Melec

STEPPING & SERVO MOTOR CONTROLLER'S OPTION

MPL-31-01V3.00/PCIW64
MPL-31-02V2.00/PCIW64
取扱説明書 応用機能編
(設計者用)

(PCI C-VX870v1シリーズ Windows用デバイスドライバ)

USER'S MANUAL

本製品を使用する前に、この取扱説明書を良く読んで十分に理解してください。
この取扱説明書は、いつでも取り出して読めるように保管してください。

MN0513

はじめに

このデバイスドライバ「取扱説明書 応用機能編 」は、「ステッピングモータ、およびサーボモータ用コントローラ C-VX870v1 シリーズ」を正しく安全に使用していただくために、ステッピングモータ、あるいはサーボモータを使った制御装置の設計を担当される方を対象に、Windows における応用機能について説明しています。各ボードコントローラの「取扱説明書」ならびにデバイスドライバ「取扱説明書」と同様に、本デバイスドライバ「取扱説明書 応用機能編 」を良く読んで十分に理解してください。
このデバイスドライバ「取扱説明書 応用機能編 」は、いつでも取り出して読めるように保管してください。

安全設計に関するお願い

■本資料に記載されている製品および製品仕様は、改良などにより予告なく変更することがあります。

■本資料に記載される技術情報は、製品の代表的動作・応用を説明するためのものであり、その使用に際して当社および第三者の知的財産権その他の権利に対する保証または実施権の許諾を行うものではありません。

■本資料に記載されている回路、ソフトウェア、およびこれらに関連する情報を使用する場合は、お客様の機器およびシステム全体で十分に評価し、お客様の責任において適用可否を判断してください。

■半導体ならびに半導体を使用した製品は、ある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。本製品の故障または誤動作により、人身事故、火災事故、社会的な損害などを生じさせないように、お客様の責任において、お客様の機器またはシステムに必要な安全設計を行うことをお願いします。

■本製品は、一般工業向けの汎用品として設計・製造されていますので、航空機器、航空宇宙機器、海底中継機器、原子力制御システム、輸送機器(車両、船舶等)、交通用信号機器、防災・防犯機器、安全装置、医療機器など、人命や財産に多大な影響が予想される用途には使用しないでください。

■本製品を改造、改変、複製等しないでください。

■輸出に際しては、「外国為替および外国貿易法」など適用される輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続きを行ってください。本製品または本資料に記載されている技術情報を、大量破壊兵器の開発等の目的、軍事利用の目的、その他軍事用途の目的で使用しないでください。
また、本製品を国内外の法令および規制により製造・使用・販売を禁止されている機器に使用することはできません。

■本製品の環境適合性などの詳細につきましては、必ず弊社営業窓口までお問い合わせください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令など適用される環境関連法令を十分調査の上、かかる法令に適合するようにご使用ください。
お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は一切その責任は負いません。

安全に関する事項の記述方法について

本製品は正しい方法で取り扱うことが大切です。
誤った方法で使用された場合、予期しない事故を引き起こし、人身への障害や財産の損壊などの被害を被るおそれがあります。
そのような事故の多くは、危険な状況を予め知っていれば回避することができます。
そのため、このデバイスドライバ「取扱説明書 応用機能編 」では危険な状況が予想できる場合には、注意事項が記述してあります。
それらの記述は、次のようなシンボルマークとシグナルワードで示しています。

⚠ 警告	取り扱いを誤った場合に死亡、または重傷を負うおそれのある警告事項を示します。
⚠ 注意	取り扱いを誤った場合に、軽傷を負うおそれや物的損害が発生するおそれがある注意事項を示します。

御使用の前に

■ 入出力仕様ならびに接続に関する取り扱いについては、各コントローラの「取扱説明書」をご覧ください。

■ デバイスドライバを用いた基本仕様については、デバイスドライバ「取扱説明書」をご覧ください。

■ C-VX870v1 シリーズは各軸を独立で制御できるため、各軸を以下のように呼称します。

製品名	軸数	1 軸目	2 軸目	3 軸目	4 軸目	5 軸目	6 軸目	7 軸目	8 軸目	9 軸目	10 軸目	11 軸目	12 軸目
C-VX870v1	4 軸	X 軸	Y 軸	Z 軸	A 軸	―	―	―	―	―	―	―	―
C-VX871v1	6 軸	X 軸	Y 軸	Z 軸	A 軸	B 軸	C 軸	―	―	―	―	―	―
C-VX872v1	8 軸	X1 軸	Y1 軸	Z1 軸	A1 軸	X2 軸	Y2 軸	Z2 軸	A2 軸	―	―	―	―
C-VX873v1	12 軸	X1 軸	Y1 軸	Z1 軸	A1 軸	B1 軸	C1 軸	X2 軸	Y2 軸	Z2 軸	A2 軸	B2 軸	C2 軸
C-VX870Ev1	4 軸	X 軸	Y 軸	Z 軸	A 軸	―	―	―	―	―	―	―	―
C-VX871Ev1	6 軸	X 軸	Y 軸	Z 軸	A 軸	B 軸	C 軸	―	―	―	―	―	―
C-VX872Ev1	8 軸	X1 軸	Y1 軸	Z1 軸	A1 軸	X2 軸	Y2 軸	Z2 軸	A2 軸	―	―	―	―
C-VX873Ev1	12 軸	X1 軸	Y1 軸	Z1 軸	A1 軸	B1 軸	C1 軸	X2 軸	Y2 軸	Z2 軸	A2 軸	B2 軸	C2 軸

以降、原則として X 軸についてのみ説明します。

はじめに
安全設計に関するお願い
安全に関する事項の記述方法について
御使用の前に

目　　次　　PAGE

目　次

目　次

目　次

PAGE

本版で改訂された主な箇所

1．ドライバ仕様

1-1. 概要

MPL-31-01v3.00/PCIW64 および MPL-31-02v2.00/PCIW64 は、Windows パソコンの Windows 上で PCI または PCI Express を介して弊社製ステッピング ＆ サーボモータコントローラボード C-VX870v1 シリーズ を動作させるための DLL ベースのドライバ関数です。
各関数を使用してボードコントローラ上の MCC07E、汎用 I/O、HARD CONFIGURATION（応用機能）の各 PORT にアクセスし、モータコントロールを行います。

MCC07E　PORT/ 各軸	汎用 I/O PORT	HARD CONFIGURATION PORT
・DRIVE COMMAND PORT ・DRIVE DATA1 PORT ・DRIVE DATA2 PORT ・STATUS1 PORT ・STATUS2 PORT ・STATUS3 PORT ・STATUS4 PORT ・STATUS5 PORT	・汎用出力 1 PORT ・汎用出力 2 PORT （$\overline{\text{OUTn0}}$ ～ $\overline{\text{OUTn3}}$） ・汎用入力 1 PORT ・汎用入力 2 PORT （$\overline{\text{INn0}}$ ～ $\overline{\text{INn3}}$）	・HARD CONFIG COMMAND PORT ・HARD CONFIG DATA1 PORT ・HARD CONFIG DATA2 PORT ・HARD CONFIG DATA3 PORT ・HARD CONFIG STATUS1 PORT ・HARD CONFIG STATUS2 PORT ・HARD CONFIG STATUS3 PORT ・HARD CONFIG STATUS4 PORT
各コントローラに対応	4 軸と 8 軸コントローラに対応 汎用入出力 2 PORT は 8 軸コントローラのみ対応	各コントローラに対応（応用機能）

■ C-VX870v1 シリーズのボードコントローラには、弊社製チップコントローラ MCC07E を搭載しています。
MCC07E へのアクセスはデバイスオープン関数で各軸のハンドルを取得した後、その受け渡しによって各 PORT へのアクセスを行います。
アプリケーション終了時はデバイスクローズ関数を実行して終了させます。

・各軸 MCC07E へのアクセスは、マルチスレッドに対応しています。

■ 4 軸、8 軸の製品には上記 MCC07E とは独立した汎用 I/O があります。
汎用 I/O へのアクセスは、汎用 I/O オープン関数でハンドルを取得した後、その受け渡しによって汎用 I/O の書き込み/読み出しを行います。
アプリケーション終了時は汎用 I/O クローズ関数を実行して終了させます。

■全ての製品に各軸 MCC07E とボードコントローラの入出力間の接続を結ぶ HARD CONFIGRATION ブロックがあります。HARD CONFIGRATION ブロックへのアクセスは、各軸ハンドルの受け渡しによって各 PORT へのアクセスを行います。

・HARD CONFIGURATION ブロックへのアクセスは、マルチスレッドに対応していません。
マルチスレッドで使用する場合、HARD CONFIGURATION ブロックへのアクセスは注意して行ってください。

■各軸 MCC07E からの割り込み発生による処理を行うための割り込み関数を用意しています。

■ MCC07E コマンドによる円弧補間ドライブを行う際に必要となる短軸パルス数を算出する関数を用意しています。

■以下についての詳細は、デバイスドライバ取扱説明書をご覧ください。
・サポート OS や言語他
・ソフト開発に必要なファイル
・デバイスドライバの制限事項
・その他のデバイス関数

1-2. 割り込み

(1)割り込みの通知

割り込みの通知方法は、次の 3 つが使用できます。

通知方法	説明
コールバック	割り込みが発生したときに、設定された関数をコールバックします。
メッセージ	割り込みが発生したときに、設定されたウィンドウのウィンドウプロシージャにメッセージをポストします。
イベント	割り込みが発生したときに、設定されたイベントオブジェクトがシグナル状態になります。

(2)割り込み処理の手順

ユーザアプリケーションで割り込みを使用する場合、次の手順で行ってください。

■ DRIVE 終了割り込み(RDYINT)使用の場合

① RDYINT の出力仕様の設定を行います。

・RDYINT ･･･ SPEC INITIALIZE2 COMMAND

注. RDYINT TYPE0_1 を次の設定にしてください。

『DRIVE STATUS1 PORT の DRVEND=0 → 1 のエッジ検出で RDYINT=1 にする』

②割り込み通知方法を設定します。

コールバック使用の場合 ･･･ 割り込みコールバック設定関数
メッセージ使用の場合 ･･･ 割り込みメッセージ設定関数
イベント使用の場合 ･･･ 割り込みイベント設定関数

③割り込みオープン関数で割り込みをオープンします。

④割り込みが検出されると通知方法に従い、通知処理が実行されます。

⑤ユーザアプリケーション終了時には、割り込みクローズ関数で割り込みをクローズした後、
デバイスをクローズし、ユーザアプリケーションを終了してください。

注 1. RDYINT TYPE0_1 を『DRIVE STATUS1 PORT の DRVEND=0 → 1 のエッジ検出で RDYINT=1 にする』以外に設定することはできません。
設定された場合、デバイスドライバでチェックを行い、『出力しない』に補正して設定します。

注 2. ORIGIN ドライブ中の DRIVE 終了割り込み(RDYINT)は、最終の DRIVE 終了割り込み(RDYINT)のみ有効となり、ユーザアプリケーションへの割り込みの通知が行われます。最終以外の DRIVE 終了割り込みは、デバイスドライバ内部で処理され割り込みの通知は行われません。

注 3. エラー出力機能で停止機能を ERROR=1 の発生要因に設定してコマンド予約機能を使用する場合、
ERROR=1 が発生して DRIVE が終了しても DRIVE 終了割り込み(RDYINT)が発生しない場合があります。
エラー出力機能で停止機能を ERROR=1 の発生要因に設定してコマンド予約機能を使用する場合、
DRIVE 終了割り込み(RDYINT)は使用しないでください。
DRIVE 終了の確認は STATUS1 PORT の BUSY=0 をポーリングで確認してください。

■カウンタ割り込み(ADRINT/CNTINT/DFLINT)使用の場合

①カウンタの COUNTER COMPARE REGISTER の設定を行います。

②カウンタの機能設定を行います。

ADRINT 使用の場合 ･･･ ADDRESS COUNTER INITIALIZE1 COMMAND
注. ADRINT_TYPE0_1 は必ず次の設定にしてください。
・COMP1,2,3 の一致出力の出力仕様 :『一致出力をエッヂラッチして出力』
・クリア条件 :『INT FACTOR CLR COMMAND の ADRINT INT CLR=1 の実行でクリア』
ADDRESS COUNTER INITIALIZE2 COMMAND

CNTINT 使用の場合 ･･･ PULSE COUNTER INITIALIZE1 COMMAND
注.CNTINT TYPE0_1 は、必ず次の設定としてください。
・COMP1,2,3 の一致出力の出力仕様 :『一致出力をエッヂラッチして出力』
・クリア条件 :『INT FACTOR CLR COMMAND の CNTINT INT CLR=1 の実行でクリア』
PULSE COUNTER INITIALIZE2 COMMAND

DFLINT 使用の場合 ･･･ DFL COUNTER INITIALIZE1 COMMAND
注.DFLINT TYPE0_1 は、必ず次の設定としてください。
・COMP1,2,3 の一致出力の出力仕様 :『一致出力をエッヂラッチして出力』
・クリア条件 :『INT FACTOR CLR COMMAND の DFLINT INT CLR=1 の実行でクリア』
DFL COUNTER INITIALIZE2 COMMAND
DFL COUNTER INITIALIZE3 COMMAND

