テクノ言語／Ｇ言語

サンプル運転プログラム説明書

Ver1.4
2009.08.28

― 目次 ―

テクノコードで説明していますが、【　　】は代替のＧコード命令です。

【注意事項】

ご説明するサンプル運転プログラムは、ご使用のシステムに応じて多少異なる部分があります。

・ＰＬＭＣ－ＭⅡＥＸでは、運転プログラムの先頭に「PNO」を記述する必要がありますが、本説明書では省略しています。

・ＰＬＭＣ－ＭⅡＥＸではマクロ変数の一般レジスタへのアクセスが、「WORDアクセス」(#1000～)と「LONGアクセス」(#5500～)の二通りあります。”１２.マクロ機能”で使用しているマクロ変数は「WORDアクセス」(#1000～)で記述してありますが、ＰＬＭＣ－ＭⅡＥＸのサンプル運転プログラムでは、「LONGアクセス」(#5500～)を使用している部分があります。

１.インクレ直線補間命令

以下のようなときに使用します。
・速度を指定して軸を移動させたい
・移動軌跡を直線にしたい
・現在位置からの移動量を指定したい
・お互いの軸を同期させて動かしたい

指令した全ての軸が、同時に移動を開始して、同時に移動を終了します。
微小な直線補間を連続させることで、自由曲線を描くことも可能です。

【参考資料】
SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-4.直線補間移動指令」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-4.直線補間移動指令」
PLMC-MⅡEX :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-4.直線補間移動指令」

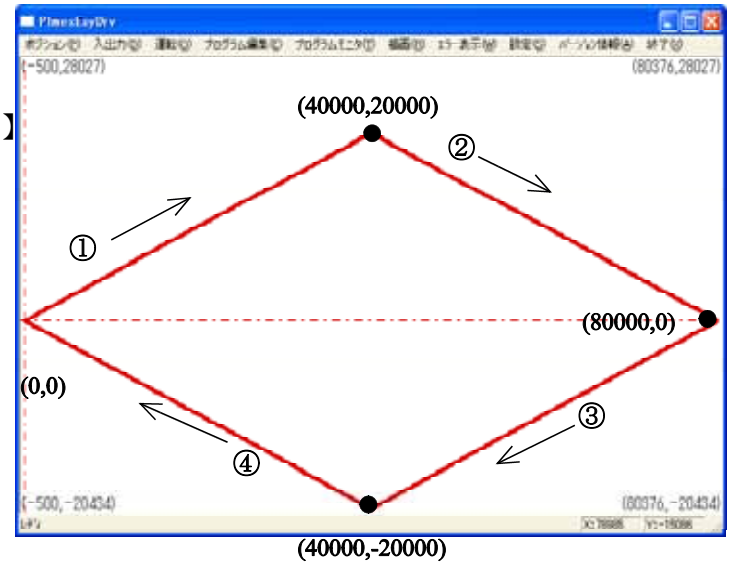
１-１.運転プログラム例

```
/****************************/
/*  L_LIN.TXT               */
/****************************/
① LIN X40000 Y20000 F40000;
①【(G91)G01 X40000 Y20000 F40000;】
② LIN X40000 Y-20000;
③ LIN X-40000 Y-20000;
④ LIN X-40000 Y20000;
⑤ END;
⑤【M30;】
```

LIN：Linear
X,Y：移動量（パルス）
F　：接線速度

【　】は、代替するＧコード命令。
（　）は省略可能です。

＜Ｘ／Ｙ軸の合成軌跡＞



１-２.運転プログラムの動き

	移動量 X	移動量 Y
① (0,0)から、(40000,20000)へ移動	40000	20000
② (40000,20000)から、(80000,00000)へ移動	40000	-20000
③ (80000,0)から、(40000,-20000)へ移動	-40000	-20000
④ (40000,-20000)から、(0,0)へ移動	-40000	20000

１-３.命令の記述

■Ｔコード：　ＬＩＮ　　Ｘ　　Ｙ　　Ｚ　　Ａ　　（Ｆ　）；
■Ｇコード：（G91）Ｇ01　Ｘ　　Ｙ　　Ｚ　　Ａ　　（Ｆ　）；
　　指令コード　　軸指定　　速度指定

指令コードの後に、各軸の移動量、速度を指定します。速度の単位は、パルス／秒です。
指定のない軸は移動を行いません。

※ テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。
（　）は省略可能です。

注　・Ｆ値を省略したときは、以前の指定値で移動します。(このような指定をモーダルと呼びます)
・移動量/座標およびＦ値の指定は、小数点を使用することができます。
・Ｆ値の単位と、移動量/座標の小数点の単位はセッティングＰＣソフトで変更できます。変更の詳細は以下の資料を参照して下さい。
SLM4000　　:「セッティングＰＣマニュアル(TB00-0802)　5-4.表示設定画面」
PLMC40　　:「セッティングＰＣマニュアル(TB00-0812)　5-4.表示設定画面」
PLMC-MⅡEX　:「セッティングＰＣマニュアル(TB00-0901)　5-4.表示設定画面」

２.インクレ位置決め命令

各軸を任意の位置に位置決めするときに使用します。
各軸、現在位置からの移動量を指定して動作します。
各軸は、サーボパラメータにて設定したＰＴＰ速度・ＰＴＰ時定数で移動します。

【参考資料】
SLM4000　　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-1.ｲﾝｸﾚPTP移動指令」
PLMC40　　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-1.ｲﾝｸﾚPTP移動指令」
PLMC-ⅡEX　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-1.ｲﾝｸﾚPTP移動指令」

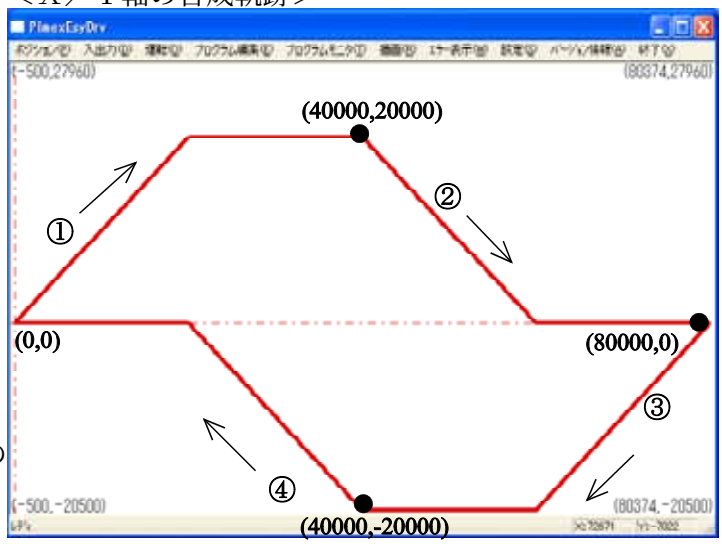
２-１.運転プログラム例

```
/*****************************/
/*   L_PTP.TXT               */
/*****************************/
① PTP X40000 Y20000;
①  【(G91)G00 X40000 Y20000;】
② PTP X40000 Y-20000;
③ PTP X-40000 Y-20000;
④ PTP X-40000 Y20000;
⑤ END;
```

【】は代替するＧコード命令です。
(　) は省略可能です。

PTP：Point To Point
X,Y：移動量（パルス）
※各軸の送り速度はセッティングＰＣの「サーボパラメータ」で設定します。

＜Ｘ／Ｙ軸の合成軌跡＞



２-２.運転プログラムの動き

	移動量 X	移動量 Y
① (0,0)から、(40000,20000)へ移動	40000	20000
② (40000,20000)から、(80000,00000)へ移動	40000	-20000
③ (80000,0)から、(40000,-20000)へ移動	-40000	-20000
④ (40000,-20000)から、(0,0)へ移動	-40000	20000

２-３.命令の記述

■Ｔコード：　ＰＴＰ　　Ｘ　　Ｙ　　Ｚ　　Ａ　　；
■Ｇコード：(Ｇ91)　Ｇ00　Ｘ　　Ｙ　　Ｚ　　Ａ　　；

（(Ｇ91)　Ｇ00：指令コード　／　Ｘ　Ｙ　Ｚ　Ａ：軸指定）

指令コードの後に、各軸の移動量を指定します。単位はパルスです。
指定のない軸は移動を行いません。

※　・テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。
　　・()は省略可能です。

３.アブソ位置決め命令

各軸を任意の位置に位置決めするときに使用します。
各軸、位置決め座標を指定して動作します。
各軸はサーボパラメータで設定したＰＴＰ速度・ＰＴＰ時定数で移動します。

【参考資料】
SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-2.論理座標系ｱﾌﾞｿ位置決め」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-2.論理座標系ｱﾌﾞｿ位置決め」
PLMC-MⅡEX :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-2.論理座標系ｱﾌﾞｿ位置決め」

３-１.運転プログラム例

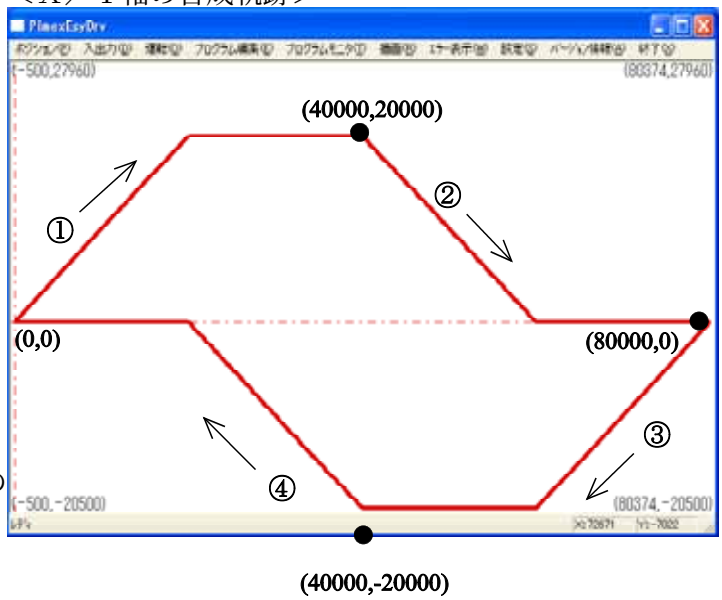
```
/*****************************/
/*  L_PTPA.TXT               */
/*****************************/
① PTPA X40000 Y20000;
① 【(G90)G00 X40000 Y20000;】
② PTPA X80000 Y0;
③ PTPA X40000 Y-20000;
④ PTPA X0 Y0;
⑤ END;
```

