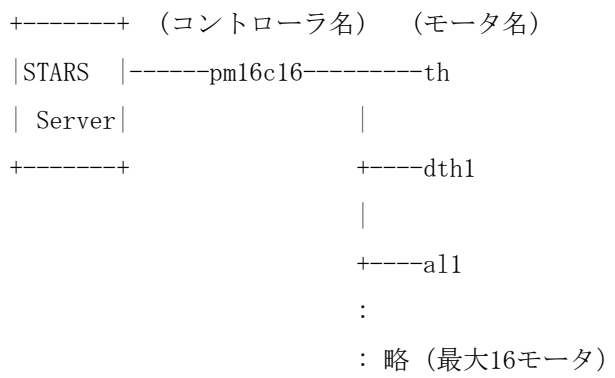
STARS pm16c16 用コマンド集

2022.4.12 版

はじめに

ツジ電子製 PM16C-16 シリーズは 16 台のパルスモータの駆動可能なパルスモータコントローラです。

PM16C-16 STARS I/O Client の作りはコントローラの先にモータが接続されているような形をイメージして、各モータへの命令は、送り先を"pm16c16.dth1"のように階層化した形式にします。



メッセージフォーマット

STARS 経由でパルスモータ PM16C-16（ツジ電子製）を制御する場合は、下記フォーマットのメッセージを送信して行います。

メッセージ配信先名□コマンド□引数(必要な場合のみ)

※□は半角スペースを意味しています

例） pm16c16.th□GetValue

メッセージを送った場合は返事（リプライメッセージを含んだ文字列）が返ってきます。

(メッセージ配信先)>(メッセージ送信元)□@コマンド□引数□値

例） pm16c16.th>term1□@GetValue□10000

対応しているメッセージ配信先名は下記の通りです。

メッセージの配信先について

[メッセージ配信先名]
Stars のノード名が pm16c16 の場合（STARS のノード名はプログラム起動時のオプションで指定することができます）

pm16c16	コントローラコマンド パルスモータコントローラに対してメッセージを配信します
pm16c16.*motorname*	モータコマンド motorname という名前のモータに対してメッセージを配信します

エラーメッセージについて

エラーが起こると返事（リプライメッセージ）として下記の形式の文字列が返ってきます。

(メッセージ配信先)>(メッセージ送信元)□@コマンド□引数□Er:□（エラー内容を表す文字列）

例） pm16c16.th>term1□@Preset□1000000000□Er:□Busy.

メッセージ配信先を誤って送った場合は下記のエラーを含んだ文字列が返ってきます。

(pm16c16 のノード名)>(メッセージ送信元)□@コマンド□引数□Er:□（誤って送ったメッセージ配信先名）□is□down.

例） pm16c16.thet□GetValue
pm16c16>term1□@GetValue□Er:□pm16c16.thet□is□down.

用意されていないコマンドもしくは適切でない引数を含んだメッセージを送った場合は下記のエラーを含んだ文字列が返ってきます。

(メッセージ配信先)>(メッセージ送信元)□@コマンド□引数□Er:□Bad□command□or□parameters.

例） pm16c16.th□GetValu
pm16c16>term1□@GetValu□Er:□Bad□command□or□parameters.

プログラムの実行について

[プログラムの起動]

プログラムファイル pypmc.py があるフォルダで、以下のコマンドを実行します。

＜Python 実行ファイル名＞□pypmc.py[□プログラム起動オプション]

プログラム起動オプションで、-h を指定するとプログラム起動オプションが確認できます。

```
$ python pypmc.py -h
usage: pypmc [-h] [--version] [-d] [--debuglevel DEBUGLEVELNUM] [--logenable]
             [--logdir LOGDIR] [--loglevel LOGLEVELNUM]
             [--nodename STARSNODENAME] [--serverhost STARSSERVERHOST]
             [--serverport STARSSERVERPORT] [--devicehost DEVICEHOST]
             [--deviceport DEVICEPORT] [--rawenable] [--config CONFIG]
             [--channelnamelist CHANNELNAMELIST]
             [--limitstatuschannellist LIMITSTATUSCHANNELLIST]
```

[主なプログラム起動オプション]

--nodename *STARSNODENAME*	*STARS ノード名　デフォルト* pm16c16
--serverhost *STARSSERVERHOST*	*STARS ホスト名 or IP アドレス　デフォルト* localhost
--devicehost *DEVICEHOST*	*制御対象とする PM16C-16 のホスト名 or IP アドレス*
--channelnamelist *CHANNELNAMELIST*	*モータの名前リスト* モータ番号 0 から順に最大 15 までのモータ名を(カンマ)区切りで指定する 指定がないモータは、モータ名を自動生成する
--limitstatuschannellist *CHANNELNAMELIST*	*モータ番号あるいはモータの名前リスト* モータイベント_ChangedLimitStatus の送信を有効にするモータ名あるいはモータ番号を,(カンマ)区切りで指定する *（アスタリスク）指定で全モータが対象となる
--rawemanle	コントローラコマンド SendRawCommand を使ってツジ電子デバイスコマンドの送受信を行えるようにする
--config *CONFIG*	*プログラム設定ファイル名　デフォルト*./config.cfg プログラム設定ファイルで指定されたプログラムパラメータが、プログラム起動オプションで同時に指定された場合は、プログラム起動オプションの値が優先する

[プログラム設定ファイルの記述例]

STARS ノード単位でプログラムパラメータをファイルに記述する

・#で始まる行はコメント

・[STARS ノード名] ⇒ STARS ノード名のパラメータの記述を開始する

・複数の STARS ノード名の記述が可能

プログラム起動時、プログラム起動オプション--nodename STARSNODENAME で指定された STARS ノード名（デフォルト pm16c16）のプログラムパラメータが読み込まれます

```
[pm16c16]
# Setting of STARS NodeName pm16c16
StarsServerHost=localhost
DeviceHost=192.168.1.55
#DevicePort=7777
ChannelNameList=th,dth1,d1,al1,Mt4,d2,al2,Mt7,Mt8,Mt9,Mta,Mtb,Mtc,Mtd,Mte,Mtf
#LimitStatusChannelList=*
#PM16C04Compatible=True
#AllReplyEnable=True
RawEnable=True
#Debug=True
[pm16c16_2]
# Setting of STARS NodeName pm16c16_2
…
```

コントローラコマンド

[メッセージ配信先名]

コントローラコマンドを送信する場合は、配信先名として STARS のノード名を指定します。
以下のコントローラコマンドの送信例は、配信先となる STARS のノード名が pm16c16、コマンドの送信元のノード名が term1 である場合を示しています。

[コマンド]

hello

STARS の通信が行われているかをチェックするコマンド。
このコマンドを送信すると'@hello Nice to meet you.'の文字列を返します。
[例]
（送信側）
pm16c16□hello
（返信されてくる文字列）
pm16c16>term1□@hello□Nice□to□meet□you.

help

引数を指定しない場合はコントローラコマンドの一覧を返します。
引数にコマンド名を指定した場合はコマンドのヘルプを返します。
[例]
（送信側）
pm16c16□help　　コントローラコマンドの一覧を表示します
（返信されてくる文字列）
pm16c16>term1□@help□GetAccRate□GetAccRateCode□GetAccRateList□GetAllReplyEnable□GetAutoChangeSpeed□GetAutoChangeSpeedReady□GetCancelBacklash□...(略)...□getversion□getversionno□hello□help□listnodes□terminate

（送信側）
pm16c16□help□hello　　hello コマンドのヘルプを表示します
（返信されてくる文字列）
pm16c16>term1□@help□hello□Return□'hello□Nice□to□meet you.'

（送信側）
pm16c16 help helo　　引数として該当しないコマンドを指定した場合
pm16c16>term1 @help helo Er: Command "helo" not found.

getversion

このコマンドを送信すると当 STARS Client プログラムのバージョン情報を返します。

[例]

（送信側）

pm16c16□getversion

（返信されてくる文字列）

pm16c16>term1□getversion□pypmc□0.1,2022-04-12,Yasuko Nagatani

getversionno

このコマンドを送信すると当 STARS Client プログラムのバージョン No.を返します。

[例]

（送信側）

pm16c16□getversionno

（返信されてくる文字列）

pm16c16>term1□getversionno 0.1

◆モータ名に関するコマンド

モータ名のセットアップは、目次「プログラム実行について」を確認してください。

GetMotorList

このコマンドを送信するとモータ名称の一覧をスペース区切りで返します。

[リプライ・メッセージ]

@GetMotorList□（モータ No.0 の名称）□（モータ No.1 の名称）□... (略)...□（モータ No.15 の名称）

[例]

（送信側）

pm16c16□GetMotorList

（返信されてくる文字列）

pm16c16>term1□@GetMotorList□th□dth1□d1□al1□Mt4□d2□al2□Mt7□Mt8□Mt9□Mta□Mtb□Mtc□Mtd□Mte□Mtf

左から順にモータ No.0 の名称'th'、No.1 の名称'dth1'…以降 No.15 の名称’Mtf’を返します

GetMotorName

モータ No.に対応するモータ名を返します。

[引数]

モータ No.	0 から 15 の数字文字列

[リプライ・メッセージ]

@GetMotorName□（引数のモータ No.）□（モータ名）	データ取得が正常におこなわれた場合
@GetMotorName□Er:□Bad□command□or□parameters.	引数のモータ No.が指定されなくてエラーの

場合

@GetMotorName□（引数）□Er:□Bad□parameters. モータ No.が間違っていてエラーの場合

[例]

（送信側）

pm16c16□GetMotorName□0 モータ No.0 のモータ名を問い合わせます

（返信されてくる文字列）

pm16c16>term1□@GetMotorName□0□th モータ No.0 のモータ名'th'が返ってきます

◆制御デバイスに関する情報

GetRomVersion

このコマンドを送信すると、制御対象 PM16C-16 デバイスのシステムプログラムのバージョンと日付、そして機種名を返します。

[例]

（送信側）

pm16c16□GetRomVersion

（返信されてくる文字列）

pm16c16>term1□GetRomVersion□1.13□17-12-13□PM16C-16

GetFirmwareVersion

このコマンドを送信すると、制御対象 PM16C-16 デバイスのシステムプログラム情報としてデバイスの VER?コマンドが返した値をそのまま返します。

[例]

