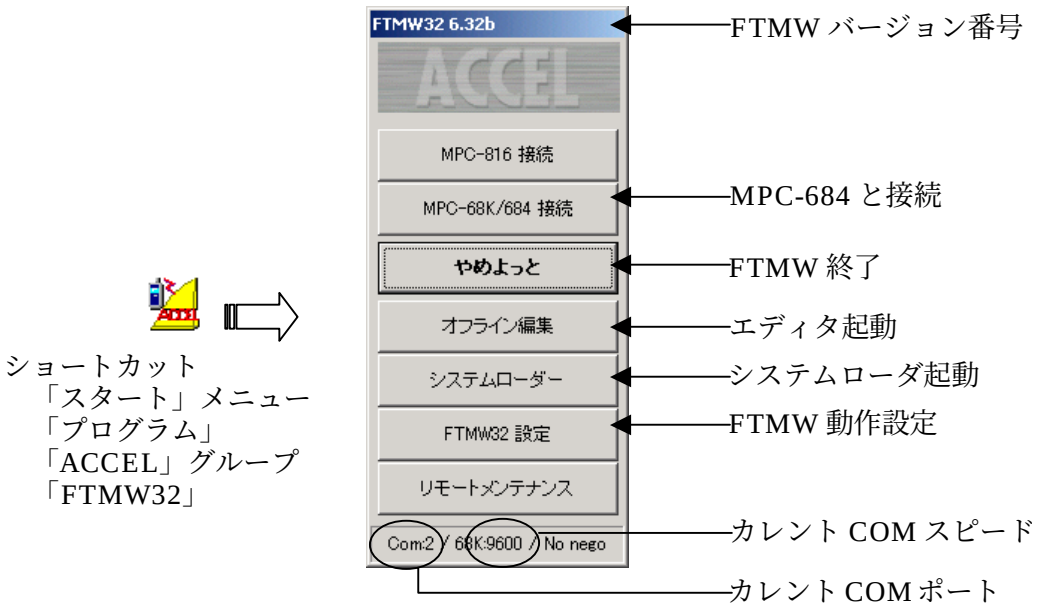
簡易コマンドリファレンスです。FTMW、MPCED からは F1 キーでヘルプになります。コマンドにカーソルを合わせて F1 を押してください。単体でも使えます。
ABC 順・分類別コマンド検索ができます。(本文「分類別コマンドリスト」の項参照)

接続

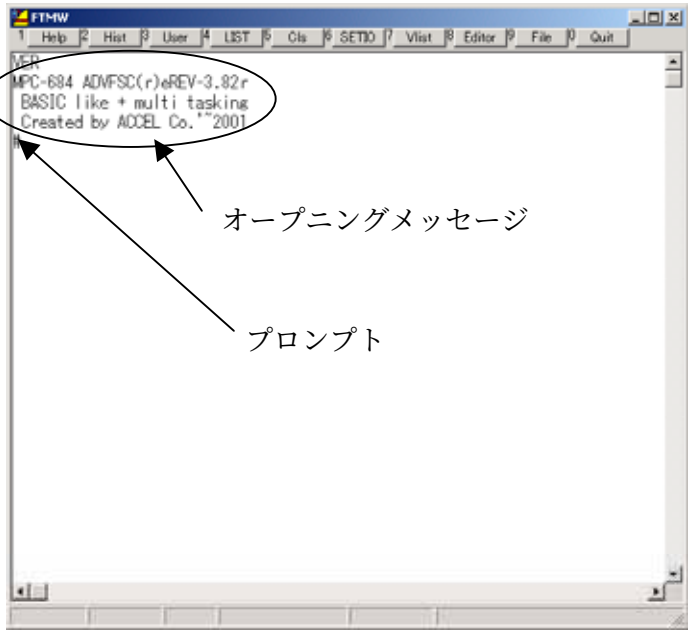
◆ MPC とパソコンをプログラミングケーブルで接続し、MPC の電源を入れます。



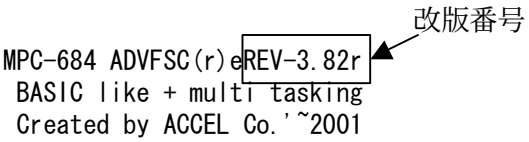
◆ FTMW 起動後、「FTMW32 設定」で COM ポート番号、COM スピードをあわせ、「MPC-684 接続」を押します。



◆　編集画面にオープニングメッセージとプロンプトが表示されれば正常接続。

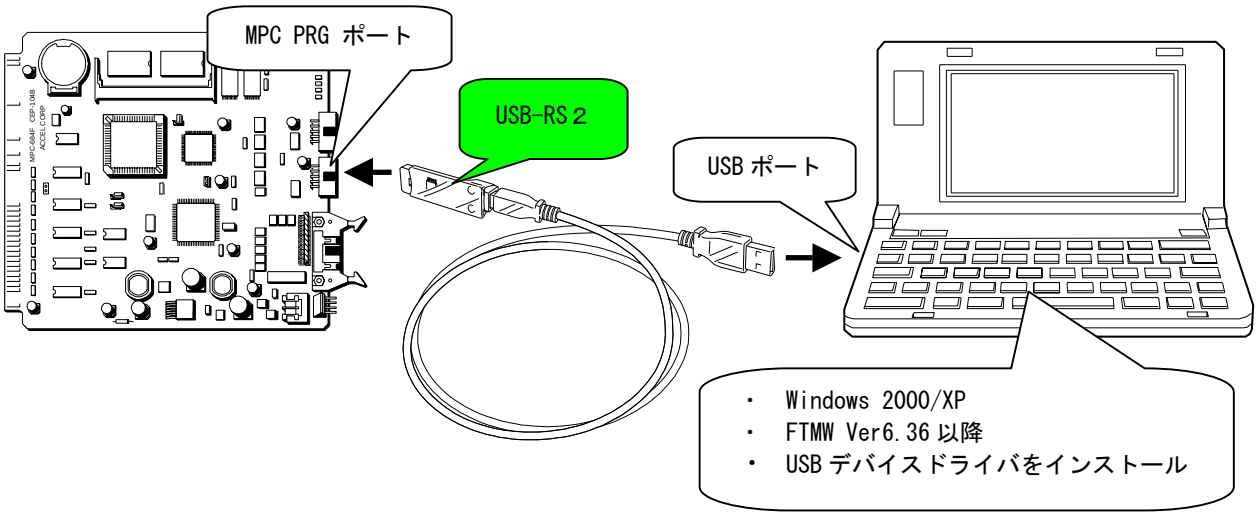


◆　オープニングメッセージの意味

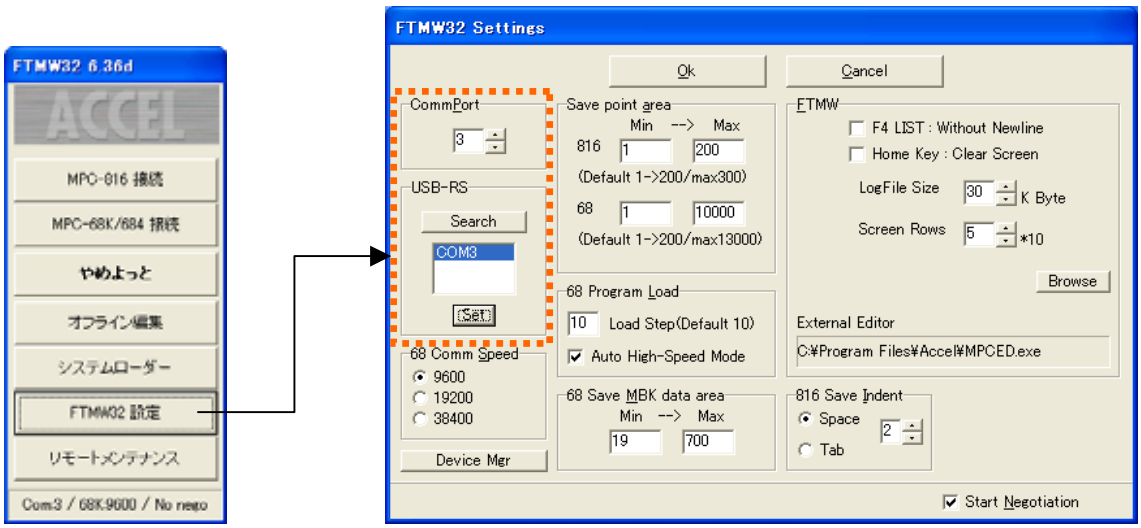


USB-RS (MPC用USB<->シリアルコンバータ)

◆ USB は有るが RS-232C が無い..というパソコンには「USB-RS」をお使いください。



◆ FTMW は USB-RS に引き当てられた COM ポートをボタン１つで検出します。



※USB-RS についての詳細は「USB-RS 製品別マニュアル」をご覧下さい。

コマンド入力

- プロンプトの後にコマンドを入力し Enter すると、その場で実行します。これをダイレクトコマンド実行といいます。
- 殆どのコマンドはダイレクトに実行することもプログラムに記述することもできますが、メンテナンス・編集関係などでダイレクトコマンドでしか使えないものや制御文などでプログラムにしか書けないものもあります。

両方で使える	ダイレクトのみ	プログラムのみ
ON 0	LIST	GOTO
OFF 0	MPCINIT	GOSUB
PRINT A	ERASE	IF ~
MOVE	RUN	FOR ~ NEXT
など	など	など

```
#ON 0<Enter>              /* ダイレクト実行　10 ON 0<Enter> とすればプログラム
#GOTO 100<Enter>          /* ダイレクト実行しても何もおこらない
#10 MPCINIT<Enter>        /* このコマンドをプログラムするとプログラムが消える！
```