【】は代替するＧコード命令です。
（　）は省略可能です。

PTPA：Point To Point Absolute
X,Y　：目標座標（パルス）

※各軸の送り速度はセッティングＰＣの「サーボパラメータ」で設定します。

＜Ｘ／Ｙ軸の合成軌跡＞



３-２.運転プログラムの動き

	移動量 X	移動量 Y
① (0,0)から、(40000,20000)へ移動	40000	20000
② (40000,20000)から、(80000,00000)へ移動	40000	-20000
③ (80000,0)から、(40000,-20000)へ移動	-40000	-20000
④ (40000,-20000)から、(0,0)へ移動	-40000	20000

３-３.命令の記述

■Ｔコード：　ＰＴＰＡ　　Ｘ　　Ｙ　　Ｚ　　Ａ　　；
■Ｇコード：(Ｇ90)　Ｇ00　Ｘ　　Ｙ　　Ｚ　　Ａ　　；
　　指令コード　　軸指定

指令コードの後に、各軸の目標座標を指定します。単位はパルスです。
指定のない軸は移動を行いません。

※　・テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。
　　・()は省略可能です。

４.インクレ円弧補間命令

移動軌跡を円弧にしたいときに使用します。インクレ円弧補間命令には、
右回りの指令(ＣＩＲＲ)と左回りの指令(ＣＩＲＬ)の２種類があります。
以下は中心指定の例ですが、半径指定も可能です。詳細は以下の資料を参照して下さい。

【参考資料】
SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-7.円弧補間移動指令」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-7.円弧補間移動指令」
PLMC-MⅡEX :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-7.円弧補間移動指令」

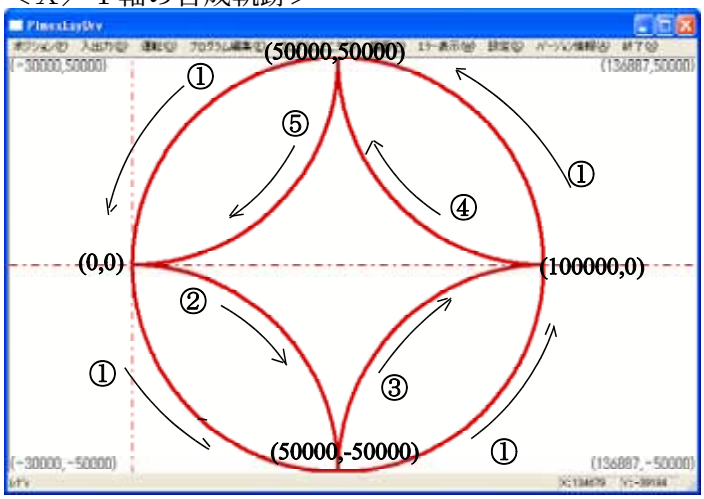
４-１.運転プログラム例

```
/*****************************/
/*  L_CIR.TXT                */
/*****************************/
① CIRL X0 Y0 I50000 J0 F50000;
①【G03 X0 Y0 I50000 J0 F50000;】
② CIRR X50000 Y-50000 I0 J-50000;
②【G02 X50000 Y-50000 I0 J-50000;】
③ CIRR X50000 Y50000 I50000 J0;
④ CIRR X-50000 Y50000 I0 J50000;
⑤ CIRR X-50000 Y-50000 I-50000 J0;
⑥ END;
```

【】は代替するＧコード命令です。

CIRL：Circle Left （左回り）
CIRR：Circle Right（右回り）
X,Y ：終点指定（始点からの移動量）
I,J ：円弧中心（始点からの移動量）

＜Ｘ／Ｙ軸の合成軌跡＞



４-２.運転プログラムの動き

	移動量 X	移動量 Y	中心位置 X	中心位置 Y
① (0,0)から、半径50000の左回りで一周円移動	0	0	50000	0
② (0,0)から、(50000,-50000)へ移動	50000	-50000	0	-50000
③ (50000,-50000)から、(100000,0)へ移動	50000	50000	50000	0
④ (100000,0)から、(50000,50000)へ移動	-50000	50000	0	50000
⑤ (50000,50000)から、(0,0)へ移動	-50000	-50000	-50000	0

４-３.命令の記述

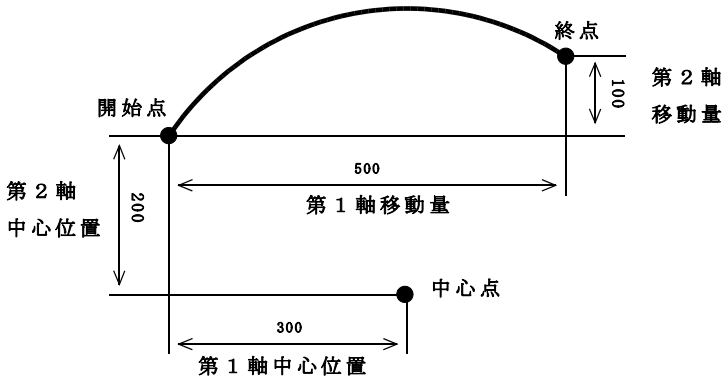
■Ｔコード：ＣＩＲＲ　X　Y　I　J　(F　);
■Ｇコード：(G91) G02　X　Y　I　J　(F　);

指令コード　軸指定　中心指定　速度指定

指令コードの後に、各軸の移動量、円弧中心の位置、速度を指定します。
右回りの指令コードはＣＩＲＲ【G02】、左回りの指令コードはＣＩＲＬ【G03】です。

※ ・テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。
・()は省略可能です。

【例】
ＣＩＲＲ　Ｘ５００　Ｙ１００　Ｉ３００　Ｊ－２００　Ｆ１０



５.サブルーチンコール

以下のようなときに使用します。
・運転プログラム中の複数の箇所で同じ動作を行わせたい
・同じ動作を繰り返し行いたい

同じ動作をサブルーチンとして用意しておき、サブルーチンコールにて呼び出します。

【参考資料】
SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-9.ｻﾌﾞﾙｰﾁﾝｺｰﾙ」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-9.ｻﾌﾞﾙｰﾁﾝｺｰﾙ」
PLMC-ⅡEX　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-9.ｻﾌﾞﾙｰﾁﾝｺｰﾙ」

５-１.運転プログラム例

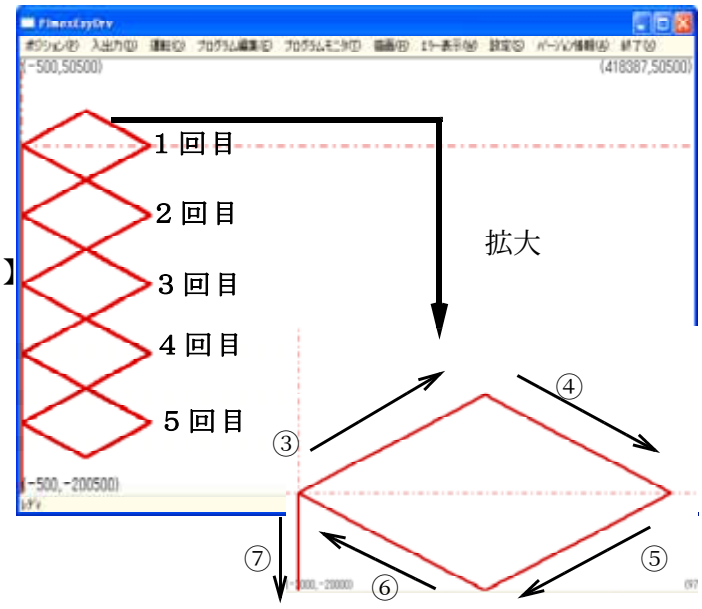
```
/*****************************/
/*  L_CALL.TXT               */
/*****************************/
① CALL SUB L5;
① 【P0100 M98 L5;】
② END;

   :SUB
③ LIN X40000 Y20000 F40000;
③ 【N0100 G01 X40000 Y20000 F40000;】
④ LIN X40000 Y-20000;
⑤ LIN X-40000 Y-20000;
⑥ LIN X-40000 Y20000;
⑦ LIN Y-40000;
⑧ END;
⑧ 【M99;】
```

【】は代替するＧコード命令です。

ＣＡＬＬ：ｻﾌﾞﾙｰﾁﾝ呼び出し
文字列　：呼び出し先のラベル
Ｌ□　　：繰り返し回数

＜Ｘ／Ｙ軸の合成軌跡＞



5-2.運転プログラムの動き

① ”③ ”～”⑧ ”の動作を５回呼び出す。（５回繰り返す）

③ Ｘ軸+40000、Ｙ軸+20000の位置へ移動
④ Ｘ軸+40000、Ｙ軸-20000の位置へ移動
⑤ Ｘ軸-40000、Ｙ軸-20000の位置へ移動
⑥ Ｘ軸-40000、Ｙ軸+20000の位置へ移動
⑦ Ｙ軸-40000の位置へ移動
⑧ メインプログラムに戻る