（送信側）

pm16c16□GetFirmwareVersion

（返信されてくる文字列）

pm16c16>term1□@GetRomVersion□1.13□17-12-13□PM16C-16

GetHardwareVersion

このコマンドを送信すると、制御対象 PM16C-16 デバイスのハードウェア情報としてデバイスの VERH?コマンドが返した値をそのまま返します。

[例]

（送信側）

pm16c16□GetHardwareVersion

（返信されてくる文字列）

pm16c16>term1□@GetRomVersion□HD-VER□9

◆コントローラ Remote／Local 操作モードコマンド

GetFunction

このコマンドを送信することで、コントローラの Remote それとも Local 操作モードどちらにあるかを取得します。

[リプライ・メッセージ]

@GetFunction□0	Local モード
@GetFunction□1	Remote モード

[例]

（送信側）

pm16c16□GetFunction	Remote/Local モードの状態を確認します

（返信されてくる文字列）

pm16c16>term1□@GetFunction□0	Local 操作モードの場合
pm16c16>term1□@GetFunction□1	Remote 操作モードの場合

Remote

このコマンドを送信することでコントローラを Remote 操作モードに設定します。

[リプライ・メッセージ]

@Remote□Ok:	正常に動作を終了した場合
@Remote□Er:□(エラー情報)	なんらかの理由で Remote に変更できなかった場合

[例]

（送信側）

pm16c16□Remote	Remote 操作モードに設定します
pm16c16□GetFunction	Remote/Local 操作モードの状態を確認します

（返信されてくる文字列）

pm16c16>term1□@Remote□Ok:	
pm16c16>term1□@GetFunction□1	Remote 操作モードです

Local

このコマンドを送信することでコントローラを Local 操作が可能な状態に設定します。

[リプライ・メッセージ]

@Local□Ok:	正常に動作を終了した場合
@Local□Er:□(エラー情報)	なんらかの理由で Local に変更できなかった場合

[例]

（送信側）

pm16c16□Local	Local 操作モード状態に設定します
pm16c16□GetFunction	Remote/Local 操作モードの状態を確認します

（返信されてくる文字列）

pm16c16>term1□@Remote□Ok:	
pm16c16>term1□@GetFunction□0	Local 操作モードです

SetFunction

このコマンドを送信することで、コントローラを Remote 操作それとも Local 操作モードいずれかに設定します。

それぞれ Remote コマンドもしくは Local コマンドと同一の機能です。

[引数]

0	Local
1	Remote

[リプライ・メッセージ]

@SetFunction□（引数）□Ok:	正常に動作を終了した場合
@SetFunction□（引数）□Er:□(エラー情報)	なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□SetFunction□1	Remote 操作モード状態に設定します

（返信されてくる文字列）

pm16c16>term1□@SetFunction□1□Ok:

pm16c16>term1□@GetFunction□1	Remote 操作モードです

◆タイミング信号出力ポートへのモータ No.の割当／本体フロントパネルディスプレイに表示するモータ No.の設定に関するコマンド

PM16C-16 シリーズのタイミング出力機能では、タイミング信号出力ポート TP0 から TP3 の４ポートに対しモータ No を割り当てて使用します。

また、本体のフロントパネルのディスプレイでは Pos.A から Pos.D までの４つのモータの情報が表示されます。

PM16C-04X シリーズでは、タイミング出力ポート TP0 から TP3 のモータ No.は、本体フロントパネルディスプレイ Pos.A から Pos.D の順に表示されるモータ No.と一致していますが、PM16C-16 の本体のファームウェアバージョン 1.01 以降、両者のモータ No.を一致させるあるいはさせない設定が可能となりました。

GetTimingOutChannelFixEnable

このコマンドを送信することで、タイミング信号出力ポートと本体フロントパネルディスプレイに表示するモータ No.の設定が連動する／しないの選択状態を取得します。

[リプライ・メッセージ]

@GetTimingOutChannelFixEnable□0	連動する
@GetTimingOutChannelFixEnable□1	連動しない

[例]

（送信側）

pm16c16□GetTimingOutChannelFixEnable	連動する／しない状態を確認します

（返信されてくる文字列）

pm16c16>term1□@GetTimingOutChannelFixEnable□1	連動しない場合

pm16c16>term1□@GetTimingOutChannelFixEnable□0　　連動する場合

GetTimingOutChannel

このコマンドを送信することで、タイミング信号出力ポート TP0 から TP3 に設定されている 4 つのモータ No.の情報を取得します。

当コマンド以外に、モータコマンド「GetSelected」を用いて指定したモータに関連付けられているタイミング信号出力ポートを確認することができます。

[リプライ・メッセージ]

@GetTimingOutChannel□(モータ No.情報)　　モータ No.情報は長さ 4 の文字列で、左から各 1 文字ずつ TP0,TP1,TP2,TP3 に相当し、モータ No.が 16 進数表記されます

[例]

（送信側）

pm16c16□GetTimingOutChannel

（返信されてくる文字列）

pm16c16>term1□@GetTimingOutChannel□0123　　TP0,TP1,TP2,TP3 の順にモータ 0,1,2,3 が選択されている場合

GetDispChannel

このコマンドを送信することで、本体のフロントパネルのディスプレイの Pos.A から Pos.D に設定されている 4 つのモータ No.の情報を取得します。

[リプライ・メッセージ]

@GetDispChannel□(モータ No.情報)　　モータ No.情報は長さ 4 の文字列で、左から各 1 文字づつ Pos.A,Pos.B,Pos.C, Pos.D に相当し、モータ No.が 16 進数表記されます

[例]

（送信側）

pm16c16□GetDispChannel

（返信されてくる文字列）

pm16c16>term1□@GetDispChannel□0123　　Pos.A,Pos.B,Pos.C, Pos.D の順にモータ 0,1,2,3 が選択されている場合

SetTimingOutChannelFixEnable

このコマンドを送信することで、タイミング信号出力ポートと本体フロントパネルディスプレイに表示するモータ No.が連動するしないを設定します。

[リプライ・メッセージ]

@SetTimingOutChannelFixEnable□（引数）□Ok:　　正常に動作した場合

@SetTimingOutChannelFixEnable□（引数）□Er: □(エラー情報).　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□SetTimingOutChannelFixEnable□0　　連動する状態に設定します

（返信されてくる文字列）

pm16c16>term1□@SetTimingOutChannelFixEnable□0□Ok:

pm16c16>term1□@ GetTimingOutChannelFixEnable□0　　連動する状態です

Select

このコマンドを送信するとタイミング信号出力ポート TP0 から TP3 に対して関連付けるモータ No. を割り当てることができます。

また、モータコマンド「GetSelected」を用いて指定したモータに関連付けられているタイミング信号出力ポートを確認することができます。

≪パターン１≫

[引数]

１．出力ポート　　A、B、C、D のいずれかの文字
(A から D：TP0 から TP3 に相当する)

２．モータ No.　　0 から 15 の数字文字列

[リプライ・メッセージ]

@Select□（引数の出力ポート）□（引数のモータ No.）□Ok: 正常に動作した場合

@Select□（引数の出力ポート）□（引数のモータ No.）□Er:□(エラー情報)
なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□Select□A□0　　TP0 にモータ 0 を割当

pm16c16.th□GetSelected　　モータ 0 の割当出力ポートを確認します
（モータ 0 のモータ名が th の場合）

（返信されてくる文字列）

pm16c16>term1□@Select□A□0□Ok

pm16c16.th>term1□@GetSelected□A

≪パターン２≫

下記、コントローラコマンド「SetTimingOutChannel）を「Select」に置き換えて使用することができます。

SetTimingOutChannel

このコマンドを送信するとタイミング信号出力ポート TP0 から TP3 に対して関連付けるモータ No. の情報を割り当てることができます。

[引数]

１．(モータ No.情報)　　長さ 4 の文字列、左から各 1 文字づつ TP0,TP1,TP2,TP3 に相当します。

各 1 文字には、設定するモータ No. を 16 進数表記で指定するかもしくは自動選択させる場合は-(ハイフン)を指定します

[リプライ・メッセージ]

@SetTimingOutChannel□(モータ No.情報) □Ok: 正常に動作した場合

@SetTimingOutChannel□(モータ No.情報) □Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□SetTimingOutChannel□1A23 TP0 にモータ 1、TP2 にモータ 10、TP3 にモータ 2 と TP4 にモータ 3 を割当

（返信されてくる文字列）

pm16c16>term1□@SetTimingOutChannel□1A23□Ok:

SetDispChannel

このコマンドを送信すると本体のフロントパネルのディスプレイの Pos.A から Pos.D に対して関連付けるモータ No.の情報を割り当てることができます。

[引数]

1．(モータ No.情報) 長さ 4 の文字列､左から各 1 文字づつ Pos.A,Pos.B,Pos.C, Pos.D に相当します。
各 1 文字には、設定するモータ No. を 16 進数表記で指定するかもしくは自動選択させる場合は-(ハイフン)を指定します

[リプライ・メッセージ]

@SetDispChannel□(モータ No.情報) □Ok: 正常に動作した場合

@SetDispChannel□(モータ No.情報) □Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□SetDispChannel□1A23 Pos.A にモータ 1、Pos.B にモータ 10、Pos.C にモータ 2 と Pos.D にモータ 3 を割当

（返信されてくる文字列）

pm16c16>term1□@SetDispChannel□1A23□Ok:

◆速度および加減速レート表確認コマンド

GetAccRateList

このコマンドを送信すると有効なモータの速度加減速レートの一覧を返します。

[リプライ・メッセージ]

@GetAccRateList□（有効なモータ速度の加減速率 1）□（有効なモータ速度の加減速率 2）

…□（有効なモータ速度の加減速率 N）

[例]

（送信側）

pm16c16□GetAccRateList

（返信されてくる文字列）

pm16c16>term1□@GetAccRateList
□1000□910□820□750□680□620□560□510□470□430□390□360□330□300□270□240□220□200□18
0□160□150□130□120□110□100□91□82□75□68□62□56□51□47□43□39□36□33□30□27□24□22□2
0□18□16□15□13□12□11□10□9.1□8.2□7.5□6.8□6.2□5.6□5.1□4.7□4.3□3.9□3.6□3.3□3.0□2.7□2.4
□2.2□2.0□1.8□1.6□1.5□1.3□1.2□1.1□1.0□0.91□0.82□0.75□0.68□0.62□0.56□0.51□0.47□0.43□0.3
9□0.36□0.33□0.3□0.27□0.24□0.22□0.2□0.18□0.16□0.15□0.13□0.12□0.11□0.1□0.091□0.082□0.07
5□0.068□0.062□0.056□0.051□0.047□0.043□0.039□0.036□0.033□0.03□0.027□0.024□0.022□0.02□
0.018□0.016