※本文中の<Enter>は、パソコンのキーボードの Enter キー押下を表します。

I/Oチェック

コマンドによるチェック

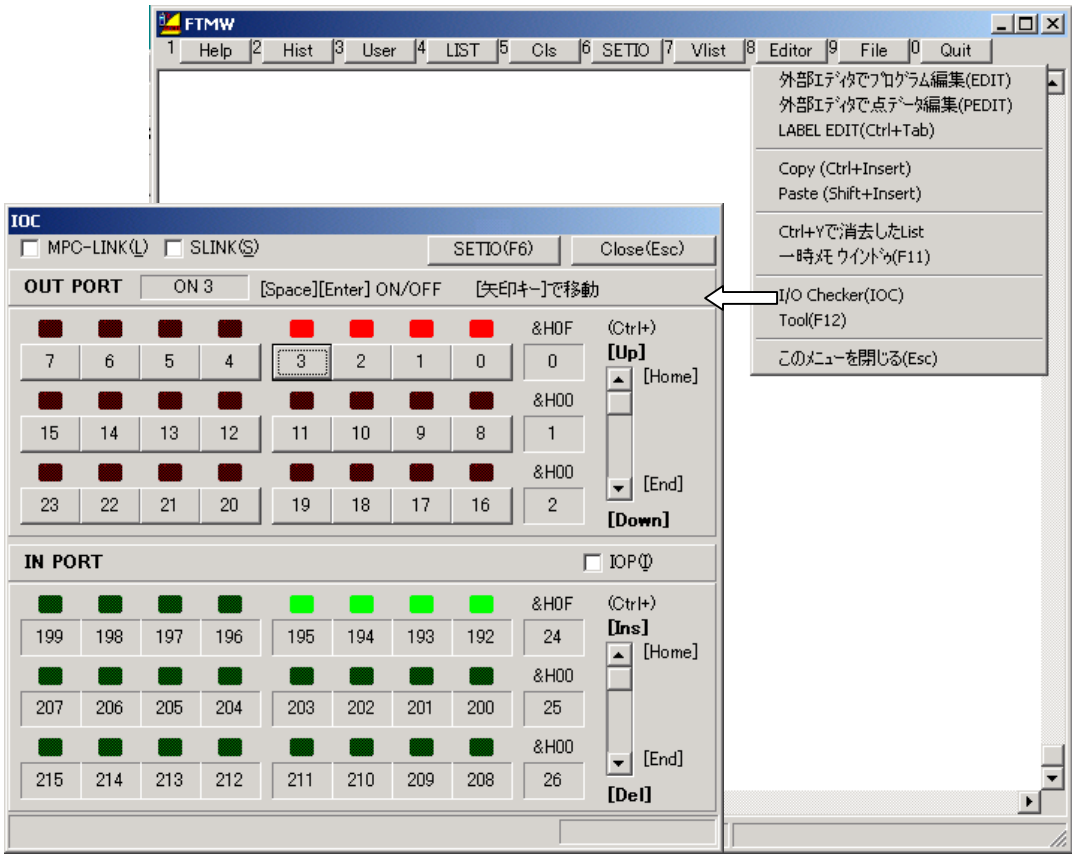
・ 基本はコマンドによる 1 ビット単位のチェックです。

```
#ON 0                    /* 出力 0 をオン
#OFF 0                   /* 出力 0 をオフ
#PR SW(0)                /* 入力 0 の状態　(PR は PRINT の省略型です)
 1                       /* 0=オフ、1=オン
#PG &H400
#PRX HPT(0)              /* MPG-314 原点センサーポート確認
 0005                    /* SX1(J4 11 番ピン)、SY1(J4 13 番ピン)がオン
```

I/Oチェッカによるチェック

・ まとめて見るなら [F8] I/O Checker で I/O チェッカを起動。

・ #IOC<Enter>　としても I/O チェッカが起動します。



・ MPG-314 の入力ポートは INCHK_314 コマンドで確認できます。

```
#INCHK_314                    /* 連続スキャン開始
X_S1  ON   Y_S1  ON
Z_S1  __   U_S1  __
X_S2  __   Y_S2  __
Z_S2  __   U_S2  __
XIN2  __   YIN2  __
ZIN2  __   UIN2  __
XIN3  __   YIN3  __
ZIN3  __   UIN3  __
X-INP __   Y-INP __
Z-INP __   U-INP __
X-ALM __   Y-ALM __
Z-ALM __   U-ALM __
XLMT+ __   XLMT- __
YLMT+ __   YLMT- __
ZLMT+ __   ZLMT- __
ULMT+ __   ULMT- __
#                             /* どれかのキーでスキャン停止
```

プログラム

◆ 文番号を付けて記述するとプログラムになります。Enter キーを押して確定(MPC へ送信)です。エラーメッセージが出たら文法に誤りが無いか確認して再入力して下さい。

```
10 'サンプルプログラム<Enter>          /* コメント文
20 DO<Enter>                          /* 制御文
30 FOR I=0 TO 48<Enter>               /* 繰り返し
40 ON I : TIME 100<Enter>             /* マルチステートメント
50 NEXT I<Enter>
60 LOOP<Enter>
70 pri I<Enter>
·····この命令は登録されてません m(__)m   ┐ エラーなら間違いを訂正して再入力
70 PRINT I<Enter>                     ←┘
```

マルチステートメント

・ ：(コロン)で区切って、1 行に複数のコマンドを書くことができます。

```
1000      WAIT SW(0)==0 : ON 0 : TIME 100 : OFF 0 : TIME 500
```

コメント

・ '(シングルコーテーション)の後ろにコメントが書けます。
・ プログラムの後ろにも書くこともできます。その場合は自動的にマルチステートメント化されます。

```
#40 ON 0 'COMMENT<Enter>
LIST 40 1<Enter>
40        ON 0 :                      'COMMENT    ←┘  ﾏﾙﾁｽﾃｰﾄﾒﾝﾄ
```

・ コメントには日本語(全角、半角文字)も使えます。
・ コメント文のコロン以降はプログラムとして実行されます。注意してください。

```
10      'COMMENT  : ON 0       /* ON 0 は実行されます
```

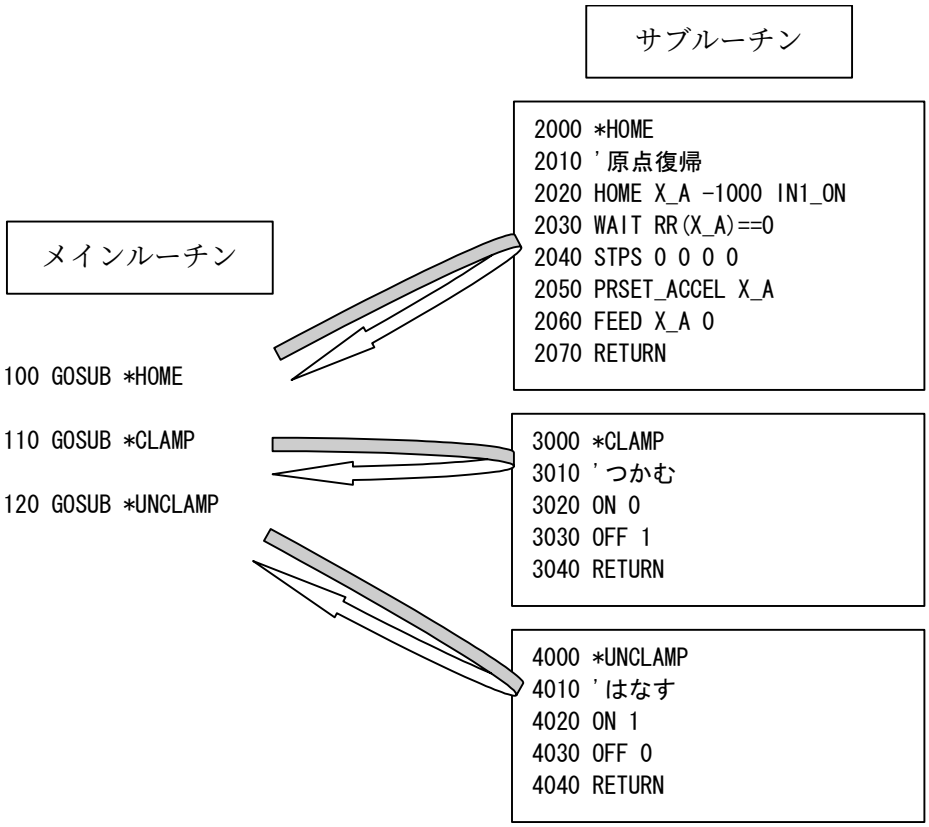
ラベル

・ 先頭に＊が付いたものはラベルです。スペースは使えません。

```
10      *MAIN                         /* 飛び先のラベル名は重複禁止
20        IF SW(0)==0 THEN : GOTO *MAIN : END_IF
```

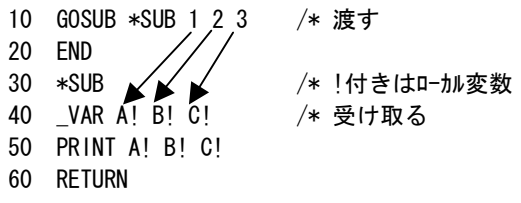
サブルーチン

- GOSUB でジャンプして RETURN で戻る構文をサブルーチンといいます。
- 仕事単位でサブルーチン化して、それをメインルーチンから呼ぶようにするとプログラムが読み易くなります。



サブルーチンの引き数、戻り値

- GOSUB にはサブルーチンへの引き数を与えることができます。サブルーチン内では _VAR コマンドで値を取得します。
- RETURN の後ろにサブルーチンからの戻り値を与えられます。戻り値の取得は _RET_VAL です。
- ローカル変数と組合わせるとタスク間でのサブルーチンの共有が可能となります。



```
10  GOSUB *SUB 1 2 3     /* 渡す
20  END
30  *SUB                 /* !付きはﾛｰｶﾙ変数
40  _VAR A! B! C!        /* 受け取る
50  PRINT A! B! C!
60  RETURN
```

```
10 GOSUB *SUB
20 _RET_VAL A           /* 戻り値の受け取り
30 PRINT A
40 END
50 *SUB
60  C!=123
70 RETURN C!            /* C!=戻り値
```

プログラム編集

◆　プログラムを編集する際、頻繁に行う操作を解説します。

LISTの表示

・ 最も頻繁に使用するのが LIST コマンドです。

■書式

LIST [n,m]

n:開始文番号或は開始ラベル

m:表示行数

・ LIST だけでも実行できます(下記①)。その場合は前回の続が表示されます。

・ 第 1 パラメータに表示開始位置を文番号またはラベルで指定できます(下記②③)。

・ 第 2 パラメータに表示する行数を指定できます(下記④)。以後この行数は保持されます。

・ LIST 0 とすると最初から表示します(下記⑤)。

①
```
#LIST
10          IF SW(192)==0 THEN
20            GOTO *MANU
30            ELSE
(以下略)
```

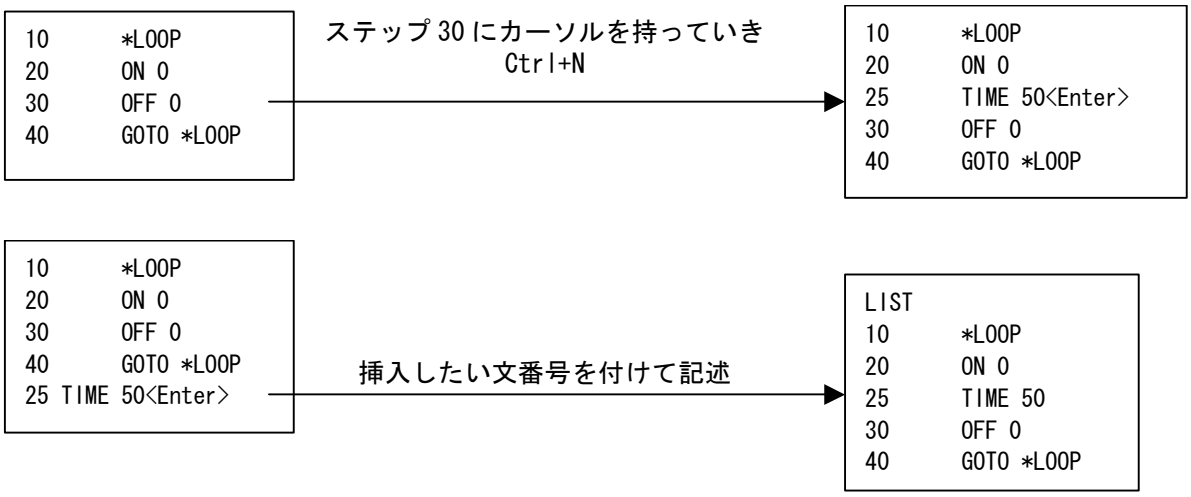
②
```
#LIST 40
40          GOTO *AUTO
50          END_IF
100      *MANU
(以下略)
```

