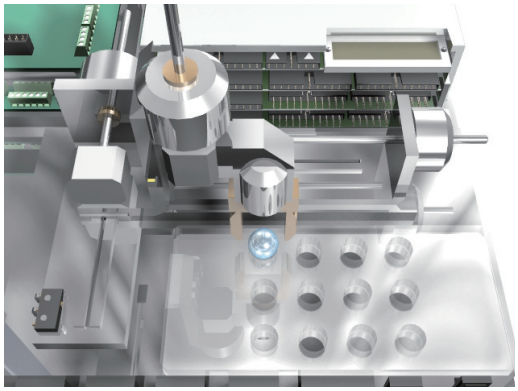
第 7 章 プログラム例

7-1 ロボットアプリケーション

ここでは、簡単なプログラム例で、基本的なプログラム記述を例示します。
想定は、以下のような 4 × 3 のマトリクス配置されたガラス玉をつかんで、原点で排出するものです。
こうしたアプリケーションでも PG の設定や原点復帰などが必要となります。



```
CONST   HAND 15
 CONST   READY 0
 CONST   BUSY 1
 CONST   START 192
 PG   1
 ACCEL   X_A|Y_A|SACL 50000
 ACCEL   Z_A|SACL 40000
 GOSUB   *HOME
 PALLET   1 P(1) P(2) P(3) P(4) 4 3

 DO
  ON   READY
  QUIT   1 : TIME   10
  OFF   BUSY
  WAIT   M_SW(START)
  OFF   READY
  FORK   1 *BUSY
  FOR   I=1 TO 12
   JUMP   PL(1;I)
   WAIT   RR(ALL_A)==0
   ON   HAND
   TIME   300
   JUMP   0 0 0 0
   OFF   HAND
   TIME   300
  NEXT
 LOOP
 END
*BUSY
DO
  ON   BUSY
  TIME   100
  OFF   BUSY
  TIME   100
```

I/O の定義
このように CONST を用いて、I/O をシンボル化しておくとプログラムの見通しが良くなる。

PG の選択。DSW1 の PG を選択
速度・加速度の設定を行う。
このように軸ごとに設定可。SACL は S 字指定
サブルーチンコール
パレット宣言。ここでは 4 点指定。

繰り返し制御文

スタートスイッチ待ち。チャタがあるときは M_SW

別タスクで、BUSY(点滅) 表示
順次処理、1 ～ 12 のワークを取り出す。

パルス発生 (移動) 終了の監視
ガラス玉をつかむ

排出位置に運ぶ
ガラス玉を放す

順次処理終端
繰り返し終端

別タスクのプログラム

LED 点滅を繰り返す単純なプログラム

```
 LOOP
*HOME                                  原点復帰プログラム
 STOP   ALL_A VOID
 RMVS   1000 1000 0 -1000              原点位置脱出
 WAIT   RR(ALL_A)==0
 SPEED   1000                          速度 1kPPS に設定
 STOP   ALL_A IN0_ON                   停止条件を指定する。( 原点センサ )
 RMVS   -100000 -100000 0 100000       原点サーチ
 WAIT   RR(ALL_A)==0                   停止したら終了
 STOP   ALL_A VOID                     停止条件を解除
 CLRPOS                                現在位置を 0 にする
 SPEED   30000                         速度をもとに戻す
 RETURN                                サブルーチンからもどる。
```