5-3.命令の記述

5-3-1.サブルーチンコール命令の記述方法

■Ｔコード： ＣＡＬＬ　　文字列　　Ｌ ；
指令コード　ラベル(サブルーチン名)　呼び出し回数

■Ｇコード： M98　　Ｐ　　Ｌ ；
指令コード　呼び出すシーケンス番号　呼び出し回数

・指令コードの後に、”サブルーチン名”、”呼び出し回数”を指定します。
　Ｇコードの場合は、”指令コード”と”呼び出すシーケンス番号”は順不同です。
・呼び出し回数は省略できます。省略した場合は呼び出し回数は１回になります。

※ テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。

5-3-2.サブルーチンの記述方法

■Ｔコードの場合
　：ラベル(サブルーチン名)　”:”(コロン)は必ず記述します。
　命令(PTP、LIN等);
　ＥＮＤ；
(例) サブルーチン名”SUB”の場合

```
:SUB                  /* 先頭には必ずラベルが付いた命令を書きます。   */
 PTP X1000 Y2000;     /* ENDまでの間に命令を記述します。              */
 END;                 /* サブルーチンの最後にはＥＮＤ命令を書きます。*/
```

※ ラベル(サブルーチン名)の後は(改行)を入れてください。

■Ｇコードの場合
　N(シーケンス番号)　命令(PTP、LIN等);
　M30；
(例) シーケンス番号”10”番の場合

```
N010 G91 G00 X1000 Y1000; /* N番号の後に命令を記述します。              */
M99;                      /* ｻﾌﾞﾙｰﾁﾝの最後には終了命令(M99)を書きます。*/
```

６.汎用出力とタイマー命令

汎用出力処理は、以下のようなときに使用します。
・外部機器の動作ＯＮ／ＯＦＦを行いたい
・ランプなどをＯＮ／ＯＦＦしたい

タイマー命令は以下のようなときに使用します。
・プログラムステップの動作時間を指定したい
・運転プログラム中にウェイトを入れたい

両方とも移動命令と同時に実行できます。

※Ｇコードには汎用出力命令はありません。

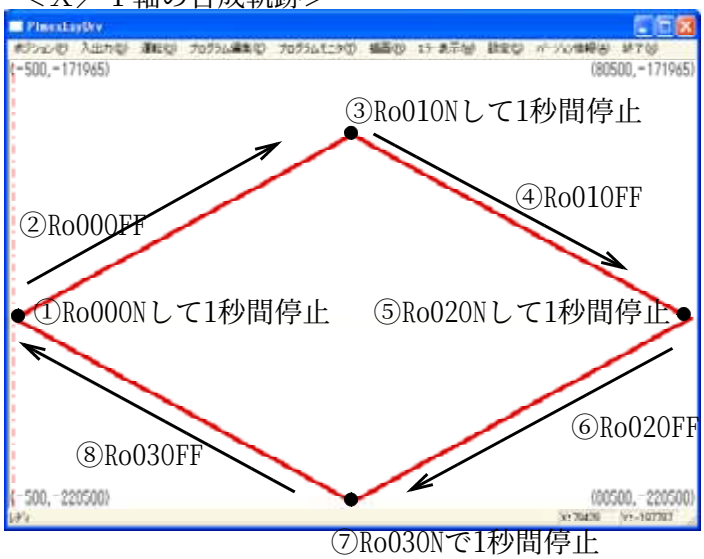
【参考資料】
SLM4000　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-18.ｽﾃｯﾌﾟ実行時間指定、6-3-20.汎用入出力処理指定」
PLMC40　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-17.ｽﾃｯﾌﾟ実行時間指定、7-3-19.汎用入出力処理指定」
PLMC-MⅡEX　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-17.ｽﾃｯﾌﾟ実行時間指定、7-3-19.汎用入出力処理指定」

６-１.運転プログラム例

```
/*****************************/
/*  L_DO.TXT                 */
/*****************************/
  ① TIM1 ONR00;
  ①【G04 P1.0;】
  ② LIN X40000 Y20000 F40000 OFR00;
  ③ TIM1 ONR01;
  ④ LIN X40000 Y-20000 OFR01;
  ⑤ TIM1 ONR02;
  ⑥ LIN X-40000 Y-20000 OFR02;
  ⑦ TIM1 ONR03;
  ⑧ LIN X-40000 Y20000 OFR03;
  ⑨ END;
```

【】は代替するＧコード命令です。

＜Ｘ／Ｙ軸の合成軌跡＞

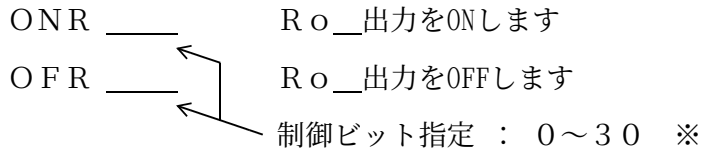


６-２.運転プログラムの動き

① Ro00出力をONして一秒間停止
② Ro00出力をOFFして、(0,0)から(40000,20000)へ移動
③ Ro01出力をONして一秒間停止
④ Ro01出力をOFFして、(40000,20000)から(80000,0)へ移動
⑤ Ro02出力をONして一秒間停止
⑥ Ro02出力をOFFして、(80000,0)から(40000,-20000)へ移動
⑦ Ro03出力をONして一秒間停止
⑧ Ro03出力をOFFして、(40000,-20000)から(0,0)へ移動

6-3.命令の記述

6-3-1.汎用出力処理



※システムによって範囲は変わります。
SLM4000 ：０～３０
PLMC40 ：０～４５
PLMC-MⅡEX：０～６３

出力をＯＮするときは”ＯＮＲ”コードに続けてＯＮする出力のビットを指定します。
出力をＯＦＦするときは”ＯＦＲ”コードに続けてＯＦＦする出力のビットを指定します。

6-3-2.タイマー命令

ＴＩＭ ＿＿

”ＴＩＭ”コードに続けて時間を指定します。
単位は秒です。最小設定単位は0.1[sec]です。

6-3-3.ＰＴＰ命令と同時に指令した例

(例)ＰＴＰ Ｘ１０００ ＯＮＲ００ ＯＦＲ０１ ＴＩＭ２．０；

このステップの場合、以下のような動作になります。

(1) 出力Ｒｏ００ＯＮ、出力Ｒｏ０１をＯＦＦ

(2) Ｘが１０００パルス移動

(3) １０００パルス移動完了時、移動開始から２秒間経過していたら次ステップへ
２秒間経過していなかったら、２秒間経過するまで待った後、次ステップへ

７.入力判別処理

センサーや外部機器からの信号のＯＮ／ＯＦＦ状態によって、運転プログラムの
実行を停止/終了/スキップしたい場合に使用します。
I/Oによる入力待ち(タイムアウト監視)や同期制御として良く使用する手法です。
移動命令と同時に実行できます。

※Ｇコードには汎用入力命令はありません。

【参考資料】
SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-20.汎用入出力処理指定」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-19.汎用入出力処理指定」
PLMC-MⅡEX :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-19.汎用入出力処理指定」

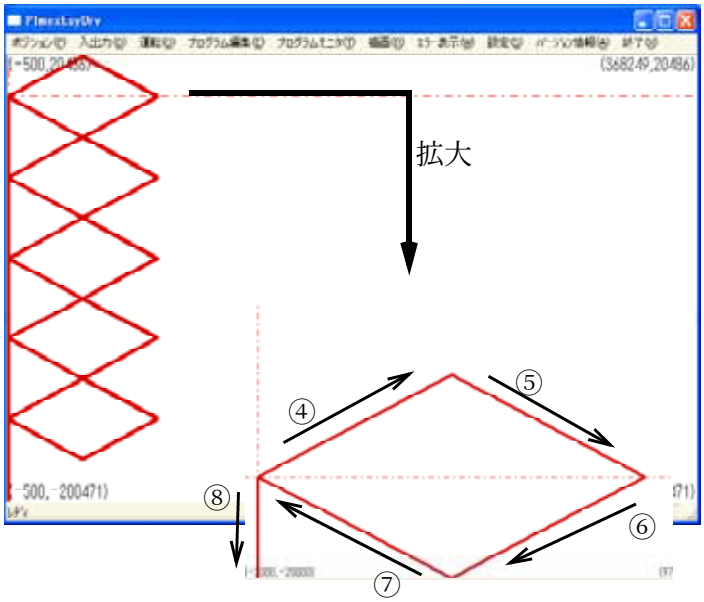
７-１.入力判別機能

７-１-１.運転プログラム例

```
/*****************************/
/*  L_DI.TXT                 */
/*****************************/
① CALL SUB L5 JR01 WR02;
② PTPA X0 Y0;
③ END;

   :SUB
④ LIN X40000 Y20000 F40000
                    ER00 SR03;
⑤ LIN X40000 Y-20000;
⑥ LIN X-40000 Y-20000;
⑦ LIN X-40000 Y20000;
⑧ LIN Y-40000;
⑨ END;
```

＜Ｘ／Ｙ軸の合成軌跡＞



７-１-２.運転プログラムの動き

７－１－２－１.運転プログラム実行開始時、Ri01がＯＮの場合　※

① このステップをスキップ
② (0,0)へ移動
③ 運転プログラム終了

※　Ri00が途中でONした場合
　サブプログラム(”④　”～”⑧　”)から”①　”に戻った時に、Ri01のONを
　確認して”②　”、”③　”を実行。

７－１－２－２.運転プログラム実行開始時、Ri02がＯＮの場合

① このステップの実行をウェイト
　Ri02がOFFになると、”④　”～”⑧　”を５回繰り返す
④ (0,0)から、(40000,20000)へ移動
⑤ (40000,20000)から、(80000,0)へ移動
⑥ (80000,0)から、(40000,-20000)へ移動
⑦ (40000,-20000)から、(0,0)へ移動
⑧ (0,0)から、(0,40000)へ移動
　”④　”～”⑧　”を５回実行
② (0,0)へ移動
③ 運転プログラム終了