③
```
#LIST *MANU
100      *MANU
110      'MANUAL JOB
200      *AUTO
(以下略)
```

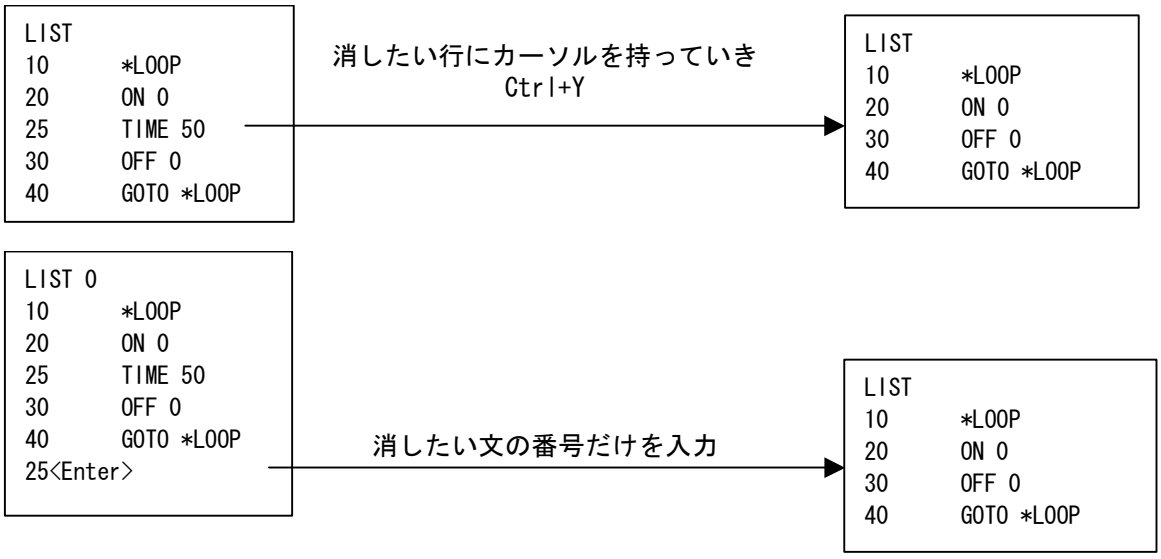
④
```
#LIST *MANU 2
100      *MANU
110      'MANUAL JOB
```

⑤
```
#LIST 0 20
10          IF SW(192)==0 THEN
20            GOTO *MANU
30            ELSE
40            GOTO *AUTO
50          END_IF
100      *MANU
110      'MANUAL JOB
200      *AUTO
210      'AUTO JOB
```

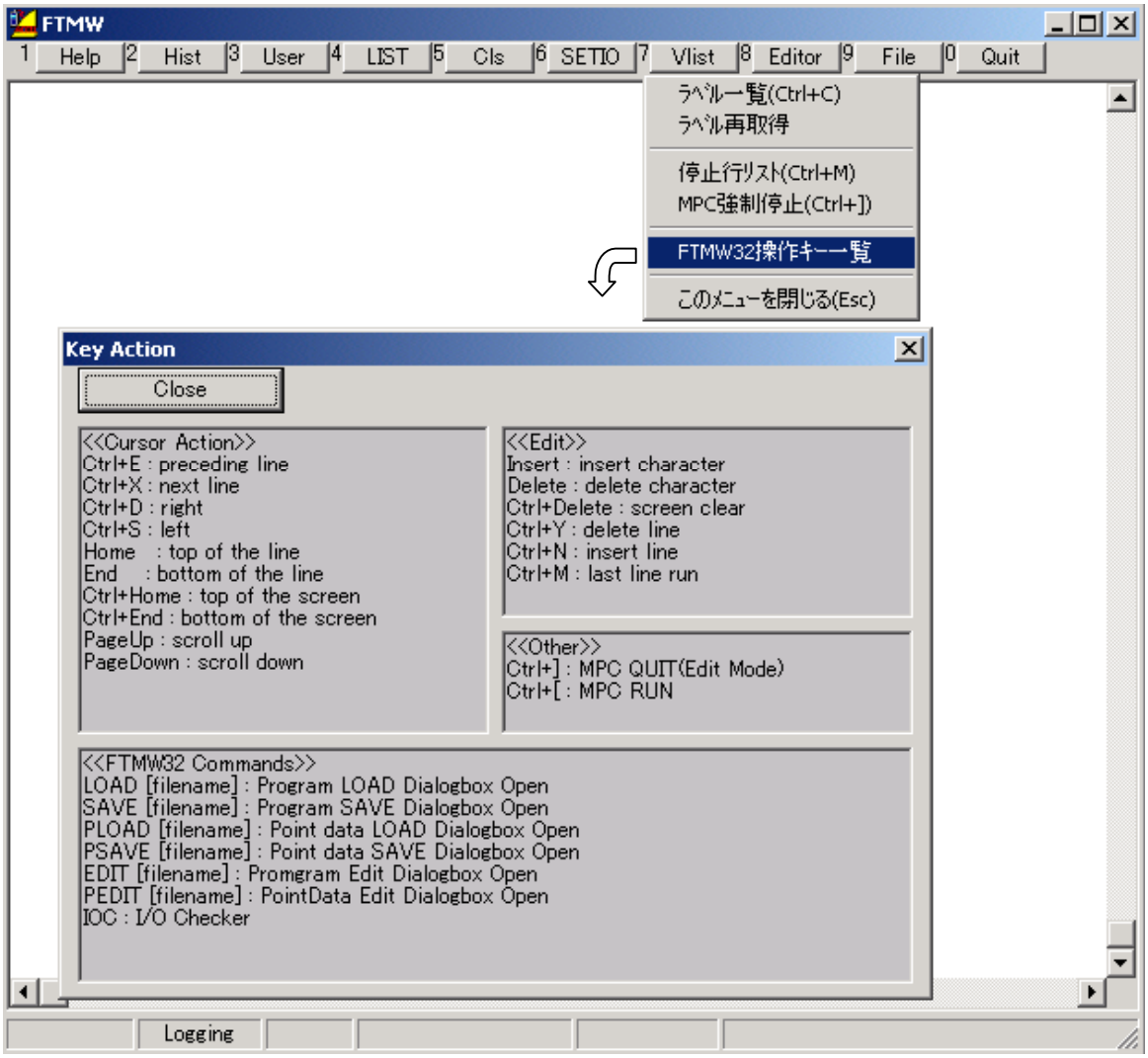
行の挿入


10 *LOOP
20 ON 0
30 OFF 0
40 GOTO *LOOP
ステップ 30 にカーソルを持っていき
Ctrl+N
10 *LOOP
20 ON 0
25 TIME 50<Enter>
30 OFF 0
40 GOTO *LOOP
10 *LOOP
20 ON 0
30 OFF 0
40 GOTO *LOOP
25 TIME 50<Enter>
挿入したい文番号を付けて記述
LIST
10 *LOOP
20 ON 0
25 TIME 50
30 OFF 0
40 GOTO *LOOP

行の削除


LIST
10 *LOOP
20 ON 0
25 TIME 50
30 OFF 0
40 GOTO *LOOP
消したい行にカーソルを持っていき
Ctrl+Y
LIST
10 *LOOP
20 ON 0
30 OFF 0
40 GOTO *LOOP
LIST 0
10 *LOOP
20 ON 0
25 TIME 50
30 OFF 0
40 GOTO *LOOP
25<Enter>
消したい文の番号だけを入力
LIST
10 *LOOP
20 ON 0
30 OFF 0
40 GOTO *LOOP

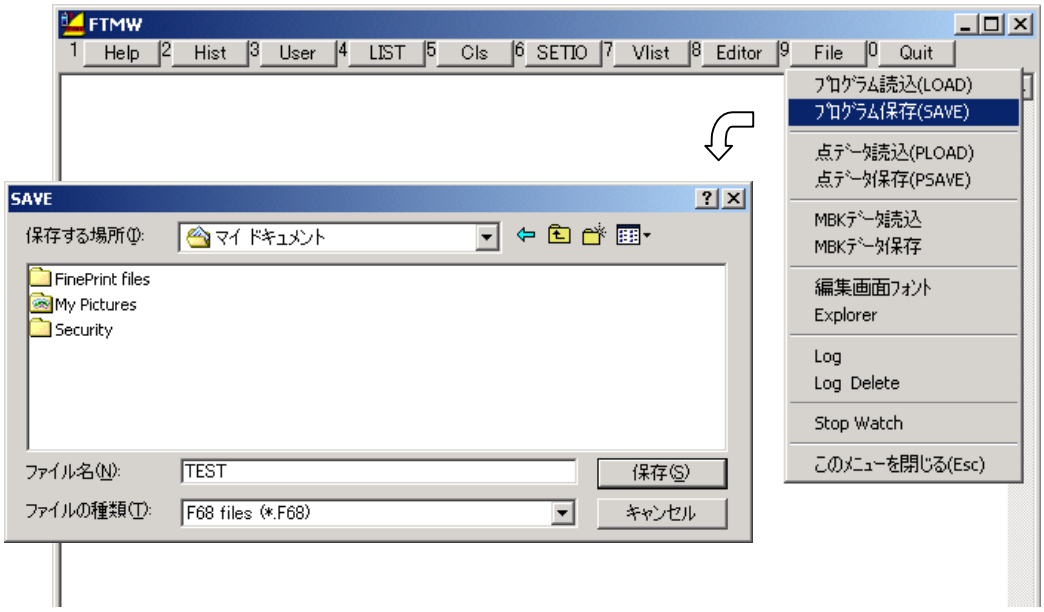
その他のキー操作


FTMW
1 Help
2 Hist
3 User
4 LIST
5 Cls
6 SETIO
7 Vlist
8 Editor
9 File
0 Quit
ﾗﾍﾞﾙ一覧(Ctrl+C)
ﾗﾍﾞﾙ再取得
停止行ﾘｽﾄ(Ctrl+M)
MPC強制停止(Ctrl+])
FTMW32操作ｷｰ一覧
このﾒﾆｭｰを閉じる(Esc)
Key Action
Close
<<Cursor Action>>
Ctrl+E : preceding line
Ctrl+X : next line
Ctrl+D : right
Ctrl+S : left
Home : top of the line
End : bottom of the line
Ctrl+Home : top of the screen
Ctrl+End : bottom of the screen
PageUp : scroll up
PageDown : scroll down
<<Edit>>
Insert : insert character
Delete : delete character
Ctrl+Delete : screen clear
Ctrl+Y : delete line
Ctrl+N : insert line
Ctrl+M : last line run
<<Other>>
Ctrl+] : MPC QUIT(Edit Mode)
Ctrl+[: MPC RUN
<<FTMW32 Commands>>
LOAD [filename] : Program LOAD Dialogbox Open
SAVE [filename] : Program SAVE Dialogbox Open
PLOAD [filename] : Point data LOAD Dialogbox Open
PSAVE [filename] : Point data SAVE Dialogbox Open
EDIT [filename] : Promgram Edit Dialogbox Open
PEDIT [filename] : PointData Edit Dialogbox Open
IOC : I/O Checker
Logging