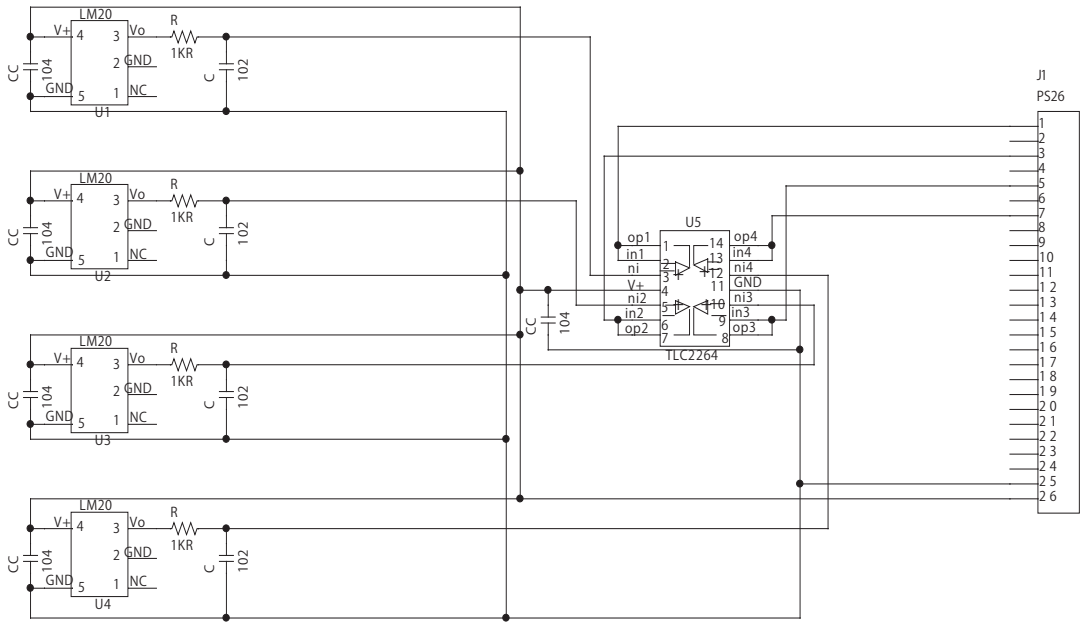
ここでは、プログラムの見通しをよくするために、いくつか条件を設定しています。
パワーオン時はほぼ原点の近くに XY があること。それにより、原点サーチにより停止すれば必ずそこが原点だという前提です。実際には、原点サーチ抜け後、センサ状態などの確認が必要です。

7-2 温度計測とデータロガー

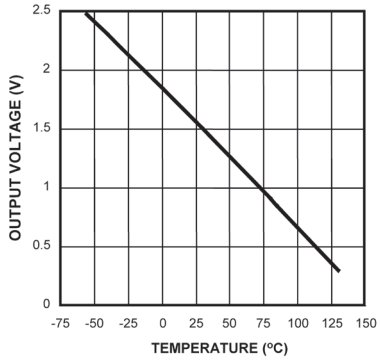
MPC-AD12 と温度センサ LM20 を組み合わせることにより、多点温度計測ロガーを構成できます。センサー部の回路図は以下のとおりです。OP アンプとセンサ IC の間は、1m 程度は引き伸ばすことができます。以下回路図で J1 は MPC-AD12 に接続、センサ部の電源は MPC-AD12 からの給電で動作します。

ここでは、カレンダ IC に同期して 10 秒ごとに温度データを計測し USB メモリに記録保存するプログラムを例示します。

【LM20 参考回路図】



Output Voltage vs Temperature



【温度データへの変換方法】

LM20 は左のような特性を持ちます。
0 度で 1.86V で、1 度ごとに -11.7mV 変化します。
MPC-AD12 は出荷状態で 1mV/1digit です。
このため、以下の演算で 3 桁の温度データを得ることができます。