◆全モータ速度レベル一括設定コマンド

全てのモータの速度レベル(High/Middle/Low)を一括で設定します。

SpeedHigh

このコマンドを送信することで、以降全てのモータの速度が’High’の設定で動きます。

[リプライ・メッセージ]

@SpeedHigh□Ok:	正常に動作した場合
@SpeedHigh□Er:□(エラー情報)	なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□SpeedHigh

（返信されてくる文字列）

pm16c16>term1□@SpeedHigh□Ok:

SpeedMiddle

このコマンドを送信することで、以降全てのモータの速度が'Middle’の設定で動きます。

[リプライ・メッセージ]

@SpeedMiddle□Ok:	正常に動作した場合
@SpeedMiddle□Er:□(エラー情報)	なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□SpeedMiddle

（返信されてくる文字列）

pm16c16>term1□@SpeedMiddle□Ok:

SpeedLow

このコマンドを送信することで、以降全てのモータの速度が'Low’の設定で動きます。

[リプライ・メッセージ]

@SpeedLow□Ok: 正常に動作した場合

@SpeedLow□Er: □(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16□SpeedLow

（返信されてくる文字列）

pm16c16>term1□@SpeedLow□Ok:

◆全モーター括停止コマンド

Stop

このコマンドを送信すると全てのモータが減速停止します。

このコマンドはコントローラが Local 操作モードの場合無視されます。

[リプライ・メッセージ]

@Stop□Ok: 正常に動作した場合

[例]

（送信側）

pm16c16□Stop 全てのモータを減速停止します

（返信されてくる文字列）

pm16c16>term1□@Stop□Ok:

StopEmergency

このコマンドを送信すると全てのモータが緊急停止します。

このコマンドはコントローラが Local 操作モードの場合無視されます。

[リプライ・メッセージ]

@StopEmergency□Ok: 正常に動作した場合

[例]

（送信側）

pm16c16□StopEmergency 全てのモータを緊急停止します

（返信されてくる文字列）

pm16c16>term1□@StopEmergency□Ok:

◆モータ同時駆動制御コマンド

モータ駆動コマンドを送信する前に、モータ駆動を待機状態に設定することができます。
当機能を使って、複数のモータ駆動の開始タイミングを合わせることができます。

Standby

モータ駆動を待機状態にします。
このコマンド実行後に送信された全てのモータ駆動コマンドは待ち状態に入ります。後述の SyncRun コマンドを送信することでモータ駆動が開始します。
[リプライ・メッセージ]

@Standby□Ok:	正常に動作した場合
@Standby□Er:□(エラー情報)	なんらかの理由で実行できなかった場合

[例]
（送信側）

pm16c16□Standby	モータ同時駆動 Stanby 状態にします。

（返信されてくる文字列）
pm16c16>term1□@Standby□Ok:

SyncRun

このコマンドを送信するとモータ駆動の待機状態を解除します。
前述の「Standby」コマンド送信後に送信されたモータ駆動コマンドがあった場合にはそれらモータ駆動が一斉に開始します。
[リプライ・メッセージ]

@SyncRun□Ok:	正常に動作した場合
@SyncRun□Er:□(エラー情報)	なんらかの理由で実行できなかった場合

[例]
（送信側）

pm16c16□SyncRun	モータ同時駆動を開始します

（返信されてくる文字列）
pm16c16>term1□@SyncRun□Ok:

IsStandby

このコマンドを送信することで、モータ駆動待機状態にあるかを取得します。
[リプライ・メッセージ]

@IsStandby□0	待機状態にない（送信したモータ駆動コマンドは即実行される）
@ IsStandby□1	待機状態にある（待機状態に入った後に送信されたモータ駆動コマンドは待機状態に入る）

[例]
（送信側）

pm16c16□IsStandby	モータ駆動待機状態を確認します

（返信されてくる文字列）

pm16c16>term1□@IsStandby□0	モータ駆動は待機状態にない
pm16c16>term1□@IsStandby□1	モータ駆動は待機状態にある

◆イベントメッセージ送信リクエストコマンド

flushdata

このコマンドを送信するとコントローラおよびモータの全てのステータス情報をイベントメッセージとして Stars の TAK サーバ’System’に返します。
ステータス情報をイベントメッセージとして受け取るには、この当コマンドを発行する前に Stars の TAK サーバ’System’に対してイベントメッセージ配信依頼のコマンドを送信しておく必要があります。

[リプライ・メッセージ]

@flushdata□Ok:
コマンドが正常に送信された場合

[例]

（送信側）

System□flgon□pm16c16	コントローラのイベントメッセージの配信を依頼します
System□flgon□pm16c16.th	モータ名’th’のイベントメッセージの配信を依頼します
pm16c16□flushdata	イベントメッセージ配信の実行を依頼します

（返信されてくる文字列）

pm16c16□@flushdata□Ok:	コマンドが正常送信されました
pm16c16>term1□_ChangedCtlIsBusy□0	モータ全てが Busy か否かがイベントメッセージの値として返ってきます
pm16c16>term1□_ChangedFunction□1	コントローラが Remote か Local かがイベントメッセージの値として返ってきます
pm16c16.th>term1□_ChangedIsBusy□0	モータ名 th の Busy 状態がイベントメッセージの値として返ってきます
pm16c16.th>term1□_ChangedValue□100	モータ名 th の現在値がイベントメッセージの値として返ってきます

flushdatatome

このコマンドを送信するとコントローラおよびモータの全てのステータス情報をイベントメッセージとして当コマンドの Stars 送信元に直接返します。

[例]

（送信側）

pm16c16□flushdatatome　　イベントメッセージ配信の実行を依頼します

（返信されてくる文字列）

pm16c16□@flushdatatome□Ok:　　コマンドが正常送信されました

pm16c16>term1□_ChangedCtlIsBusy□0　　モータ全てが Busy か否かがイベントメッセージの値として返ってきます

pm16c16>term1□_ChangedFunction□1　　コントローラが Remote か Local かがイベントメッセージの値として返ってきます

pm16c16.th>term1□_ChangedIsBusy□0　　モータ０の th の Busy 状態がイベントメッセージの値として返ってきます

pm16c16.th>term1□_ChangedValue□100　　モータ０の th の現在値がイベントメッセージの値として返ってきます

pm16c16.DTH>term1□_ChangedIsBusy□0　　モータ１の DTH の Busy 状態がイベントメッセージの値として返ってきます

pm16c16.DTH>term1□_ChangedValue□100　　モータ１の DTH の現在値がイベントメッセージの値として返ってきます

・・・

以下、モータ No.２から１５までの_ChangedIsBusy イベントと_ChangedValue イベントが返されます。（例では省略しています）

◆デバイスコマンドの送信

SendRawCommand（オプション機能）

当コマンドを用いて、PM16C-16 の制御コマンドの送受信が行えます。
当コマンドはデフォルト設定では送信されません。有効にするには、プログラム起動オプション「--rawemanle」を指定してください。
また、制御コマンドのチェックは行わないので注意して使用してください。

[引数]

PM16C-16 のデバイスコマンド

[リプライ・メッセージ]

@SendRawCommand□（引数）□Ok:（□あればデバイスコマンドの応答結果）

@SendRawCommand□（引数）□Er:□(エラー情報)　　デバイスコマンドの送信に失敗した等エラー

[例]

（送信側）

pm16c-4c-2 SendRawCommand SETCH0123

pm16c-4c-2 SendRawCommand SETCH?

（返信されてくる文字列）

pm16c-4c-2>term1 @SendRawCommand SETCH0123 Ok:

pm16c-4c-2>term1 @SendRawCommand SETCH? Ok: 0123

◆コントローラの現在状況確認コマンド

GetCtlIsBusy（PM16C-04 製品群互換用）

PM16C-04 製品群では、16 個あるモータのうち同時に駆動できるモータ数は 4 つまででしたが、PM16C-16 からは 16 個すべてのモータの同時駆動が可能になりました。

元々PM16C-04 制御用に駆動用の空きがあるかどうか確認するためのコマンドで、PM16C-16 では、常に 0 を返します。

[リプライ・メッセージ]

@Get CtlIsBusy□0	駆動上限に達していない

◆コントローライベント

_ChangedCtlIsBusy イベント（オプション機能）

このイベントは、駆動用の空きがあるかどうか否かが切り替わった場合、もしくは”flushdata”コマンド、”flushdatatome”コマンドを発行した場合に STARS Server から送られます。

PM16C-16 は全モータ同時駆動が可能なので、値は常に 0 となります。

当イベントはデフォルト設定では送信されません。有効にするには、プログラム起動オプション「--pm16c04compatible」を指定してください。

[イベント・メッセージ]

_ChangedCtlIsBusy□0	空きチャンネルがある場合

[例]

（送信側）

System flgon pm16c16	コントローラのイベント監視を開始します
pm16c16 flushdata	イベントメッセージの強制送信を指示します

（返信されてくる文字列）

System>term1□@flgon Node pm16c16 has been registered.

pm16c16>term1□_ChangedCtlIsBusy□0

pm16c16>term1□_ChangedFunction□1

pm16c16>term1□@flushdata□Ok:

_ChangedFunction イベント

このイベントは Remote もしくは Local 状態が切り替わった場合、もしくは”flushdata”コマンド、”flushdatatome”コマンドを発行した場合に STARS Server から送られます。

[イベント・メッセージ]

_ChangedFunction□1	Remote 状態である場合
_ChangedFunction□0	Local 状態である場合

[例]

（送信側）

System flgon pm16c16	コントローラのイベント監視を開始します
pm16c16 flushdata	イベントメッセージの強制送信を指示します

（返信されてくる文字列）

System>term1□@flgon Node pm16c16 has been registered.

pm16c16>term1□_ChangedCtlIsBusy□0

pm16c16>term1□_ChangedFunction□1

pm16c16>term1□@flushdata□Ok:

モータコマンド

[メッセージ配信先名]

モータに対して STARS コマンドを送信する場合は、配信先名として、STARS ノード名＋”.”（ドット）＋モータ名を指定します。
使用可能なモータ名の一覧は、コントローラコマンド「GetMotorList」で確認できます。
以下のモータコマンドの送信例は、配信先となる STARS のノード名が pm16c16、モータ名が th、コマンドの送信元のノード名が term1 である場合を示しています。

[コマンド]

hello

STARS の通信が行われているかをチェックするコマンド。
このコマンドを送信すると’@hello Nice to meet you.’の文字列を返します。

[例]
（送信側）
pm16c16.th□hello
（返信されてくる文字列）
pm16c16.th>term1□@hello□@hello□Nice□to□meet□you.

help

引数を指定しない場合はモータコマンドの一覧を返します。
引数にコマンド名を指定した場合はコマンドのヘルプを返します。

[例]
（送信側）
pm16c16.th□help　　コントローラコマンドの一覧を表示します
（返信されてくる文字列）
pm16c16.th>term1□@help□GetAccRate□GetAccRateCode□GetAccRateList□...(略)...□SetValue□SetValueREL□SpeedHigh□SpeedLow□SpeedMiddle□Stop□StopEmergency□_ChangedIsBusy□_ChangedLimitStatus□_ChangedValue□hello□help
（送信側）
pm16c16.th□help□hello　　hello コマンドのヘルプを表示します
（返信されてくる文字列）
pm16c16.th>term1□@help□hello□Return□'hello□Nice□to□meet you.'