プログラムの保存・読み込み

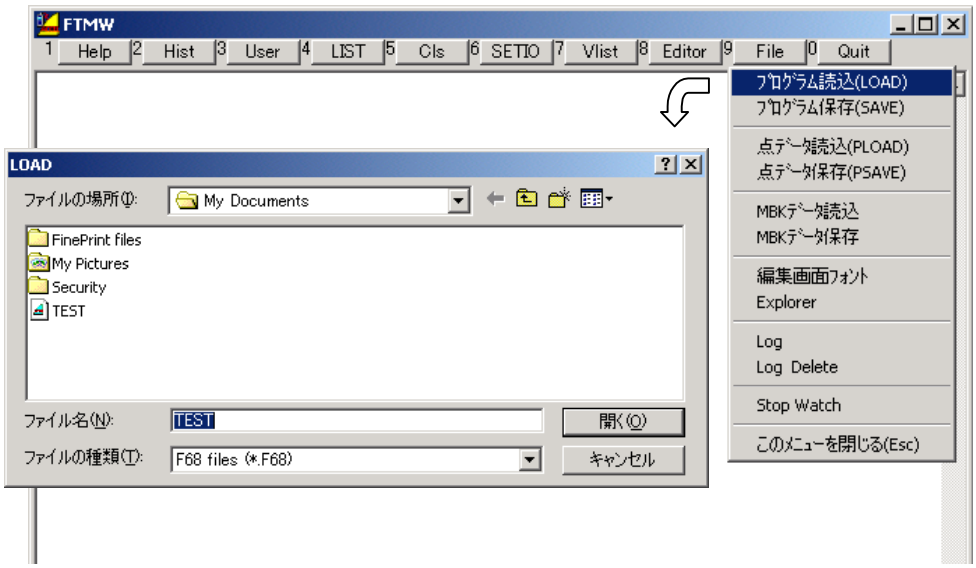
保存

- F9 「プログラム保存」でパソコンに保存します。
- 拡張子‘ F68 ’で保存されます。
- 保存したプログラムには文番号はありません。



読み込み

- F9 「プログラム読込」でパソコンからプログラムを読み込みます。
- 文番号は初期状態で 10 間隔です。60000 ステップを越えると自動的に 5 で RENUM します。



オフライン作成

- FTMW は MPC 内のデータを直接操作するもので、接続していないと使うことはできませんが、プログラムはオフラインで作成することができます。
- エディタをご用意ください。MPCED は MPC 専用ですが、汎用のエディタやワープロでも使えます。その場合、プログラムはテキストデータ、拡張子を”F68”として保存して下さい。
- プログラムのエラーは MPC に読み込み、実行するまではわかりません。

印刷

- FTMW には印刷機能はありません。パソコンに保存したファイルを MPCED などのエディタやワープロソフトで印刷してください。

I/O制御

- I/O はビット単位、またはバイト単位(バンク)で扱うことができます。
- ビットでは 0 か 1 の状態しかありませんが、バンクでは 0~255(&HFF)のデータを持ちます。主に、ビット操作はソレノイドバルブ・リレー・スイッチなどの機器の単体制御、バイト操作は DSW 読み込みやシーケンサなどの外部機器とのデータ授受に利用されます。
- バンクとはバイト単位の表現です。入出力 0~7 がバンク 0、8~15 がバンク 1、16～23 がバンク 2 というようになります

ビット処理

```
ON 0                                    /* 出力 0 をオン
OFF 1                                   /* 出力 1 をオフ
WAIT SW(192)==1                         /* 入力 192 が 1(オン)になるまで待つ
IF SW(1)==1 THEN : GOTO *LABEL : END_IF /* 条件分岐
ON -1                                   /* マイナス数字はメモリ I/O
WAIT SW(-2)==0                          /*       〃
```

バイト処理

```
OUT 170 0                                    /* 10 進数 170 をバンク 0 に出力。
A=IN(0)                                      /* 入力バンク 0 の値を変数 A に入れます。
OUT A B                                      /* A の値をバンク B にパラレル出力
WAIT IN(1)==255                              /* 条件待ち
IF IN(2)==&H0F THEN : GOTO *LABEL : END_IF   /* 条件分岐
OUT &HFF -1                                  /* マイナス数字はメモリ I/O
WAIT IN(-1)==&HAA                            /*          〃
```

変数・配列変数・メモリI/O・文字列配列

- ◆ MPC-684 には多くの変数・配列変数・メモリ I/O・文字列配列があります。
- ◆ 変数・配列変数は 4 バイト長整数(約±21 億)です。
- ◆ 変数・配列変数は主に演算結果・カウント値などのデータ格納やタスク間のインターロックに、メモリ I/O はタスク間のインターロックに利用されます。
- ◆ 殆どのコマンド・関数には定数・変数・配列変数を与えることができます。
- ◆ メモリ I/O は通常の I/O と同様に ON、OFF、OUT、SW、IN でビット操作・バイト操作ができます。
- ◆ 現在の使用状況は VLIST コマンドで一覧表示します。

変数

・ MPC-684 の変数は自動変数です。パラメータにコマンド・関数・定数・予約文字列以外の文字列を与えると変数と解釈されます。C コンパイラのように変数名、型などの宣言は要りませんが、初期化はプログラムで行います。

```
10          SOL1=0              /* SOL1 は変数
20        ON SOL1               /* 変数を使って出力
```

・ CONST で定数化します。

```
10        CONST SOL1 0          /* 変数を定数化
20        ON SOL1
```

・ 一度定数化すると値は変更できません。

```
10        CONST SOL1 0
20        ON SOL1
30          SOL1=1              /* 値を変更しようとしている
#RUN
#30     ・・・・・定数は変更できません
```

ローカル変数

・ ローカル変数とはタスク単位の変数です。同じ名前でもタスク毎に違うメモリエリアに割り当てられるので、１つのサブルーチンを複数のタスクで共有することができます。

```
A!=B!+C!                        /* ‘!’を付ければローカル変数
```

配列変数

・ DIM 宣言で配列変数を確保します。

```
10        DIM ARRY(100)          /* ARRY(0)～ARRY(99)の 100 個を確保
20        FOR I=0 TO 99
30            ARRY(I)=I
40        NEXT I
```

・ DIM 配列は 2 次元も可能です。

```
10        DIM ARRY(2,3)
```

・ ポイントデータも一種の配列変数です。
ポイントデータは FTMW でパソコンに保存・読み込みできます。現在の点データの内容は PLS コマンドで一覧表示します。

```
#NEWP                            /* 全点データクリア
#X(1)=999
#Y(1)=998
#U(1)=997
#Z(1)=996
#PLS
点 1      X= 999 Y= 998 U= 997 Z= 996
(中略)
点 21     X= 0 Y= 0 U= 0 Z= 0
点 22                            /* ”Q”ｷｰで表示終了。その他のキーで継続
```

文字列配列

・ AR$という文字列配列があります。DIM_AR$で確保します。

```
10        DIM_AR$ 35 4000        /* 半角 35 文字 AR$(0)～AR$(3999)の 4000 個を確保
20          AR$(3000)="ACCEL"
730        PRINT AR$(3000)
#run
ACCEL
```

・ 注意：文字列配列は点データエリアを使っています。

```
#DIM_AR$                                  /* パラメータ無しで配列エリアの確認
Length=35 Count =4000 P(MAX)=4249         /* 35 文字*4000 個、使用可能点データは 4249 個
```

メモリI/O

・ 負の I/O 番号を与えるとメモリ I/O として扱います。

```
ON - 1                           /* ビット-1 をオン
WAIT SW(-1)==1                   /* ビット-1 がオンになるまで待つ
OUT 255 - 2                      /* バンク-2(ビットでは-9～-16)に 255 を出力
```

演算

+	加算	A=B+C
-	減算	A=B-C
*	乗算	A=B*C
/	除算	A=B/C
%	剰余	A=B%C
&	論理積(AND)	A=B&C
|	論理和(OR)	A=B|C
¥	排他的論理和(XOR)	A=B¥C

◆ 全て４バイト長整数です。小数点以下切り捨てです。

```
10          A=5
20          B=A/2
30        PRINT B
RUN
2
```

◆ 一般的な算術演算に従っています。

```
10          A=2 :   B=3 :   C=4
20          D=A*(B+C)
30        PRINT D
RUN
14
```

◆ オプションのコプロを搭載すると、三角関数、浮動小数点演算ができます。

パルス発生

◆ MPG-314 を使ったパルス発生の例です。

◆ XY 直行座標型ロボットをイメージして解説します。

初期設定

・ MPG-314 を搭載しただけでは正常にパルスは出ません。初期設定が必要です。

・ 最初に PG コマンドでタスクに MPG を引き当てます。次に ACCEL 等で初期設定をします。

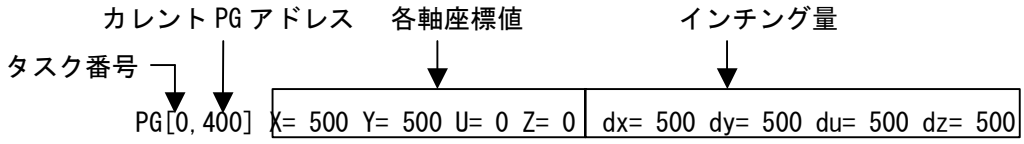
・ ダイレクトコマンドでも可能ですが、最終的にプログラムに反映させてください。

・ 設定例

```
PG &H400                          /* MPG-314 ボード選択。MPG-314 は DSW でアドレス設定。
ACCEL ALL_A 5000                  /* 最高速・加減速設定。
FEED ALL_A 0                      /* 使用速度設定。
INSET_314 ALL_A ALM_ON|INP_OFF /* 入力機能設定。ｱﾗｰﾑは ON で有効、INPOS は OFF で有効とする。
STPS 0 0 0 0                      /* 現在点設定。
```