７－１－２－３.”④ ”実行中、Ri00がＯＮした場合

④ Ri00がONになるとプログラム運転終了

７－１－２－４.”④ ”実行中、Ri03がＯＮした場合

④ Ri03がONになるとプログラム途中停止
Ri03がOFFになるとプログラム運転を再開します。

７-１-３.命令の記述

● 汎用入力処理

□□R□□
- 制御ビット指定　：　０～３４　※
- 正論理・負論理指定　：　なし／Ｎ
- 処理内容指定　：　Ｅ，Ｊ、Ｗ，Ｓ

※システムによって範囲は変わります。
SLM4000 ：０～３４
PLMC40 ：０～４８
PLMC-MⅡEX：０～６３

(1) 処理内容指定

Ｅｘｉｔ ： プログラム強制終了
この信号がアクティブになると、強制的に動作プログラムを完了して停止します。

Ｗａｉｔ ： ステップ実行開始待ち
各動作ステップの先頭でこの入力をチェックします。アクティブであれば停止します。

Ｊｕｍｐ ： スキップ
この信号がアクティブになるとそのステップの動作を終了し、次のステップへ進行します。　移動指令をスキップした時、座標のズレが発生するかどうかは、スキップしたステップの次ステップの指令形式によって変わります。

Ｓｔｏｐ ： 一時停止
この信号がアクティブな間、動作を停止します。

アクティブ ・・・ 正論理の時、入力信号ＯＮ（回路閉 ―○―○― ）
負論理の時、入力信号ＯＦＦ（回路開 ―○　○― ）

(2) 正論理・負論理指定

無し ： ＯＮのときアクティブ。　（正論理）
Ｎ ： ＯＦＦのときアクティブ。　（負論理）

(3) 制御ビット指定

判別を行うビットの番号を指定します。
範囲は０～６３です。

７-２.タイムアウト監視処理

I/Oによる同期制御や入力待ち(タイムアウト監視)では、「タイムアウト」の例外処理は重要です。一定時間以内に条件が成立しない時は、例外処理に移行する必要もあります。

このサンプルでは、汎用入力の待ち時間を制限する処理です。
指定時間経過するまでに汎用入力がONするかのチェックを行います。

７-２-１.運転プログラム例

以下の３つのサンプルは、汎用入力Ri01が時間内にONすれば正常な待ち合わせとし、次ステップを実行します。時間内にONしない場合(タイムアウト発生時)は、各サンプルで動作を変えています。

”７－２－１－１.”:タイムアウト時、プログラムを終了。
”７－２－１－２.”:タイムアウト時、プログラム実行エラー。
”７－２－１－３.”:タイムアウト時、タイムアウト用サブプログラムを実行。

各サンプルに記述してある番号のステップで、以下の処理を行っています。
① ：Ri01を待ち合わせ
② ：Ri01のON/OFFをチェック
　Ri01がONの場合 ：制限時間内に条件成立(正常な待ち合わせ)
　Ri01がOFFの場合：タイムアウトで③ を実行(７－２－１－１.はプログラム終了)

７－２－１－１.タイムアウト時に運転プログラムを終了する場合

汎用入力待ちがタイムアウトになると、運転プログラムを終了します。
エラーは発生しません。

【プログラム例】

```
① TIM5.0 JR01;          /* タイムアウト時間経過か、Ri01のONで次ステップへ */
② ENR01;                /* Ri01がOFFなら、タイムアウトなのでプログラム終了*/
  LIN X1000 F1000;      /* Ri01がONなら、このステップを実行              */
  END;                  /* プログラム終了                                */
```

７－２－１－２.タイムアウト時にプログラム実行エラーにする場合

汎用入力待ちがタイムアウトになると、プログラム実行エラーにして運転プログラムを終了します。
未定義のマクロ変数への代入を行うことにより、意図的にプログラム実行エラーを発生させます。

【使用マクロ変数】
・専用レジスタ
　#6001: 汎用入力HEXデータ Ri00～Ri03
　　汎用入力状態は、汎用入力モニタ用マクロ変数の対応bitの状態で判断しています。

【プログラム例】

```
① TIM5.0 JR01;          /* タイムアウト時間経過か、Ri01のONで次ステップへ */
② IF ~#6001 & 2;        /* Ri01をチェックし、Ri01がOFFだったらタイムアウト*/
③  #1 = 0;              /* プログラム実行エラー
  ENDIF;
  END;                  /* プログラム終了                                */
```

７－２－１－３.タイムアウト時にサブプログラムへジャンプする場合

汎用入力待ちがタイムアウトになると、サブプログラムへジャンプしてタイムアウト発生時用サブプログラムを実行します。

【プログラム例】

```
① TIM5.0 JR01;          /* タイムアウト時間経過か、Ri01のONで次ステップへ */
② JMP TIMEOUT JR01;     /* Ri01がOFFならタイムアウトなのでジャンプ        */
   END;                 /* プログラム終了                                 */

   /* サブプログラム    */
   :TIMEOUT             /* タイムアウト時の処理                           */
③ LINA X0 Y0 F40000;    /* X0Y0へ位置決め                                 */
   END;
```

7-2-2.応用例

タイムアウト時間をマクロ変数にし、”７-２-１.”の例を使用した応用例です。
直線移動後に汎用入力のチェックを行い、タイムアウトになると
タイムアウト発生箇所判別用のエラーコードを設定して、タイムアウト時の
サブプログラムへジャンプします。
サブプログラムで、プログラム実行エラーにしてプログラムを終了します。

使用マクロ変数：

汎用レジスタ　運転プログラム内で自由に使用できる変数です。
＃１０５０　：　タイムアウト時間　制御周期単位で設定します。
＃１０９０　：　エラーコード　Ri01のONがタイムアウトの場合：1を設定
Ri02のONがタイムアウトの場合：2を設定
Ri03のONがタイムアウトの場合：3を設定
Ri04のONがタイムアウトの場合：4を設定
プログラム実行エラー時に#1090をチェックする事で
タイムアウトエラー発生の有/無および、発生箇所を
確認できます。

専用レジスタ　パラメータ情報やIO情報など機能が固定の変数です。
＃６００１　：　汎用入力HEXデータ　Ri00～Ri03
＃６００２　：　汎用入力HEXデータ　Ri04～Ri07

マクロ変数の詳細は、以下の資料を参照して下さい。

【参考資料】

SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-4.マクロ機能」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-4.マクロ機能」
PLMC-MⅡEX　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-4.マクロ機能」

【プログラム例】

```
/*****************************************/
/*  L_TIMEOUT.TXT                        */
/*                                       */
/* [使用マクロ変数]                      */
/*  ・汎用レジスタ                       */
/*    #1050: タイムアウト時間            */
/*    #1090: エラーコード                */
/*                                       */
/*  ・専用レジスタ                       */
/*    #6001: 汎用入力HEXデータ Ri00～Ri03 */
/*    #6002: 汎用入力HEXデータ Ri04～Ri07 */
/*****************************************/

① #1050 = 2500;                 /* タイムアウト時間[制御周期単位]                  */
                                /* (TIM命令にマクロを使用する場合は制御周期で設定)*/
② #1090 = 0;                    /* エラーコード初期化                              */
```

```
③ LIN X40000 Y20000 F40000;      /* 現在位置から、X+40000、Y+20000の位置へ移動      */
④ TIM#1050 JR01;                 /* タイムアウト時間経過か、Ri01のONで次ステップへ  */
⑤ IF ~#6001 & 2;                 /* Ri01をチェックし、Ri01がOFFだったらタイムアウト*/
⑥  #1090 = 1;                    /* エラーコード1を設定                          */
⑦  JMP ERR;                      /* エラー発生時の処理へジャンプ                 */
⑧ ENDIF;
   LIN X40000 Y-20000;           /* 現在位置から、X+40000、Y-20000の位置へ移動      */
   TIM#1050 JR02;                /* タイムアウト時間経過か、Ri02のONで次ステップへ  */
   IF ~#6001 & 4;                /* Ri02をチェックし、Ri02がOFFだったらタイムアウト*/
    #1090 = 2;                   /* エラーコード2を設定                          */
    JMP ERR;                     /* エラー発生時の処理へジャンプ                 */
   ENDIF;
   LIN X-40000 Y-20000;          /* 現在位置から、X-40000、Y-20000の位置へ移動      */
   TIM#1050 JR03;                /* タイムアウト時間経過か、Ri03のONで次ステップへ  */
   IF ~#6001 & 8;                /* Ri03をチェックし、Ri03がOFFだったらタイムアウト*/
    #1090 = 3;                   /* エラーコード3を設定                          */
    JMP ERR;                     /* エラー発生時の処理へジャンプ                 */
   ENDIF;
   LIN X-40000 Y20000;           /* 現在位置から、X-40000、Y+20000の位置へ移動      */
   TIM#1050 JR04;                /* タイムアウト時間経過か、Ri04のONで次ステップへ  */
   IF ~#6002 & 1;                /* Ri04をチェックし、Ri04がOFFだったらタイムアウト*/
    #1090 = 4;                   /* エラーコード4を設定                          */
    JMP ERR;                     /* エラー発生時の処理へジャンプ                 */
   ENDIF;
⑨ END;                           /* プログラム終了                               */

   :ERR                          /* エラー発生時の処理                           */
⑩ LINA X0 Y0 F40000;             /* X0Y0へ位置決め                               */
⑪ #1 = 0;                        /* プログラム実行エラー                         */
   END;
```

【運転プログラムの動き】

① マクロ変数#1050にタイムアウト時間を設定。
② マクロ変数#1090のエラーコードを０で初期化。
③ 補間移動を行います。。
④ タイムアウト時間経過か、汎用入力Ri01のONで次ステップへジャンプ。
⑤ 汎用入力HEXデータ”#6001”で、Ri01がONしているかチェックします。
　Ri01がONの場合 ：”⑧ ”へジャンプ　タイムアウト無し
　Ri01がOFFの場合：”⑥ ”へジャンプ　タイムアウトエラー発生時処理へ
⑥ エラーコードとして1をマクロ変数#1090にセットします。
⑦ ”⑩ ”へジャンプ。エラー発生時の処理「:ERR」にジャンプします。
⑧ 次の移動処理へ。
Ri02～Ri04も、上記”③ ”～”⑧ ”と同じ処理でタイムアウト監視処理を行います。
⑨ Ri01～Ri04全てがタイムアウト無しの場合、エラー発生せずにプログラムを
終了します。