③割り込み通知方法を設定します。
コールバック使用の場合 ･･･ 割り込みコールバック設定関数
メッセージ使用の場合 ･･･ 割り込みメッセージ設定関数
イベント使用の場合 ･･･ 割り込みイベント設定関数

④割り込みオープン関数で割り込みをオープンします。

⑤割り込みがデバイスドライバ内部で検出されると通知方法に従い、通知処理が実行されます。

⑥ユーザアプリケーション終了時には、割り込みクローズ関数で割り込みをクローズした後、
デバイスをクローズし、ユーザアプリケーションを終了してください。

■ DALM/nCOMREG FL/COMREG EP 割り込み使用の場合

①割り込み通知方法を設定します。
コールバック使用の場合 ･･･ 割り込みコールバック設定関数
メッセージ使用の場合 ･･･ 割り込みメッセージ設定関数
イベント使用の場合 ･･･ 割り込みイベント設定関数

②割り込みオープン関数で割り込みをオープンします。

③割り込みがデバイスドライバ内部で検出されると通知方法に従い、通知処理が実行されます。

④ユーザアプリケーション終了時には、割り込みクローズ関数で割り込みをクローズした後、
デバイスをクローズし、ユーザアプリケーションを終了してください。

注 1. ORIGIN ドライブ中に nCOMREG FL、COMREG EP 割り込みは、無効となりますので
御注意ください。

２．関数リファレンス

2-1. デバイス関数

2-1-1. 割り込み関数

デバイス割り込みオープン関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたデバイスの割り込みをオープンします。
当関数が実行されると割り込みコールバック設定関数、割り込みメッセージ設定関数、割り込みイベント設定関数で設定されている処理が有効になります。

書　式

C言語　BOOL MCO7_OpenDevInt(DWORD *hDev*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_OpenDevInt(ByVal *hDev* As Integer, ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.OpenDevInt(uint *hDev*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ デバイスハンドルを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

デバイス割り込みクローズ関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたデバイスの割り込みをクローズします。
当関数が実行されると割り込みコールバック設定関数、割り込みメッセージ設定関数、割り込みイベント設定関数で設定されている処理が無効になります。

書　式

C言語　BOOL MCO7_CloseDevInt(DWORD *hDev*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_CloseDevInt(ByVal *hDev* As Integer, ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.CloseDevInt(uint *hDev*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ デバイスハンドルを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

割り込みコールバック設定関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたデバイスで、指定された割り込みが発生した場合の、コールバック関数を設定します。
当関数を複数回実行することにより、割り込みごとにそれぞれ異なるコールバック関数を設定することができます。

書　式

C言語　BOOL MC07_SetIntCallBack(DWORD *hDev*, DWORD *IntFactor*, PLPMC07CALLBACK *Func*, DWORD *UserData*, MC07_S_RESULT **psResult*);

VB.NET　Function MC07_SetIntCallBack(ByVal *hDev* As Integer, ByVal *IntFactor* As Integer, ByVal *Func* As PLPMC07CALLBACK, ByVal *UserData* As Integer, ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET　bool MC07.SetIntCallBack(uint *hDev*, uint *IntFactor*, PLPMC07CALLBACK *Func*, uint *UserData*, ref MC07_S_RESULT *psResult*);

引　数

hDev　　･･･ デバイスハンドルを指定します。
IntFactor　　･･･ 割り込みを指定します。(複数の割り込みの指定が可能です。)

*IntFactor*の値	割り込み	*IntFactor*の値	割り込み
MC07_INT_FACTOR_RDYINT	RDYINT	MC07_INT_FACTOR_ADRINT	ADRINT
MC07_INT_FACTOR_COMREG_EP	COMREG EP	MC07_INT_FACTOR_CNTINT	CNTINT
MC07_INT_FACTOR_nCOMREG_FL	nCOMREG FL	MC07_INT_FACTOR_DFLINT	DFLINT
MC07_INT_FACTOR_DALM	DALM		

Func　　･･･ コールバック関数を指定します。
UserData　　･･･ ユーザーデータを指定します。
psOutResult　　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

コールバック関数

書　式

C言語　VOID WINAPI Func(DWORD *Status*, DWORD *UserData*);
C#.NET　void Func(int *Status*, int UserData);
VB.NET　Sub Func(ByVal *Status* As Integer, ByVal UserData As Integer)

引　数

Status　　･･･ 割り込みステータスを格納します。

割り込み	引数*Status*の値
RDYINT	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0 注1.
COMREG EP	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
nCOMREG FL	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
DALM	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
ADRINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0
CNTINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0
DFLINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0

UserData　　･･･ ユーザデータを格納します。
注1. ORIGINドライブ終了時は、ORIGINドライブSTATUSの内容が下位16ビットに格納されます。

割り込みメッセージ設定関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたデバイスで、指定された割り込みが発生した場合にポストするメッセージを設定します。
当関数を複数回実行することにより、割り込みごとにそれぞれに、異なるメッセージを設定することもできます。

書 式

C言語 BOOL MC07_SetIntMessage(DWORD *hDev*, DWORD *IntFactor*, HWND *hWnd*, UINT *Message*,
DWORD *UserData*, MC07_S_RESULT **psResult*);

引 数

hDev ・・・ デバイスハンドルを指定します。

IntFactor ・・・ 割り込みを指定します。(複数の割り込みの指定が可能です。)

*IntFactor*の値	割り込み	*IntFactor*の値	割り込み
MC07_INT_FACTOR_RDYINT	RDYINT	MC07_INT_FACTOR_ADRINT	ADRINT
MC07_INT_FACTOR_COMREG_EP	COMREG EP	MC07_INT_FACTOR_CNTINT	CNTINT
MC07_INT_FACTOR_nCOMREG_FL	nCOMREG FL	MC07_INT_FACTOR_DFLINT	DFLINT
MC07_INT_FACTOR_DALM	DALM		

hWnd ・・・ メッセージのポスト先のウィンドウハンドルを指定します。
Message ・・・ メッセージコードを指定します。
UserData ・・・ ユーザーデータを指定します。(LPARAMに格納されます。)
psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

■ 割り込みステータス
メッセージのポスト先で、割り込みステータスを確認することができます。
割り込みステータスは、第一メッセージパラメータ (WPARAM) に格納されます。

割り込み	割り込みステータス
RDYINT	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0 注1.
COMREG EP	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
nCOMREG FL	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
DALM	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
ADRINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0
CNTINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0
DFLINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0

注1. ORIGINドライブ終了時は、ORIGINドライブSTATUSの内容が下位16ビットに格納されます。

割り込みイベント設定関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたデバイスで、指定された割り込みが発生した場合にシグナル状態になるイベントを設定します。
当関数を複数回実行することにより、割り込みそれぞれに異なるイベントを設定することができます。
イベントがシグナル状態になった後は割り込みステータス読み出し関数により、割り込みステータスを確認します。

書　式

C言語　BOOL MCO7_SetIntEvent(DWORD *hDev*, DWORD *IntFactor*, HANDLE *hEvent*,
　　MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_SetIntEvent(ByVal *hDev* As Integer, ByVal *IntFactor* As Integer,
　　ByVal *hEvent* As Integer, ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.SetIntEvent(uint *hDev*, uint *IntFactor*, uint *hEvent*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ デバイスハンドルを指定します。
IntFactor　　･･･ 割り込みを指定します。(複数の割り込みの指定が可能です。)

*IntFactor*の値	割り込み	*IntFactor*の値	割り込み
MCO7_INT_FACTOR_RDYINT	RDYINT	MCO7_INT_FACTOR_ADRINT	ADRINT
MCO7_INT_FACTOR_COMREG_EP	COMREG EP	MCO7_INT_FACTOR_CNTINT	CNTINT
MCO7_INT_FACTOR_nCOMREG_FL	nCOMREG FL	MCO7_INT_FACTOR_DFLINT	DFLINT
MCO7_INT_FACTOR_DALM	DALM		

hEvent　　･･･ イベントオブジェクトのハンドルを指定します。
psResult　　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

割り込みステータス読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

割り込みステータスを読み出します。
割り込みイベント設定関数により割り込みの設定をした場合に、この関数により割り込みステータスを確認します。

書 式

C言語 BOOL MCO7_ReadIntStatus(DWORD *hDev*, DWORD *IntFactor*, DWORD **pStatus*,
MCO7_S_RESULT **psResult*);

VB.NET Function MCO7_ReadIntStatus(ByVal *hDev* As Integer, ByVal *IntFactor* As Integer,
ByRef *pStatus* As Integer, ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET bool MCO7.ReadIntStatus(uint *hDev*, uint *IntFactor*, ref uint *pStatus*,
ref MCO7_S_RESULT *psResult*);

引 数

hDev ・・・ デバイスハンドルを指定します。
IntFactor ・・・ 割り込みを指定します。

*IntFactor*の値	割り込み	*IntFactor*の値	割り込み
MCO7_INT_FACTOR_RDYINT	RDYINT	MCO7_INT_FACTOR_ADRINT	ADRINT
MCO7_INT_FACTOR_COMREG_EP	COMREG EP	MCO7_INT_FACTOR_CNTINT	CNTINT
MCO7_INT_FACTOR_nCOMREG_FL	nCOMREG FL	MCO7_INT_FACTOR_DFLINT	DFLINT
MCO7_INT_FACTOR_DALM	DALM		

pStatus ・・・ 割り込みステータスが格納される変数のポインタを指定します。

割り込み	割り込みステータス
RDYINT	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0 注1.
COMREG EP	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
nCOMREG FL	下位16ビットにDRIVE STATUS1 PORTの内容、上位16ビットは0
DALM	下位16ビットにDRIVE STATUS2 PORTの内容、上位16ビットは0
ADRINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0
CNTINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0
DFLINT	下位16ビットにDRIVE STATUS4 PORTの内容、上位16ビットは0

注1. ORIGINドライブ終了時は、ORIGINドライブSTATUSの内容が下位16ビットに
格納されます。

psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

割り込み設定クリア関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

割り込みコールバック設定関数、割り込みメッセージ設定関数、割り込みイベント設定関数の設定をクリアします。
各設定関数で設定内容を変更したい場合、割り込み設定クリア関数でクリアしないと、設定の変更を行うことができません。

書　式

C言語　BOOL MCO7_ClearIntSet(DWORD *hDev*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_ClearIntSet(ByVal *hDev* As Integer, ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.ClearIntSet(uint *hDev*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ デバイスハンドルを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

2-1-2. HARD CONFIG 関数

■ HARD CONFIG COMMAND PORT

HARD CONFIG COMMAND を書き込む PORT です。この PORT に HARD CONFIG COMMAND を書き込むと、データの設定 または機能の実行を行います。
書き込む HARD CONFIG COMMAND は下位 8 ビットのみ有効です。上位 8 ビットは無視します。
HARD CONFIG COMMAND の書き込みは常時可能です。

■ HARD CONFIG DATA 1,2,3 PORT（書き込み）

HARD CONFIG COMMAND の設定データ、または機能を行うデータを書き込む PORT です。
この PORT への書き込みは常時可能です。

■ HARD CONFIG STATUS PORT

● HARD CONFIG STATUS1 PORT

MANUAL 信号の切り替えが可能かホストから確認できる $\overline{\text{MAN RDY}}$ 出力信号の状態を読み出す PORT です。
読み出しは常時可能です。

● HARD CONFIG STATUS2 PORT

各軸の PAUSE 状態を表示する PORT です。
読み出しは常時可能です。

● HARD CONFIG STATUS3 PORT

各軸の STBY 状態を表示する PORT です。
読み出しは常時可能です。

● HARD CONFIG STATUS4 PORT

特殊 I/O コネクタ（J3）の $\overline{\text{SIGNAL INx}}$ 入力信号と $\overline{\text{SIGNAL\_OUTx}}$ 出力信号の状態を表示する PORT です。
読み出しは常時可能です。

■ HARD CONFIG DATA 1,2,3 PORT（読み出し）

HARD CONFIG の設定データを読み出す PORT です。読み出しは常時可能です。
HARD CONFIG SET DATA READ コマンドを HARD CONFIG COMMAND PORT に書き込むと、該当データを HARD CONFIG DATA1,2,3 PORT にセットします。
HARD CONFIG DATA1,2 3 PORT にセットしたデータは、次の HARD CONFIG SET DATA READ コマンドの書き込みまで保持します。

HARD CONFIG COMMAND一括書き込み関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG DATA1 PORT、HARD CONFIG DATA2 PORT、HARD CONFIG DATA3 PORTにデータ構造体のデータを書き込んだ後、HARD CONFIG COMMAND PORTにコマンドコードを書き込みます。
書き込みは常時可能です。

書　式

C言語　BOOL MCO7_IWHardConfig(DWORD *hDev*, WORD *Cmd*, MCO7_S_DATA **psData*,
　　MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_IWHardConfig(ByVal *hDev* As Integer, ByVal *Cmd* As Short,
　　ByRef *psData* As MCO7_S_DATA, ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.IWHardConfig(uint *hDev*, ushort *Cmd*, ref MCO7_S_DATA *psData*,
　　ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
Cmd　　･･･ 書き込むコマンドコードを指定します。
psData　　･･･ 書き込むデータが格納されているデータ構造体のポインタを指定します。
psResult　　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG COMMAND PORT書き込み関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG COMMAND PORTにコマンドコードを書き込みます。
書き込みは常時可能です。