ティーチモードでの動作確認

・ パルス出力の最も簡単な確認方法はティーチモードです。FTMW 画面で T<Enter>でティーチングモードに入ります



・ インチング量(1 回のパルス出力数)は 0~3 のキーで切り替えます。この値は SET コマンドで変更できます。

初期値 0：10 パルス／ 1：50 パルス ／ 2：100 パルス ／ 3：500 パルス

・ X,x,Y,y,U,u,Z,z キーで各軸が動作します。

・ P キーでポイント番号入力。教示する点番号を入力して下さい。

・ Tab、＋、－キーで PG が切り替わります。

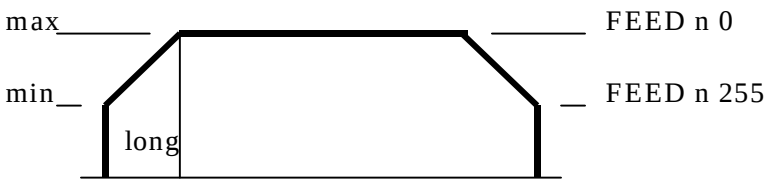
・ Q キーでティーチングモードから抜けます。

最高速・加減速の設定

- 最高速と加減速の設定は ACCEL コマンド、パルス発生時の最高速度の選択は FEED で行います。
- 通常、ACCEL はプログラム先頭で１回だけ初期設定し、稼動中のスピード変更は FEED で行います。
- 書式

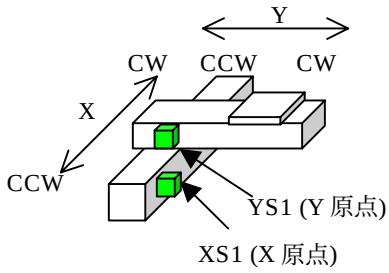
```
ACCEL [n] [s] max [long min]
  n:軸選択予約定数 X_A～Z_A もしくは論理和
  s:S 字加減速ﾊﾟﾗﾒｰﾀ -1～-100
  max:最大ｽﾋﾟｰﾄﾞ 1～4Mpps(円弧 2M)
  long:加減速領域ﾊﾟﾙｽ数
  min:立ち上がりﾊﾟﾙｽ数

FEED n m
  n:軸指定予約定数 X_A～Z_A
  m:速度指定 0～255
```

- ACCEL [n] [s] max [long min] と FEED n m の関係



原点復帰

・ HOME コマンドで軸、方向、スピード、原点センサパターンを指定して原点復帰を行います。

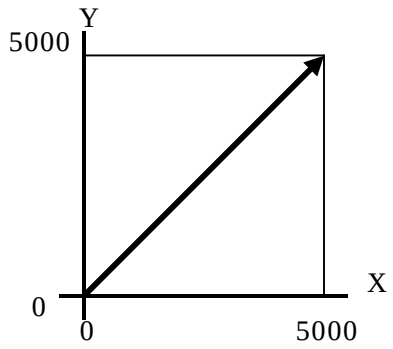


XS1 は MPG-314 J4 11 番ピン
YS1 は MPG-314 J4 13 番ピン
へ接続

```
10    PG &H400
20    ACCEL ALL_A 5000
30    FEED ALL_A 0
40    GOSUB *HOME
50    END
60  *HOME
70    RMVL 500 500 0 0                        /* CW 方向に退避移動
80    WAIT RR(ALL_A)==0                       /* パルス終了待ち
90    HOME X_A -500 IN0_ON Y_A -500 IN0_ON    /* 各軸原点ｾﾝｻがｵﾝするまで CCW に原点復帰
100   WAIT RR(ALL_A)==0
110   STPS 0 0 0 0                            /* 現在の座標値をクリア
120   PRSET_ACCEL ALL_A                       /* ACCEL プリセット
130   FEED ALL_A 0                            /* FEED 再設定
140   RETURN
```

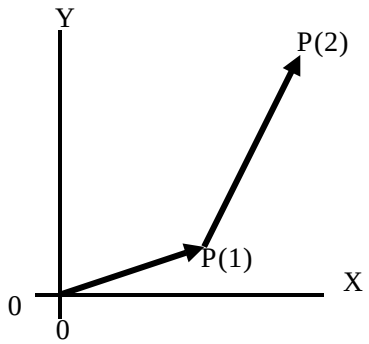
絶対座標移動

・ ①定数、変数で座標を指定して移動します。MOVL は直線補間します。



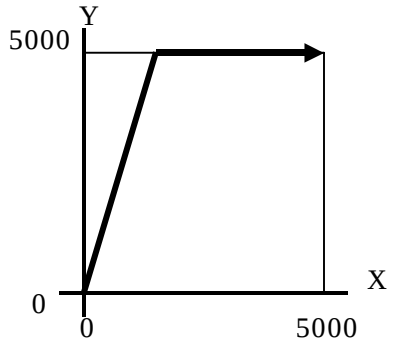

```
10  PG &H400
20  ACCEL 5000       /* 全軸加減速設定
30  FEED ALL_A 128   /* 全軸スピード設定
40  CLRPOS
50  MOVL 5000 5000 VOID VOID
60  WAIT RR(ALL_A)==0
```

・ ②ティーチングした点を指定して移動します。点はティーチングモードやプログラムで設定できます。点番号を変数で指定することもできます。




```
10  PG &H400
20  ACCEL 5000
30  FEED ALL_A 128
40  CLRPOS
50  MOVL P(1)
60  WAIT RR(ALL_A)==0
70  MOVL P(2)
80  WAIT RR(ALL_A)==0
```

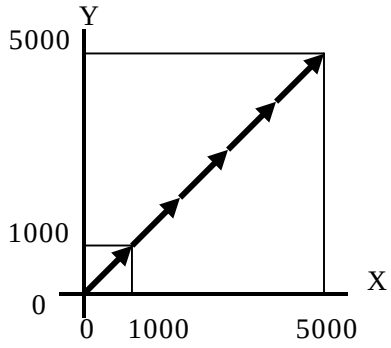
・ ③ 到達点は①と同じですが MOVS は直線補間しません。ステップモータを使ったロボットの振動防止、ステップ・サーボを組み合わせたロボットで軸毎に異なるスピードを設定したい場合などに応用できます。




```
10  PG &H400
20  ACCEL X_A 5000    /* X 軸加減速設定
30  ACCEL Y_A 10000   /* Y 軸加減速設定
40  FEED X_A 128      /* X 軸スピード設定
50  FEED Y_A 0        /* Y 軸スピード設定
60  CLRPOS
70  MOVS 5000 5000 VOID VOID
80  WAIT RR(ALL_A)==0
```

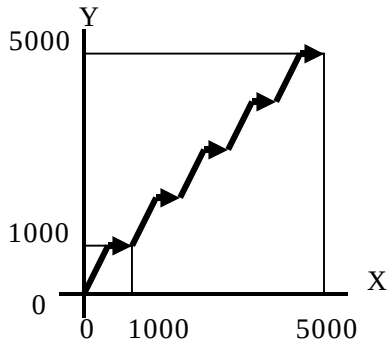
相対座標移動

・ ①定数、変数で現在位置からの移動距離を指定して移動します。RMVL は直線補間します。




```
10  PG &H400
20  ACCEL 5000
30  FEED ALL_A 0
40  CLRPOS
50  FOR I=1 TO 5
60    RMVL 1000 1000 0 0
70    WAIT RR(ALL_A)==0
80  NEXT I
```

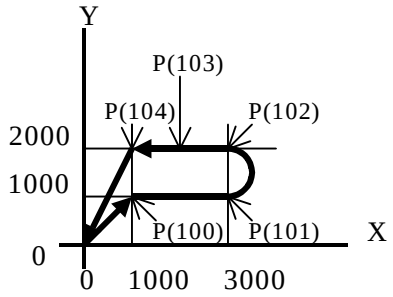
・ ② 到達点は①と同じですが RMVS は直線補間しません。




```
10  PG &H400
20  ACCEL X_A 5000
30  ACCEL Y_A 10000
40  FEED X_A 128
50  FEED Y_A 0
60  CLRPOS
70  FOR I=1 TO 5
80    RMVS 1000 1000 0 0
90    WAIT RR(ALL_A)==0
10  NEXT I
```

連続補間

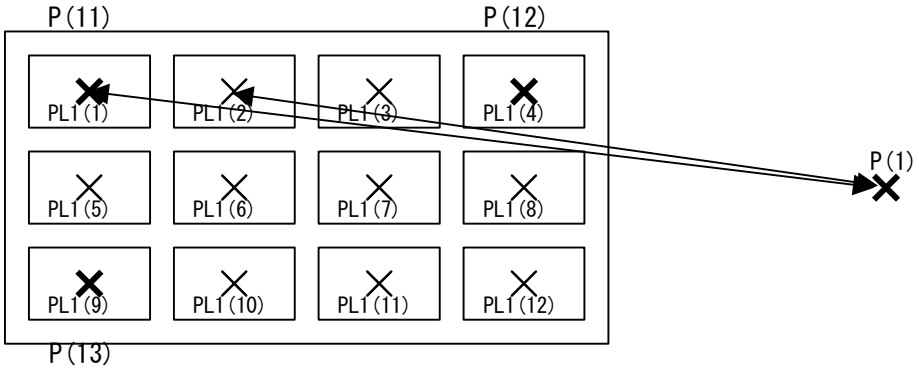
- MOVT コマンドを使った連続補間の例です。
- 10～50 行で点を設定しています。この場合 P(100)が基準点となり P(101)～P(104)はローカル座標(基準点からの相対座標)になります。
- MOVT コマンドは１つずつ先読みされるので、途中で I/O 制御を行う場合は、作業位置とプログラム実行ステップのズレに注意して下さい。下記では P(101)から P(102)まで出力 0 をオンします。




```
10   SETP 100 0 0 0 0          /* 基準点
20   SETP 101 2000 0 0 0
30   SETP 102 2000 1000 2000 500
40   SETP 103 1000 1000 0 0
50   SETP 104 0 1000 0 0
60   PG &H400
70   ACCEL 5000
80   STPS 0 0 0 0
90   MOVL 1000 1000 VOID VOID  /* 基準点へ移動
100  WAIT RR(ALL_A)==0
110  HOUT X_A;DS_DACL          /* 加減速無効
120  MOVT X_A|Y_A P(101)
130  MOVT X_A|Y_A P(102) CCW
140  MOVT X_A|Y_A P(103)       /* P(102)で OFF 0 するためのダミー点
150  ON 0                      /* P(101)～P(102)間オン 0
160  MOVT X_A|Y_A P(104)
170  OFF 0
180  HOUT X_A;EN_DACL          /* 加減速有効
190  WAIT RR(ALL_A)==0
200  MOVL 0 0 VOID VOID
```