A=(1866-AD(0))*100/117

A が 254 であれば、25.4 度ということになります。

```
 DIM   DATA(4)
 TIME   500
FILE$="log_S.txt"                          USB メモリのログファイルを指定。
err_cnt=0                                  ( 八文字 .TXT   DOS 形式 )
err$="  "
*ondo
 FORK   1 *disp_lcd                        LCD への表示タスクを起動
 ON_ERROR  *usb_err                        USB メモリ書き込みエラー規定
 DO                                        繰り返し
  WAIT   TIME(255)%16==0                   10 秒ごとののタイミング待ち
  FOR   I=0 TO 3                           4ch の繰り返し
DATA(I)=(1866-AD(1,I))*100/117             センサ電圧を温度に変換
  NEXT
DISP$=""
  FOR   I=0 TO 3                           カウント繰り返し CH0 ～ CH3
   FORMAT   " CH0_"
DISP$=DISP$+STR$(I)
   FORMAT   "00.0"
DISP$=DISP$+STR$(DATA(I))                  チャンネルごとに文字列に変換
  NEXT
  FORMAT   "000"                           LCD 用の文字列生成
OND1$=STR$(DATA(0))                        日付データ文字列生成
OND2$=STR$(DATA(1))
I$=DATE$(3)+" "+TIME$(1)+DISP$+"¥n"        日付、時間、温度データを合成
  ON   768                                 書き込みランプ点灯
  USB_WRITE   I$                           USB メモリに書き込む
  PRINT   I$                               FTM に表示
  TIME   1000
  OFF   768                                書き込みランプ消灯
 LOOP
*usb_err                                   ここは、USB メモリ書き込みエラーの処理
 RST_USB
 INC   err_cnt
 FORMAT   "00"
err$=STR$(err_cnt)                         エラーカウントを文字列に変換
 PRINT   "ERRORUSB"
 TIME   500
 RESUME                                    エラー発生コマンドにもどる。
 END
*disp_lcd
 DO
lcd$=err$+OND1$+OND2$                      MPC の LCD にエラー回数と
  PR_LCD   lcd$                            CH0,CH1 の温度を表示
  TIME   5000
 LOOP
```

以上のようにして簡単なデータロガーを構成することができます。
USB メモリに書き込まれるデータは以下のようになります。普通の PC で参照できます。

```
"log_S.txt"
2009-05-29 10:36:30 CH0_22.2 CH1_21.5 CH2_21.6 CH3_22.0
2009-05-29 10:36:40 CH0_22.2 CH1_21.5 CH2_21.6 CH3_22.1
2009-05-29 10:36:50 CH0_22.2 CH1_21.5 CH2_21.6 CH3_22.1
```

　なお、USB メモリはフラッシュメモリを内臓したメモリデバイスのため、10 万回程度の書き変えにしか耐えられません (メーカーによってバラつきあり)。1 秒に 1 回の書き込みを実行すると、24 時間で、86400 回の書き込みが発生し、メモリを劣化させます。
当社の試験では、1 秒間隔の書き込みで 1 週間ほど連続運転すると破損するものもありました。
このため、なるべく書き込みは MPC 内にバッファして USB_WRITE を実行する回数を減らすようにしてください。
　また、USB メモリは、書き込みエラーが発生する可能性のあるデバイスです。
稼動を停止させないために、適切なエラー処理 (ON_ERROR) を組み込んでください。

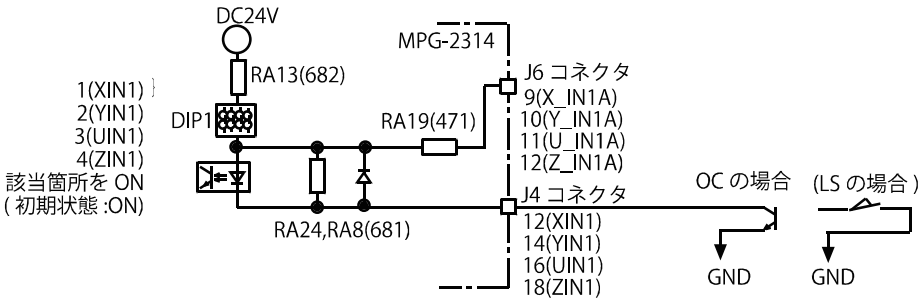
7-3 MPG-2314 サーボドライバ接続例

サーボドライバは各種入出力を備え、接続方法、名称などに相違があります。
ここでは、安川電機とパナソニックの代表的なサーボドライバと MPG-2314 の接続例を挙げ
それに対応した原点復帰方法を例示します。

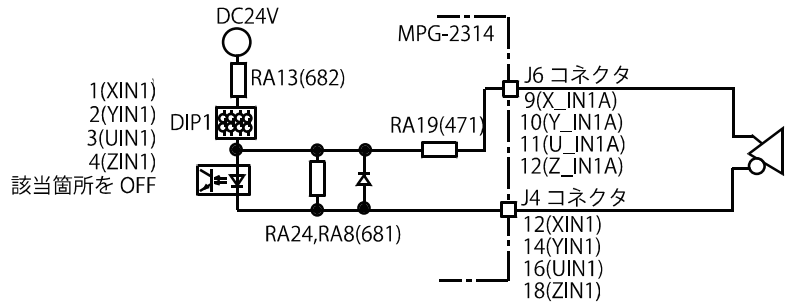
原点入力について

各軸の原点 IN1 入力はオープンコレクタ出力・接点、または差動出力を接続することが出来ます。
オープンコレクタ出力・接点の場合は DIP1 を ON(デフォルト) して J4 コネクタに接続します→ (図 1)。
差動入力は、DIP1 を OFF して、J6 コネクタと J4 コネクタに接続します→ (図 2)。
各軸 IN0 入力はオープンコレクタ出力または接点のみです→ (図 3)。

