Panasonic AC Servo driver

RSI-ECAT-Master 動作確認報告書

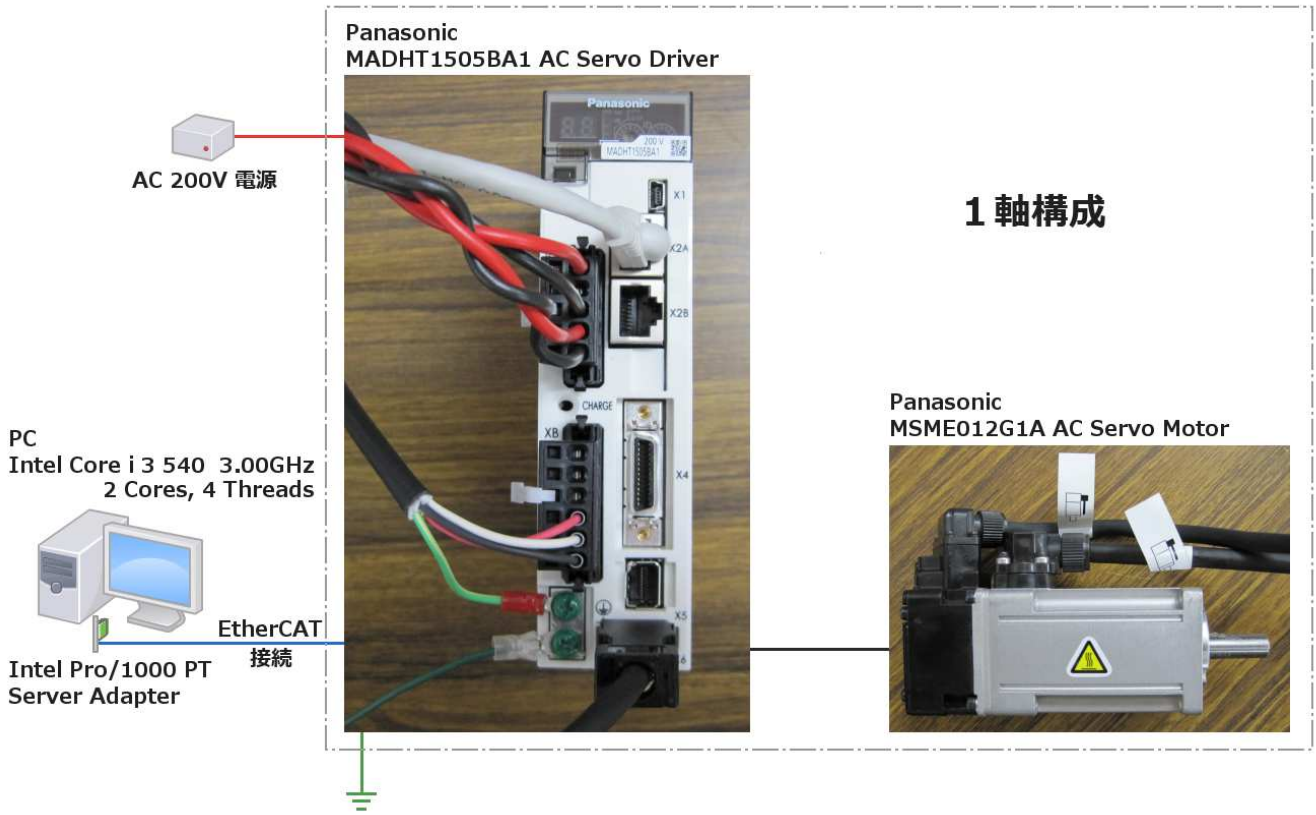
1. 動作結果

RSI-ECAT-Master において、Panasonic AC Servo driver が **正常に動作する事を確認しました。**

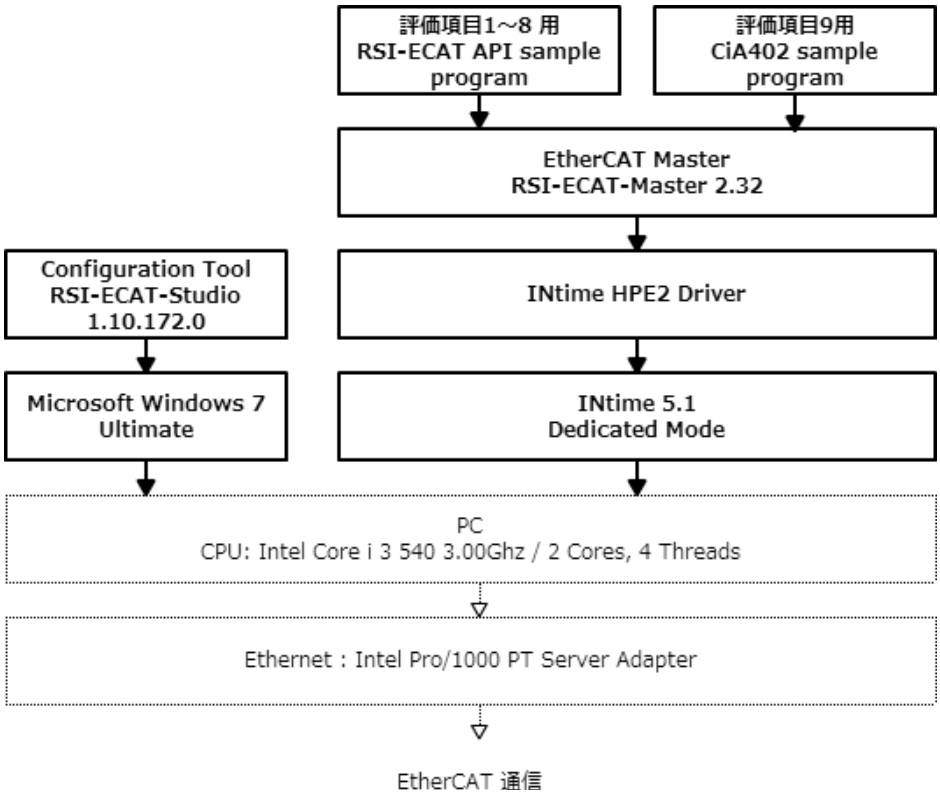
2. RSI-ECAT- Master INtime において、正常に動作すると判断した基準

- ☑ ＜評価項目１＞ スレーブとして認識でき、アタッチできる事。
- ☑ ＜評価項目２＞ CoE の Mailbox が利用でき、設定が変更できる事。
- ☑ ＜評価項目３＞ INIT から Operetional へ遷移できる事。
- ☑ ＜評価項目４＞ サーボドライバの初期化ができる事。
- ☑ ＜評価項目５＞ サーボモータの正転回転ができる事。
- ☑ ＜評価項目６＞ サーボモータの逆転回転ができる事。
- ☑ ＜評価項目７＞ サーボモータの停止ができる事。
- ☑ ＜評価項目８＞ サーボドライバの速度変更ができる事。
- ☑ ＜評価項目９＞ CiA402 汎用サンプルプログラムで一連の動作ができる事。

3. 機器構成図



4. ソフトウェア構成図



5. 環境構築手順

ネットワークカード Intel Pro/1000 PT Server Adapter を PC へ導入しました。

Panasonic サーボドライバおよびモータと、EtherCAT 接続する PC を Cat5 以上のネットワークケーブルで接続しました。

Panasonic サーボドライバ設定ツール PANATERM を使用するため、Panasonic サーボドライバと PC を USB ケーブルで接続しました。

配線が正しく接続されている事を確認後、PC および Panasonic サーボドライバの電源を投入しました。

今回の評価では、リミットセンサーを取り付けずに評価するため、**PANATERM ver.5.0** (*1)を起動し、メニューの、その他 より ピンアサイン設定 を開き、リミットセンサーの正転(07(SI2) POT_B 接)/逆転(08(SI3) NOT_B 接)の接点をデフォルトの B 接点から A 接点(POT_A 接/NOT_A 接)に変更しました。