⑩ 補間移動で、X0Y0へ移動します。
⑪ プログラム実行エラーにします。

８.論理座標設定命令

現在の論理座標系の位置を指定された値に変更(論理座標系のセットアップ)します。

【参考資料】
SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-8.論理座標値設定」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-8.論理座標値設定」
PLMC-MⅡEX :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-8.論理座標値設定」

８-１.運転プログラム例

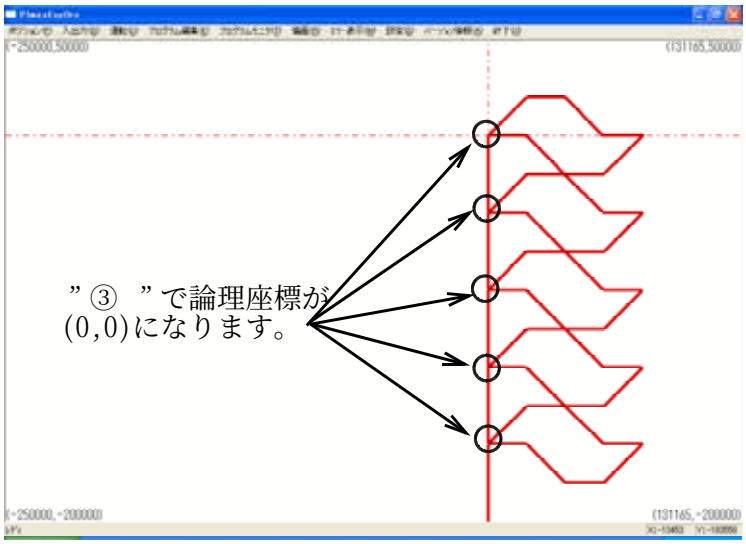
```
/****************************/
/*  L_CSET.TXT              */
/****************************/
① CALL SUB L5;
② END;

   :SUB
③ CSET X0 Y0;
③ 【G92 X0 Y0;】
④ PTPA X40000 Y20000;
④ 【G90 G00 X40000 Y20000;】
⑤ PTPA X80000 Y0;
⑥ PTPA X40000 Y-20000;
⑦ PTPA X0 Y0;
⑧ PTPA Y-40000;
⑨ END;
```

【】は代替するＧコード命令です。

CSET：論理座標系セット
X,Y　：現在位置の新しい論理座標

＜Ｘ／Ｙ軸の合成軌跡(機械座標系アブソ位置)＞



８-２.運転プログラムの動き

①　”③　”～”⑨　”の動作を５回呼び出す。（５回繰り返す）
②　運転プログラム終了

③　論理座標を(0,0)でセットアップ
④　Ｘ軸+40000、Ｙ軸+20000の位置へ移動
⑤　Ｘ軸+40000、Ｙ軸-20000の位置へ移動
⑥　Ｘ軸-40000、Ｙ軸-20000の位置へ移動
⑦　Ｘ軸-40000、Ｙ軸+20000の位置へ移動
⑧　Ｙ軸-40000の位置へ移動
⑨　メインプログラムに戻る

８-３.命令の記述

■Ｔコード：　ＣＳＥＴ　　Ｘ　　Ｙ　　Ｚ　　Ａ　　；
■Ｇコード：　Ｇ９２　　Ｘ　　Ｙ　　Ｚ　　Ａ　　；
　　指令コード　　軸指定

指令コードの後に、各軸の論理座標を指定します。単位はパルスです。
軸指定は、任意の第１～４軸(PLMC-MⅡEXは第１～９軸)で、指定軸のみ座標設定します。

※ テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。

９.ポイント位置決め命令

以下のようなときに使用します。
・位置決めポイントを番号で指定して位置決めしたい
・位置決めポイントの座標値が変わっても、プログラムステップの変更をしたくない

ポイントの指定を行うことにより各軸の目標位置を指定します。

【参考資料】
SLM4000　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-6.ﾎﾟｲﾝﾄ指定PTP位置決め移動、6-3-21.位置決めﾎﾟｲﾝﾄ設定」
PLMC40　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-6.ﾎﾟｲﾝﾄ指定PTP位置決め移動、7-3-20.位置決めﾎﾟｲﾝﾄ設定」
PLMC-MⅡEX　：「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-6.ﾎﾟｲﾝﾄ指定PTP位置決め移動、7-3-20.位置決めﾎﾟｲﾝﾄ設定」

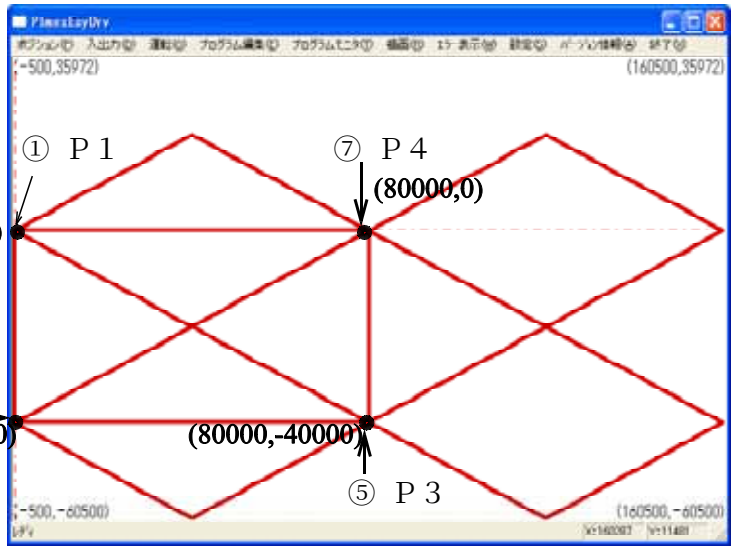
９-１.運転プログラム例

```
/****************************/
/*  L_PTMA.TXT              */
/****************************/
① PTMA P1;
① 【G100 P1;】
② CALL SUB L1;
③ PTMA P2;
④ CALL SUB L1;
⑤ PTMA P3;
⑥ CALL SUB L1;
⑦ PTMA P4;
⑧ CALL SUB L1;
⑨ PTMA P1;
⑩ END;

  :SUB
⑪ LIN X40000 Y20000 F40000;
⑫ LIN X40000 Y-20000 F40000;
⑬ LIN X-40000 Y-20000 F40000;
⑭ LIN X-40000 Y20000 F40000;
⑮ END;

⑯ PNT P1 X0 Y0;
⑰ PNT P2 X0 Y-40000;
⑱ PNT P3 X80000 Y-40000;
⑲ PNT P4 X80000 Y0;
```

＜Ｘ／Ｙ軸の合成軌跡＞



・Gコードでも”⑯ ”～”⑲ ”は同一表記です。
・ＰＬＭＣ－ＭⅡＥＸでは、”⑯ ”～”⑲ ”の様なポイント定義を別途位置決めポイントテーブルとしてダウンロードしておく必要があります。
(運転プログラムには”⑯ ”～”⑲ ”を記述しません。)
詳細は、「PLMC-MⅡEXｾｯﾃｨﾝｸﾞ ＰＣﾏﾆｭｱﾙ(TB00-0901)<5-3-5-2>ﾎﾟｲﾝﾄ位置決めﾃｰﾌﾞﾙ」を参照して下さい。

PTMA：ポイント位置決め
P□　：ポイント番号

PNT　：ポイント定義
P□　：ポイント番号
X,Y　：指定位置（Absolute）

9-2.運転プログラムの動き

① ポイント１(0,0)へ位置決め
② :SUBを1回呼び出す。”⑪ ”～”⑮ ”の動作を行う。
③ ポイント２(0,-400000)へ位置決め
④ :SUBを1回呼び出す。”⑪ ”～”⑮ ”の動作を行う。
⑤ ポイント３(80000,-400000)へ位置決め
⑥ :SUBを1回呼び出す。”⑪ ”～”⑮ ”の動作を行う。
⑦ ポイント４(80000,0)へ位置決め
⑧ :SUBを1回呼び出す。”⑪ ”～”⑮ ”の動作を行う。
⑨ ポイント１(0,0)へ位置決め
⑩ 終了

⑪ 現在位置からX+40000、Y+20000の位置へ移動
⑫ 現在位置からX+40000、Y-20000の位置へ移動
⑬ 現在位置からX-40000、Y-20000の位置へ移動
⑭ 現在位置からX-40000、Y+20000の位置へ移動
⑮ ﾒｲﾝﾌﾟﾛｸﾞﾗﾑに戻る

⑯ 位置決めポイント設定　Ｐ１(　　0,　　0)
⑰ 位置決めポイント設定　Ｐ２(　　0,-40000)
⑱ 位置決めポイント設定　Ｐ３(80000,-40000)
⑲ 位置決めポイント設定　Ｐ４(80000,　　0)

9-3.命令の記述

9-3-1.ポイント位置決め命令

■Ｔコード：ＰＴＭＡ　　Ｐ＿＿；
■Ｇコード：Ｇ100　　Ｐ＿＿；
　　指令コード　ポイント指定

※ テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。

9-3-2.ポイント指定命令

ＰＮＴ　Ｐ＿　Ｘ　Ｙ　Ｚ　Ａ；
指令コード　ポイント番号　軸指定

”ＰＮＴ”コードに続けて、各軸の位置決め座標を指定します。単位はパルスです。
指定のない軸は、移動を行いません。

※ ・テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。

・ＰＬＭＣ－ＭⅡＥＸでは、ポイント定義を別途位置決めポイントテーブルとして
ダウンロードしておく必要があります。
(上記のポイント指定命令は記述しません。)