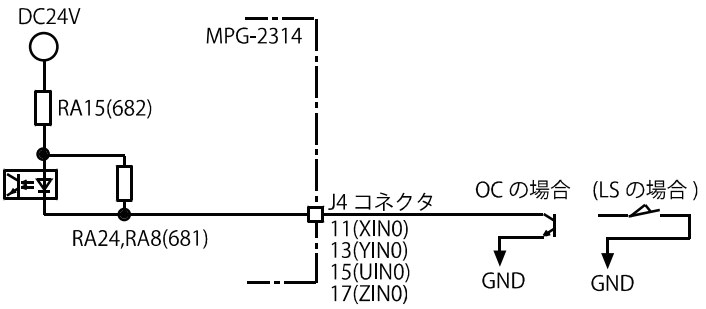
(図 1) オープンコレクタ出力、接点との接続



(図 2) 差動出力との接続

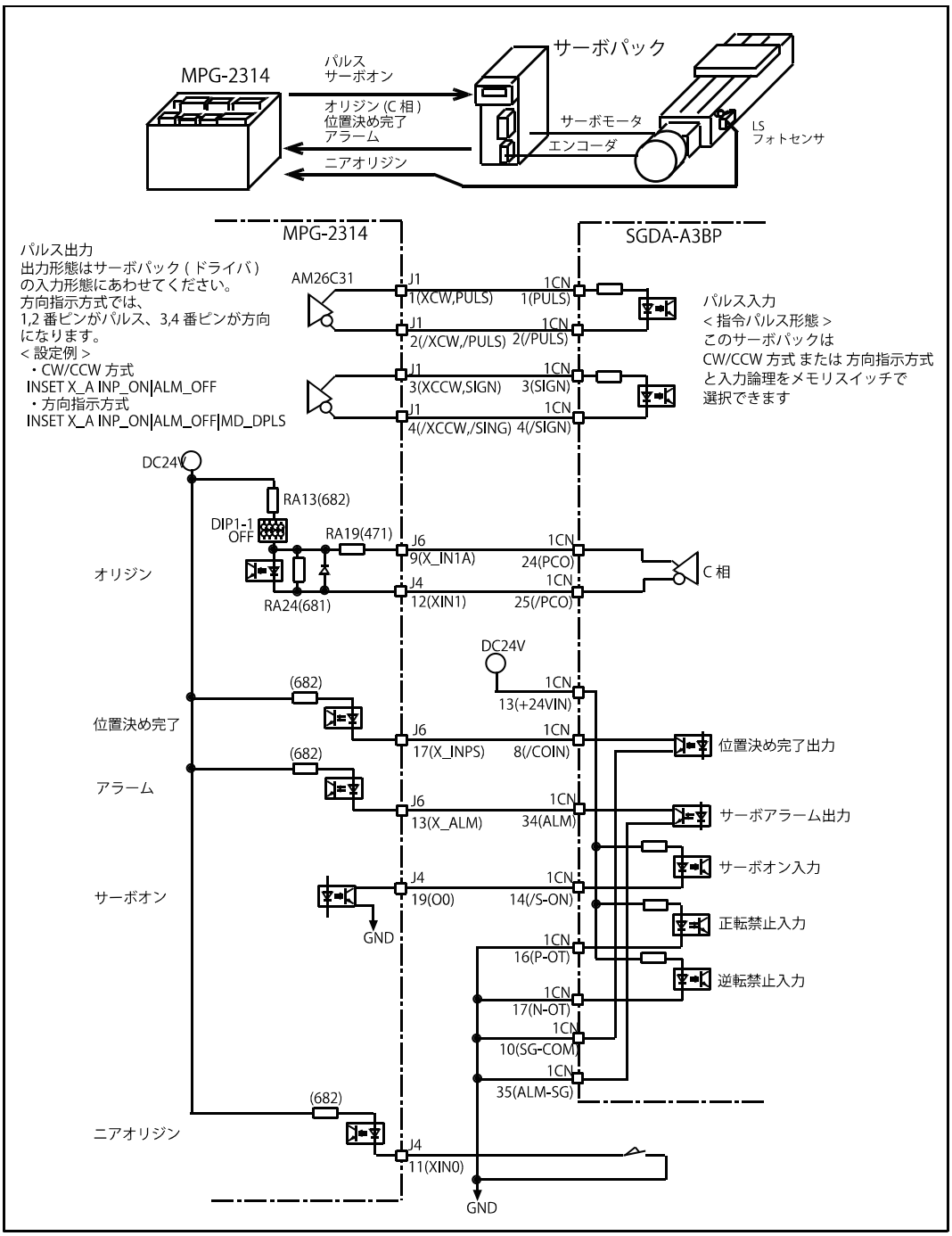


(図 3)IN0 の接続

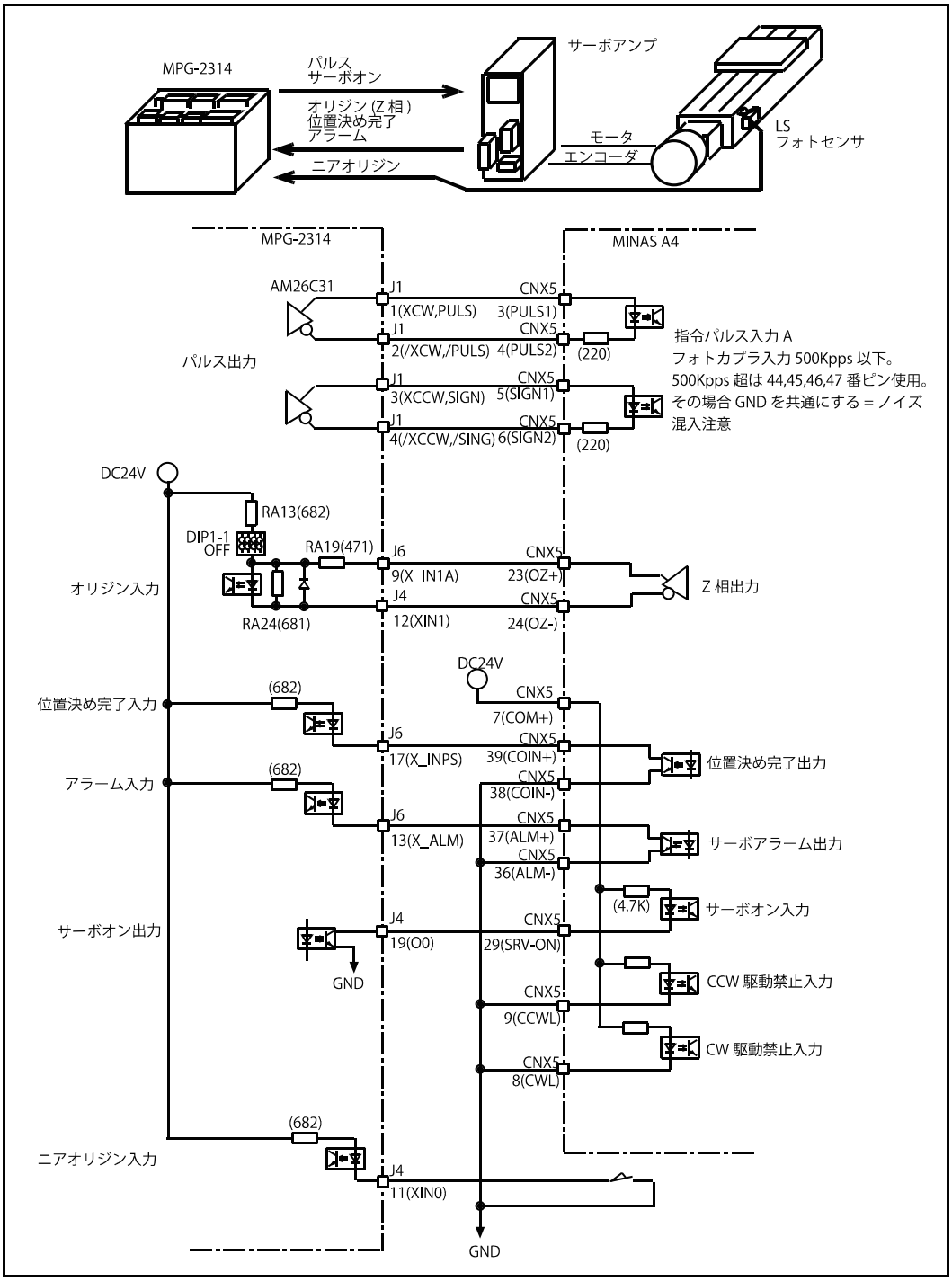


※ RA24,8 は 2 線式センサー対応用抵抗アレイ
※ RA19 は SIP ソケット実装。必要に応じて交換可能

接続例 (㈱安川電機 SGDA-A3BP)



接続例 (パナソニック㈱ MINAS A4 位置制御モード時)



プログラム例

```
/* START SW 押下 → サーボオン → ニアオリジンがオンしていれば CW 退避移動 →
/* やや速く CCW 方向へニアオリジン検出まで回転 → ゆっくり CCW へ C 相 (Z 相 ) 検出まで回転し座標クリア
/* その後ピッチ送り移動を繰り返す。
DO
  H_OFF 0                              /* サーボフリー
  PULSE_OUT 0 5                        /* START SW 点滅
  WAIT SW(192)==0
  WAIT SW(192)==1                      /* START SW 押下待ち
  PULSE_OUT VOID
  ON 0
  PG 1                                 /* MPG-2314 DSW=1