(*1) **PANATERM ver 5.0** :

Panasonic サーボドライバの設定、モニタできる Windows ツール です。

http://industrial.panasonic.com/jp/i/25000/fa_minas_a5_panaterm/fa_minas_a5_panaterm.html

6. ＜評価項目１＞ スレーブとして認識でき、アタッチできる事。

① Panasonic 社より提供された ESI ファイルをスレーブライブラリフォルダ(※1)へ配置しました。

(※1 スレーブライブラリフォルダ : C:¥ProgramData¥KPA¥EtherCAT Studio¥slavelib¥)

② Panasonic サーボドライバ スレーブ検出評価の為に RSI-ECAT-Studio を起動しました。

③ RSI-ECAT-Studio の画面からスレーブへアタッチボタンを実行しました。

④ RSI-ECAT-Studio のスレーブ情報ツリーへ Panasonic サーボドライバが追加されたことを確認しました。

⇒ 結果 : Panasonic サーボドライバが検出された為、正常と判断しました。

7. ＜評価項目2＞ CoE の Mailbox が利用でき、設定が変更できる事。

① <評価項目 1> の状態から、RSI-ECAT-Studio の State タブより、マスター状態を INIT から Pre-Operetional へ移行させました。マスター状態の変化は、RSI-ECAT-Studio の State タブから確認しました。

② スレーブ情報ツリーの Panasonic サーボドライバを選択し、Mailbox タブにある CoE 機能を選択しました。

③ CoE により読み出された Object Dictionary の一覧から、次の項目を変更し設定が反映されるかを確認しました。

設定項目	OD Index 値	設定した値	結果
目標速度	6081h	2000000 (指令単位/s)	○
加速度	6083h	5000000 (指令単位/s)	○
減速度	6084h	2500000 (指令単位/s)	○

④ RSI-ECAT-Studio から値情報をリフレッシュし、それぞれ設定した値に変化しているかを確認しました。

⇒ 結果 ： 同画面にて値が更新されている事を確認した為、正常と判断しました。

8. ＜評価項目3＞ INIT から Operetional へ遷移できる事。

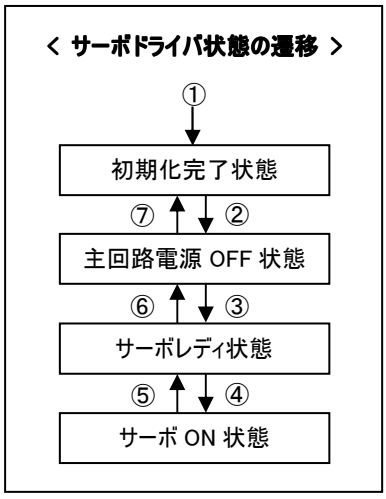
① <評価項目 2> の状態から、RSI-ECAT-Studio の State タブより、マスターステート状態を Pre-Operetional から Operetional へ移行要求をかけ、マスターステートおよびスレーブステートが、Operational へ移行するかを確認しました。

② マスターステート状態、スレーブステート状態の変化は、RSI-ECAT- Studio の State タブから確認しました。

⇒ 結果 ： マスターおよびスレーブ全てが、Operational へ移行された為、正常と判断しました。

9. ＜評価項目4＞ サーボドライバの初期化ができる事。

① <評価項目 3> のマスターステートおよびスレーブステートが Operational 状態から、サーボドライバ状態の遷移指令を要求しました。



② 初期化完了状態から主回路電源 OFF 状態に遷移要求をしました。

要求前状態	遷移要求コード	想定遷移状態	要求後状態	結果
初期化完了(70h)	Controlword : 6h	主回路電源 OFF(31h)	主回路電源 OFF(31h)	○

③ 主回路電源 OFF 状態からサーボレディ状態に遷移要求をしました。

要求前状態	遷移要求コード	想定遷移状態	要求後状態	結果
主回路電源 OFF(31h)	Controlword : 7h	サーボレディ(33h)	サーボレディ(33h)	○