（送信側）
pm16c16.th help helo　　引数として該当しないコマンドを指定した場合
pm16c16.th>term1 @help helo Er: Command "helo" not found.

GetMotorNumber

このコマンドを送信することでモータ名に対応するモータ No.を取得します。

[例]

（送信側）

pm16c16.th□GetMotorNumber　　モータ th のモータ No.番号を取得します

（返信されてくる文字列）

pm16c16.th>term1□@GetMotorNumber□0　　モータ th のモータ No.0 が返ってきます

◆モータ速度レベル選択コマンド

SpeedHigh

このコマンドを送信することで、以降対応するモータの速度は’High’の設定で動きます。

[リプライ・メッセージ]

@SpeedHigh□Ok:　　正常に動作した場合

@SpeedHigh□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SpeedHigh

（返信されてくる文字列）

pm16c16.th>term1□@SpeedHigh□Ok:

SpeedMiddle

このコマンドを送信することで、以降対応するモータの速度は’Middle’の設定で動きます。

[リプライ・メッセージ]

@SpeedMiddle□Ok:　　正常に動作した場合

@SpeedMiddle□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SpeedMiddle

（返信されてくる文字列）

pm16c16.th>term1□@SpeedMiddle□Ok:

SpeedLow

このコマンドを送信することで、以降対応するモータの速度は’Low’の設定で動きます。

[リプライ・メッセージ]

@SpeedLow□Ok:　　正常に動作した場合

@SpeedLow□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SpeedLow

（返信されてくる文字列）

pm16c16.th>term1□@SpeedLow□Ok:

GetSpeedSelected

このコマンドを送信すると対応するモータの速度がどの設定（Low/Middle/High)で動くのかを返します

[リプライ・メッセージのコマンド]

@GetSpeedSelected□H : 選択されている速度が'High'の場合

@GetSpeedSelected□M : 選択されている速度が'Middle'の場合

@GetSpeedSelected□L : 選択されている速度が'Low'の場合

[例]

（送信側）

pm16c16□GetSpeedSelected

（返信されてくる文字列）

pm16c16>term1□@GetSpeedSelected□M

◆モータ設定コマンド

以下のコマンドは Stars の GUI プログラム‘pm16cconfig’から参照・更新できます。

SetHighSpeed

このコマンドを送信することで対応するモータの速度’High’の値（PPS）を設定します。

[引数]

1 から 5000000 までの数字文字列（PPS）（＋符号の指定は不可）

[リプライ・メッセージ]

@SetHighSpeed□（引数）□Ok: 正常に値を設定した場合

@SetHighSpeed□（引数）□Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetHighSpeed□10000 モータ th の速度’High’の値（PPS）を 10000 で設定します

（返信されてくる文字列）

pm16c16.th>term1□@SetHighSpeed□10000□Ok:

SetMiddleSpeed

このコマンドを送信することで対応するモータの速度’Middle’の値（PPS）を設定します。

[引数]

1 から 5000000 までの数字文字列（PPS）（＋符号の指定は不可）

[リプライ・メッセージ]

@SetMiddleSpeed□（引数）□Ok: 正常に値を設定した場合

@SetMiddleSpeed□（引数）□Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetMiddleSpeed□10000　　モータ th の速度'Middle'の値（PPS）を10000 で設定します

（返信されてくる文字列）

pm16c16.th>term1□@SetMiddleSpeed□10000□Ok

SetLowSpeed

このコマンドを送信することで対応するモータの速度’Low’の値（PPS）を設定します。

[引数]

1 から 5000000 までの数字文字列（PPS）（＋符号の指定は不可）

[リプライ・メッセージ]

@SetLowSpeed□（引数）□Ok:　　正常に値を設定した場合

@SetLowSpeed□（引数）□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetLowSpeed□10000　　モータ th の速度'Low'の値（PPS）を10000 で設定します

（返信されてくる文字列）

pm16c16.th>term1□@SetLowSpeed□10000□Ok

SetAccRate

このコマンドを送信することで対応するモータの速度の加減速レートの値（mS/1000PPS）を設定します。

[引数]

モータの速度の加減速レートの値※　　正の数値文字列（＋符号なし）

※設定可能な速度の加減速レートは、GetAccRateList コマンドで確認できる速度の加減速レートデータ表に定義されている値です。

引数の速度の加減速レートに一致する値が速度の加減速レートデータ表にない場合は、コード表に定義されている速度の加減速レートの中から引数の値未満の近似値を使用します。

引数の速度の加減速レートが、速度の加減速レートデータ表に定義されている最小値より小さい場合は最小値を使用します。

[リプライ・メッセージ]

@SetAccRate□（引数）□Ok:　　正常に値を設定した場合

@SetAccRate□（引数）□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetAccRate□300　　モータ th の速度の加減速レートを設定

（返信されてくる文字列）

pm16c16.th>term1□@SetAccRate□300□Ok:

SetAccRateCode

このコマンドを送信することで対応するモータの速度の加減速レートをコード番号で設定します。

[引数]

モータの速度の加減速レートのコード番号　　0 から始まる数値

[リプライ・メッセージ]

@SetAccRateCode□（引数）□Ok:　　正常に値を設定した場合

@SetAccRateCode□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetAccRateCode□13　　モータ th の速度の加減速レートをコード番号 13 で設定

（返信されてくる文字列）

pm16c16.th>term1□@SetAccRateCode□13□Ok:

SetDigitalCwLs

このコマンドを送信することで対応するモータの CW ソフトウェアリミットスイッチの値を設定します。

[引数]

モータの CW ソフトウェアリミットスイッチの値（パルス）

-2147483647 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetDigitalCwLs□（引数）□Ok:　　正常に値を設定した場合

@SetDigitalCwLs□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetDigitalCwLs□40000

モータ th の CW ソフトウェアリミットスイッチの値を設定

（返信されてくる文字列）

pm16c16.th>term1□@SetDigitalCwLs□40000　　正常に値を設定した場合

SetDigitalCcwLs

このコマンドを送信することで対応するモータの CCW ソフトウェアリミットスイッチの値を設定します。

[引数]

モータの CCW ソフトウェアリミットスイッチの値（パルス）

-2147483647 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetDigitalCcwLs□（引数）□Ok:　　正常に値を設定した場合

@SetDigitalCcwLs□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetDigitalCcwLs□-40000

モータ th の CCW ソフトウェアリミットスイッチの値を設定

（返信されてくる文字列）

pm16c16.th>term1□@SetDigitalCcwLs□-40000　正常に値を設定した場合

SetCancelBacklash

このコマンドを送信することで対応するモータのバックラッシュ補正ステップ数の値を設定します。

[引数]

バックラッシュ補正ステップ数（パルス）　-9999～9999 の数字文字列

[リプライ・メッセージ]

@SetCancelBacklash□（引数）□Ok:　正常に値を設定した場合

@SetCancelBacklash□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetCancelBacklash□100

モータ th のバックラッシュ補正ステップ数の値を設定

（返信されてくる文字列）

pm16c16.th>term1□@SetCancelBackllash□100□Ok:

SetJogPulse

このコマンドを送信することで、ローカルモードで Jog 操作をする際の、モータの 1 Jog 単位の値を設定します。

[引数]

JOG 単位の値（パルス）　1～9999 の数字文字列

[リプライ・メッセージ]

@SetJogPulse□（引数）□Ok:　正常に値を設定した場合

@SetJogPulse□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetJogPulse□10　モータ th の 1 Jog 単位の値を設定

（返信されてくる文字列）

pm16c16.th>term1□@SetJogPulse□10□Ok:

SetLimits

このコマンドを送信することで、対応する対応するモータのリミットスイッチ関連フラグの値を設定します。

[引数]

下記のフォーマット※の８桁の数字文字列

※ 数字文字列のフォーマット　ABCDEFGH

A	Digital LS Enable Bit	（1: enable　0: disable）
B	HP LS Enable Bit	（1: enable　0: disable）
C	CCW LS Enable Bit	（1: enable　0: disable）
D	CW LS Enable Bit	（1: enable　0: disable）
E	未使用	（0）
F	HP LS Invert Bit	（1:Normaly Close　0: Normaly Open）
G	CCW LS Invert Bit	（1:Normaly Close　0: Normaly Open）
H	CW LS Invert Bit	（1:Normaly Close　0: Normaly Open）

[リプライ・メッセージ]

@SetLimits□（引数）□Ok:　　正常に値を設定した場合

@SetLimits□（引数）□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetLimits□11110000

（返信されてくる文字列）

pm16c16.th>term1□@SetLimits□11110000□Ok:

SetMotorSetup

このコマンドを送信することで、対応する対応するモータの基本特性の値を設定します。

[引数]

下記のフォーマット※の４桁の数字文字列

※ 数字文字列のフォーマット　ABCD

A	モータの有効無効	（1: drive enable　0: disable）
B	Hold 特性	（1: hold on　0: hold off）
C	モータの移動形式	（2: S 字　1:台形　0:定速）
D	外部設置のモータドライバの信号入力	（1: P-D 方式　0: P-P 方式）

[リプライ・メッセージ]

@SetMotorSetup□（引数）□Ok:　　正常に値を設定した場合

@SetMotorSetup□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetMotorSetup□1010

（返信されてくる文字列）

pm16c16.th>term1□SetMotorSetup□1010□Ok:

SetHold

このコマンドを送信することで、対応する対応するモータの Hold 特性の値を設定します。

[引数]