１０.回転動作指令

主軸やターンテーブルを一定速度で回転させたいときに使用します。
回転を停止させたい場合は、回転数０の回転動作を指令します。
この命令は、無限回転軸(メカ機構１回転パルス数を設定している軸)にのみ指令できます。
加減速はサーボパラメータで指定します。

※ メカ機構１回転パルス数は、ＲＯＭＳＷ設定ソフトで設定します。
詳細は、以下のマニュアルを参照して下さい。

【参考資料】
SLM4000 :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800E) 4-16-2.回転速度指令」
「ROMSW設定ｿﾌﾄﾏﾆｭｱﾙ(TB00-0801) 4-4.軸設定ROMSW ﾒｶ機構1回転ﾊﾟﾙｽ数」
PLMC40 :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810E) 5-16-2.回転速度指令」
:「ROMSW設定ｿﾌﾄﾏﾆｭｱﾙ(TB00-0811) 4-4.軸設定ﾊﾟﾗﾒﾀ ﾒｶ機構1回転ﾊﾟﾙｽ数」
PLMC-MⅡEX :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900E) 5-15-2.回転速度指令」
:「ROMSW設定ｿﾌﾄﾏﾆｭｱﾙ(TB00-0902) 4-4.軸設定ﾊﾟﾗﾒﾀ ﾒｶ機構1回転ﾊﾟﾙｽ数」

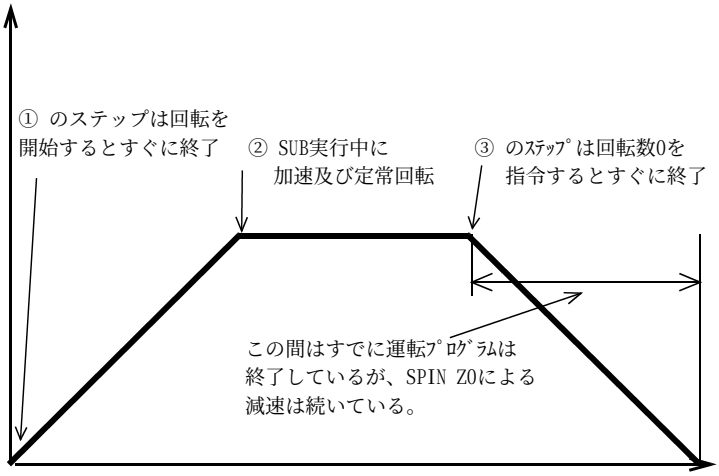
１０-１.運転プログラム例

```
/***************************/
/*  L_SPIN.TXT             */
/***************************/
① SPIN Z1000;
① 【G120 Z1000;】
② CALL SUB L5;
③ SPIN Z0;
④ END;

   :SUB
⑤ LIN X40000 Y20000 F40000;
⑥ LIN X40000 Y-20000;
⑦ LIN X-40000 Y-20000;
⑧ LIN X-40000 Y20000;
⑨ LIN Y-40000;
⑩ END;
```

【】は代替するＧコード命令です。

＜Z軸の速度線図＞



SPIN：主軸回転指令
Z□ ：回転速度(0.1RPM単位)

※Ｇコードでは、主軸命令があります。
M03 主軸正転
M04 主軸逆転
M05 主軸停止
S指令 主軸回転速度

詳細は、以下のマニュアルを参照して下さい。
ＳＬＭ４０００ :「ユーザーズマニュアル機能編(TB00-0800E) 4-17.主軸機能」
ＰＬＭＣ４０ :「ユーザーズマニュアル機能編(TB00-0810E) 5-17.主軸機能」
ＰＬＭＣ－ＭⅡＥＸ :「ユーザーズマニュアル機能編(TB00-0900E) 5-16.主軸機能」

１０-２.運転プログラムの動き

① Ｚ軸が１００ｒｐｍで回転開始（加速を開始し、次ステップへ実行が移る）
② :SUBを5回呼び出す。”⑤ ”～”⑩ ”の動作を行う。(Ｚ軸回転中)
③ Ｚ軸の回転停止（減速を開始）
④ 運転プログラム終了

⑤ 現在位置からX+40000、Y+20000の位置へ移動
⑥ 現在位置からX+40000、Y-20000の位置へ移動
⑦ 現在位置からX-40000、Y-20000の位置へ移動
⑧ 現在位置からX-40000、Y+20000の位置へ移動
⑨ 現在位置からY-40000の位置へ移動
⑩ ﾒｲﾝﾌﾟﾛｸﾞﾗﾑに戻る

１０-３.命令の記述

回転動作命令は、以下のように書きます。

■Ｔコード：ＳＰＩＮ　X　Y　Z　A　;
■Ｇコード：G120　X　Y　Z　A　;
　　↑指令コード　　軸指定

指令コードに続けて、各軸の回転数を指定します。　単位は、０．１ｒｐｍです。

※ テクノコードの場合、指令コード、各種指定の間には、ｽﾍﾟｰｽを必ず入れてください。

１１.INPOS有効／無効

インポジションチェックの有効/無効を指定します。
命令によって、ＩＮＰＯＳ有効/無効が固定(切換不可)のものがあります。
詳細は以下を参照して下さい。

【参考資料】
SLM4000　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-3-16.INPOS有効ﾓｰﾄﾞ指定、6-3-17.INPOS無効ﾓｰﾄﾞ指定」
PLMC40　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-3-15.INPOS有効ﾓｰﾄﾞ指定、7-3-16.INPOS無効ﾓｰﾄﾞ指定」
PLMC-MⅡEX　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-3-15.INPOS有効ﾓｰﾄﾞ指定、7-3-16.INPOS無効ﾓｰﾄﾞ指定」

１１-１.運転プログラム例

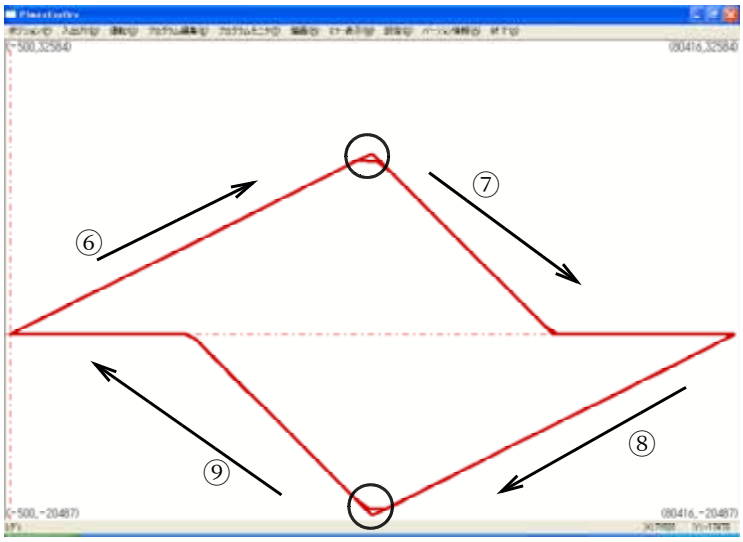
```
/*****************************/
/*  L_INPD.TXT               */
/*****************************/
① INPD;
①【G64;】
② CALL SUB;
③ INPE;
③【G61;】
④ CALL SUB;
⑤ END;

   :SUB
⑥ LIN X40000 Y20000 F40000;
⑦ PTP X40000 Y-20000;
⑧ LIN X-40000 Y-20000;
⑨ PTP X-40000 Y20000;
⑩ END;
```

【】は代替するＧコード命令です。

INPD：Inpos Check Disable
INPE：Inpos Check Enable

＜Ｘ／Ｙ軸の合成軌跡＞



１２.マクロ機能

【参考資料】
SLM4000　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0800F) 6-4.マクロ機能」
PLMC40　　:「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0810F) 7-4.マクロ機能」
PLMC-MⅡEX :「ﾕｰｻﾞｰｽﾞﾏﾆｭｱﾙ機能編(TB00-0900F) 7-4.マクロ機能」

１２-１.代入

マクロ変数に対し、マクロ変数や即値の代入ができます。

１２-１-１.運転プログラム例

この例では、
・ロングワード形式への代入、
・マクロ変数経由で入力ポート0、タスクステータスの状態読出
・マクロ変数経由による出力ポート0の汎用出力を制御
を行っております。

```
/****************************/
/*  L_MCR_SST.TXT           */
/****************************/
① #1000 = 3840;     /* ﾛﾝｸﾞﾜｰﾄﾞ形式のﾏｸﾛ変数へ値(3840=0x700)代入     */
② #1001 = #1600;    /* ﾛﾝｸﾞﾜｰﾄﾞ形式のﾏｸﾛ変数へ入力ﾎﾟｰﾄ0の状態を代入 */
③ #1610 = #1000;    /* 出力ﾎﾟｰﾄ0にある汎用出力を制御。                 */
                     /* Ro00～Ro03をON,Ro04～Ro07をOFF                  */
                     /* 汎用出力以外の信号は変化しない。                */
④ #1052 = 6400000; /* ﾛﾝｸﾞﾜｰﾄﾞ形式のﾏｸﾛ変数へ値代入                  */
⑤ #1054 = #1500;    /* ﾛﾝｸﾞﾜｰﾄﾞ形式のﾏｸﾛ変数へﾀｽｸｽﾃｰﾀｽを代入          */
⑥ #1056 = -#1052;   /* ﾛﾝｸﾞﾜｰﾄﾞ形式のﾏｸﾛ変数同士の代入                */
⑦ END;
```