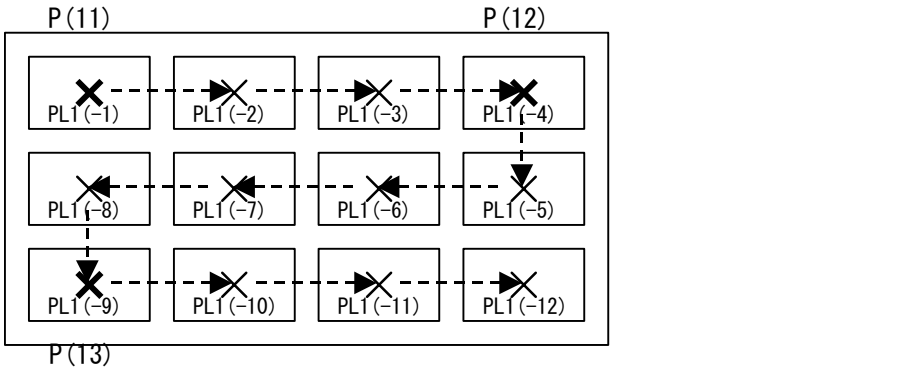
パレタイズ

・ パレット間の移動に利用します。ティーチングした 3 点とマトリックス数からパレット上の作業点 PL1(N)を算出します。




```
10  PG &H400
20  ACCEL ALL_A 5000
30  PALET1 P(11) P(12) P(13) 4 3          /* パレット宣言
35  MOVL 0 0 VOID VOID
37  WAIT RR(ALL_A)==0
40  FOR N=1 TO 12
50    MOVL P(1)
60    WAIT RR(ALL_A)==0
70    MOVL PL1(N)                         /* パレットへ移動
80    WAIT RR(ALL_A)==0
90  NEXT N
```

・ N が負の時、ZIGZAG モードになります。列間の移動距離が短くなります。



途中停止

・ ①ソフトによる途中停止。移動開始後に入力を監視してスイッチが入ったら STOP コマンドを発行しています。

```
10  PG &H400
20  ACCEL 1000
30  FEED ALL_A 0
40  MOVL 5000 5000 VOID VOID
50  WAIT SW(192)==1
60  STOP ALL_A STP_I            /* 急停止。STP_D なら減速停止
70  WAIT RR(ALL_A)==0
```

・ ②ハードによる途中停止。下記は MPG-314 の汎用入力を利用した停止です。移動前に停止条件を設定しています。このほか、リミット入力、アラーム入力による停止もできます。

```
10  PG &H400
20  ACCEL 1000
30  FEED ALL_A 0
40  STOP ALL_A IN3_ON           /* 停止条件設定。X-IN3(J2 ｺﾈｸﾀ 1 番ﾋﾟﾝ)がｵﾝしたら停止
50  MOVL 5000 5000 VOID VOID
60  WAIT RR(ALL_A)==0
70  IF RR(IN3_)<>0 THEN : PRINT "IN3 停止" : END_IF          /* 停止原因確認
80  STOP ALL_A VOID             /* 停止条件解除
```

マルチタスク

- ◆ タスク数は 32 本 (0(メイン)～31)です。
- ◆ 実行するとメインタスクでプログラムが走り、メインタスクから子タスクを起動します。
- ◆ 子タスク同士の優先順位はありません。子タスク同士で起動、終了が可能です。

マルチタスク関係のコマンド

・ FORK

タスクを起動します。メインタスクから子タスク、子タスクから子タスクを起動できます。END コマンド、他のタスクからの QUIT で終了します。実行中のタスクを FORK すると、そのタスクは最初から再実行されます。

・ PAUSE

実行中の他のタスクを一時停止します。

・ CONT

PAUSE で一時停止中のタスクを再実行します。

・ QUIT

タスク終了

```
10   FORK 1 *TASK1              /* タスク１起動
20   WAIT SW(192)==1
30   QUIT 1                     /* タスク１終了
40   OFF 0
50   END
60 *TASK1
70   DO
80     ON 0 : TIME 500
90     OFF 0 : TIME 500
100  LOOP
```

RS-232通信

- MPC-684 にはユーザーでプログラム制御可能な RS-232 ポートが 2CH あり、パソコンや画像機器との通信ができます。そのうち 1CH はデジタル社タッチパネル GP シリーズとダイレクトアクセスが可能です。
- MRS-402 で最大 4CH を増設できます。
- どのタスクでも通信できますが、複数のタスクから同時にアクセスすると正常に送受信できません。通信タスクは 1 本に限定するべきです。

主要コマンド

- CNFG#n　　CHn 通信設定
- PRINT#n　　CHn 出力
- INPUT#n　　CHn 入力

```
CNFG#0 "9600b8pns1XON"      /* CH0 初期化 9600bps 8ﾋﾞｯﾄ ﾉﾝﾊﾟﾘ  1ｽﾄｯﾌﾟ XON/OFF
INPUT#0 a$                  /* CH0 から文字列入力
  b$="&H"+a$                /* 文字列処理
  v=VAL(b$)                 /* HEX 文字列から数値へ
PRINT "&H"+a$ "=" v         /* FTMW 画面表示
PRINT#0 HEX$(v)+"¥n"        /* CH0 へ出力
```

デバッグ

基本形(実行・停止・確認)

・ MPC のデバッグの基本形は 実行・停止・確認 です。例えば、実行(RUN)して装置が止まったら停止(Ctrl+A)してプログラムの停止位置から停止原因を調べます。停止行を調べるには Ctrl+M です。

```
10        ON 0
20        WAIT SW(192)==1
30        OFF 0
#RUN                          /* 実行して装置が止まったところで Ctrl+A でプログラム停止。
        *0   [20]
#                             /* Ctrl+M して各タスクの停止行を確認
TASK0 20     WAIT SW(192)==1  /* ﾀｽｸ 0 の 20 行で SW(192)オン待ち。どうして?
```

PRINTを仕込む

・ プログラムの必要個所に PRINT 文を仕込んで変数や I/O の状態を表示させます。

```
10          C=0
20        DO
30            C=C+1
40          PRINT "count=" C   /* C の値をモニタ
50        LOOP WHILE C<3
#RUN
count=1
count=2
```

サブルーチン単位で実行

・ ＲＵＮにサプルーチンを指定するとサブルーチン単位で実行します。仕事単位でサブルーチン化しておくと部分的なデバッグができます。

```
10        DO
20          GOSUB *FLICK
30        LOOP
40      *FLICK
50        ON 0
60        TIME 50
70        OFF 0
80        TIME 50
90        RETURN
#RUN *FLICK                   /* *FLICK サブルーチンだけ実行
#90     ・・・・・GOSUB と対応しない RETURN が実行されました      /* 終了
```

自動実行中の停止個所の確認方法

・ 自動実行中に停止した場合、電源を切らずにそのままケーブルを差込み FTMW と接続してください。

```
<<装置が止まった！どうしたこったい？。プログラムケーブルを差込み FTMW と接続>>
VER
MPC-684 ADVFSC(r)eREV-3.82v
 BASIC like + multi tasking
 Created by ACCEL Co.'~2001              /* MPC と接続完了
#MON                                     /* MON コマンドでも停止位置が判る
        *0   [20]
#                                        /* Ctrl+M で各タスクのリストを表示
TASK0 20        WAIT SW(192)==1          /* タスク 0 が 20 行で SW(192)オン待ち。
#LIST 0
10        ON 0
20        WAIT SW(192)==1
30        OFF 0
```

プログラムポートの出力記録

・ LOG<Enter>とすると RS-232 CH1(プログラムポート)の出力記録を表示します。これにより自動実行時のランタイムエラーの事後確認ができます。

・ 記録メモリは 1k バイトのリングバッファです。

・ LOG 0<Enter>とするとバッファをクリアします。

・ タスク 0 でのランタイムエラーのメッセージは記録できません。

```
LIST
10        FOR I=1 TO 2
20          PRINT I
30        NEXT I
#LOG 0                          /* LOG データクリア
#RUN
1                               /* プログラム実行中の PRINT 表示
2
#LOG                            /* LOG データを見る

1                               ┐ 記録されたデータ
2                               ┘

#
```

特殊なプログラム

- タスク 0 を END で終了すると FTMW にプロンプトが返ってきます。この状態で MPC へコマンドを発行することが可能になり、リアルタイムで変数などのモニタ、実行位置の確認ができるようになります。
- 注意
  - ・プログラム中に PRINT 表示があると表示が重なります。
  - ・プログラムを変更すると停止します。
  - ・Ctrl+A では停止できなくなります。Ctrl+]で停止してください。

```
10        FORK 1 *JOB1
20        END                  /* ﾀｽｸ 0 を END で終わらしてしまう
30      *JOB1
35        DO
40          ON 0
50          TIME 50
60          OFF 0
70          TIME 50
80        LOOP
#RUN                           /* 実行
#                              /* ﾌﾟﾛﾝﾌﾟﾄが返ってくる。ﾀｽｸ 1 は動いている
TASK0 20          END          /* Crtl+M で実行位置を表示
TASK1 70          TIME 50
#PRINT SW(0)                   /* PRINT 文で I/O 状態や変数を確認できる
1
#       *0   [20]              /* Ctrl+]で停止
        *1   [50]
#
```