Hold 特性　　（1: hold on　0: hold off）

＊上記モータコマンド「SetMotorSetup」の「B」と同じ値です。

[リプライ・メッセージ]

@SetHold□（引数）□Ok:　　正常に値を設定した場合

@SetHold□（引数）□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetHold□0

（返信されてくる文字列）

pm16c16.th>term1□SetHold□0□Ok:

SetStopMode

このコマンドを送信することで、対応する対応するモータの停止方法の値を設定します。

[引数]

下記のフォーマット※の２桁の数字文字列

※ 数字文字列のフォーマット　AB

A　　CW/CCW リミットスイッチ　　（1: 緊急停止　0: 減速停止）

B　　パネルの STOP スイッチ　　（1: 緊急停止　0: 減速停止）

[リプライ・メッセージ]

@SetStopMode□（引数）□Ok:　　正常に値を設定した場合

@SetStopMode□（引数）□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetStopMode□00

（返信されてくる文字列）

pm16c16.th>term1□@SetStopMode□00□Ok:

GetHighSpeed

このコマンドを送信することで対応するモータの速度’High’の設定値（PPS）を返します。

[戻り値]

1 から 5000000 までの数字文字列（PPS）

[例]

（送信側）

pm16c16.th□GetHighSpeed　　モータ th の速度’High’の設定値を取得

（返信されてくる文字列）

pm16c16.th>term1□@GetHighSpeed□1000　　正常に動作した場合

GetMiddleSpeed

このコマンドを送信することで対応するモータの速度’Middle’の設定値（PPS）を返します。

[戻り値]

1 から 5000000 までの数字文字列（PPS）

[例]

（送信側）

pm16c16.th□GetMiddleSpeed　　モータ th の速度’Middle’の設定値を取得

（返信されてくる文字列）

pm16c16.th>term1□@GetMiddleSpeed□500　　正常に動作した場合

GetLowSpeed

このコマンドを送信することで対応するモータの速度’Low’の設定値（PPS）を返します。

[戻り値]

1 から 5000000 までの数字文字列（PPS）

[例]

（送信側）

pm16c16.th□GetLowSpeed　　モータ th の速度’Low’の設定値を取得

（返信されてくる文字列）

pm16c16.th>term1□@GetLowSpeed□100　　正常に動作した場合

GetAccRate

このコマンドを送信することで対応するモータの速度の加減速レートの値（mS/1000PPS）を返します。

[戻り値]

モータの速度の加減速レートの値※

※加減速レートとして取りうる値は、GetAccRateList コマンドで確認できる速度の加減速レートデータ表に定義されている値です。

[例]

（送信側）

pm16c16.th□GetAccRate　　モータ th の速度の加減速レートの設定値を取得

（返信されてくる文字列）

pm16c16.th>term1□@GetAccRate□300　正常に動作した場合

GetAccRateCode

このコマンドを送信することで対応するモータの速度の加減速レートのコード番号を返します

[戻り値]

モータの速度の加減速レートのコード値

[例]

（送信側）

pm16c16.th□GetAccRateCode　　モータ th の速度の加減速レートのコード番号 13 で設定

（返信されてくる文字列）

pm16c16.th>term1□@GetAccRateCode□13

GetDigitalCwLs

このコマンドを送信することで対応するモータの CW ソフトウェアリミットスイッチの値を返します。

[戻り値]

モータの CW ソフトウェアリミットスイッチの値（パルス）

-2147483647 から 2147483647 までの数値文字列

[例]

（送信側）

pm16c16.th□GetDigitalCwLs

モータ th の CW ソフトウェアリミットスイッチの値を取得

（返信されてくる文字列）

pm16c16.th>term1□@GetDigitalCwLs□40000　　正常に動作した場合

GetDigitalCcwLs

このコマンドを送信することで対応するモータの CCW ソフトウェアリミットスイッチの値を返します。

[戻り値]

モータの CCW ソフトウェアリミットスイッチの値（パルス）

-2147483647 から 2147483647 までの数値文字列

[例]

（送信側）

pm16c16.th□GetDigitalCcwLs

モータ th の CCW ソフトウェアリミットスイッチの値を取得

（返信されてくる文字列）

pm16c16.th>term1□@GetDigitalCcwLs□-40000　　正常に動作した場合

GetCancelBacklash

このコマンドを送信することで対応するモータのバックラッシュ補正ステップ数の値を返します。

[戻り値]

バックラッシュ補正ステップ数（パルス）　-9999～9999 の数字文字列

[例]

（送信側）

pm16c16.th□GetCancelBacklash

モータ th のバックラッシュ補正ステップ数の値を取得

（返信されてくる文字列）

pm16c16.th>term1□@GetCancelBackllash□100　正常に動作した場合

GetJogPulse

このコマンドを送信することで、ローカルモードで Jog 操作をする際の、モータの 1 Jog 単位の値を返します。

[戻り値]

JOG 単位の値（パルス）　1～9999 の数字文字列

[例]

（送信側）

pm16c16.th□GetJogPulse　モータ th の 1 Jog 単位の値を設定

（返信されてくる文字列）

pm16c16.th>term1□@GetJogPulse□10　正常に動作した場合

GetLimits

このコマンドを送信することで対応するモータのリミットスイッチ関連フラグの値を返します。

[戻り値]

下記のフォーマット※の８桁の数字文字列を返します。

※ 数字文字列のフォーマット　ABCDEFGH

A	Digital LS Enable Bit	（1: enable　0: disable）
B	HP LS Enable Bit	（1: enable　0: disable）
C	CCW LS Enable Bit	（1: enable　0: disable）
D	CW LS Enable Bit	（1: enable　0: disable）
E	未使用	（0）
F	HP LS Invert Bit	（1:Normaly Close　0: Normaly Open）
G	CCW LS Invert Bit	（1:Normaly Close　0: Normaly Open）
H	CW LS Invert Bit	（1:Normaly Close　0: Normaly Open）

[例]

（送信側）

pm16c16.th□GetLimits

（返信されてくる文字列）

pm16c16.th>term1□@GetLimits□11110000

GetMotorSetup

このコマンドを送信することで対応するモータの基本特性の値を返します。

[戻り値]

下記のフォーマット※の４桁の数字文字列を返します。

※ 数字文字列のフォーマット　ABCD

A	モータの有効無効	（1: drive enable　0: disable）
B	Hold 特性	（1: hold on　0: hold off）
C	モータの移動形式	（2: S 字　1:台形　0:定速）
D	外部設置のモータドライバの信号入力	（1: P-D 方式　0: P-P 方式）

[例]

（送信側）

pm16c16.th□GetMotorSetup

（返信されてくる文字列）

pm16c16.th>term1□@GetMotorSetup□1010

GetHold

このコマンドを送信することで対応するモータの Hold 特性の値を返します。

[戻り値]

Hold 特性　（1: hold on　0: hold off）

＊上記、GetMotorSetup コマンドの値「B」と同じ値です。

[例]

（送信側）

pm16c16.th□GetHold

（返信されてくる文字列）

pm16c16.th>term1□@GetHold□1

GetStopMode

このコマンドを送信することで対応するモータの停止方法の値を返します。

[戻り値]

下記のフォーマット※の２桁の数字文字列を返します。

※ 数字文字列のフォーマット　AB

A	CW/CCW リミットスイッチ	（1: 緊急停止　0: 減速停止）
B	パネルの STOP スイッチ	（1: 緊急停止　0: 減速停止）

[例]

（送信側）

pm16c16.th□GetStopMode

（返信されてくる文字列）

pm16c16.th>term1□@GetStopMode□00

◆HP 関連フラグ設定・読み出しコマンド

HP 検索機能の実行にあたっては、ツジ電子（株）が提供する取扱説明書の該当箇所をよく読んでからおこなってください。

SetHPMode

このコマンドを送信することで対応するモータの原点位置設定方法や検出状況の強制書き換えをおこないます。

[引数]

下記のフォーマット※の４桁の数字文字列

※ 数字文字列のフォーマット　ABCD

A	未使用	（0）
B	原点検出状況	（1: 検出あり　0: 検出なし）
C	原点検出方向	（1: CCW　0: CW）
D	自動原点検出開始方向	（1: CCW　0: CW）

[リプライ・メッセージ]

@SetHPMode□（引数）□Ok:　　正常に動作した場合

@SetHPMode□（引数）□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetHPMode□0001

（返信されてくる文字列）

pm16c16.th>term1□@SetHPMode□0001□Ok:

SetHPOffset

このコマンドを送信することでモータの原点オフセットの値を設定します。

[引数]

原点オフセット値（パルス）　0～9999 の数字文字列

[リプライ・メッセージ]

@SetHPOffset□（引数）□Ok:　　正常に動作した場合

@SetHPOffset□（引数）□Er:□(エラー情報)　　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetHPOffset□100□Ok:　　モータ th の原点オフセットの値を設定します

（返信されてくる文字列）

pm16c16.th>term1□@SetHPOffset□100□Ok:

SetHomePosition

このコマンドを送信することで対応するモータの原点位置データを書き換えます。

[引数]

原点位置データの値（パルス）

-2147483647 から～2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetHomePosition□（引数）□Ok:	正常に動作した場合
@SetHomePosition□（引数）□Er:□(エラー情報)	なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetHomePosition□0　　モータ th の原点位置データの値を書き換えます

（返信されてくる文字列）

pm16c16.th>term1□@SetHomePosition□0□Ok:

GetHPMode

このコマンドを送信することで対応するモータの原点位置設定方法や検出状況の値を返します。

[戻り値]

下記のフォーマット※の４桁の数字文字列

※ 数字文字列のフォーマット　ABCD

A	未使用	(0)
B	原点検出状況	(1: 検出あり　0: 検出なし)
C	原点検出方向	(1: CCW　0: CW)
D	自動原点検出開始方向	(1: CCW　0: CW)

[例]

（送信側）

pm16c16.th□GetHPMode

（返信されてくる文字列）

pm16c16.th>term1□@GetHPMode□0001

GetHPOffset

このコマンドを送信することで対応するモータの原点オフセットの値を取得します。

[戻り値]

原点オフセット値（パルス）　0～9999 の数字文字列

[例]

（送信側）

pm16c16.th□GetHPOffset　　モータ th の原点オフセットの値を取得します

（返信されてくる文字列）

pm16c16.th>term1□@GetHPOffset□100 モータ th の原点オフセットは 100 です

GetHomePosition

このコマンドを送信することで対応するモータの原点位置データを取得します。

[戻り値]