書　式

C言語　BOOL MC07_BWHardConfigCommand(DWORD *hDev*, WORD **pCmd*, MC07_S_RESULT **psResult*);

VB.NET　Function MC07_BWHardConfigCommand(ByVal *hDev* As Integer, ByRef *pCmd* As Short,
ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET　bool MC07.BWHardConfigCommand(uint *hDev*, ref ushort *pCmd*, ref MC07_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pCmd　　･･･ 書き込むコマンドコードが格納されている変数のポインタを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG DATA1 PORT書き込み関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG DATA1 PORTにデータを書き込みます。
書き込みは常時可能です。

書　式

C言語　BOOL MCO7_BWHardConfigData1(DWORD *hDev*, WORD **pData*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_BWHardConfigData1(ByVal *hDev* As Integer, ByRef *pData* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.BWHardConfigData1(uint *hDev*, ref ushort *pData*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pData　　･･･ 書き込むデータが格納されている変数のポインタを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG DATA2 PORT書き込み関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG DATA2 PORTにデータを書き込みます。
書き込みは常時可能です。

書　式

C言語　BOOL MCO7_BWHardConfigData2(DWORD *hDev*, WORD **pData*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_BWHardConfigData2(ByVal *hDev* As Integer, ByRef *pData* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.BWHardConfigData2(uint *hDev*, ref ushort *pData*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pData　　･･･ 書き込むデータが格納されている変数のポインタを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG DATA3 PORT書き込み関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG DATA3 PORTにデータを書き込みます。
書き込みは常時可能です。

書　式

C言語　BOOL MCO7_BWHardConfigData3(DWORD *hDev*, WORD **pData*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_BWHardConfigData3(ByVal *hDev* As Integer, ByRef *pData* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.BWHardConfigData3(uint *hDev*, ref ushort *pData*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pData　　･･･ 書き込むデータが格納されている変数のポインタを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG STATUS1 PORT読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたモータコントローラボードのHARD CONFIG STATUS1 PORTを読み出します。
MAN RDY出力の状態を読み出すPORTです。読み出しは常時可能です。

書 式

C言語 BOOL MCO7_BRHardConfigStatus(DWORD *hDev*, WORD **pStatus*, MCO7_S_RESULT **psResult*);

VB.NET Function MCO7_BRHardConfigStatus(ByVal *hDev* As Integer, ByRef *pStatus* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET bool MCO7.BRHardConfigStatus(uint *hDev*, ref ushort *pStatus*,
ref MCO7_S_RESULT *psResult*);

引 数

hDev ・・・ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pStatus ・・・ 読み出した内容が格納される変数のポインタを指定します。
HARD CONFIG STATUS1 PORTの内容は、次に示す通りです。

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	(不定)

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	$\overline{\text{MAN RDY}}$	0	0	0	(不定)

D0 : $\overline{\text{MAN RDY}}$
1 : MAN入力信号受付可能の状態を示します。(初期値)
0 : MAN入力信号受付不可の状態を示します。

・MAN RDYの状態はMAN MASKコマンドで操作することができます。

psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG STATUS2 PORT読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたモータコントローラボードのHARD CONFIG STATUS2 PORTを読み出します。
各軸のPAUSE状態を表示するPORTです。読み出しは常時可能です。

書 式

C言語 BOOL MCO7_BRHardConfigPauseStatus(DWORD *hDev*, WORD **pStatus*, MCO7_S_RESULT **psResult*);

VB.NET Function MCO7_BRHardConfigPauseStatus(ByVal *hDev* As Integer, ByRef *pStatus* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET bool MCO7.BRHardConfigPauseStatus(uint *hDev*, ref ushort *pStatus*,
ref MCO7_S_RESULT *psResult*);

引 数

hDev ・・・ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pStatus ・・・ 読み出した内容が格納される変数のポインタを指定します。
HARD CONFIG STATUS2 PORTの内容は、次に示す通りです。

D15	D14	D13	D12	D11	D10	D9	D8	
0	0	C2 PAUSE	B2 PAUSE	A2 PAUSE	Z2 PAUSE	Y2 PAUSE	X2 PAUSE	C-VX872(E)v1 C-VX873(E)v1
		0	0	0	0	0	0	C-VX870(E)v1 C-VX871(E)v1

D7	D6	D5	D4	D3	D2	D1	D0	
0	0	C1 PAUSE	B1 PAUSE	A1 PAUSE	Z1 PAUSE	Y1 PAUSE	X1 PAUSE	C-VX872(E)v1 C-VX873(E)v1
		C PAUSE	B PAUSE	A PAUSE	Z PAUSE	Y PAUSE	X PAUSE	C-VX870(E)v1 C-VX871(E)v1

C-VX870(E)v1,C-VX872(E)v1
C-VX871(E)v1,C-VX873(E)v1

D0-D5 : X-C PAUSE(X1-C1 PAUSE)
D8-D13 : X2-C2 PAUSE
1：該当軸が待機指令中であることを示します。

・各軸DRIVE STATUS1 PORTのPAUSEと同じです。

psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG STATUS3 PORT読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたモータコントローラボードのHARD CONFIG STATUS3 PORTを読み出します。
各軸のSTBY状態を表示するPORTです。読み出しは常時可能です。

書 式

C言語 BOOL MCO7_BRHardConfigStbyStatus(DWORD *hDev*, WORD **pStatus*, MCO7_S_RESULT **psResult*);

VB.NET Function MCO7_BRHardConfigStbyStatus(ByVal *hDev* As Integer, ByRef *pStatus* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET bool MCO7.BRHardConfigStbyStatus(uint *hDev*, ref ushort *pStatus*,
ref MCO7_S_RESULT *psResult*);

引 数

hDev ･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pStatus ･･･ 読み出した内容が格納される変数のポインタを指定します。
HARD CONFIG STATUS3 PORTの内容は、次に示す通りです。

D15	D14	D13	D12	D11	D10	D9	D8	
0	0	C2 STBY (C2 OUT A)	B2 STBY (B2 OUT A)	A2 STBY (A2 OUT A)	Z2 STBY (Z2 OUT A)	Y2 STBY (Y2 OUT A)	X2 STBY (X2 OUT A)	C-VX872(E)v1 C-VX873(E)v1
		0	0	0	0	0	0	C-VX870(E)v1 C-VX871(E)v1

D7	D6	D5	D4	D3	D2	D1	D0	
0	0	C1 STBY (C1 OUT A)	B1 STBY (B1 OUT A)	A1 STBY (A1 OUT A)	Z1 STBY (Z1 OUT A)	Y1 STBY (Y1 OUT A)	X1 STBY (X1 OUT A)	C-VX872(E)v1 C-VX873(E)v1
		C STBY (C OUT A)	B STBY (B OUT A)	A STBY (A OUT A)	Z STBY (Z OUT A)	Y STBY (Y OUT A)	X STBY (X OUT A)	C-VX870(E)v1 C-VX871(E)v1

C-VX870(E)v1,C-VX872(E)v1
C-VX871(E)v1,C-VX873(E)v1

D0-D5 : X-C STBY(X1-C1 STBY)
D8-D13 : X2-C2 STBY
1：該当軸のパルス出力の準備が完了した状態を示します。

・各軸DRIVE STATUS3 PORTのOUT Aと同じです。
・STBY状態を表示するために、全軸のOUT A信号にそれぞれSTBYフラグを出力する必要があります。

psResult ･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG STATUS4 PORT読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたモータコントローラボードのHARD CONFIG STATUS4 PORTを読み出します。
特殊I/Oコネクタ(J3)のSIGNAL INnx入力信号とSIGNAL_OUTnx出力信号の状態を表示するPORTです。
読み出しは常時可能です。

書 式

C言語 BOOL MCO7_BRHardConfigSigStatus(DWORD *hDev*, WORD **pStatus*, MCO7_S_RESULT **psResult*);

VB.NET Function MCO7_BRHardConfigSigStatus(ByVal *hDev* As Integer, ByRef *pStatus* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET bool MCO7.BRHardConfigSigStatus(uint *hDev*, ref ushort *pStatus*,
ref MCO7_S_RESULT *psResult*);

引 数

hDev ・・・ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pStatus ・・・ 読み出した内容が格納される変数のポインタを指定します。
HARD CONFIG STATUS4 PORTの内容は、次に示す通りです。

D15	D14	D13	D12	D11	D10	D9	D8	
0	0	SIGNAL IN21	SIGNAL IN20	SIGNAL OUT21 LATCH	SIGNAL OUT20 LATCH	SIGNAL OUT21	SIGNAL OUT20	C-VX872(E)v1 C-VX873(E)v1
		0	0	0	0	0	0	C-VX870(E)v1 C-VX871(E)v1

D7	D6	D5	D4	D3	D2	D1	D0	
0	0	SIGNAL IN11	SIGNAL IN10	SIGNAL OUT11 LATCH	SIGNAL OUT10 LATCH	SIGNAL OUT11	SIGNAL OUT10	C-VX872(E)v1 C-VX873(E)v1
		SIGNAL IN1	SIGNAL IN0	SIGNAL OUT1 LATCH	SIGNAL OUT0 LATCH	SIGNAL OUT1	SIGNAL OUT0	C-VX870(E)v1 C-VX871(E)v1

D0-D1 : SIGNAL OUT x(SIGNAL OUT1x)
D8-D9 : SIGNAL OUT2x
1 : SIGNAL OUTnx出力信号がアクティブレベルであることを示します。

D2-D3 : SIGNAL OUT x LATCH(SIGNAL OUT1x LATCH)
D10-D11 : SIGNAL OUT2x LATCH
1 : SIGNAL OUTnx出力信号のアクティブエッジを検出したことを示します。

・SIGNAL OUTnx LATCHはSIGNAL OUT LATCH CLRコマンドの実行でクリアします。

D4-D5 : SIGNAL IN x(SIGNAL IN1x)
D12-D13 : SIGNAL IN2x
1 : SIGNAL INnx入力信号がアクティブレベルであることを示します。

・MANUALモード時はSIGNAL INnxは0となります。

psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG DATA1 PORT読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG DATA1 PORTを読み出します。読み出しは常時可能です。

書　式

C言語　BOOL MCO7_BRHardConfigData1(DWORD *hDev*, WORD **pData*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_BRHardConfigData1(ByVal *hDev* As Integer, ByRef *pData* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.BRHardConfigData1(uint *hDev*, ref ushort *pData*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pData　　･･･ 読み出した内容が格納される変数のポインタを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG DATA2 PORT読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG DATA2 PORTを読み出します。読み出しは常時可能です。

書　式

C言語　BOOL MCO7_BRHardConfigData2(DWORD *hDev*, WORD **pData*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_BRHardConfigData2(ByVal *hDev* As Integer, ByRef *pData* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.BRHardConfigData2(uint *hDev*, ref ushort *pData*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pData　　･･･ 読み出した内容が格納される変数のポインタを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

HARD CONFIG DATA3 PORT読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたモータコントローラボードのHARD CONFIG DATA3 PORTを読み出します。読み出しは常時可能です。

書　式

C言語　BOOL MCO7_BRHardConfigData3(DWORD *hDev*, WORD **pData*, MCO7_S_RESULT **psResult*);

VB.NET　Function MCO7_BRHardConfigData3(ByVal *hDev* As Integer, ByRef *pData* As Short,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET　bool MCO7.BRHardConfigData3(uint *hDev*, ref ushort *pData*, ref MCO7_S_RESULT *psResult*);

引　数

hDev　　･･･ 対象とするモータコントローラボードのいずれかの軸のデバイスハンドルを指定します。
pData　　･･･ 読み出した内容が格納される変数のポインタを指定します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

2-1-3. WAIT 関数

COMREG NOT FULL WAIT関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたデバイスのコマンド予約レジスタが予約可能な状態(MCC07E STATUS1 PORT COMREG FULL BIT = 0) になるまで待機します。最大待ち時間を超えるとエラー終了します。
また、待機中にMCC07E STATUS1 PORT ERROR BIT = 1を検出した場合もエラー終了します。

書　式

C言語　BOOL MC07_BWaitComregNotFull(DWORD *hDev*, WORD *WaitTime*, MC07_S_RESULT **psResult*);

VB.NET　Function MC07_BWaitComregNotFull(ByVal *hDev* As Integer, ByVal *WaitTime* As Short,
ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET　bool MC07.BWaitComregNotFull(uint *hDev*, ushort *WaitTime*, ref MC07_S_RESULT *psResult*);

引　数

hDev　　･･･ デバイスハンドルを指定します。
WaitTime　･･･ 最大待ち時間を1ms単位で設定します。0を指定するとコマンド予約レジスタが予約可能な状態になるまで無限に待機します。
psResult　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

2-1-4. MPL 補間関数

円弧補間短軸PULSE数ゲット関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

MCO7Eコマンドによる円弧補間ドライブを行う際に必要となる短軸パルス数を算出する関数です。
指定された円弧の中心点相対アドレス、目的地相対アドレス、回転方向をもとに目的地の短軸座標までの
パルス数を計算し格納します。
座標は、モータの現在位置を座標中心(0,0)とした相対アドレスです。
計算の詳細については、「4-1-5.(4)円弧補間ドライブ」をご覧ください。
注. 次の場合、関数がエラー終了します。
- ・円弧の中心点座標が(0, 0)、または中心点と目的地が同一座標の場合
- ・円弧補間で求めた短軸PULSE数が-2,147,483,647～+2,147,483,647の範囲内でない場合
- ・円弧補間で指定出来ない目的地座標が指定された場合