④ サーボレディ状態からサーボ ON 状態に遷移要求をしました。

要求前状態	遷移要求コード	想定遷移状態	要求後状態	結果
サーボレディ(33h)	Controlword : Fh	サーボ ON(37h)	サーボ ON(37h)	○

⑤ サーボ ON 状態からサーボレディ状態に遷移要求をしました。

要求前状態	遷移要求コード	想定遷移状態	要求後状態	結果
サーボ ON(37h)	Controlword : 7h	サーボレディ(33h)	サーボレディ(33h)	○

⑥ サーボレディ状態から主回路電源 OFF 状態に遷移要求をしました。

要求前状態	遷移要求コード	想定遷移状態	要求後状態	結果
サーボレディ(33h)	Controlword : 6h	主回路電源 OFF(31h)	主回路電源 OFF(31h)	○

⑦ 主回路電源 OFF 状態から初期化完了状態に遷移要求をしました。

要求前状態	遷移要求コード	想定遷移状態	要求後状態	結果
主回路電源 OFF(31h)	Controlword : 0h	初期化完了(70h)	初期化完了(70h)	○

⇒ 結果 ： サーボドライバ状態がサーボ ON 状態まで遷移できトルクがかかっていることを確認し、その後、初期化完了状態に戻る事も成功した為、正常と判断しました。

10. ＜評価項目 5＞ サーボモータの正転回転ができる事。

① <評価項目 4> の状態から、PI の 制御モード: Mode of Operation を 1: PP 制御(Profile position mode) に設定しました。

② PI の 目標位置: Target Position に 位置情報 50000000 の値をセットしました。

③ PI の コントロールワード: Controlword を Fh から 1Fh に変更し、動作を開始しました。

④ PI の 現在位置: Position Actual Value を確認し、50000000 の近似値内に変化しているかを確認しました。

⇒ 結果 ： 正転回転し、PDI : Position Actual Value 値も近似値内の為、正常と判断しました。

11. ＜評価項目 6＞ サーボモータの停止ができる事。

① <評価項目 5> の状態から、PI の 目標位置: Target Position に 位置情報 100000000 の値をセットしました。

② PI の コントロールワード: Controlword を Fh → 1Fh に変更し、動作を開始しました。

③ その後直ぐに、PI の コントロールワード: Controlword を 1Bh に変更し、停止するかを確認しました。

⇒ 結果 ： 直ぐに停止した為、正常と判断しました。

12. ＜評価項目 7＞ サーボモータの逆転回転ができる事。

① <評価項目 6> の状態から、PI の 目標位置: Target Position に 位置情報 0 の値をセットしました。

② PI の コントロールワード: Controlword を Fh → 1Fh に変更し、動作を開始しました。

③ PI の 現在位置: Position Actual Value を確認し、0 の近似値内に変化しているかを確認しました。

⇒ 結果 ： 逆転回転し、PDI : Position Actual Value 値も近似値内の為、正常と判断しました。

13. ＜評価項目 8＞ サーボドライバの速度変更ができる事。

① ＜評価項目 7＞ の状態から、スレーブ情報ツリーの Panasonic サーボドライバを選択し、Mailbox タブにある CoE 機能を選択しました。

② 目標速度(6081h)を変更し、速度の変化をみました。

※ 目標位置の 50000000 へ移動後は、目標位置を 0 に戻して確認しました。

OD 名	インデックス	設定した値	動作確認内容	結果
目標速度	6081h	1000000	低速でモータ回転することを確認	○
目標速度	6081h	3000000	高速でモータ回転することを確認	○

⇒ 結果 ： 速度が変更されサーボモータが動作した為、正常と判断しました。

14. ＜評価項目 9＞ CiA402 汎用サンプルプログラムで一連の動作ができる事。

① CiA402 汎用サンプルプログラムを動作させました。

② EtherCAT 通信開始から始め、動作モード = 8 : CSP 制御 (Cyclic synchronous position mode) で実行しました。