原点位置データがある場合は、原点位置データの値（パルス）

-2147483647 から～2147483647 までの数値文字列

原点位置データがない場合、デフォルト設定では、文字列“-”を返します。プログラム起動オプション「--pm16c04compatible」を指定することで、STARS PM16C-04 用 STARS プログラムと同様”Er: NO H.P”を返すようにすることも可能です。

[リプライ・メッセージ]

[原点位置データが設定されている場合]

@GetHomePosition□（パルス値）　原点位置として設定されているパルス値

[原点位置データが設定されていない場合]

@GetHomePosition□-　プログラムデフォルト起動時

@GetHomePosition□Er: □NO□H.P　プログラム起動オプション「--pm16c04compatible」指定時

[例]

（送信側）

pm16c16.th□GetHomePosition　モータ th の原点位置データの値を取得します

（返信されてくる文字列）

pm16c16.th>term1□@GetHomePosition□100　モータ th の原点位置データは 100 です

pm16c16.th>term1□@GetHomePosition□-　モータ th の原点位置データは未設定です

ScanHome

このコマンドを送信することでモータの自動原点検出を開始します

[リプライ・メッセージ]

@ScanHome□Ok:　正常に動作を開始した場合

@ScanHome□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□ScanHome　モータ th の自動原点検出を開始します

（返信されてくる文字列）

pm16c16.th>term1□@ScanHome□Ok:

ReScanHome

あらかじめモータの原点が検出されている時、このコマンドを送信することで、原点位置近傍に移

動しそこから低速で原点を検出します

[リプライ・メッセージ]

@ReScanHome□Ok: 正常に動作を開始した場合

@ReScanHome□Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□ReScanHome モータ th の原点の再検出を開始します

（返信されてくる文字列）

pm16c16.th>term1□@ReScanHome□Ok:

◆タイミングアウト出力機能関連コマンド

タイミングアウト出力機能の実行にあたっては、ツジ電子（株）が提供する取扱説明書の該当箇所をよく読んでからおこなってください。

SetTimingOutMode

このコマンドを送信することで対応するモータのタイミングアウトモードの設定を設定します。

[引数]

0 タイミングパルス出力をおこないません

1 開始点から終了点の間、TTL のゲートパルスを出力

2 開始点から終了点の間、設定インターバル毎に、200ns の TTL パルスを出力

3 開始点から終了点の間、設定インターバル毎に、10μs の TTL パルスを出力

4 開始点から終了点の間、設定インターバル毎に、100μs の TTL パルスを出力

5 開始点から終了点の間、設定インターバル毎に、1ms の TTL パルスを出力

[リプライ・メッセージ]

@SetTimingOutMode□（引数）□Ok: 正常に動作した場合

@SetTimingOutMode □（引数）□Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetTimingOutMode□1

（返信されてくる文字列）

pm16c16.th>term1□@SetTimingOutMode□1□Ok:

SetTimingOutStart

このコマンドを送信することで対応するモータのタイミングアウトの開始点を設定します。

[引数]

-2147483647 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetTimingOutStart□（引数）□Ok: 正常に動作した場合

@SetTimingOutStart□（引数）□Er:□(エラー情報) なんらかの理由で変更できなか

った場合

[例]

（送信側）

pm16c16.th□SetTimingOutStart□-10000

（返信されてくる文字列）

pm16c16.th>term1□@SetTimingOutStart□-10000□Ok:

SetTimingOutEnd

このコマンドを送信することで対応するモータのタイミングアウトの終了点を設定します。

[引数]

-2147483647 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetTimingOutEnd□（引数）□Ok:	正常に動作した場合
@SetTimingOutEnd□（引数）□Er:□(エラー情報)	なんらかの理由で変更できなかった場合

た場合

例]

（送信側）

pm16c16.th□SetTimingOutEnd□10000

（返信されてくる文字列）

pm16c16.th>term1□@SetTimingOutEnd□10000□Ok:

SetTimingOutInterval

このコマンドを送信することで対応するモータのタイミングアウトのインターバルを設定します。

[引数]

0 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetTimingOutInterval□（引数）□Ok:	正常に動作した場合
@SetTimingOutInterval□（引数）□Er:□(エラー情報)	なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetTimingOutInterval□1000

（返信されてくる文字列）

pm16c16.th>term1□@SetTimingOutInterval□1000□Ok:

GetTimingOutMode

このコマンドを送信することで対応するモータのタイミングアウトモードの値を返します。

[戻り値]

0　タイミングパルス出力をおこないません
1　開始点から終了点の間、TTL のゲートパルスを出力
2　開始点から終了点の間、設定インターバル毎に、200ns の TTL パルスを出力
3　開始点から終了点の間、設定インターバル毎に、10μs の TTL パルスを出力
4　開始点から終了点の間、設定インターバル毎に、100μs の TTL パルスを出力
5　開始点から終了点の間、設定インターバル毎に、1ms の TTL パルスを出力

[例]

（送信側）

pm16c16.th□GetTimingOutMode

（返信されてくる文字列）

pm16c16.th>term1□@GetTimingOutMode□0

GetTimingOutStart

このコマンドを送信することで対応するモータのタイミングアウトの開始点の値を返します。

[戻り値]

-2147483647 から 2147483647 までの数値文字列

[例]

（送信側）

pm16c16.th□GetTimingOutStart

（返信されてくる文字列）

pm16c16.th>term1□@GetTimingOutStart□-10000

GetTimingOutEnd

このコマンドを送信することで対応するモータのタイミングアウトの終了点の値を返します。

[戻り値]

-2147483647 から 2147483647 までの数値文字列

[例]

（送信側）

pm16c16.th□GetTimingOutEnd

（返信されてくる文字列）

pm16c16.th>term1□@GetTimingOutEnd□10000

GetTimingOutInterval

このコマンドを送信することで対応するモータのタイミングアウトのインターバルの値を返します。

[戻り値]

0 から 2147483647 までの数値文字列

[例]

（送信側）

pm16c16.th□GetTimingOutInterval

（返信されてくる文字列）

pm16c16.th>term1□@GetTimingOutInterval□1000

SetTimingOutReady

このコマンドを送信することで対応するモータのタイミングアウトの ready を設定します。

[引数]

0　タイミングアウトの ready 状態を解除します

1　タイミングアウトの ready 状態を設定します

[リプライ・メッセージ]

@SetTimingOutReady□1□Ok:	正常に動作した場合
@SetTimingOutReady□（引数）□Er:□(エラー情報)	なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetTimingOutReady□1

pm16c16.th□GetTimingOutReady　　ready 状態の確認をおこないます

（返信されてくる文字列）

pm16c16.th>term1□@SetTimingOutReady□1□Ok:

pm16c16.th>term1□@GetTimingOutReady□1　　ready 状態が設定されています

GetTimingOutReady

このコマンドを送信することで対応するモータのタイミングアウトの ready の値を返します。

[戻り値]

0　タイミングアウトの ready 状態は解除されています

1　タイミングアウトの ready 状態は設定されています

[例]

（送信側）

pm16c16.th□GetTimingOutReady

（返信されてくる文字列）

pm16c16.th>term1□@GetTimingOutReady□1

Select

このコマンドを送信することで対応するモータに対しタイミング信号出力ポートを割り当てることができます。

[引数]

1．出力ポート	A、B、C、D のいずれかの文字 (A から D：TP0 から TP3 に相当する)

[リプライ・メッセージ]

@Select□（引数の出力ポート）□Ok: 正常に動作した場合

@Select□（引数の出力ポート）□Er:□(エラー情報)

なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□Select□A モータ th に対し TP0 を割り当てる

pm16c16.th□GetSelected モータ th の割当出力ポートを確認します

（返信されてくる文字列）

pm16c16.th>term1□@Select□A□Ok

pm16c16.th>term1□@GetSelected□A

GetSelected

このコマンドを送信することで、対応するモータのタイミング信号出力ポート TP0 から TP3 の割り当て状況が確認できます。

[戻り値]

チャンネルが選択されている場合 A、B、C、D のいずれかの文字
(A から D：TP0 から TP3 に相当する)

チャンネルが選択されていない場合 N

[リプライ・メッセージ]

@GetSelected□A TP0 に割り当てられている場合

@GetSelected□B TP1 に割り当てられている場合

@GetSelected□C TP2 に割り当てられている場合

@GetSelected□D TP3 に割り当てられている場合

@GetSelected□N いずれにも割り当てられていない場合

[例]

（送信側）

pm16c16.th□GetSelected モータ th のタイミング信号出力ポート割当状況を取得します

（返信されてくる文字列）

pm16c16.th>term1□@GetSelected□A TP0 に割り当てられています

◆自動速度変更機能関連コマンド

自動速度変更機能の実行にあたっては、ツジ電子（株）が提供する取扱説明書の該当箇所をよく読んでからおこなってください。

SetAutoChangeSpeed

このコマンドを送信することで対応するモータの自動速度変更テーブルを設定します。

[引数]

１．行番号 0～127 までの数字文字列

２．速度条件コード

ADD　開始位置から指定された相対座標（引数３.）に達したら

TIM　前行の条件が実行されてから指定秒数（引数３.）が経過したら

ACC　一定の速度（引数３.）に達したら（但し加速中の場合）

DCC　一定の速度（引数３.）に達したら（但し減速中の場合）

END　条件設定終了　（引数３．以降の引数指定はありません）

３．速度条件コード値

・速度条件コードが ADD の場合座標値指定、-2147483647 から 2147483647（パルス）までの数値文字列（＋符号の指定は不可）

・速度条件コードが TIM の場合秒数指定、0 から 65535 までの数値文字列（ms）（＋符号の指定は不可）

・速度条件コードが ACC もしくは DCC の場合速度 pps 指定、1 から 5000000（pps）までの数値文字列（＋符号の指定は不可）

４．動作種別

SPD　指定速度（引数５.）への変更を開始します。

RTE　指定加減速レート（引数５.）への変更を開始します。

SLW　減速停止を開始します（引数５．以降の引数指定はありません）

FST　緊急停止を開始します（引数５．以降の引数指定はありません）

NOP　コメント行にします。当行の設定は無視されます。（引数５．以降の引数指定はありません）

５．動作種別値

・動作種別が SPD の場合速度 pps 指定、1 から 5000000（pps）までの数値文字列（＋符号の指定は不可）

・動作種別が RTE の場合加減速レートコード指定、0 から 115（RateCode）までの数値文字列（＋符号の指定は不可）

但し、GetAccRate コマンドの値に依存して動作可能となる加減速レートコードが絞り込まれます。詳細はツジ電子（株）の取扱説明書をお読みください。

[リプライ・メッセージ]