書　式

C言語
```
BOOL MCO7_GetCirShortPulse( WORD Dir, MCO7_S_XY_POSITION *psCenterPosition,
        MCO7_S_XY_POSITION *psTragetPosition, LONG *pShortPulse,
        MCO7_S_RESULT *psResult );
```

VB.NET
```
Function MCO7_GetCirShortPulse(ByVal Dir As Short,
        ByRef psCenterPosition As MCO7_S_XY_POSITION,
        ByRef psTragetPosition As MCO7_S_XY_POSITION, ByRef pShortPulse As Integer,
        ByRef psResult As MCO7_S_RESULT) As Boolean
```

C#.NET
```
bool MCO7.GetCirShortPulse(ushort Dir, ref MCO7_S_XY_POSITION psCenterPosition,
        ref MCO7_S_XY_POSITION psTragetPosition, ref int pShortPulse,
        ref MCO7_S_RESULT psResult);
```

引　数

Dir	･･･ 円弧の回転方向を指定します。 MCO7_CCW : +(CCW)方向　　MCO7_CW : +(CW)方向
psCenterPosition	･･･ 中心点のX･Y相対座標(-8,388,607～+8,388,607)が格納されているポジション構造体のポインタを指定します。 中心点のX･Y座標は、現在位置を座標の中心(0,0)とした相対座標です。
psTragetPosition	･･･ 目的地のX･Y相対座標(-16,777,214～+16,777,214)が格納されているポジション構造体のポインタを指定します。 目的地のX･Y座標は、現在位置を座標の中心(0,0)とした相対座標です。
pShortPulse	･･･ 目的地の短軸座標までのパルス数が格納される変数のポインタを指定します。
psResult	･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

2-1-5. その他の関数

従来のデバイスドライバで用意されていたデータ構造体を使用した一括書き込み／読み出し関数です。
MCC07Eへの一括アクセスについては原則、基本機能に用意しているデータ構造体を使用しないで一括アクセスできる関数の使用を推奨します。

データ構造体

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

説　明

データを一括で読み書きするときに使用します。

■データを一括で読み書きするとき
・DRIVE DATA1 PORT、DRIVE DATA2 PORTのデータを一括で書き込むとき
・DRIVE DATA1 PORT、DRIVE DATA2 PORTのデータを一括で読み出すとき
・HARD CONFIG DATA1 PORT、HARD CONFIG DATA2 PORT、HARD CONFIG DATA3 PORTのデータを一括で書き込むとき

書　式

C言語
```
typedef struct MCO7_TAG_S_DATA {
    WORD MCO7_Data[4];
} MCO7_S_DATA;
```

VB.NET
```
Structure MCO7_S_DATA
    <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> Public MCO7_Data() As Short
    Public Sub Initialize()
        ReDim MCO7_Data(4)
    End Sub
End Structure
```

C#.NET
```
struct MCO7_S_DATA
{
     [MarshalAs( UnmanagedType.ByValArray, SizeConst=4 )] public ushort[] MCO7_Data;
           MCO7_S_DATA( ushort dummy )
     {
           MCO7_Data = new ushort[4];
     }
 }
```

メンバ

次に示すメンバは、C言語で表記しています。C言語の*MCO7_Data[0]*～*MCO7_Data[3]*は、Visual Basic.NETでは*MCO7_Data(0)*～*MCO7_Data(3)*、C#.NETでは*MCO7_Data[0]*～*MCO7_Data[3]* に対応します。

MCO7_Data[0]　・・・ DRIVE DATA1 PORT、HARD CONFIG DATA1 PORTのいずれかの内容を格納します。
MCO7_Data[1]　・・・ DRIVE DATA2 PORT、HARD CONFIG DATA2 PORTのいずれかの内容を格納します。
MCO7_Data[2]　・・・ HARD CONFIG DATA3 PORTの内容を格納します。
MCO7_Data[3]　・・・ 拡張用です。

データセット関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

32ビットデータを次の形式でデータ構造体に格納します。

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←					引数*Data*の下位16ビット（D15～D0）										→ D0

引数*psData*で示されるデータ構造体のメンバ*MCO7_Data[1]*（C言語表記）［各種DATA2 PORTに対応］

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D31 ←					引数*Data*の下位16ビット（D31～D16）										→ D16

書　式

C言語　VOID MCO7_SetData(DWORD *Data*, MCO7_S_DATA **psData*);

VB.NET　Sub MCO7_SetData(ByVal *Data* As Integer, ByRef *psData* As MCO7_S_DATA)

C#.NET　void MCO7.SetData(int *Data*, ref MCO7_S_DATA *psData*);

引　数

Data　　･･･ 32ビットのデータを指定します。
psData　　･･･ データ構造体のポインタを指定します。

戻り値

この関数に、戻り値はありません。

データゲット関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

データ構造体の内容を、次の形式で32ビットデータに変換し返します。

変換後の32ビットデータ

D31	D30	D29	D28	D27	D26	D25	D24	D23	D22	D21	D20	D19	D18	D17	D16
D15 ←			引数*psData*で示されるデータ構造体のメンバ*MC07_Data[1]*（C言語表記） ［各種DATA1 PORTに対応］											→	D0

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←			引数*psData*で示されるデータ構造体のメンバ*MC07_Data[0]*（C言語表記） ［各種DATA2 PORTに対応］											→	D0

書　式

C言語　DWORD MC07_GetData(MC07_S_DATA **psData*);

VB.NET　Function MC07_GetData(ByRef *psData* As MC07_S_DATA) As Integer

C#.NET　int MC07.GetData(ref MC07_S_DATA *psData*);

引　数

psData　・・・　データ構造体のポインタを指定します。

戻り値

32ビットに変換されたデータを返します。

DRIVE COMMAND データ構造体書き込み関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたデバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTにデータ構造体のデータを書き込んだ後、
コマンドコードを書き込みます。

書 式

C言語 BOOL MCO7_IWDrive(DWORD *hDev*, WORD *Cmd*, MCO7_S_DATA **psData*, MCO7_S_RESULT **psResult*);

VB.NET Function MCO7_IWDrive(ByVal *hDev* As Integer, ByVal *Cmd* As Short, ByRef *psData* As MCO7_S_DATA,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET bool MCO7.IWDrive(uint *hDev*, ushort *Cmd*, ref MCO7_S_DATA *psData*,
ref MCO7_S_RESULT *psResult*);

引 数

hDev ・・・ デバイスハンドルを指定します。
Cmd ・・・ 書き込むコマンドコードを指定します。
psData ・・・ 書き込むデータが格納されているデータ構造体のポインタを指定します。
psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

DRIVE DATA データ構造体書き込み関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたデバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTにデータ構造体の内容を書き込みます。

書 式

C言語 BOOL MC07_IWData(DWORD *hDev*, MC07_S_DATA **psData*, MC07_S_RESULT **psResult*);

VB.NET Function MC07_IWData(ByVal *hDev* As Integer, ByRef *psData* As MC07_S_DATA,
ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET bool MC07.IWData(uint *hDev*, ref MC07_S_DATA *psData*, ref MC07_S_RESULT *psResult*);

引 数

hDev ・・・ デバイスハンドルを指定します。
psData ・・・ 読み出した内容が格納されるデータ構造体のポインタを指定します。
psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

DRIVE DATA データ構造体読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機 能

指定されたデバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTを読み出し、データ構造体に格納します。

書 式

C言語 BOOL MCO7_IRDrive(DWORD *hDev*, MCO7_S_DATA **psData*, MCO7_S_RESULT **psResult*);

VB.NET Function MCO7_IRDrive(ByVal *hDev* As Integer, ByRef *psData* As MCO7_S_DATA,
ByRef *psResult* As MCO7_S_RESULT) As Boolean

C#.NET bool MCO7.IRDrive(uint *hDev*, ref MCO7_S_DATA *psData*, ref MCO7_S_RESULT *psResult*);

引 数

hDev ・・・ デバイスハンドルを指定します。
psData ・・・ 読み出した内容が格納されるデータ構造体のポインタを指定します。
psResult ・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

MPLリセット関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたボードのMPL内部の状態を初期状態に戻します。

書　式

C言語　BOOL MC07_Reset(WORD *BoardNo*, MC07_S_RESULT **psResult*);

VB.NET　Function MC07_Reset(ByVal *BoardNo* As short, ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET　bool MC07.Reset(ushort *BoardNo*, ref MC07_S_RESULT *psResult*);

引　数

BoardNo　　･･･ ボード番号(0～9)を指定します。
psResult　　･･･ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

16ビット符号なし変換関数

―

機　能

指定された16ビット符号付きデータを16ビット符号なしデータに変換します。

書　式

C言語　WORD MCO7_Unsigned16(SHORT *Data*);

VB.NET　Function MCO7_Unsigned16(ByVal *Data* As Short) As UShort

C#.NET　ushort MCO7.Unsigned16(short *Data*);

引　数

Data　　･･･ 16ビット符号付きデータを指定します。

戻り値

16ビット符号なしデータを返します。

16ビット符号付き変換関数

―

機　能

指定された16ビット符号なしデータを16ビット符号つきデータに変換します。

書　式

C言語　SHORT MCO7_Signed16(WORD *Data*);

VB.NET　Function MCO7_Signed16(ByVal *Data* As UShort) As Short

C#.NET　short MCO7.Signed16(ushort *Data*);

引　数

Data　　･･･ 16ビット符号なしデータを指定します。

戻り値

16ビット符号付きデータを返します。

32ビット符号なし変換関数

―

機　能

指定された32ビット符号付きデータを32ビット符号なしデータに変換します。

書　式

C言語　DWORD MCO7_Unsigned32(LONG *Data*);

VB.NET　Function MCO7_Unsigned32(ByVal *Data* As Integer) As UInteger

C#.NET　uint MCO7.Unsigned32(int *Data*);

引　数

Data　　･･･ 32ビット符号付きデータを指定します。

戻り値

32ビット符号なしデータを返します。

32ビット符号付き変換関数

―

機　能

指定された32ビット符号なしデータを32ビット符号つきデータに変換します。

書　式

C言語　LONG MCO7_Signed32(DWORD *Data*);

VB.NET　Function MCO7_Signed32(ByVal *Data* As UInteger) As Integer

C#.NET　int MCO7.Signed32(uint *Data*);

引　数

Data　　･･･ 32ビット符号なしデータを指定します。

戻り値

32ビット符号付きデータを返します。

ボードタイプ読み出し関数

C-VX870v1 | C-VX871v1 | C-VX872v1 | C-VX873v1 | C-VX870Ev1 | C-VX871Ev1 | C-VX872Ev1 | C-VX873Ev1

機　能

指定されたボードのボードタイプ及びバージョンを読み出します。

書　式

C言語　BOOL MC07_ReadBoardType(WORD *BoardNo*, WORD **pBoardType*, WORD **pVersion*,
MC07_S_RESULT **psResult*);

VB.NET　Function MC07_ReadBoardType(ByVal *BoardNo* As Short,ByRef *pBoardType* As Short,ByRef *pVersion*
As Short,ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET　bool MC07.ReadBoardType(ushort *BoardNo*, ref ushort *pBoardType*, ref ushort *pVersion*
ref MC07_S_RESULT *psResult*);

引　数

BoardNo　・・・ ボード番号(0～9)を指定します。
pBoardType　・・・ ボードタイプが格納される変数のポインタを指定します。

格納される値	意味
H'00	C-VX870v1
H'01	C-VX871v1
H'02	C-VX872v1
H'03	C-VX873v1
H'10	C-VX870Ev1
H'11	C-VX871Ev1
H'12	C-VX872Ev1
H'13	C-VX873Ev1

pVersion　・・・ ボードのバージョンが格納される変数のポインタを指定します。

格納される値	意味
H'01	v1製品

psResult　・・・ この関数を実行した結果が格納されるRESULT構造体のポインタを指定します。

戻り値

この関数を実行した結果、正常終了したときはTRUE、エラーが発生したときはFALSEを返します。

3．コマンド仕様

3-1. ドライブコマンド

3-1-1. 入出力仕様の設定

(1) HARD INITIALIZE1

ステータス信号(OUT A,B 信号)に出力する機能を設定します。
このコマンドの実行は常時可能です。

DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
1	1	1	0	1	1	1	0

D7	D6	D5	D4	D3	D2	D1	D0
OUT B TYPE3	OUT B TYPE2	OUT B TYPE1	OUT B TYPE0	OUT A TYPE3	OUT A TYPE2	OUT A TYPE1	OUT A TYPE0

●リセット後の初期値は H'EE21（アンダーライン側）です。

D3--D0：OUT A TYPE3--0　　初期値 ＝ CNTINT　出力
D7--D4：OUT B TYPE3--0　　初期値 ＝ DFLINT　出力

OUT A,B 信号に出力するステータスを選択します。

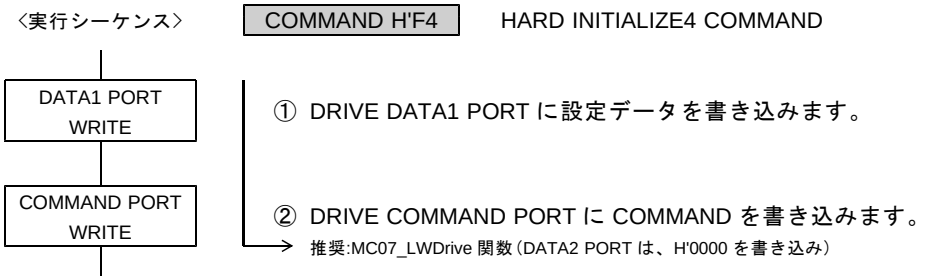
TYPE3	TYPE2	TYPE1	TYPE0	OUT A,B 信号に出力する機能	
0	0	0	0	ADRINT	カウンタ割り込み要求の ADRINT
0	0	0	1	CNTINT	カウンタ割り込み要求の CNTINT
0	0	1	0	DFLINT	カウンタ割り込み要求の DFLINT
0	0	1	1	RDYINT	コマンド終了割り込み要求の RDYINT
0	1	0	0	STBY	STATUS1 の STBY フラグ
0	1	0	1	nDRIVE	STATUS1 の DRIVE フラグの反転
0	1	1	0	nSPEED CBUSY	STATUS5 の SPEED CBUSY フラグの反転
0	1	1	1	nINDEX CBUSY	STATUS5 の INDEX CBUSY フラグの反転
1	0	0	0	UP	STATUS1 の UP フラグ
1	0	0	1	DOWN	STATUS1 の DOWN フラグ
1	0	1	0	CONST	STATUS1 の CONST フラグ
1	0	1	1	EXT PULSE	STATUS1 の EXT PULSE フラグ
1	1	0	0	nPULSE MASK	STATUS2 の PULSE MASK フラグの反転
1	1	0	1	ORG SIGNAL	STATUS2 の ORG SIGNAL フラグ
1	1	1	0	汎用出力	汎用出力として使用する
1	1	1	1	PULSE OVF	STATUS4 の PULSE OVF フラグ