③ 軸の回転～停止までの一連の動作の確認をしました。

⇒ 結果 ： 一連の動作が確認できた為、正常と判断しました。

＜評価項目 9＞で使用した CiA402 サンプルプログラムコード

```
/*****************************************************************************
* Description         : CiA402 ﾓｰｼｮﾝﾌﾟﾛｸﾞﾗﾑｻﾝﾌﾟﾙ          ｻｲｸﾘｯｸ同期位置ﾓｰﾄﾞ
*                     : (for Panasonic MADHT1507BA1)
* Software            : INtime 4.2 + RSI-ECAT 2.3.2
¥*****************************************************************************/
#include <stdio.h>
#include <string.h>
#include <rt.h>
#include <EhApi.h>

/*****************************************************************************
* Description : CiA402 ｻｰﾎﾞﾄﾞﾗｲﾊﾞ状態遷移処理
¥*****************************************************************************/
WORD    fnChgMotionGoState(WORD woStWd, WORD *pCrWd)
{
        switch( (0x6F & woStWd) ) {
        case    0: *pCrWd = (0xFF70 & *pCrWd) | 0;    break; // 未初期化状態        -> 初期化要求
        case 0x40: /* (0x60 と同じ) */
        case 0x60: *pCrWd = (0xFF70 & *pCrWd) | 0x06; break; // 初期化完了状態      -> 主回路電源 OFF 要求
        case 0x21: *pCrWd = (0xFF70 & *pCrWd) | 0x07; break; // 主回路電源 OFF 状態  -> ｻｰﾎﾞﾚﾃﾞｨ要求
        case 0x23: *pCrWd = (0xFF70 & *pCrWd) | 0x0F; break; // ｻｰﾎﾞﾚﾃﾞｨ状態        -> ｻｰﾎﾞ ON 要求
        case 0x27:                                    break; // ｻｰﾎﾞ ON 状態 (運転開始 OK 状態)
        case 0x2F:                                    break; // 異常処理動作中状態
        case 0x28: *pCrWd = (0xFF70 & *pCrWd) | 0x80; break; // 異常状態            -> 初期化要求
        }
        return (0x6F & woStWd);
}

/*****************************************************************************
* Description : main 処理
¥*****************************************************************************/
void    main(int argc, char* argv[])
{
        WORD                    woStatus;
        WORD                    woReqState = 0;
        WORD                    woNowState = 0;
        EHHANDLE                hAPI;
        BYTE                    byValue;
        WORD                    woStatasWord;
        WORD                    woControlWord;
        DWORD                   dwTargetPos = 0;
        SLAVE_DETAIL            stSlave;
```

```
BYTE                    bySize;


// RSI-ECAT 起動状態ﾁｪｯｸ (RSI-ECAT API ｺｰﾙ)
woStatus = EhGetEhNodeStatus("NodeA", 100);
printf("EhGetEhNodeStatus() Status = %04xH¥n", woStatus);
if( 0 != woStatus ) {
        printf("start check failed¥n");
        return ;
}

// RSI-ECAT API 対話ﾊﾝﾄﾞﾙ OPEN (RSI-ECAT API ｺｰﾙ)
hAPI = EhOpen("NodeA", &woStatus);
if( 0 != woStatus ) {
        printf("EH Handle Open failed¥n");
        return ;
}

// Master ｽﾃｰﾄﾁｪﾝｼﾞ要求 (RSI-ECAT API ｺｰﾙ)
// MST_OPERATIONAL : ｵﾍﾟﾚｰｼｮﾅﾙへ遷移
woStatus = EhRqState(hAPI, MST_OPERATIONAL);
if( 0 != woStatus ) {
        printf("EhRqState() failed. Status = %04xH¥n", woStatus);