/* 入力とパルス形態の設定 : (a) は CW/CCW 方式、(b) は方向指示方式
/*   (a) 位置決め完了 =on で有効 | サーボアラーム =off で有効 ( パルス出力は CW/CCW 方式 )
  INSET X_A INP_ON|ALM_OFF
/*   (b) 位置決め完了 =on で有効 | サーボアラーム =off で有効 | パルス出力 = 方向指示方式
  'INSET X_A INP_ON|ALM_OFF|MD_DPLS
  H_ON 0                               /* サーボオン
  TIME 1000
  GOSUB *HOME_X
  ACCEL   X_A 50000 1000 1000
  FEED 100
  DO
    FOR I=1 TO 5
      RMVS X_A 10000
      WAIT RR(X_A)==0
      GOSUB *STOP_STATUS
      TIME 100
    NEXT I
    MOVS X_A 0
    WAIT RR(X_A)==0
    GOSUB *STOP_STATUS
    IF SW(192)==1 THEN
      BREAK
    END_IF
    TIME 1000
  LOOP
LOOP
*HOME_X                                /* X 軸原点復帰
  ACCEL X_A 500 100 100                /* 原点復帰スピード
  FEED 100
  IF HPT(XIN0)!=0 THEN                 /* X 軸 IN0 がオンなら退避移動
    RMVS X_A 1000
    WAIT RR(X_A)==0
    TIME 100
  END_IF
  SHOM   X_A IN0_ON|IN1_ON|CCW         /* ニアオリジン → CCW 方向に Z 相検出
  TMOUT 20000
  HOME -100000 0 0 0                   /* CCW 方向にニアオリジン検出
  IF timer_==0 THEN
    PRINT "TIME OUT"
    END
  END_IF
  STPS 0 VOID VOID VOID                /* X ここを '0' にセット
  PRINT "X HOME" X(0)
  TIME 1000
  RETURN
*STOP_STATUS                           /* 停止状態確認
  IF RR(X_E)<>0 THEN                   /* エラーステータスで停止状態判断
  /*IF LMT(X_A,ALM)==1 THEN            /* エラー入力で停止状態判断
    PRINT " 異常停止 "
    PRX RR(X_E)
    END
  END_IF
  RETURN
```