「汎用出力」を選択した場合は、SIGNAL OUT コマンドで出力レベルを操作します。

●ステータス信号(OUT A,B 信号)はステータス外部出力機能で特殊 I/O コネクタから外部出力することができます。
また、多用途センサ機能、同期スタート機能に使用することもできます。
詳しくは 4-3. HARD CONFIG 仕様をご覧ください。

(2) HARD INITIALIZE4

CWLM, CCWLM 信号、DEND/PO 信号 および DALM 信号(INnx 信号)入力のデジタルフィルタ機能の時定数を設定します。このコマンドの実行は常時可能です。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
—	SERVO FILTER2	SERVO FILTER1	SERVO FILTER0	—	0	0	0

D7	D6	D5	D4	D3	D2	D1	D0
—	LIMIT FILTER2	LIMIT FILTER1	LIMIT FILTER0	—	0	0	0

●リセット後の初期値は H'0000（アンダーライン側）です。

D6--D4 : LIMIT FILTER2--0 : CWLM, CCWLM 信号入力の選択
D14--D12 : SERVO FILTER2--0 : DEND/PO, DALM 信号入力の選択
D2--D0 : 0 に設定する
D10--D8 : 0 に設定する

デジタルフィルタ機能の時定数を選択します。

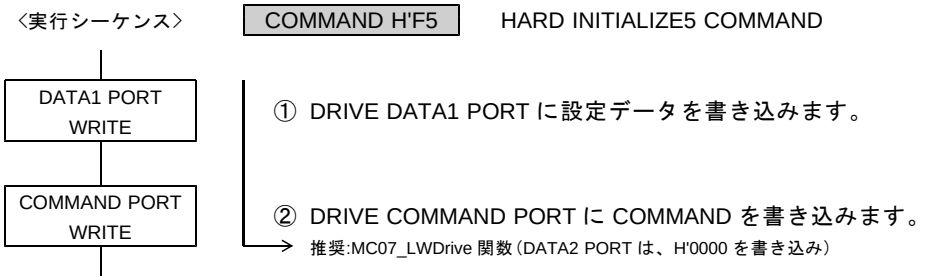
FLT2	FLT1	FLT0	デジタルフィルタ
0	0	0	0 ～ 50 ns
0	0	1	50 µs
0	1	0	100 µs
0	1	1	200 µs
1	0	0	500 µs
1	0	1	1.0 ms
1	1	0	5.0 ms
1	1	1	10.0 ms

（誤差：+10, −0 µs）

●ボードの入力回路には、ハード的なノイズフィルタが入っており、原則設定不要です。
環境下によって不要な信号を拾うようなとき、デジタルフィルタを設定してください。
但し、デジタルフィルタの設定により、信号の応答性は低下します。

(3) HARD INITIALIZE5

ORG 信号、NORG 信号, ± ZORG 信号入力のデジタルフィルタ機能の時定数を設定します。
このコマンドの実行は常時可能です。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
−	0	0	0	−	0	0	0

D7	D6	D5	D4	D3	D2	D1	D0
−	ZPO FILTER2	ZPO FILTER1	ZPO FILTER0	−	ORIGIN FILTER2	ORIGIN FILTER1	ORIGIN FILTER0

●リセット後の初期値は H'0000（アンダーライン側）です。

D2--D0 : ORIGIN FILTER2--0 : ORG,NORG 信号
D10--D8 : 設定禁止(0)
D14--D12 : 設定禁止(0)

デジタルフィルタ機能の時定数を選択します。

FLT2	FLT1	FLT0	デジタルフィルタ
0	0	0	0 ～ 50 ns
0	0	1	50 µs
0	1	0	100 µs
0	1	1	200 µs
1	0	0	500 µs
1	0	1	1.0 ms
1	1	0	5.0 ms
1	1	1	10.0 ms

（誤差：+10, −0 µs）

D6--D4 : ZPO FILTER2--0 : ± ZORG 信号

デジタルフィルタ機能の時定数を選択します。

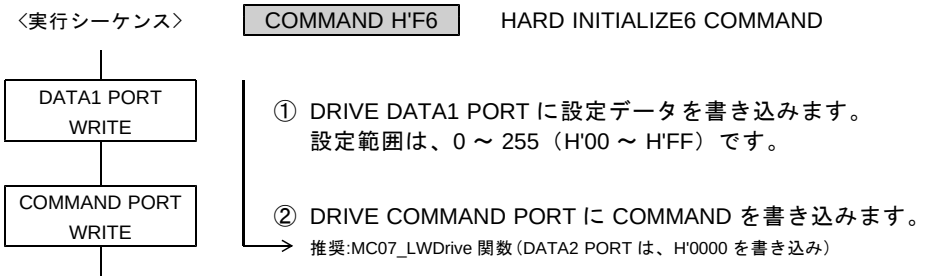
FLT2	FLT1	FLT0	デジタルフィルタ
0	0	0	0 ～ 50 ns
0	0	1	5 µs
0	1	0	10 µs
0	1	1	20 µs
1	0	0	50 µs
1	0	1	100 µs
1	1	0	500 µs
1	1	1	1.0 ms

（誤差：+1, −0 µs）

●ボードの入力回路には、ハード的なノイズフィルタが入っており、原則設定不要です。
環境下によって不要な信号を拾うようなとき、デジタルフィルタを設定してください。
但し、デジタルフィルタの設定により、信号の応答性は低下します。

(4) HARD INITIALIZE6

± EA,± EB 信号入力のデジタルフィルタ機能の時定数を設定します。
このコマンドの実行は常時可能です。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	D7	⟵ デジタルフィルタのデータ ⟶						D0

●リセット後の初期値は H'00（0 ～ 50 ns）です。

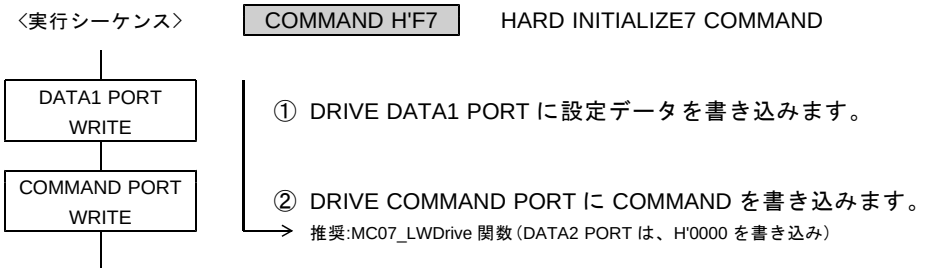
± EA, ± EB 信号入力のデジタルフィルタ機能の時定数を設定します。
デジタルフィルタの時定数 ＝ 設定データ x 50 ns（0 ～ 12.75 µs）です。
（誤差：+50, −0 ns）

●環境下によって不要な信号を拾うようなとき、デジタルフィルタを設定してください。
但し、デジタルフィルタの設定により、信号の応答性は低下します。

(5) HARD INITIALIZE7

CWLM, CCWLM 信号、DALM 信号(INnx 信号)の入力アクティブ論理を設定します。
このコマンドの実行は常時可能です。

> ⚠ 注意
> 設定を誤ると停止機能が働かなくなります。
> 入力信号のアクティブ論理を設定するデータ部には、重要な機能(FSSTOP 信号など)の信号も割り付いています。指定された "1" の論理は間違えないようにしてください。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
1	1	ORG ACTIVE(1)	1	1	NORG ACTIVE(1)	1	1

D7	D6	D5	D4	D3	D2	D1	D0
1	DALM ACTIVE(1)	1	1	CCWLM ACTIVE(1)	CWLM ACTIVE(1)	1	1

●リセット後の初期値は H'FFFF(アンダーライン側) です。

D2 : CWLM ACTIVE
CWLM 入力信号のアクティブ論理を選択します。
0 : 負論理(ローアクティブ)
1 : 正論理(ハイアクティブ)

D3 : CCWLM ACTIVE
CCWLM 入力信号のアクティブ論理を選択します。
0 : 負論理(ローアクティブ)
1 : 正論理(ハイアクティブ)

D6 : DALM ACTIVE
DALM 入力信号(INnx 入力信号)のアクティブ論理を選択します。
0 : 正論理(ハイアクティブ)
1 : 負論理(ローアクティブ)

D10 : NORG ACTIVE
NORG 入力信号のアクティブ論理を選択します。
0 : 正論理(ハイアクティブ)
1 : 負論理(ローアクティブ)

D13 : ORG ACTIVE
ORG 入力信号のアクティブ論理を選択します。
0 : 正論理(ハイアクティブ)
1 : 負論理(ローアクティブ)

・HARD INITIALIZE7 コマンドの実行で、各信号のアクティブ論理を変更します。
アクティブ論理を変更すると、変更した信号のデジタルフィルタ機能が動作します。
デジタルフィルタ機能の時定数経過後に、アクティブ論理の変更が確定します。

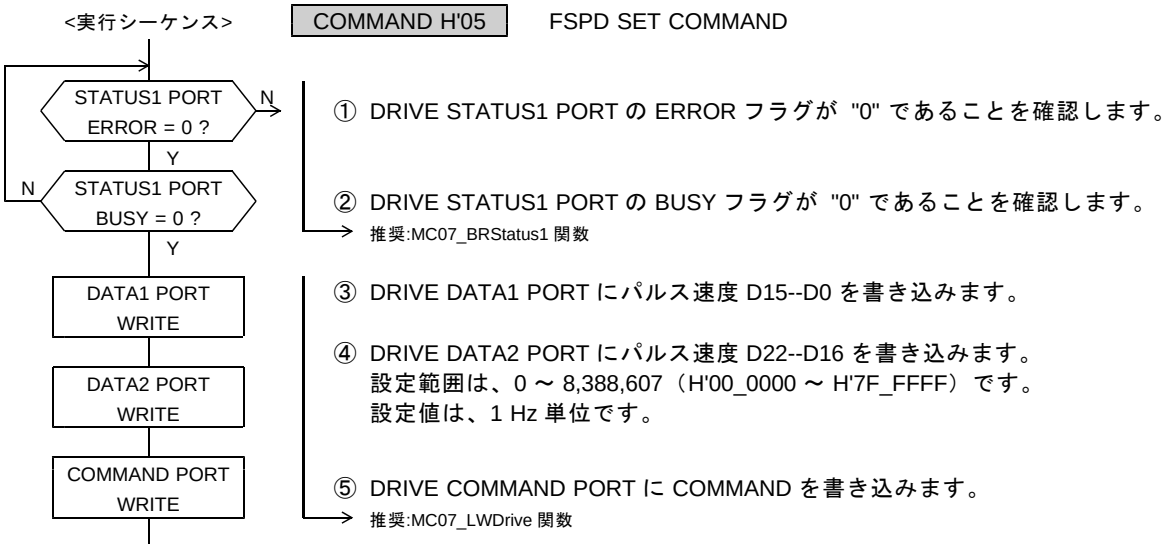
3-1-2. ドライブ パラメータの設定

ドライブのパラメータを設定します。各設定は変更が必要な場合に設定します。

※ SPEED･RATE 関数はこれらのコマンドを使用してドライブパラメータを設定しています。

(1) FSPD SET

ドライブパルス出力の第 1 パルス目のパルス周期（パルス速度）を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←							FSPD							→	D0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	D22 ←		FSPD			→	D16