        return ;
}

// ｵﾍﾟﾚｰｼｮﾅﾙへ遷移したかどうかの確認処理
while(1) {
        // 現在の Master ｽﾃｰﾄ状態取得 (RSI-ECAT API ｺｰﾙ)
        woStatus = EhGetState(hAPI, &woReqState, &woNowState);

        // ｵﾍﾟﾚｰｼｮﾅﾙへ遷移完了かをﾁｪｯｸ
        if( woNowState == MST_OPERATIONAL ) {
                break; // 遷移完了の為ﾌﾞﾚｲｸ
        }

        // ｽﾃｰﾄﾁｪﾝｼﾞ待機用ｽﾘｰﾌﾟ
        RtSleep(500);
}

// ｽﾚｰﾌﾞ検索 : Panasonic Servo Driver 用の VendorID, ProductCode, RevisionNo をｾｯﾄ
memset(&stSlave, 0, sizeof(stSlave));
stSlave.dwSize        = sizeof(stSlave);
   stSlave.dwVendorID    = 0x066F;
   stSlave.dwProductCode = 0x515070a1;
   stSlave.dwRevisionNo  = 0x00010000;
woStatus = EhFindSlave(hAPI, &stSlave);
if( 0 != woStatus ) {
        printf("EhFindSlave() failed. Status = %04xH¥n", woStatus);
        return ;
}

// CiA402 動作ﾓｰﾄﾞの変更
// 8 : CSP 制御 (Cyclic synchronous position mode) ｻｲｸﾘｯｸ同期位置ﾓｰﾄﾞ へ変更要求
byValue = 8;
   bySize = 1;
woStatus = EhWriteByte(hAPI, stSlave.dwViosOutBaseOffset + 2, byValue);
 if( 0 != woStatus ) {
        printf("EhWriteByte() failed. Status = %04xH¥n", woStatus);
        return ;
}

// CiA402 ｻｰﾎﾞﾄﾞﾗｲﾊﾞ状態を、ｻｰﾎﾞ ON に遷移
while(1) {
        woControlWord = 0;

        // ｻｲｸﾘｯｸ処理待ち
        woStatus = EhWaitForCyclic(hAPI, WAIT_FOREVER);
        if( 0 != woStatus ) {
                break;  // ｻｲｸﾘｯｸ処理が中断された為、終了
        }

        // 現在のｻｰﾎﾞﾄﾞﾗｲﾊﾞ状態取得 (RSI-ECAT API ｺｰﾙ)
```

```
                woStatasWord = EhReadWord(hAPI, stSlave.dwViosInBaseOffset + 2, &woStatus);


                if( 0 != woStatus ) {
                        printf("EhReadWord() failed. Status = %04xH¥n", woStatus);
                        return ;
                }

                // 現在のｻｰﾎﾞﾄﾞﾗｲﾊﾞ状態から、次のｻｰﾎﾞﾄﾞﾗｲﾊﾞ状態へ遷移させる関数ｺｰﾙ
                woStatus = fnChgMotionGoState(woStatasWord, &woControlWord);
                if( woStatus == 0x27 ) {
                        break;
                }

                // 次のｻｰﾎﾞﾄﾞﾗｲﾊﾞ遷移値を書き込み (RSI-ECAT API ｺｰﾙ)
                woStatus = EhWriteWord(hAPI, stSlave.dwViosOutBaseOffset, woControlWord);
                if( 0 != woStatus ) {
                        printf("EhWriteWord() failed. Status = %04xH¥n", woStatus);
                        return ;
                }
        }

        // ﾓｰﾀｰ動作処理
        while(1) {
                // ｻｲｸﾘｯｸ処理待ち
                woStatus = EhWaitForCyclic(hAPI, WAIT_FOREVER);
                if( 0 != woStatus ) {
                        break;  // ｻｲｸﾘｯｸ処理が中断された為、終了
                }

                // ﾙｰﾌﾟ毎に 1000 分移動要求
                EhWriteDword(hAPI, stSlave.dwViosOutBaseOffset + 3, dwTargetPos);

                dwTargetPos += 1000;

                if(dwTargetPos > 1000000)       {
                        break;
                }
        }
        printf("program end.");
}
```