１２-１-２.記述方法

変数　=　{変数｜即値}；

変数："#"に続けて、レジスタの番号を指定します。
即値：整数を指定します。
　ﾛﾝｸﾞﾜｰﾄﾞ形式：±2147483647
　ﾜｰﾄﾞ形式　　：±32767

※　マクロ変数、即値、演算子の間には、ｽﾍﾟｰｽを必ず入れてください。

１２-２.計算

マクロ変数に対し、演算を行う事ができます。
四則演算や論理演算を使用し、カウンタや設定値の計算だけでなく、
入力やステータスの特定ビットの状態の確認なども可能です。

１２-２-１.運転プログラム例

この例では、マクロ変数の計算を行い、計算結果をオーバーライドとして設定しています。
繰り返し計算して０～１２７の範囲でｵｰﾊﾞﾗｲﾄﾞが1%刻みで変化していきます。

```
/*****************************/
/*  L_MCR_CLC.TXT            */
/*****************************/
① #1000 = 100;              /* 初期値の代入                                 */
② :LOOP
③ 【N100】#1502 = #1000;   /* オーバライドの変更                           */
④ #1000 = #1000 + 1;        /* ﾛﾝｸﾞﾜｰﾄﾞ形式のﾏｸﾛ変数をインクリメント      */
⑤ #1000 = #1000 & 127;      /* 0～127の範囲で変化するように変更。           */
⑥ TIM1.0;
⑦ JMP LOOP;                 /* LOOPﾗﾍﾞﾙへｼﾞｬﾝﾌﾟ。                            */
⑦ 【JMP100】
⑧ END;
```

ＰＬＭＣ４０のＧコードの場合は、「JMP」命令に対応していないため、
「GOTO」命令を使用します。

```
GOTO100;
```

１２-２-２.記述方法

変数　＝　{変数｜即値}　＋　{変数｜即値}；

└ 演算子：+、-、*、/、%、&、|、^、<<、>>、
<、<=、>、>=、==、!=

※　マクロ変数、即値、演算子の間には、ｽﾍﾟｰｽを必ず入れてください。

１２-３.条件判断

IF命令を使った条件判断を行うことが出来ます。
マクロ変数を経由することで、演算結果だけでなく、ステータスや入出力状態などの
内部情報での分岐も行えます。

１２-３-１.運転プログラム例

この例では、”１２-２.”で使った例に条件判断を加え、0～200の間でオーバーライドを
変化させます。

```
/*****************************/
/*  L_MCR_CNDJDG.TXT         */
/*****************************/
① #1000 = 1;                /* オーバライド増分値 */
② :LOOP
③ 【N100】IF #1502 == 200; /* オーバライドが200%の場合                     */
④     #1000 = -1;           /* 増分値の符号反転                             */
⑤ ENDIF;
⑥ IF #1502 == 0;            /* オーバライドが0%の場合                       */
⑦     #1000 =  1;           /* 増分値の符号反転                             */
⑧ ENDIF;
⑨ #1502 = #1502 + #1000;   /* ﾏｸﾛ変数経由でオーバライドを変更              */
⑩ TIM1.0;
⑪ JMP LOOP;                 /* LOOPﾗﾍﾞﾙへｼﾞｬﾝﾌﾟ。                            */
⑪ 【JMP100】
⑫ END;
```

１２-３-２.記述方法

```
ＩＦ　{変数｜即値}　　演算子　　{変数｜即値}；
   (1)
ＥＬＳＥ；
   (2)
ＥＮＤＩＦ；
```

ＩＦの後に条件の式を書きます。
（１)、（２）には、それぞれ条件式の結果が０以外の場合の処理、条件式の結果が０の場合の処理を書きます。
”ＥＬＳＥ”と（２）の処理は必要なければ、省略できます。

※　マクロ変数、即値、演算子の間には、ｽﾍﾟｰｽを必ず入れてください。

１２-４.変数指定

移動命令の軸指定や速度などをマクロ変数で指定することができます。
マクロ変数で指定することで、運転プログラム内の演算結果で動きを変化させることや、
ＰＣやラダーからマクロ変数を指定することで動きを変更することもできます。

１２-４-１.運転プログラム例

この例では、外部から#1050に１辺の長さを設定します。
#1050から移動量を計算し、正方形の動作を繰り返します。

```
/****************************/
/*  L_MCR_PARM.TXT          */
/****************************/
/*#1050に1辺の長さを設定*/
① #1054 = 30000;  /* F値は30kpps */
② :LOOP
③ 【N100】#1052 = #1050;  /* 移動量を指定 */
④ LIN X#1052 Y0  F#1054;
⑤ LIN X0 Y#1052  F#1054;
⑥ LIN X-#1052 Y0 F#1054;
⑦ LIN X0 Y-#1052 F#1054;
⑧ JMP LOOP;       /* LOOPﾗﾍﾞﾙへｼﾞｬﾝﾌﾟ。    */
⑧ 【JMP100】
⑨ END;
```

１２-４-２.記述方法

```
ＰＴＰ　X#1000　Y#1001　Z#1002　A#1003；
```

各軸の軸指令にて、軸のコードに続けて変数を指定します。
変数に入っている値が、各軸の移動量（目標位置）になります。

※　マクロ変数、即値、演算子の間には、ｽﾍﾟｰｽを必ず入れてください。

１２-５.応用例１

汎用入力の状態で、実行する運転プログラムを切り替えます。

１２-５-１.運転プログラム例

```
/****************************/
/*  L_MACRO.TXT             */
/****************************/
  :LOOP
/* 入力i#00(#1600)のD13,D14  */
/* ﾋﾞｯﾄをﾚｼﾞｽﾀ#1000に代入     */
① 【N100】#1000 = #1600 & 24576;
② IF #1000 == 8192;
③   CALL SANKAKU L1;
④ ENDIF;
```

```
⑤ IF #1000 == 16384;
⑥   CALL SIKAKU L1;
⑦ ENDIF;
⑧ IF #1600 & 16;
⑨   JMP LOOP;
⑨ 【JMP100】;
⑩ ENDIF;
⑪ END;

   :SANKAKU
⑫ LIN X25000 Y50000 F25000;
⑬ LIN X25000 Y-50000;
⑭ LIN X-50000;
⑮ END;

   :SIKAKU
⑯ LIN Y50000 F25000;
⑰ LIN X50000;
⑱ LIN Y-50000;
⑲ LIN X-50000;
   END;
```

１２-５-２.運転プログラムの動き

① 入力i#00(#1600)のD13,D14ビットをレジスタ#1000に代入
② レジスタ#1000のD13ビット(Ri01)のみONの場合”③ ”へ、OFFの場合”④ ”へ
③ ３角形動作プログラム（SANKAKU）呼び出し
④ ”② ”のIFの終わり
⑤ レジスタ#1000のD14ビット(Ri02)のみONの場合”⑥ ”へ、OFFの場合”⑦ ”へ
⑥ 四角形動作プログラム（SIKAKU）呼び出し
⑦ ”⑤ ”のIFの終わり
⑧ レジスタ#1600のD4ビット(Ri00)がONの場合”⑨ ”へ、OFFの場合”⑩ ”へ
⑨ ”① ”へジャンプ
⑩ ”⑧ ”のIFの終わり
⑪ 運転プログラム終了

⑫ (0,0)から(25000,50000)へ移動
⑬ (25000,50000)から(50000,0)へ移動
⑭ (50000,0)から(0,0)へ移動
⑮ メインプログラムに戻る

⑯ (0,0)から(0,50000)へ移動
⑰ (0,50000)から(50000,50000)へ移動
⑱ (50000,50000)から(50000,0)へ移動
⑲ (50000,0)から(0,0)へ移動

使用変数：

汎用レジスタ
＃１０００　：　入力i#00(#1600)のD13,D14ビットの状態です。

専用レジスタ
＃１６００　：　ｉ＃００の内容を読み出します。

上記運転プログラムの入力信号は、以下のように割り付けられているものとします。
I#00 D04　：Ri00
I#00 D13　：Ri01
I#00 D14　：Ri02

・Ｒｉ０１～Ｒｉ０２の変化で、以下のように動作が変わります。

Ｒｉ０２	Ｒｉ０１	描画する図形
ＯＦＦ	ＯＦＦ	無し
ＯＦＦ	ＯＮ	三角形
ＯＮ	ＯＦＦ	四角形

・Ｒｉ００のＯＦＦでプログラム運転を終了します。

１２-６.応用例２

汎用入力で指定された分割数で、Ａ軸モータの割り出しを行います。

１２-６-１.運転プログラム例

```
/****************************/
/*  L_INDEX-X.TXT           */
/****************************/
① #1050 = 1000;            /* ﾓｰﾀ一回転ﾊﾟﾙｽ数          */
② #1002 = 1;               /* 割出数初期化             */
③ #1003 = 1;               /* ﾙｰﾌﾟｶｳﾝﾀ初期化           */
  :LOOP
④ 【N100】#1003 = #1003 + 1; /* ﾙｰﾌﾟｶｳﾝﾀｶｳﾝﾄｱｯﾌﾟ       */
⑤ IF #1003 > #1002;        /* １回転終了の場合         */
⑥   IF #1600 & 16;         /* Ri00 ONの場合            */
⑦     JMP EXIT;            /* 運転プログラム終了       */
⑧   ENDIF;
⑨   #1002 = #1600 & 57344;   /* 割出数ｾｯﾄ(Max:8分割)    */
⑩   #1002 = #1002 >> 13;     /* 13bitシフト             */
⑪   #1002 = #1002 + 1;       /* 0～7 → 1～8             */
⑫   #1003 = 1;               /* ﾙｰﾌﾟｶｳﾝﾀﾘｾｯﾄ            */
⑬ ENDIF;
⑭ #1054 = #1050 * #1003;   /* 目標位置計算             */
⑮ #1056 = #1054 / #1002;
⑯ PTPA A#1056;             /* 位置決め                 */
⑰ TIM0.5;                  /* 0.5秒ｳｪｲﾄ               */
⑱ JMP LOOP;
⑱ 【JMP100】;
  :EXIT
⑲ END;
```