●リセット後の初期値は H'00 1388 （5,000 Hz ： 1 周期 200 µs） です。

・FSPD の設定値が "0" の場合は、"1" に補正します。

・第 1 パルスのパルス周期の計算式

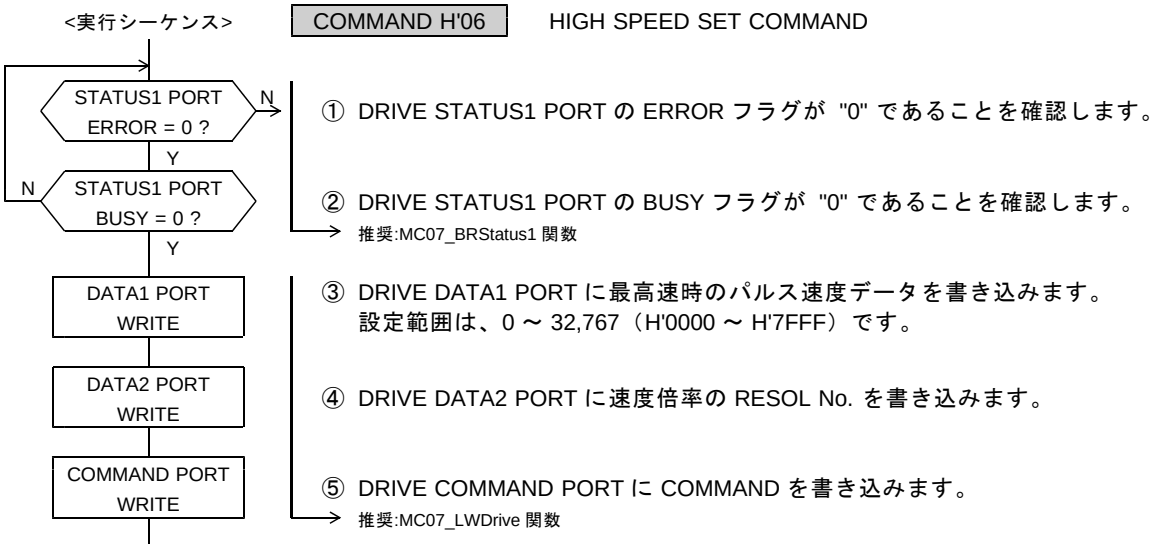
10,000,000 ／ FSPD ＝ 商 A ＋余り B	→	OFF 周期	＝商 A x 50 ns
(10,000,000 ＋余り B)／ FSPD ＝ 商 C ＋余り D	→	ON 周期	＝商 C x 50 ns

FSPD の設定値と実際に出力する第 1 パルスのパルス周期

FSPD の設定値		第 1 パルスのパルス周期（パルス速度）
8,388,607 ～ 6,666,667 Hz	→	OFF 周期 = 50 ns　ON 周期 = 50 ns （10,000,000 Hz）
6,666,666 ～ 5,000,001 Hz	→	OFF 周期 = 50 ns　ON 周期 = 100 ns（6,666,666 Hz）
5,000,000 ～ 4,000,001 Hz	→	OFF 周期 = 100 ns　ON 周期 = 100 ns（5,000,000 Hz）
4,000,000 ～ 3,333,334 Hz	→	OFF 周期 = 100 ns　ON 周期 = 150 ns（4,000,000 Hz）
3,333,333 ～ 2,857,143 Hz	→	OFF 周期 = 150 ns　ON 周期 = 150 ns（3,333,333 Hz）

(2) HIGH SPEED SET

加減速ドライブの最高速時のパルス速度データ(HSPD) および 速度データの速度倍率(RESOL)を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	D14 ←						HSPD							→	D0

●リセット後の初期値は H'0BB8（3,000：3,000 Hz） です。

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	–	–	← RESOL No.			→

●リセット後の初期値は H'3（速度倍率＝ 1） です。

・HSPD の設定値が "0" の場合は、HSPD を HSPD = LSPD に補正します。
最高速時の速度（Hz） ＝ HSPD x RESOL

・減速ドライブと一定速ドライブの 1 パルス目は、FSPD の第 1 パルスです。
2 パルス目から HSPD x RESOL の速度になります。

・RESOL No.を選択して、速度倍率（RESOL）を設定します。

RESOL No.	速度倍率 (RESOL)
H'0	0.1
H'1	0.2
H'2	0.5
H'3	1

RESOL No.	速度倍率 (RESOL)
H'4	2
H'5	5
H'6	10
H'7	20

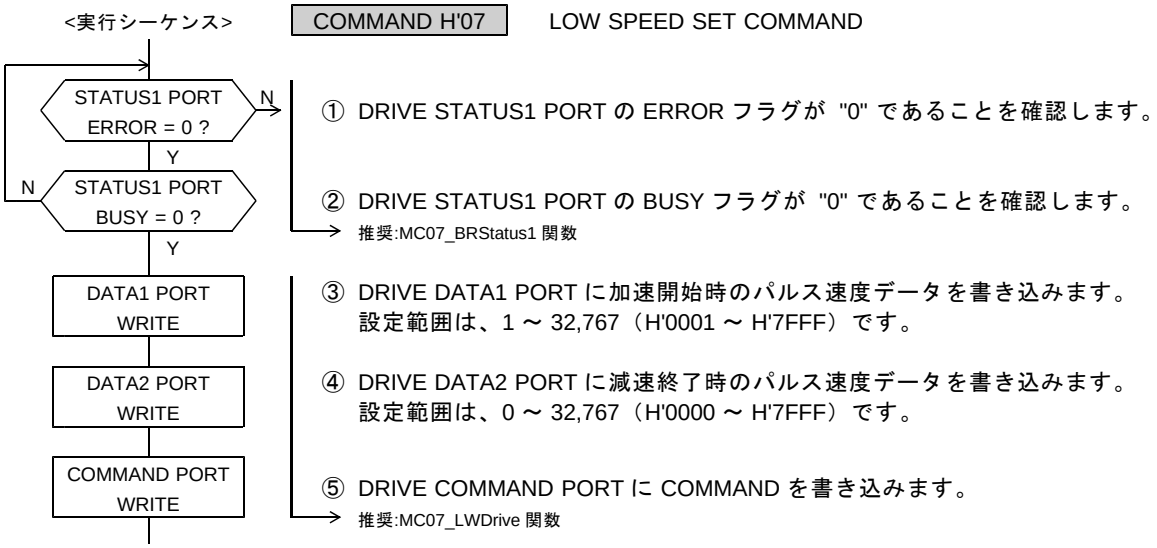
RESOL No.	速度倍率 (RESOL)
H'8	50
H'9	100
H'A	200
H'B	200

RESOL No.	速度倍率 (RESOL)
H'C	200
H'D	200
H'E	200
H'F	200

速度設定値 ＝ 速度データ × 速度倍率（RESOL）：0.1 ～ 6,553,400 Hz

(3) LOW SPEED SET

加減速ドライブの加速開始時のパルス速度データ(LSPD) および 減速終了時のパルス速度データ(ELSPD)を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
−	D14 ←						LSPD							→	D0

●リセット後の初期値は H'012C（300：300 Hz） です。

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
−	D14 ←						ELSPD							→	D0

●リセット後の初期値は H'0000（LSPD と同じ） です。

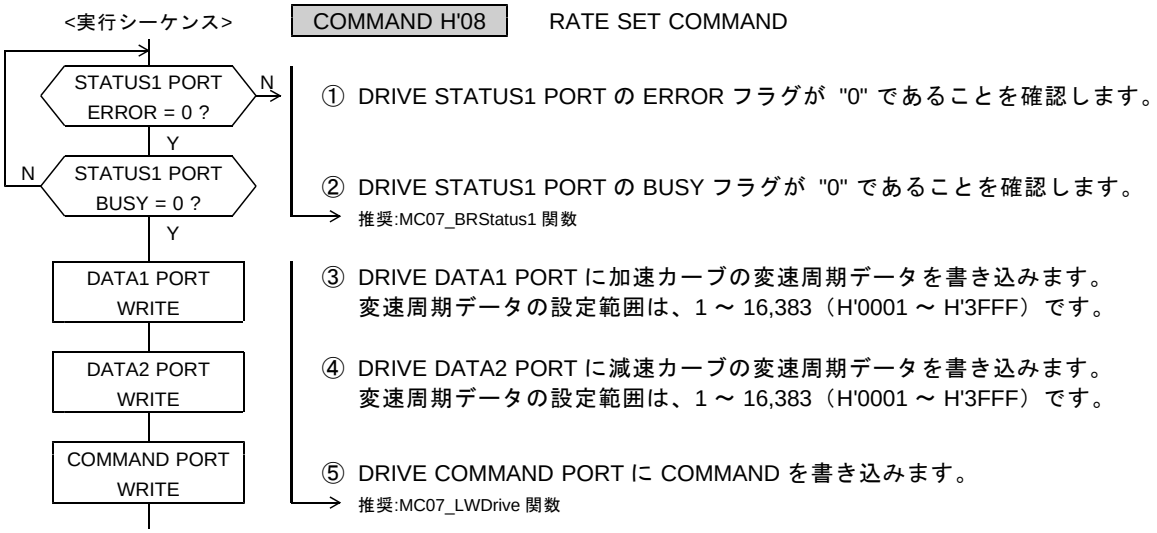
・LSPD の設定値が "0" の場合は、"1" に補正します。
　加速開始時の速度（Hz）＝ LSPD x RESOL

・ELSPD の設定値が "0" の場合は、ELSPD を ELSPD = LSPD に補正します。
　減速終了時の速度（Hz）＝ ELSPD x RESOL

・加減速ドライブと加速ドライブの 1 パルス目は、FSPD の第 1 パルスです。
　2 パルス目から LSPD x RESOL の速度になります。

(4) RATE SET

加速カーブの変速周期データ(UCYCLE) および 減速カーブの変速周期データ(DCYCLE)を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	D13	←				UCYCLE							→	D0

●リセット後の初期値は H'00C8（200：100 µs 周期） です。

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	D13	←				DCYCLE							→	D0

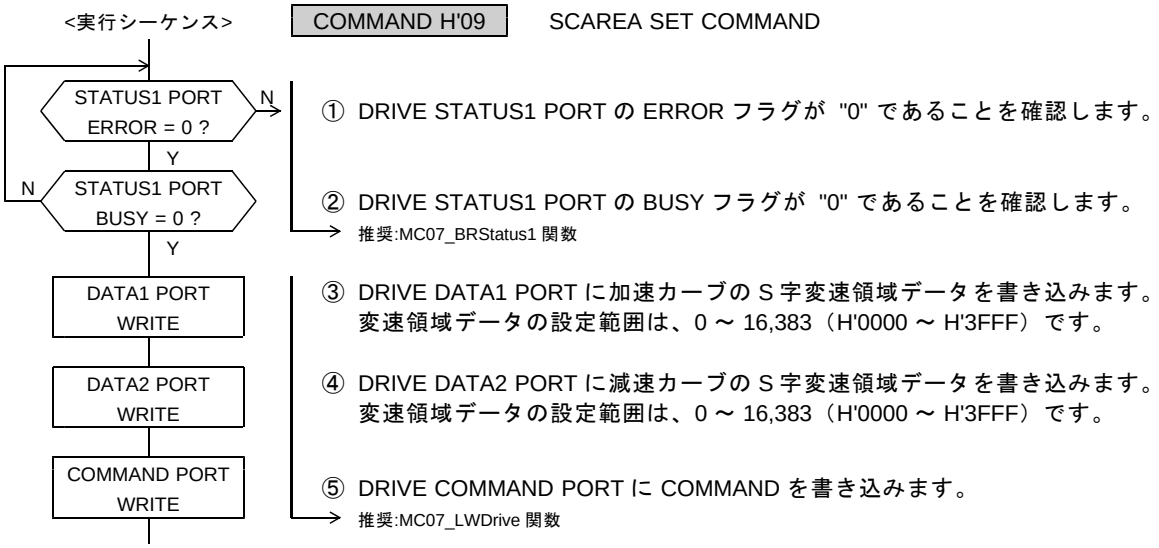
●リセット後の初期値は H'00C8（200：100 µs 周期） です。

・UCYCLE の設定値が "0" の場合は、"1" に補正します。
　　変速周期 (µs) ＝ UCYCLE x 0.5 µs　：0.5 µs ～ 8.1915 ms

・DCYCLE の設定値が "0" の場合は、"1" に補正します。
　　変速周期 (µs) ＝ DCYCLE x 0.5 µs　：0.5 µs ～ 8.1915 ms

(5) SCAREA SET

加速カーブの S 字変速領域データ(SUAREA) および 減速カーブの S 字変速領域データ(SDAREA)を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
−	−	D13 ←					SUAREA							→	D0

●リセット後の初期値は H'0000（0：SUAREA の変速領域なし） です。

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
−	−	D13 ←					SDAREA							→	D0

●リセット後の初期値は H'0000 （0：SDAREA の変速領域なし） です。

・SUAREA の設定値が "(HSPD − LSPD)／ 2" より大きい場合は、重複した変速領域を重ねます。
SUAREA の変速領域（Hz） ＝ SUAREA × RESOL ：0 ～（HSPD − LSPD）x RESOL ／ 2 Hz
S 字加速開始部の変速領域（Hz） ＝ SUAREA × RESOL
S 字加速終了部の変速領域（Hz） ＝ SUAREA × RESOL