@SetAutoChangeSpeed□0□ADD□10000□SPD□2000□Ok:　正常に動作した場合(ADDxSPD)

@SetAutoChangeSpeed□0□ADD□10000□RTE□10□Ok:　正常に動作した場合(ADDxRTE)

@SetAutoChangeSpeed□0□ADD□10000□SLW□Ok:　正常に動作した場合(ADDxSLW)

@SetAutoChangeSpeed□0□ADD□10000□FST□Ok:　正常に動作した場合(ADDxFST)

@SetAutoChangeSpeed□0□ADD□10000□NOP□Ok:　正常に動作した場合(ADDxNOP)

@SetAutoChangeSpeed□0□TIM□1000□SPD□2000□Ok:　正常に動作した場合(TIMxSPD)

@SetAutoChangeSpeed□0□TIM□1000□RTE□10□Ok:　正常に動作した場合(TIMxRTE)

@SetAutoChangeSpeed□0□TIM□1000□SLW□Ok:　正常に動作した場合(TIMxSLW)

@SetAutoChangeSpeed□0□TIM□1000□FST□Ok:　正常に動作した場合(TIMxFST)

@SetAutoChangeSpeed□0□TIM□1000□NOP□Ok:　正常に動作した場合(TIMxNOP)

@SetAutoChangeSpeed□0□ACC□1000□SPD□2000□Ok:　正常に動作した場合(ACCxSPD)

@SetAutoChangeSpeed□0□ACC□1000□RTE□16□Ok: 正常に動作した場合(ACCxRTE)

@SetAutoChangeSpeed□0□ACC□1000□SLW□Ok: 正常に動作した場合(ACCxSLW)

@SetAutoChangeSpeed□0□ACC□1000□FST□Ok: 正常に動作した場合(ACCxFST)

@SetAutoChangeSpeed□0□ACC□1000□NOP□Ok: 正常に動作した場合(ACCxNOP)

@SetAutoChangeSpeed□0□DCC□300□SPD□10□Ok: 正常に動作した場合(DCCxSPD)

@SetAutoChangeSpeed□0□DCC□300□RTE□0□Ok: 正常に動作した場合(DCCxRTE)

@SetAutoChangeSpeed□0□DCC□300□SLW□Ok: 正常に動作した場合(DCCxSLW)

@SetAutoChangeSpeed□0□DCC□300□FST□Ok: 正常に動作した場合(DCCxFST)

@SetAutoChangeSpeed□0□DCC□300□NOP□Ok: 正常に動作した場合(DCCxNOP)

@SetAutoChangeSpeed□1□END□Ok: 正常に動作した場合(END)

@SetAutoChangeSpeed□（与えられた引数）□Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetAutoChangeSpeed□0□ADD□10000□SPD□2000

自動速度条件０：開始座標から+10000 で速度を 2000pps にあげる。

pm16c16.th□SetAutoChangeSpeed□1□ADD□20000□SPD□10000

自動速度条件１：開始座標から+20000 で速度を 10000pps にあげる。

pm16c16.th□SetAutoChangeSpeed□2□ADD□120000□SLW

自動速度条件２：開始座標から+120000 で減速停止する。

pm16c16.th□SetAutoChangeSpeed□3□END

自動速度条件３：END なので自動速度変更の条件設定はここで終了。

（返信されてくる文字列）

pm16c16.th>term1□@SetAutoChangeSpeed□0□ADD□10000□SPD□2000□Ok:

pm16c16.th>term1□@SetAutoChangeSpeed□0□ADD□20000□SPD□10000□Ok:

pm16c16.th>term1□@SetAutoChangeSpeed□2□ADD□120000□SLW□Ok:

pm16c16.th>term1□@ SetAutoChangeSpeed□3□END□Ok:

GetAutoChangeSpeed

このコマンドを送信することで対応するモータの自動速度変更テーブルの値を返します。

[引数]

行番号 0～127 までの数字文字列

[戻り値]

１．速度条件コード

ADD 開始位置から指定された相対座標（引数２.）に達したら

TIM 前行の条件が実行されてから指定秒数（引数２.）が経過したら

ACC 一定の速度（引数２.）に達したら（但し加速中の場合）

DCC　一定の速度（引数２.）に達したら（但し減速中の場合）

END　条件設定終了　（引数３．以降はありません）

２．速度条件コード値

・速度条件コードが ADD の場合座標値、-2147483647 から 2147483647（パルス）までの数値文字列

・速度条件コードが TIM の場合秒数、0 から 65535（ms）までの数値文字列

・速度条件コードが ACC もしくは DCC の場合速度 pps、1 から 5000000（pps）までの数値文字列

３．動作種別

SPD　指定速度（引数４.）への変更を開始します。

RTE　指定加減速レート（引数４.）への変更を開始します。

SLW　減速停止を開始します（引数４．以降はありません）

FST　緊急停止を開始します（引数４．以降はありません）

NOP　コメント行にします。当行の設定は無視されます。（引数４．以降はありません）

４．動作種別値

・動作種別が SPD の場合速度 pps、1 から 5000000（pps）までの数値文字列

・動作種別が RTE の場合加減速レートコード、0 から 115（RateCode）までの数値文字列

[リプライ・メッセージ]

@GetAutoChangeSpeed□0□ADD□10000□SPD□2000	正常に動作した場合(ADDxSPD)
@GetAutoChangeSpeed□0□ADD□10000□RTE□10	正常に動作した場合(ADDxRTE)
@GetAutoChangeSpeed□0□ADD□10000□SLW	正常に動作した場合(ADDxSLW)
@GetAutoChangeSpeed□0□ADD□10000□FST	正常に動作した場合(ADDxFST)
@GetAutoChangeSpeed□0□ADD□10000□NOP	正常に動作した場合(ADDxNOP)
@GetAutoChangeSpeed□0□TIM□1000□SPD□2000	正常に動作した場合(TIMxSPD)
@GetAutoChangeSpeed□0□TIM□1000□RTE□10	正常に動作した場合(TIMxRTE)
@GetAutoChangeSpeed□0□TIM□1000□SLW	正常に動作した場合(TIMxSLW)
@GetAutoChangeSpeed□0□TIM□1000□FST	正常に動作した場合(TIMxFST)
@GetAutoChangeSpeed□0□TIM□1000□NOP	正常に動作した場合(TIMxNOP)
@GetAutoChangeSpeed□0□ACC□1000□SPD□2000	正常に動作した場合(ACCxSPD)
@GetAutoChangeSpeed□0□ACC□1000□RTE□16	正常に動作した場合(ACCxRTE)
@GetAutoChangeSpeed□0□ACC□1000□SLW	正常に動作した場合(ACCxSLW)
@GetAutoChangeSpeed□0□ACC□1000□FST	正常に動作した場合(ACCxFST)
@GetAutoChangeSpeed□0□ACC□1000□NOP	正常に動作した場合(ACCxNOP)
@GetAutoChangeSpeed□0□DCC□300□SPD□10	正常に動作した場合(DCCxSPD)
@GetAutoChangeSpeed□0□DCC□300□RTE□0	正常に動作した場合(DCCxRTE)
@GetAutoChangeSpeed□0□DCC□300□SLW	正常に動作した場合(DCCxSLW)
@GetAutoChangeSpeed□0□DCC□300□FST	正常に動作した場合(DCCxFST)

@GetAutoChangeSpeed□0□DCC□300□NOP　　正常に動作した場合(DCCxNOP)

@GetAutoChangeSpeed□1□END　　正常に動作した場合(END)

[例]

（送信側）

pm16c16.th□GetAutoChangeSpeed□0

pm16c16.th□GetAutoChangeSpeed□1

pm16c16.th□GetAutoChangeSpeed□2

pm16c16.th□GetAutoChangeSpeed□3

（返信されてくる文字列）

pm16c16.th>term1□@SetAutoChangeSpeed□0□ADD□10000□SPD□2000

pm16c16.th>term1□@SetAutoChangeSpeed□0□ADD□20000□SPD□10000

pm16c16.th>term1□@SetAutoChangeSpeed□2□ADD□120000□SLW

pm16c16.th>term1□@ SetAutoChangeSpeed□3□END

SetAutoChangeSpeedReady

このコマンドを送信することで対応するモータの自動速度変更機能の ready を設定します。

[引数]

0　自動速度変更機能の ready 状態を解除します

1　自動速度変更機能の ready 状態を設定します

[リプライ・メッセージ]

@SetAutoChangeSpeedReady□（引数）□Ok:　　正常に動作した場合

@SetAutoChangeSpeedReady□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetAutoChangeSpeedReady□1

pm16c16.th□GetAutoChangeSpeedReady　　ready 状態の確認をおこないます

（返信されてくる文字列）

pm16c16.th>term1□@SetAutoChangeSpeedReady□1□Ok:

pm16c16.th>term1□@GetAutoChangeSpeedReady□1　　ready 状態が設定されてます

GetAutoChangeSpeedReady

このコマンドを送信することで対応するモータの自動速度変更機能の ready の値を返します。

[戻り値]

0　自動速度変更機能の ready 状態は解除されています

1　自動速度変更機能の ready 状態は設定されています

[例]

（送信側）

pm16c16.th□GetTimingOutReady

（返信されてくる文字列）

pm16c16.th>term1□@GetTimingOutReady□1

◆モータ座標のプリセットコマンド

Preset

このコマンドを送信することでモータの現在位置を指定値で書き換えます。

[引数]

モータ位置の絶対値

-2147483647 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@Preset□（引数）□Ok: 正常に値を設定した場合

@Preset□（引数）□Er:□(エラー情報) なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□Preset□10000 モータ th の現在値を 10000 に設定する

（返信されてくる文字列）

pm16c16.th>term1□@Preset□10000□Ok:

◆モータ駆動コマンド

以下、モータを駆動するコマンドです。

コントローラコマンド「Standby」を実行して同時駆動 Standby 状態に入っている場合、コントローラコマンド「SyncRun」を実行するまで移動動作はおこなわれません。

JogCw

このコマンドを送信することで CW の方向にモータを 1 パルス動かします。

[リプライ・メッセージ]