タッチパネル

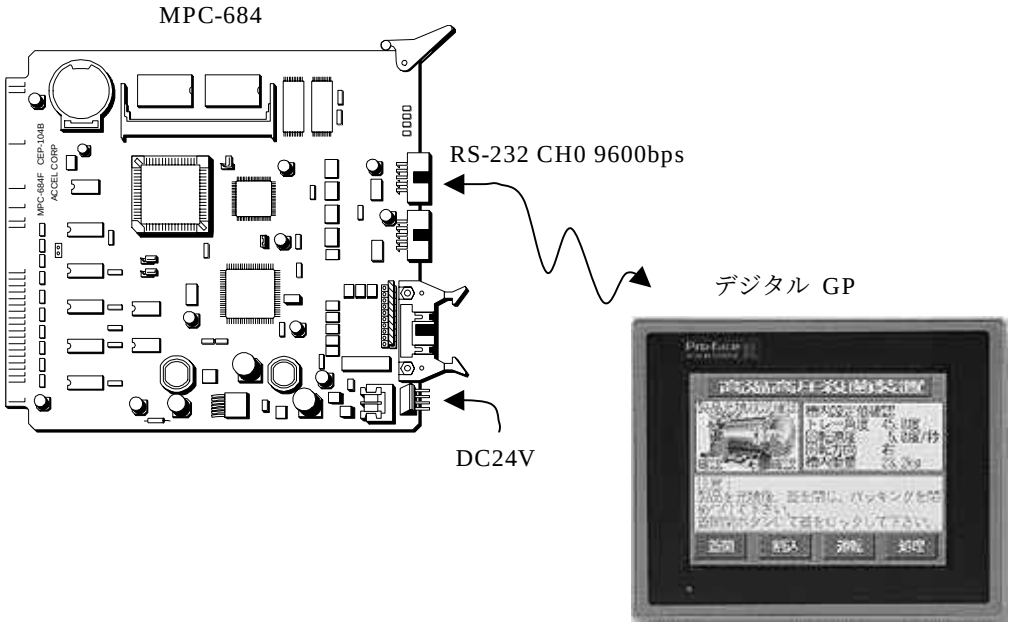
- ◆ デジタル社タッチパネル GP シリーズを「ダイレクトアクセス方式」で接続できます。
- ◆ 接続は、MPC-684 の RS-232 CH0 、または MBK-SH の 2 通りです。
- ◆ データエリア 1000 ワード、I/O エリア 100 ワードのメモリ空間をダイレクトに読み書きすることができます。
- ◆ プログラムでは通信を一切意識することなく、I/O 感覚でタッチパネルを制御できます。

MPC-684のRS-232 CH0 との接続

・ この機能を「MBK-RS」といいます。プログラム先頭に「PROTOCOL MEWNET」と記述することによりこのモードに入ります。

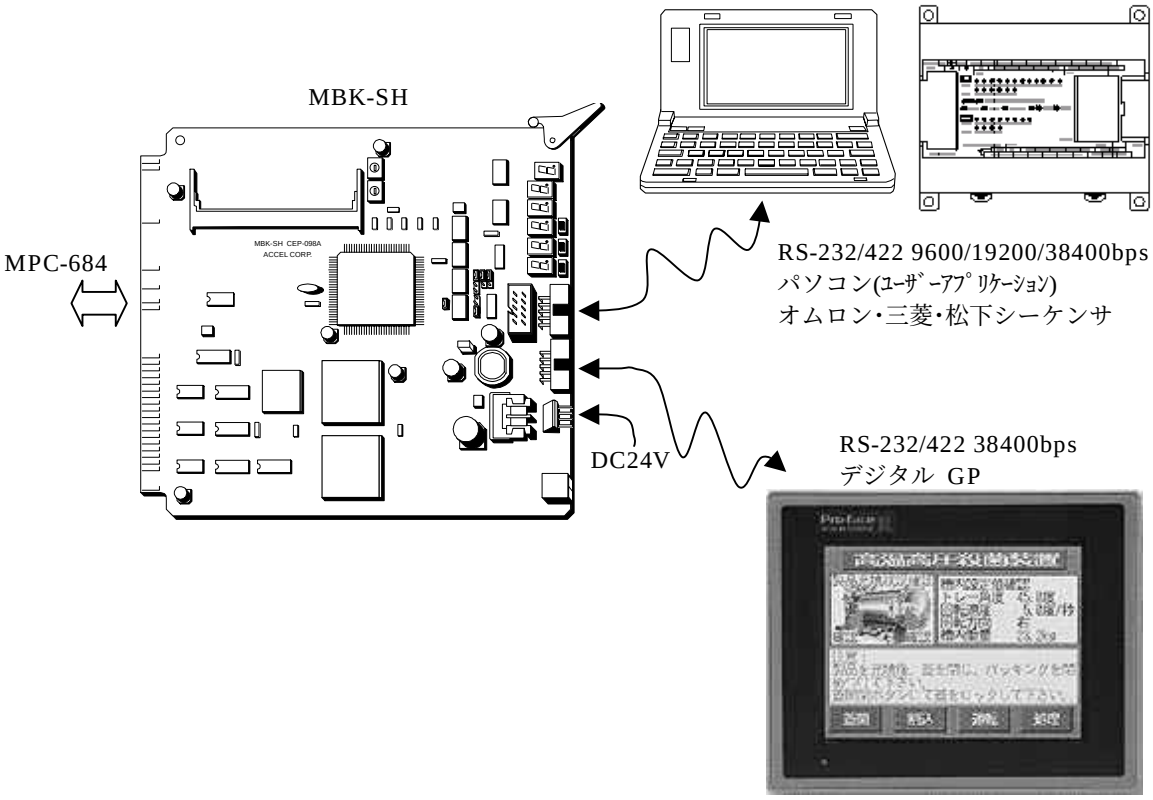
```
10  PROTOCOL MEWNET
```

・ MBK-RS と CH0 通常モードの併用はできません。



MBK-SH との接続

- MBK-SH は RS-422 38400bps で通信します。
- 独立した CPU 制御なのでメイン CPU に負担をかけません。
- シーケンサとの上位リンク(HOST 側)もできます。
- プログラム実行位置を表示する 7 セグメント付き。



MBK-RSとMBK-SHの選択目安

- MBK-RS は 9600bps ですが通常使用での応答性に問題はありません。
- RS-232 CH0 の使用状況、空スロット数、信号線の引き伸ばし等を考慮の上、機種を選択してください。
  例えば、RS-232 CH0 を他の機器で使用する、信号線引き回しが長く減衰やノイズ混入の心配がある、プログラムが重く CPU の負担が大きい(タスク、ループが多い)、タッチパネルを 2 台接続する、などの場合は MBK-SH です。
- プログラムは”PROTOCOL MEWNET” を除きコンパチブルです。

分類別コマンドリスト

I/O

?	‥ビットの読み込み
ALT	‥ON/OFF 反転
B_OUT	‥パラレル出力
CLR_OUTP	‥領域別出力クリア
HIN	‥パラレル入力
HOUT	‥パルスボート汎用出力
HPT	‥パルスボード原点ポート入力
HSW	‥ビットの読み込み
IN	‥パラレル入力
IO	‥I/O モニタ
IOR	‥バス読み込み
IOW	‥バス書き込み
OFF	‥出力オフ
ON	‥出力オン
OUT	‥パラレル出力
P_HSW	‥MPC-68K J4 コネクタの入出力
P_IN	‥MPC-68K J4 コネクタの入出力
P_IN	‥MPC-684 J4 コネクタの高速入力
P_OFF	‥MPC-68K J4 コネクタの入出力
P_ON	‥MPC-68K J4 コネクタの入出力
P_OUT	‥MPC-68K J4 コネクタの入出力
P_SW	‥MPC-68K J4 コネクタの入出力
PULSE_OUT	‥出力ポート パルス ON/OFF
SENSE_SW	‥入力検出で出力操作
SETIO	‥出力初期化
SW	‥ビット入力
WS0	‥タイムアウト付入力
WS1	‥タイムアウト付入力

MBK-SH/RS

ADD_MBK	‥データインクリメント
CHR$	‥MBK より文字列の取得
DIMCPY	‥データエリアコピー
LD_M	‥メモリー括コピー(MPC→MBK)
MBK	‥データエリア読み出し
MBK	‥バックアップ状態・機種確認
PROTOCOL	‥MBK-RS
S_MBK	‥データエリア書き込み
S_MBK	‥指定エリアー括書き込み
S_MBK	‥データリスト表示
S_MBK	‥文字列転送
SV_M	‥メモリー括コピー(MBK→MPC)

MPCLNK

CMN	‥共有変数参照
LNK	‥LNK アドレス取得

M_INP　　‥アドレス&H1000 の LNK からデータ取出
M_INP1　　‥アドレス&H2000 の LNK からデータ取出
M_PR　　‥アドレス&H1000 の LNK へデータ送出
M_PR1　　‥アドレス&H2000 の LNK へデータ送出
M_R　　‥メール受信
M_X　　‥メール送信
MR　　‥受信データの有無を確認
MX　　‥相手受信バッファデータの有無確認
S_CMN　　‥共有変数変更
S_LCL　　‥ローカル変数変更
S_RNG　　‥子 LNK の共有変数エリア獲得宣言
S_SCN　　‥LNK サポートアドレス変更

MPG-314専用

ACCEL　　‥加減速テーブルの作成
CH_MOVS　　‥MOVS 到達点途中変更
ERR_PAUSE　　‥エラー発生時のタスク制御
FEED　　‥スピード設定
FEED　　‥スピード設定(微細設定)
HOME　　‥定則低速原点復帰
HOME　　‥高速加減速原点復帰
HOUT　　‥ポート出力
HOUT　　‥レジスタ制御
HPT　　‥入力ポート読み込み
INCHK_314　　‥入力モニタ
INSET_314　　‥入力ポート機能設定
M_RMVS　　‥非対称加減速移動(相対移動)
MOVL　　‥直線補間(絶対座標移動)
MOVS　　‥軸独立パルス発生(絶対座標移動)
MOVT　　‥連続補間(絶対座標移動)
PG　　‥PG 宣言
PLSC　　‥一定速パルス発生
PRSET_ACCEL　　‥ACCEL パラメータ復旧
RANGE　　‥動作範囲の制限(MPG-68K 互換)
RANGE　　‥動作範囲の制限(MPG-314 拡張)
RMVC　　‥無限パルス発生
RMVL　　‥直線補間(相対座標移動)
RMVS　　‥軸独立パルス発生(相対座標移動)
RMVT　　‥連続補間(相対座標移動)
RR　　‥レジスタ読み取り
SETP　　‥円弧補間(MOVT)の点データ設定
STOP　　‥停止条件
STPS　　‥現在位置指定
STPS　　‥カウンタ設定
U　　‥カウンタ値読込み
U　　‥カウンタ値読込み&クリア
U　　‥カウンタ値読込み&クリア(PG 指定)
WARP　　‥ワープジャンプ
X　　‥カウンタ値読み込み
X　　‥カウンタ値読み込み&クリア
X　　‥カウンタ値読み込み&クリア(PG 指定)
Y　　‥カウンタ値読み込み

Y　　‥カウンタ値読み込み&クリア
Y　　‥カウンタ値読み込み&クリア(PG 指定)
Z　　‥カウンタ値読み込み
Z　　‥カウンタ値読み込み&クリア
Z　　‥カウンタ値読み込み&クリア(PG 指定)