・SUAREA の設定値が "0" の場合は、UCYCLE と RESOL による直線加速カーブのみで加速します。

・SUAREA の設定値が "(HSPD − LSPD)／ 2" の場合は、S 字加速カーブのみで加速します。

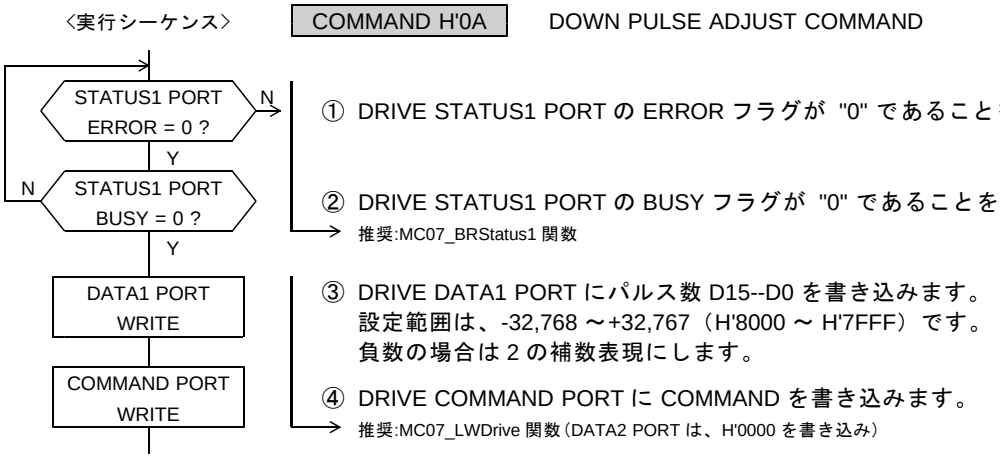
・SDAREA の設定値が "(HSPD − ELSPD)／ 2" より大きい場合は、以下のようになります。
減速停止指令の減速停止時には、重複した変速領域を重ねます。
INDEX ドライブの自動減速停止時には、SDAREA ＝(HSPD − ELSPD)／ 2 に補正します。
SDAREA の変速領域（Hz） ＝ SDAREA × RESOL ：0 ～（HSPD − ELSPD）x RESOL ／ 2 Hz
S 字減速開始部の変速領域（Hz） ＝ SDAREA × RESOL
S 字減速終了部の変速領域（Hz） ＝ SDAREA × RESOL

・SDAREA の設定値が "0" の場合は、DCYCLE と RESOL による直線減速カーブのみで減速します。

・SDAREA の設定値が "(HSPD − ELSPD)／ 2" の場合は、S 字減速カーブのみで減速します。

(6) DOWN PULSE ADJUST

INDEX ドライブの自動減速停止動作を開始する減速パルス数のオフセットパルス数を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←						オフセットパルス数								→	D0

●リセット後の初期値は H'0001（1 パルス） です。

・設定したオフセットパルス数は、MCC07E が自動検出する減速パルス数に加算します。
　オフセットパルス数を正数にすると、減速パルス数は増加します。
　オフセットパルス数を負数にすると、減速パルス数は減少します。

・通常の INDEX ドライブでは、自動減速停止動作開始後に、停止位置を検出した時点で停止します。
INDEX CHANGE 指令を検出した場合は、自動減速停止動作開始後に、終了速度に達してから停止位置を検出して停止します。
このため、負数のオフセットパルス数を設定している場合に INDEX CHANGE 指令を実行すると、停止位置を通過してから停止する可能性があります。
この場合は、ERROR STATUS の INDEX CHANGE ERROR = 1 になります。

3-1-3. ORIGIN ドライブの設定と実行

ORIGIN ドライブの動作仕様を設定して、ORIGIN ドライブを実行します。

※ ORIGIN 関数はこれらのコマンドを使用して、従来製品と同様の機械原点検出ドライブを実現しています。

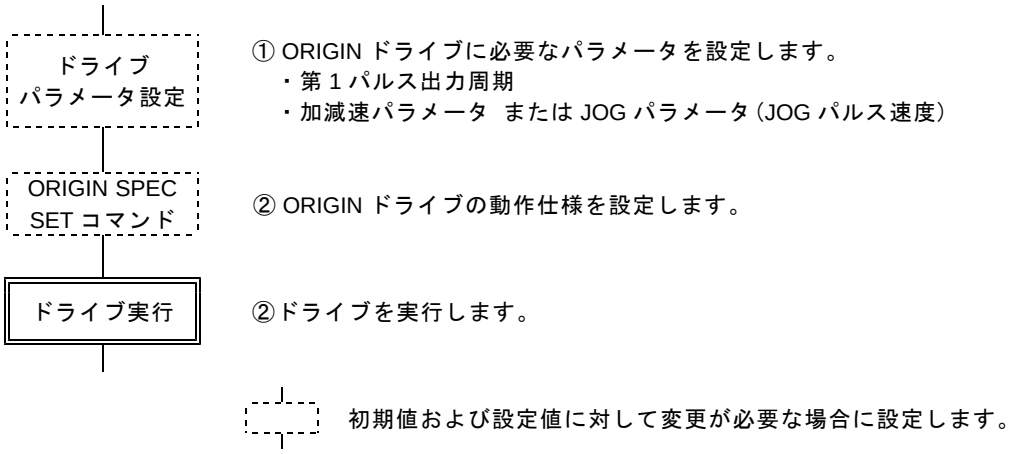
■ ORIGIN ドライブの検出工程

ドライブ工程は、ORIGIN SCAN ドライブと ORIGIN CONSTANT SCAN ドライブが選択できます。

● ORIGIN SCAN ドライブ
加減速ドライブのパラメータで、SCAN ドライブを行います。
ORG 検出信号の指定エッジを指定のカウント数検出すると減速停止します。

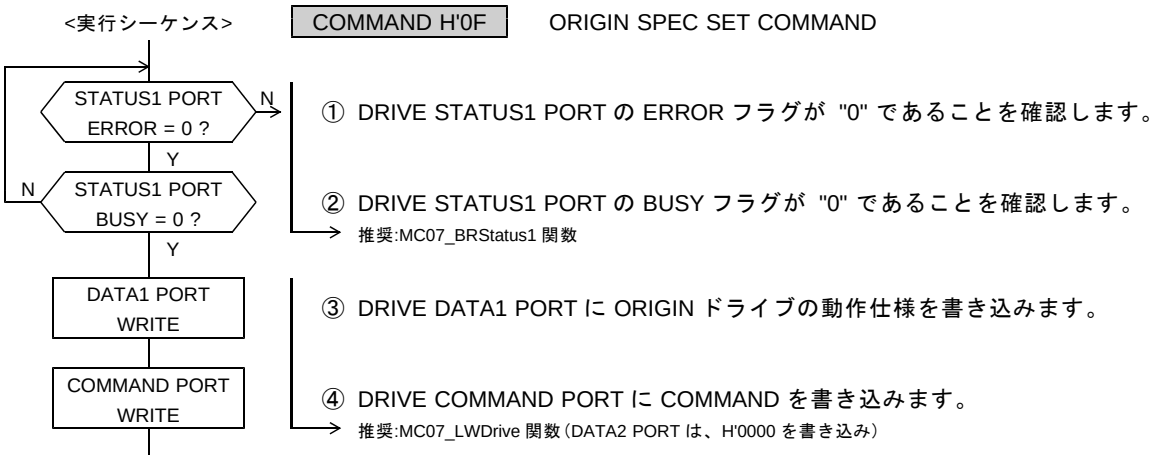
● ORIGIN CONSTANT SCAN ドライブ
JOG ドライブのパルス速度（JSPD）で、一定速ドライブを行います。
ORG 検出信号の指定エッジを指定のカウント数検出すると即時停止します。

■ ORIGIN ドライブの実行シーケンス



(1) ORIGIN SPEC SET

ORIGIN ドライブの動作仕様を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
−	−	−	AUTO DRST ENABLE	ORG COUNT D3	ORG COUNT D2	ORG COUNT D1	ORG COUNT D0

D7	D6	D5	D4	D3	D2	D1	D0
−	−	ORIGIN START DIRECTION	ORG DETECT EDGE	ORG SIGNAL TYPE3	ORG SIGNAL TYPE2	ORG SIGNAL TYPE1	ORG SIGNAL TYPE0

●リセット後の初期値は H'0003（アンダーライン側）です。

【注意】
ORIGIN ドライブ終了後は、D5 ビット（ORIGIN START DIRECTION）を "0" に設定してください。

＊ ORIGIN START DIRECTION = 1 の状態で ORIGIN 以外のドライブを実行すると、－(CCW)方向のドライブを実行しても、＋(CW)方向にドライブパルスを出力してしまいます。

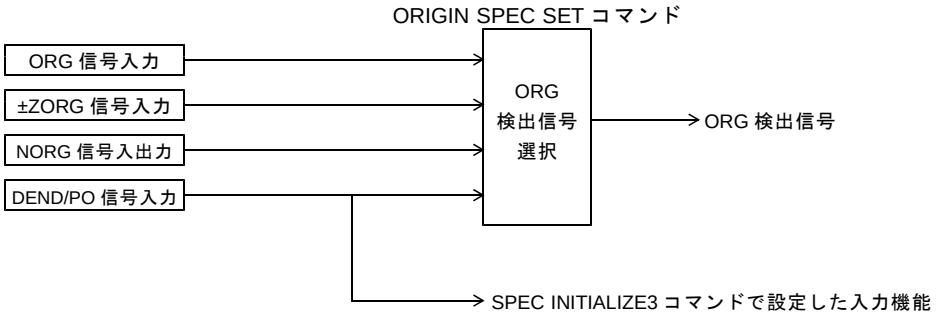
D3--D0：ORG SIGNAL TYPE3-0
ORIGIN ドライブで検出する ORG 検出信号を選択します。

TYPE3	TYPE2	TYPE1	TYPE0	ORG 検出信号
0	0	0	0	$\overline{\text{ORG}}$ 信号
0	0	0	1	± ZORG 信号
0	0	1	0	$\overline{\text{ORG}}$ 信号と± ZORG 信号の AND（論理積）
0	0	1	1	$\overline{\text{ORG}}$ 信号と± ZORG 信号の OR（論理和）
0	1	0	0	$\overline{\text{ORG}}$ 信号
0	1	0	1	$\overline{\text{DEND/PO}}$ 信号
0	1	1	0	$\overline{\text{ORG}}$ 信号と $\overline{\text{DEND/PO}}$ 信号の AND（論理積）
0	1	1	1	$\overline{\text{ORG}}$ 信号と $\overline{\text{DEND/PO}}$ 信号の OR（論理和）
1	0	0	0	$\overline{\text{NORG}}$ 信号
上記以外				設定禁止

・各信号入力のアクティブレベルを合成したものが、ORG 検出信号になります。

■ ORG 検出信号の構成

$\overline{\text{DEND}}/\overline{\text{PO}}$ 信号を ORG 検出信号に選択した場合は、$\overline{\text{DEND}}/\overline{\text{PO}}$ 信号の入力機能と ORG 検出信号の停止機能の両方が有効になります。



D4 ：ORG DETECT EDGE

ORG 検出信号の検出エッジを選択します。

0 ：ORG 検出信号の 0→1（アクティブ）エッジを検出する

1 ：ORG 検出信号の 1→0（OFF）エッジを検出する

D5 ：ORIGIN START DIRECTION

ORIGIN ドライブの起動方向を選択します。

0 ：－(CCW) 方向に起動する

1 ：＋(CW) 方向に起動する

【注意】

ORIGIN ドライブ終了後は、ORIGIN START DIRECTION を "0" に設定してください。

＊ ORIGIN START DIRECTION = 1 の状態で ORIGIN 以外のドライブを実行すると、－(CCW)方向のドライブを実行しても、＋(CW)方向にドライブパルスを出力してしまいます。

D11-D8：ORG COUNT D3-D0

ORG 検出信号の検出エッジのカウント数を設定するビットです。

ORG 検出信号を指定のカウント数検出すると、ORIGIN ドライブの停止機能が動作します。

ORG COUNT D3--D0 = H'0 ：１カウント目のエッジ検出で、停止機能を動作させる

ORG COUNT D3--D0 = H'1 ：２カウント目のエッジ検出で、停止機能を動作させる

ORG COUNT D3--D0 = H'2 ：３カウント目のエッジ検出で、停止機能を動作させる

: :