１２-６-２.運転プログラムの動き

指定した分割数毎に０．５秒停止し、一回転したら分割数を読み出して、
再度分割動作を行います。
汎用入力Ｒｉ０１～Ｒｉ０３の３ビットを使用して、Ａ軸モータの一回転を分割する数を指定します。
汎用入力Ｒｉ００が、ＯＮになるとプログラムが終了します。
Ａ軸の一回転を１サイクルとして、Ｒｉ００のチェック（プログラム終了チェック）は、１サイクル毎に行います

使用変数：

汎用レジスタ
＃１０５０　：　モーター回転パルス数です。
＃１００２　：　回転の分割数です。
＃１００３　：　Ａ軸を一回転させるためのプログラムループの繰り返し回数です。
＃１０５４　：　一回の割り出しの位置を計算するためのワーク変数です。
＃１０５６　：　一回の割り出し位置です。

専用レジスタ
＃１６００　：　ｉ＃００の内容を読み出します。

上記運転プログラムの入力信号は、以下のように割り付けられているものとします。
I#00 D04　：Ri00
I#00 D13　：Ri01
I#00 D14　：Ri02
I#00 D15　：Ri03

汎用入力Ｒｉ０１～Ｒｉ０３の変化で以下のように動作が変わります。

Ｒｉ０３	Ｒｉ０２	Ｒｉ０１	分割数
ＯＦＦ	ＯＦＦ	ＯＦＦ	１分割
ＯＦＦ	ＯＦＦ	ＯＮ	２分割
ＯＦＦ	ＯＮ	ＯＦＦ	３分割
ＯＦＦ	ＯＮ	ＯＮ	４分割
ＯＮ	ＯＦＦ	ＯＦＦ	５分割
ＯＮ	ＯＦＦ	ＯＮ	６分割
ＯＮ	ＯＮ	ＯＦＦ	７分割
ＯＮ	ＯＮ	ＯＮ	８分割

Ｒｉ００のＯＮでプログラム運転を終了します。

１２-７.ストッカー動作(搬送装置)の事例

マクロ変数を使用して、ストッカー座標(棚の位置)から各動作の目標位置を自動生成する例です。搬送機械等でストッカー座標はティーチングで決定し、棚へのアプローチや、ワークの取り上げ等の動作を運転プログラム内で自動生成する例がよくあります。
以下の例は、マクロ変数にストッカー座標を設定して実現します。

ストッカー座標は、ＰＣやラダーで管理しマクロ変数へ設定します。

【マクロ変数を使用するメリット】
・アプローチ点など前後の位置決めポイントは、ストッカー座標から運転プログラム内で自動生成します。そのため、ストッカー座標のデータテーブルは、ストック数のみの管理で充分です。ストック数が増えても管理するデータテーブルは大幅に増えることなく実現できます。

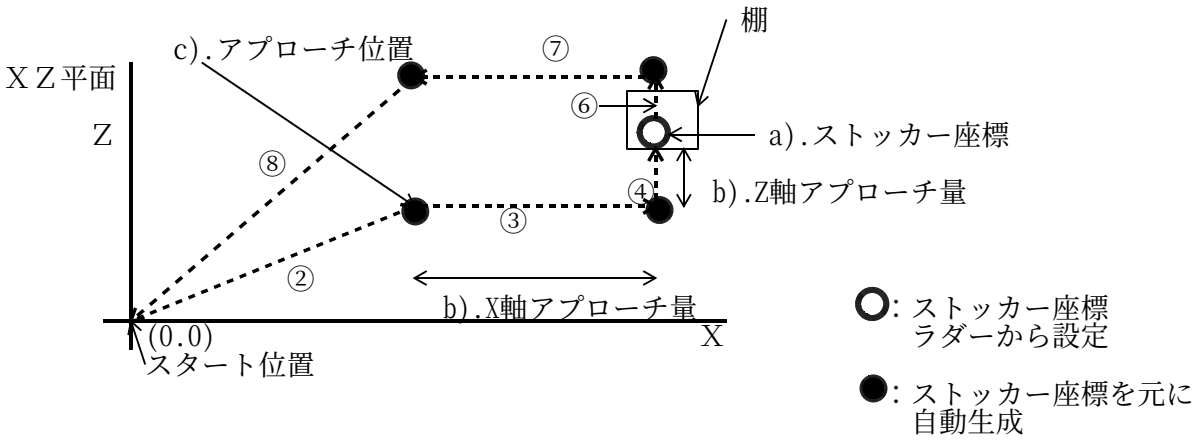
１２-７-１.運転プログラム例

ストッカー座標とアプローチ量から､途中のアプローチ位置を自動生成して移動を行います。

a).ストッカー座標　　棚の座標

b).アプローチ量
ストッカー座標から、アプローチ位置までのオフセット量

c).アプローチ位置
アプローチ位置 =ストッカー座標 - アプローチ量



※　・図の点線の矢印は、サンプル運転プログラムの移動軌跡です。
・点線横の数字は、以下の運転プログラム例に記述してある番号に対応しています。

【プログラム例】

```
/*****************************************/
/*  L_MCR_STOCK.TXT                      */
/*                                       */
/* [使用マクロ変数]                      */
/*  ・汎用レジスタ ロングワード          */
/*    #5500: X軸ストッカー座標           */ ┐
/*    #5502: Y軸ストッカー座標           */ ├ストッカー座標
/*    #5504: Z軸ストッカー座標           */ ┘
/*                                       */
/*    #5512: Z軸ピックアップ量           */ ┐
/*    #5560: 移動速度                    */ ┘パラメータ
/*                                       */
/*    #5520: X軸アプローチ量             */ ┐パラメータ
/*    #5524: Z軸アプローチ量             */ ┘運転プログラム内の計算で使用
/*                                       */
/*    #5550: X軸アプローチ位置           */ ┐
/*    #5554: Z軸アプローチ位置           */ ┘運転プログラム内で自動生成
/*                                       */
/*****************************************/

/* ストッカー座標設定                                          */
/* 運転プログラムを起動する前に、ＰＣやラダーから設定しておきます。 */
/* #5500 = 20000;              /* X軸ストッカー座標設定        */
/* #5502 = 20000;              /* Y軸ストッカー座標設定        */
/* #5504 = 10000;              /* Z軸ストッカー座標設定        */

/* 各パラメータ設定                                            */
#5520 = 10000;                 /* X軸アプローチ量設定          */
#5524 = 4000;                  /* Z軸アプローチ量設定          */
#5512 = 2000;                  /* Z軸ピックアップ量            */
#5560 = 10000;                 /* 移動速度10000pps設定         */

/* アプローチ位置を自動生成                                    */
#5550 = #5500 - #5520;         /* X軸アプローチ位置設定        */
#5554 = #5504 - #5524;         /* Z軸アプローチ位置設定        */


① LINA X0 Y0 Z0  F#5560;       /* スタート位置へ移動           */
② LINA X#5550 Y#5502 Z#5554 F#5560; /* 各軸アプローチ位置へ移動 */
③ PTP X#5520;                  /* X軸アプローチ量移動          */
④ PTP Z#5524;                  /* Z軸アプローチ量移動          */
⑤ TIM1.0;                      /* 1秒待ち                      */
⑥ PTP Z#5512;                  /* Z軸ピックアップ量移動        */
⑦ PTP X-#5520;                 /* X軸アプローチ位置へ移動      */
⑧ LINA X0 Y0 Z0  F#5560;       /* 各軸スタート位置へ移動       */
⑨ END;                         /* プログラム終了               */
```

【運転プログラムの動き】

① スタート位置へ移動
② 各軸、アプローチ位置へ移動します。
③ X軸がストッカー座標に移動します。
④ Z軸がストッカー座標に移動します。
⑤ 1秒待ちます。
⑥ Z軸が上昇します。
⑦ X軸がアプローチ位置へ移動します。
⑧ 各軸スタート位置へ移動します。
⑨ プログラムを終了します。

１２-７-２.ストックが複数ある場合

ストックが複数ある場合は、”１２-７-１.”の運転プログラムをストックの数だけ繰り返し実行します。ストッカー座標も、ストックの数だけ管理しておきます。

最初のストッカー座標を、ストッカー座標用のマクロ変数に書き込み、運転プログラムを実行します。実行完了したら、次のストッカー座標をマクロ変数に書き込み、運転プログラムを実行します。
「ストッカー座標をマクロ変数へ書き込み→運転プログラム実行」を繰り返し行うことでストックが複数ある場合の動作を実現します。

１２-７-３.ラダーによるティーチングの例

タッチパネルやＰＣアプリからティーチングによって、複数のストッカー座標をＰＣやラダーのレジスタに記憶します。

【ティーチングの事例】

① ストッカー座標まで手動操作(JOG,インチング等)で移動
タッチパネルやＰＣアプリでストックがある棚の位置まで移動します。

② ストッカー座標を記憶
ストッカー座標(棚の位置)をＰＣやラダーのレジスタに記憶します。
各軸の座標は、以下で読み込みできます。
・ステータスデータ(DAT_STATUS)の各軸座標
・マクロ変数のポジションデータ(#4000～)

ストックの数だけ”①　”→”②　”を繰り返し、ストッカー座標を記憶します。

③ 作業対象のストッカー座標をマクロ変数に設定
ＰＣやラダーのレジスタに記憶したストッカー座標を、運転プログラムで使用するマクロ変数に設定します。
#5500 ： X軸ストッカー座標設定
#5502 ： Y軸ストッカー座標設定
#5504 ： Z軸ストッカー座標設定
} ”１２-７-１.”の例の場合は、このマクロ変数に設定します。

④ 運転プログラム起動
運転プログラムを起動します。

ストックの数だけ”③　”→”④　”を繰り返します。

ラダー処理の詳細は、「ＰＬＭＣ－ＭⅡＥＸ対応サンプルラダー説明書」(TB040-0917)を参照して下さい。