@JogCw□Ok: 正常に動作を開始した場合

@JogCw□Er:□(エラー情報) なんらかの理由で開始できなかった場合

[例]

（送信側）

pm16c16.th□JogCw モータ th を CW の方向に 1Jog 移動する

（返信されてくる文字列）

pm16c16.th>term1□@JogCw□Ok:

JogCcw

このコマンドを送信することで CCW の方向にモータを 1 パルス動かします。

使用方法およびリプライ・メッセージはコマンド JogCw の説明を JogCcw、CW を CCW に置き換えてお読みください。

ScanCwConst

このコマンドを送信することで CW の方向に一定速走行でモータを連続して動かします。

[リプライ・メッセージ]

@ScanCwConst□Ok:	正常に動作を開始した場合
@ScanCwConst□Er:□(エラー情報)	なんらかの理由で開始できなかった場合

[例]

（送信側）

pm16c16.th□ScanCwConst	モータ th を CW の方向に一定速走行で連続して動かします

（返信されてくる文字列）

pm16c16.th>term1□@ScanCwConst□Ok:

ScanCcwConst

このコマンドを送信することで CCW の方向に一定速走行でモータを連続して動かします。

使用方法およびリプライ・メッセージはコマンド ScanCwConst の説明を ScanCcwConst、CW を CCW に置き換えてお読みください。

ScanCw

このコマンドを送信することで CW の方向にモータを連続して動かします。

[リプライ・メッセージ]

@ScanCw□Ok:	正常に動作を開始した場合
@ScanCw□Er:□(エラー情報)	なんらかの理由で開始できなかった場合

[例]

（送信側）

pm16c16.th□ScanCw	モータ th を CW の方向に連続して動かします

（返信されてくる文字列）

pm16c16.th>term1□@ScanCw□Ok:

ScanCcw

このコマンドを送信することで CW の方向にモータを連続して動かします。

使用方法およびリプライ・メッセージはコマンド ScanCw の説明を ScanCcw、CW を CCW に置き換えてお読みください。

ScanCwHome

このコマンドを送信することで CW の方向に HP（原点）を検出するまでモータを連続して動かします

[リプライ・メッセージ]

@ScanCwHome□Ok:	正常に動作を開始した場合
@ScanCwHome□Er:□(エラー情報)	なんらかの理由で開始できなかった場合

[例]

（送信側）

pm16c16.th□ScanCwHome　モータ th を CW の方向に HP 検出まで連続して動かします

（返信されてくる文字列）

pm16c16.th>term1□@ScanCwHome□Ok:

ScanCcwHome

このコマンドを送信することで CCW の方向に HP（原点）を検出するまでモータを連続して動かします

使用方法およびリプライ･メッセージはコマンド ScanCwHome の説明を ScanCcwHome､CW を CCW に置き換えてお読みください。

SetValue

このコマンドを送信することでモータの位置を絶対値で指定された値まで移動します。

[引数]

モータ位置の絶対値

-2147483647 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetValue□（引数）□Ok:	正常に動作を開始した場合
@SetValue□（引数）□Er:□(エラー情報)	なんらかの理由で開始できなかった場合

[例]

（送信側）

pm16c16.th□SetValue□10000　モータ th を絶対値 10000 まで移動する

（返信されてくる文字列）

pm16c16.th>term1□@SetValue□10000□Ok:

SetValueREL

このコマンドを送信することでモータの位置を現在位置からの相対値で指定された値まで移動します。

[引数]

モータ位置の絶対値

-2147483647 から 2147483647 までの数値文字列（＋符号の指定は不可）

[リプライ・メッセージ]

@SetValueREL□（引数）□Ok:	正常に動作を開始した場合
@SetValueREL□（引数）□Er:□(エラー情報)	なんらかの理由で開始できなかった場合

[例]

（送信側）

pm16c16.th□SetValueREL□100　　モータ th を現在位置から+の方向に 100 移動する

（返信されてくる文字列）

pm16c16.th>term1□@SetValueREL□10000□Ok:

◆モータ停止コマンド

Stop

このコマンドを送信すると対応するモータが減速停止します。

このコマンドはコントローラが Local 状態の場合操作は無視されます。

[リプライ・メッセージ]

@Stop□Ok:　　正常に動作した場合

[例]

（送信側）

pm16c16.th□Stop　　モータ th を減速停止します

（返信されてくる文字列）

pm16c16.th>term1□@Stop□Ok:

StopEmergency

このコマンドを送信すると対応するモータが緊急停止します。

このコマンドはコントローラが Local 状態の場合操作は無視されます。

[リプライ・メッセージ]

@StopEmergency□Ok:　　正常に動作した場合

[例]

（送信側）

pm16c16.th□StopEmergency　　モータ th を緊急停止します

（返信されてくる文字列）

pm16c16.th>term1□@StopEmergency□Ok:

◆モータ駆動中の速度変更コマンド

SetSpeedCurrent

このコマンドを送信することで駆動中のモータの速度を変更します。

[引数]

モータの速度の値（PPS）※　　正の数値文字列（＋符号なし）

[リプライ・メッセージ]

@SetSpeedCurrent□（引数）□Ok:　　正常に動作した場合

@SetSpeedCurrent□（引数）□Er:□(エラー情報)　なんらかの理由で変更できなかった場合

[例]

（送信側）

pm16c16.th□SetSpeedCurrent□10000　　モータ th の速度値（PPS）を
10000 で設定します

（返信されてくる文字列）

pm16c16.th>term1□@SetSpeedCurrent□10000□Ok:

◆モータ現在状況確認コマンド

以下、モータの現在の状況を確認するコマンドです。

GetValue

このコマンドを送信することでモータの現在位置を絶対値で取得します。

[戻り値]

モータ位置の絶対値

-2147483647 から 2147483647 までの数値文字列

[例]

（送信側）

pm16c16.th□GetValue　　モータ th の現在位置の値を取得します

（返信されてくる文字列）

pm16c16.th>term1□@GetValue□10000　モータ th の現在位置の値 10000 が返ってきます

IsBusy

このコマンドを送信することでモータが駆動しているか否かのデータを取得します。

[リプライ・メッセージ]

@Busy□0　　モータが停止状態

@Busy□1　　モータが駆動中の状態

[例]

（送信側）

pm16c16.th□IsBusy　　モータ th の駆動状況を確認します

（返信されてくる文字列）

pm16c16.th >term1□@IsBusy□0　　モータ th が停止状態の場合

pm16c16.th>term1□@IsBusy□1　　モータ th が駆動中の状態の場合

GetLimitStatus

このコマンドを送信することでモータの現在のリミットステータス値を取得します。

[戻り値]

0 から 7 の数値　　下記 bit より構成されます

4bit	Unused	(0: unused)
3bit	HP Status	（1: on　0: off）
2bit	CCW LS Status	（1: on　0: off）
1bit	CW LS Status	（1: on　0: off）

[例]

（送信側）

pm16c16.th□GetLimitStatus　　モータ th の現在のリミットステータス値を取得します

（返信されてくる文字列）

pm16c16.th>term1□@GetLimitStatus□0

◆モータイベント

_ChangedValue イベント

このイベントは対応するモータの現在位置が変化した場合、もしくは”flushdata”コマンド、”flushdatatome”コマンドを発行した場合に STARS Server から送られます。

[イベント・メッセージ]

_ChangedValue□（モータの現在位置）

[例]

（送信側）

System flgon pm16c16.th	モータ th のイベント監視を開始します
pm16c16 flushdata	イベントメッセージの強制送信を指示します

（返信されてくる文字列）

System>term1□@flgon Node pm16c16.th has been registered.

pm16c16.th>term1□_ChangedIsBusy□0

pm16c16.th>term1□_ChangedValue□10000

pm16c16>term1□@flushdata□Ok:

_ChangedIsBusy イベント

このイベントは対応するモータの駆動状態が切り替わった場合、もしくは”flushdata”コマンド、”flushdatatome”コマンドを発行した場合に STARS Server から送られます。

[イベント・メッセージ]

_ChangedIsBusy□1	モータが駆動中の状態
_ChangedIsBusy□0	モータが停止状態

[例]

（送信側）

System flgon pm16c16.th	モータ th のイベント監視を開始します
pm16c16 flushdata	イベントメッセージの強制送信を指示します

（返信されてくる文字列）

System>term1□@flgon Node pm16c16.th has been registered.

pm16c16.th>term1□_ChangedIsBusy□0

pm16c16.th>term1□_ChangedValue□10000

pm16c16>term1□@flushdata□Ok:

_ChangedLimitStatus イベント（オプション機能）

このイベントは対応するモータのリミットステータスが変化した場合、もしくは”flushdata”コマンド、”flushdatatome”コマンドを発行した場合に STARS Server から送られます。
当イベントはデフォルト設定では送信されません。有効にするには、プログラム起動オプション「--limitstatuschannellist」を用いて、当イベント送信の対象とするモータのリストを指定してください。

[イベント・メッセージ]

_ChangedLimitStatus□（モータの現在のリミットステータス）

[例]

（送信側）

System flgon pm16c16.th	モータ th のイベント監視を開始します
pm16c16 flushdata	イベントメッセージの強制送信を指示します

（返信されてくる文字列）

System>term1□@flgon Node pm16c16.th has been registered.
pm16c16.th>term1□_ChangedIsBusy□0
pm16c16.th>term1□_ChangedValue□10000
pm16c16.th>term1□_ChangedLimitStatus□0
pm16c16>term1□@flushdata□Ok:

＜補足＞PM16C-16 について PM16C-04X からの主な変更点

全モータの同時駆動が可能

PM16C-04 製品群までは、16 あるモータのうち同時駆動可能なモータの数が 4 つまでに制限されていましたが、PM16C-16 製品群では同時駆動数の制限がなくなりました。

タイミングアウト出力機能で使用するモータ No.について

PM16C-04X 製品群 同様、PM16C-16 でも最大 4 ポート(TP0 から TP4 まで)のモータ動作中の TTL 出力がおこなえます。

デフォルト設定では、4 ポートのモータ No.は本体のフロントパネルのディスプレイの Pos.A から Pos.D のモータ No.に相当しますが、両者のモータ No.を別々に設定することが可能になりました。

２補軸ドライブ機能関連（コントローラコマンド）が削除されました

PM16C-04X 製品群にあった、直線、円、円弧の軌跡を描くようにモータを動作させる機能は、PM16C-16 製品群では削除されました。これに伴い PM16C-04X 製品群で使用可能だった当機能関連の STARS コマンドを削除しました。