MPG-68K互換

ACCEL　　‥加減速テーブルの作成
BSY　　‥パルス発生状態入力
CLRPOS　　‥現在位置クリア
CURPOS　　‥現在位置表示
FEDD　　‥スピード設定
FEDH　　‥スピード設定
FEDT　　‥スピード設定
FEDZ　　‥スピード設定
FEED　　‥スピード設定
GO　　‥4 軸同時パルス発生
HOME　　‥原点復帰
HOMZ　　‥原点復帰
JMPZ　　‥ゲートモーション移動
JUMP　　‥ゲートモーション移動
LIMZ　　‥ゲートモーション規制
MOVE　　‥XYU 絶対座標移動
MOVZ　　‥Z 絶対座標移動
P　　‥点データ
PALET1　　‥パレット宣言
PALET2　　‥パレット宣言
PALET3　　‥パレット宣言
PALET4　　‥パレット宣言
PG　　‥PG 宣言
PGSEL　　‥PG ボード選択
PL1　　‥パレットポイント
PL2　　‥パレットポイント
PL3　　‥パレットポイント
PL4　　‥パレットポイント
PULSE　　‥定速パルス発生
Q_PAUSE　　‥クイックポーズ
RM　　‥4 軸相対座標移動
RMOV　　‥XYU 軸相対座標移動
RMVZ　　‥Z 軸相対座標移動
SET　　‥インチング量設定
SETP　　‥点データ設定
SETPOS　　‥現在位置変更
SHMZ　　‥原点復帰設定
SHOM　　‥原点復帰設定
STOP　　‥パルス発生停止
T　　‥ティーチングモード
TEACH　　‥ティーチングモード
U　　‥U 軸点データ
X　　‥X 軸点データ
Y　　‥Y 軸点データ
Z　　‥Z 軸点データ

RS-232

CNFG#0	‥通信モード設定
CNFG#2	‥通信モード設定
INP$#0	‥n 文字読み込み
INP$#2	‥n 文字読み込み
INPBLK#	‥バイナリ固定フォーマット入力
INPUT	‥データ入力
INPUT#0	‥データ入力
INPUT#2	‥データ入力
INPUT$	‥n 文字読み取り
LOF	‥バッファの文字数
PR	‥データ表示
PRINT	‥データ表示
PRINT#0	‥データ出力
PRINT#2	‥データ出力
PRX	‥データ HEX 表示
PUT	‥データ表示
PUT#0	‥データ出力
PUT#2	‥データ出力
RS	‥バッファ表示
RSE	‥RS-232C エラー
RSE	‥CH1 キャラクタ入力
SLOW	‥CH1 キャラクタ送信間隔
ST192	‥CH1 ボーレート変更

カレンダ

CLK	‥時間表示
DATE	‥日付表示

コプロ演算

CALF	‥演算
GETF	‥データ取り出し
PRF	‥内部データ表示
SETF	‥データ引き渡し

システム

FCLK	‥クロックスピード変更
MPC	‥MPC-816 互換

タイマ

TIME	‥時間待ち
TMOUT	‥入力時間設定
TMOUT	‥入力時間設定 (WAIT 文)

タスク操作

CONT	‥タスク継続
FORK	‥タスク実行
LIFE_TIME	‥タスク寿命を設定
PAUSE	‥タスク一時停止
QUIT	‥タスク停止
RLS	‥セマフォ解放
RSV	‥セマフォ獲得
SWAP	‥実行権の放棄

デバッグ

CNT	‥実行継続
DUMP	‥メモリ表示
FIND	‥文字列検索
FIX	‥フラッシュ ROM へ書き込み
LOG	‥プログラムポート出力記録
MON	‥停止状態確認
RAM	‥RAM モード
ROM	‥フラッシュ ROM モード
RUN	‥プログラム実行
TASK	‥タスク状態表示
TOFF	‥トレースモード
TON	‥トレースモード

バスアクセス

WIR	‥ワード読み取り
WOW	‥ワード書き込み

ファイルメモリ

P_LD	‥FROM から点データを読込む
P_SV	‥FROM に点データを保存

メモリアクセス

PEEK	‥ﾕｰｻﾞｰﾒﾓﾘ読みだし
POKE	‥ﾕｰｻﾞｰﾒﾓﾘの書き込み

メンテナンス

ERASE	‥フラッシュ ROM のプログラム消去
HISTORY	‥改版履歴表示
MEM	‥メモリーテスト

MPCINIT ‥SRAM 初期化
TMON ‥タスクモニタ
VER ‥改版データの表示

ユーザーコマンド

ADR ‥アドレス取得
COMSET ‥コマンド名設定

演算

@ ‥論理否定
ABS ‥絶対値
AND ‥論理結合式/論理積
ATAN ‥三角関数
ATAN2 ‥三角関数
CONST ‥変数の定数化
COS ‥三角関数
DIM ‥配列宣言
DIM ‥配列宣言 二次元
LET ‥式実行
NOT ‥補数
OR ‥論理結合式/論理和
SFTL ‥配列変数ローテート
SFTR ‥配列変数ローテート
SIN ‥三角関数
SQ ‥自乗
SQR ‥平方根
SWP ‥上下位バイト交換
TAN ‥三角関数

制御文

BREAK ‥制御フロー終了 IF 文から BREAK
BREAK ‥制御フロー終了 繰返し文から BREAK
CASE ‥多値分岐
CASE_ELSE ‥多値分岐
DO ‥繰り返し
ELSE ‥条件分岐
END ‥プログラムの停止
END_IF ‥条件分岐
END_SELECT ‥多値分岐
FAST ‥SWAP 機能停止
FOR ‥繰り返し
GOSUB ‥サブルーチンコール
GOSUB ‥サブルーチンコール 引数渡し
GOTO ‥無条件分岐
IF ‥条件分岐
LOOP ‥繰り返し
NEXT ‥繰り返し

_RET_VAL　　‥戻り値の受け取り
RETURN　　‥リターン
RETURN　　‥リターン 戻り値渡し
SELECT_CASE　‥多値分岐
THEN　　‥条件分岐
UNTIL　　‥条件文
_VAR　　‥引数の受け取り
WAIT　　‥条件待ち
WAIT　　‥タイムアウト付き条件待ち
WEND　　‥条件ループ
WHILE　　‥条件ループ

文字列

AR$　　‥文字配列
ASC　　‥文字からコード
CHR$　　‥コードから文字へ変換
DIM_AR$　　‥文字列配列宣言
DIMCPY　　‥配列データのコピー
GET_VAL　　‥文字列からの数値自動取り出し
HEX$　　‥数値からヘキサ表現文字列
LEN　　‥文字数取得
STR$　　‥数値から文字列
STRCPY　　‥文字列の複写
VAL　　‥数字文字列から数値

編集

DELETE　　‥プログラムの削除
FREE　　‥残りメモリの表示
LIST　　‥プログラム表示
NEW　　‥プログラム初期化
NEWP　　‥点データ初期化
PLIST　　‥点データ表示
RENUM　　‥文番号ふりなおし
TAIL　　‥文番号の最大値
VLIST　　‥プログラムリファレンスの表示

予約定数

MPG-314 関係
  NOP　　‥[指定]ノーオペレーション。
  VOID　　‥[指定]入力無効。
  CLR_ERR　　‥[指定]コマンドエラー解除。
  STP_I　　‥[指定]パルス停止。急停止
  STP_D　　‥[指定]パルス停止。減速停止
  X_A　　‥[指定]軸指定。X 軸
  Y_A　　‥[指定]軸指定。Y 軸
  U_A　　‥[指定]軸指定。U 軸
  Z_A　　‥[指定]軸指定。Z 軸

```
  ALL_A         ‥[指定]軸指定。全軸
  X_C           ‥[指定]カウンタ指定。X 軸
  Y_C           ‥[指定]カウンタ指定。Y 軸
  U_C           ‥[指定]カウンタ指定。U 軸
  Z_C           ‥[指定]カウンタ指定。Z 軸
  ALL_C         ‥[指定]カウンタ指定。全軸
  CW            ‥[指定]円弧補間方向指定。時計方向
  CCW           ‥[指定]円弧補間方向指定。反時計方向
  DS_DACL       ‥[指定]自動減速指定。無効
  EN_DACL       ‥[指定]自動減速指定。有効
  IN0_ON        ‥[指定]停止入力設定。
  IN1_ON        ‥[指定]停止入力設定。
  IN2_ON        ‥[指定]停止入力設定。
  IN3_ON        ‥[指定]停止入力設定。
  IN0_OFF       ‥[指定]停止入力設定。
  IN1_OFF       ‥[指定]停止入力設定。
  IN2_OFF       ‥[指定]停止入力設定。
  IN3_OFF       ‥[指定]停止入力設定。
  INP_ON        ‥[指定]インポジション設定。Low で Active
  INP_OFF       ‥[指定]インポジション設定。Hi で Active
  ALM_ON        ‥[指定]アラーム設定。Low で Active
  ALM_OFF       ‥[指定]アラーム設定。Hi で Active
  LMT_ON        ‥[指定]リミット入力設定。Low で Active
  LMT_OFF       ‥[指定]リミット入力設定。Hi で Active
  SLMT_ON       ‥[指定]RANGE によるソフトリミット有効
  MD_2PLS       ‥[指定]CW/CCW パルス発生指定。
  MD_DPLS       ‥[指定]DIR パルス発生指定。
  CHG_MOVT      ‥[指定]MOVT 優先実行。
  MSK_MOVT      ‥[指定]MOVT 継ぎ目確認。
  STS_MOVT      ‥[指定]MOVT 継ぎ目確認。
  EMG@          ‥[状態監視]EMG がイネーブル検出
  ALM@          ‥[状態監視]サーボアラーム出力 ON 検出
  LMTP@         ‥[状態監視]リミット入力(+)検出
  LMTM@         ‥[状態監視]リミット入力(-)検出
  SLMP@         ‥[状態監視]ソフトリミット(+)検出
  SLMM@         ‥[状態監視]ソフトリミット(-)検出
  EMG_          ‥[停止要因]EMG によって停止
  ALM_          ‥[停止要因]アラーム入力によって停止
  LMTP_         ‥[停止要因]リミット入力によって停止
  LMTM_         ‥[停止要因]リミット入力によって停止
  IN0_          ‥[停止要因]STOP 条件の IN0 によって停止
  IN1_          ‥[停止要因]STOP 条件の IN1 によって停止
  IN2_          ‥[停止要因]STOP 条件の IN2 によって停止
  IN3_          ‥[停止要因]STOP 条件の IN3 によって停止
その他
  MEWNET        ‥CH0 通信プロトコル設定。
  Int           ‥I/O サイズ指定
  Wrd           ‥I/O サイズ指定
  Lng           ‥I/O サイズ指定
```

予約変数

date$	‥日付文字列取得
SYSCLK	‥システムクロック
TASKN	‥タスク番号取得
time$	‥時刻文字列取得
timer	‥時間取得
VER$	‥改版データの取得

--- End of Document ---