ORG COUNT D3--D0 = H'F ：16 カウント目のエッジ検出で、停止機能を動作させる

D12 ：AUTO DRST ENABLE

SERVO SPEC SET コマンドで、DRST 信号を〈サーボ対応〉に設定している場合に有効です。

ORG 検出信号の停止機能が動作して ORIGIN ドライブを停止した時に、

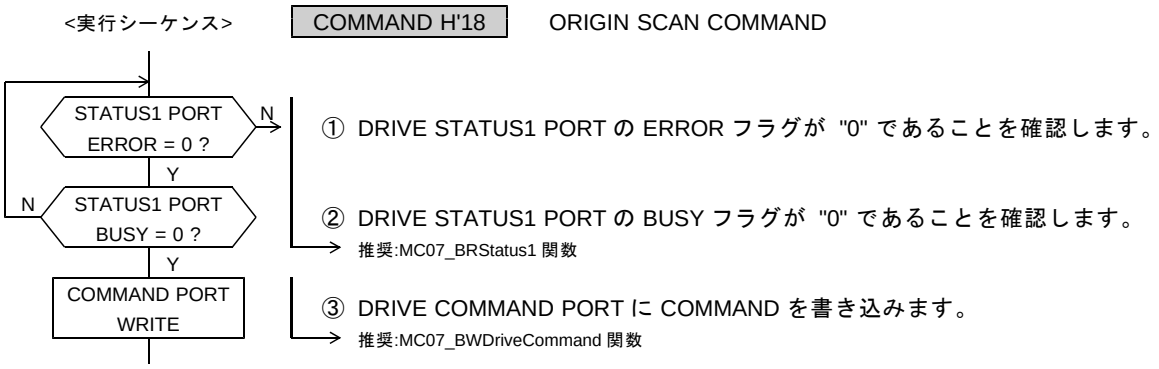
DRST 信号を「出力する／出力しない」を選択します。

0 ：DRST 信号を出力しない

1 ：DRST 信号を出力する（10 ms 間アクティブレベルにする）

(2) ORIGIN SCAN

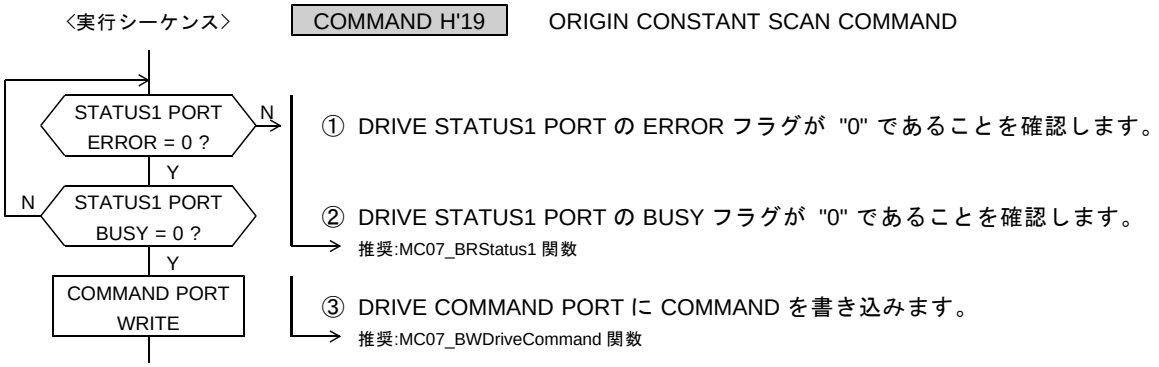
ORIGIN SCAN ドライブを実行します。



● ORIGIN SCAN ドライブ
加減速ドライブのパラメータで、SCAN ドライブを行います。
ORG 検出信号の指定エッジを指定のカウント数検出すると減速停止します。
AUTO DRST ENABLE = 1 の場合は、減速停止後に DRST 信号を出力します。

(3) ORIGIN CONSTANT SCAN

ORIGIN CONSTANT SCAN ドライブを実行します。



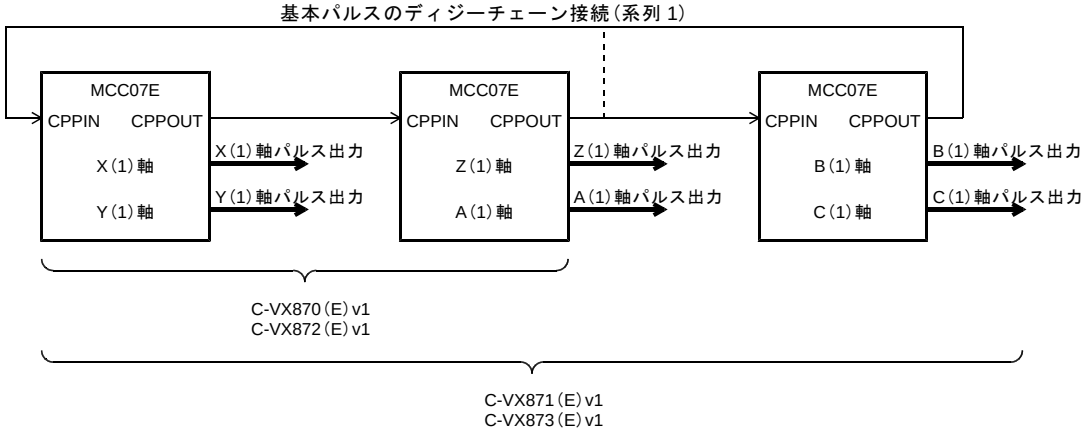
● ORIGIN CONSTANT SCAN ドライブ
JOG パルス速度（JSPD）で、一定速ドライブを行います。
ORG 検出信号の指定エッジを指定のカウント数検出すると即時停止します。
AUTO DRST ENABLE = 1 の場合は、即時停止後に DRST 信号を出力します。

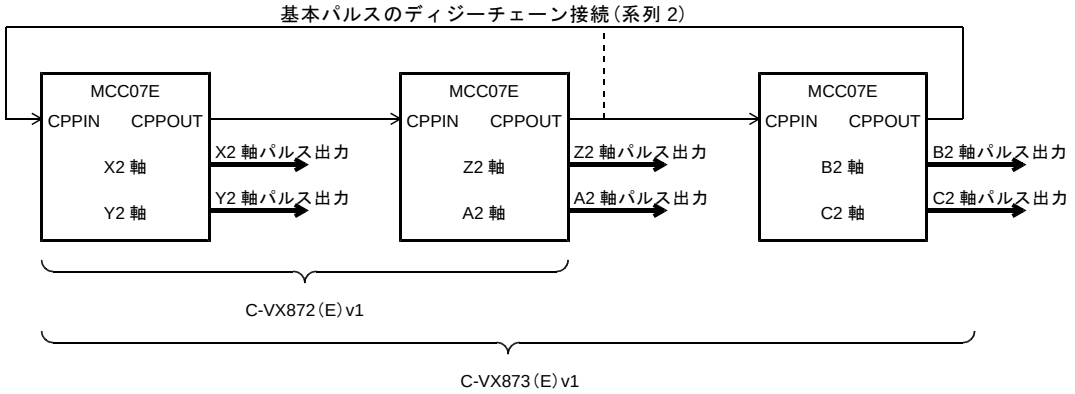
3-1-4. 任意軸補間ドライブの設定

MCC07E の CPPOUT 端子から出力するパルスを設定します。

各軸 MCC07E の CPPOUT 端子と CPPIN 端子はディジーチェーン接続で繋がっています。
任意軸補間ドライブではメイン軸のチップが CPPOUT 端子に補間ドライブの基本パルスを出力します。
サブ軸のチップは基本パルスを CPPIN 端子から入力して CPPOUT 端子に出力します。

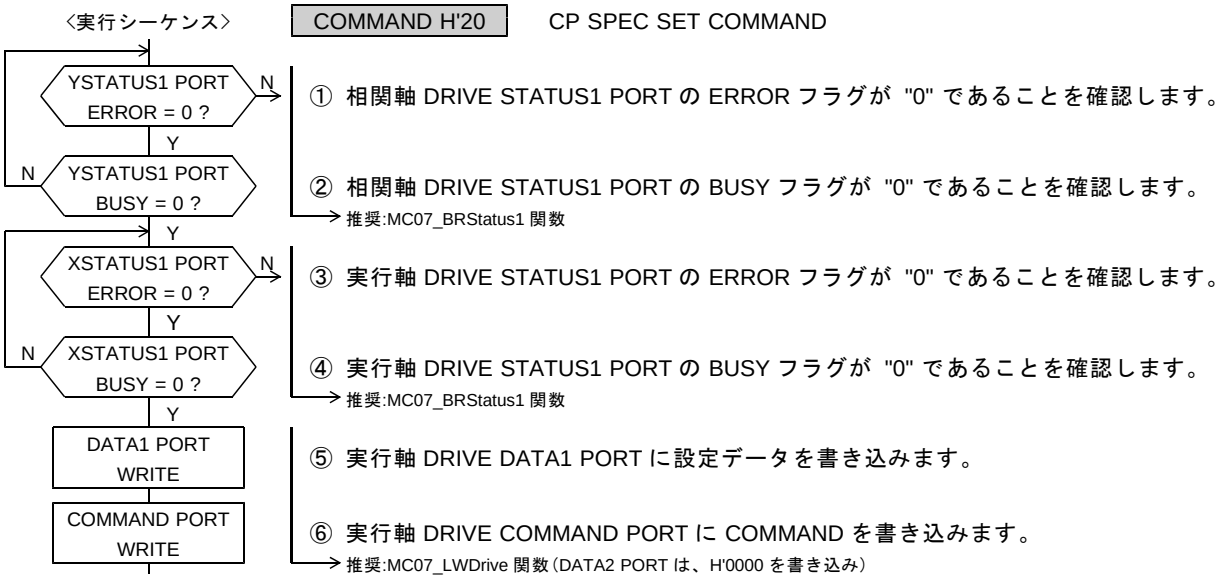
・C-VX872(E)v1, C-VX873(E)v1 にはディジーチェーン接続が 2 系列あります。
　任意軸補間ドライブはディジーチェーン接続の系列内の軸間で行います。





(1) CP SPEC SET

2 軸相関コマンドです。相関軸両軸が BUSY = 0, ERROR = 0 のときにコマンドを実行します。
相関 2 軸 1 チップの CPPOUT 端子から出力するパルスを設定します。
このコマンドの設定は相関軸で共有します。実行軸のどちらの軸に設定しても有効です。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	CPPOUT SEL2	CPPOUT SEL1	CPPOUT SEL0

●リセット後の初期値は H'0（アンダーライン側）です。

D0 ：CPPOUT SEL0
D1 ：CPPOUT SEL1
D2 ：CPPOUT SEL2
CPPOUT 端子から出力するパルスを選択します。

SEL2	SEL1	SEL0	CPPOUT から出力するパルス	
0	0	0	CPPIN 端子から入力するパルス	
0	0	1	メインチップ 2 軸補間ドライブの基本パルス	*1
0	1	0	X 軸の出力パルス（XOP）	
0	1	1	Y 軸の出力パルス（YOP）	
1	0	0	CPPIN 端子から入力するパルス	
1	0	1	メインチップ 2 軸補間ドライブの基本パルス	*1
1	1	0	X 軸のメイン軸補間ドライブの基本パルス	*1
1	1	1	Y 軸のメイン軸補間ドライブの基本パルス	*1

*1：メイン軸補間ドライブまたはメインチップ 2 軸補間ドライブを実行するときに、
コマンド実行軸が発生する補間ドライブの基本パルスを出力します。
その他のドライブを実行する場合は、パルス出力なし（ハイレベル出力）になります。

3-1-5. 直線補間ドライブの設定と実行

長軸と短軸の座標アドレスとドライブ仕様を指定して、相関 2 軸直線補間ドライブ または
任意多軸直線補間ドライブを実行します。

・直線補間ドライブは実行軸の加減速パラメータでドライブします。
・直線補間ドライブでは、長軸パルスを補間ドライブの基本パルスとし、短軸側は長軸パルスを
補間演算して補間パルスを出力します。
・現在位置を座標中心（0, 0）とした長軸と短軸の相対アドレスを、座標アドレスとします。
座標アドレスは、正数が ＋(CW)方向、負数が －(CCW)方向です。

※補間関数のメインチップ 2 軸相対アドレス直線補間ドライブ関数は相関 2 軸直線補間ドライブを
使用しています。

■直線補間ドライブ仕様

● DRIVE MODE
直線補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。

・直線補間 SCAN ドライブ
各補間軸は、長軸と補間軸のパルス比で、目的地の指定方向に補間パルス出力を続けます。
停止指令を検出すると、補間ドライブを終了します。

・直線補間 INDEX ドライブ
各補間軸は、長軸と補間軸のパルス比で、目的地の指定方向に補間パルス出力を続けます。
長軸パルスをカウントして、カウント数が長軸の目的地のパルス数になると、補間ドライブを終了します。

● CONST CP ENABLE
相関 2 軸直線補間ドライブとメイン軸直線補間ドライブで有効です。
線速一定制御を「無効にする／有効にする」を選択します。

・線速一定制御
直線補間ドライブしている２軸の合成速度を一定にする制御です。

コマンド実行軸が発生する補間ドライブの長軸パルスを線速一定制御します。
コマンド実行軸の長軸と短軸の２軸間で、２軸同時にパルス出力したときに、次の長軸パルスの
出力周期を 1.414 倍にします。
線速一定で加減速ドライブを行うと、減速後の終了速度でのドライブが長くなります。

● CPP STOP ENABLE
相関 2 軸直線補間ドライブとメイン軸直線補間ドライブで有効です。
メイン軸の CPP STOP 機能を「有効にする／無効にする」を選択します。

・CPP STOP ENABLE = 1 を選択した場合は、CPPIN 端子に入力できるパルス速度の最高値は、
2.5 MHz になります。

【メイン軸の CPP STOP 機能】
相関 2 軸直線補間ドライブとメイン軸直線補間ドライブ実行中に機能します。

・メイン軸の CPP STOP ENABLE = 1 にすると、メイン軸が発生する補間ドライブの基本パルスと CPPIN
から入力するパルスを偏差カウントします。

・CPPIN のパルス数が、メイン軸の基本パルス数より 2 パルス分少なくなると、メイン軸のドライブを
終了して、メイン軸が発生する補間ドライブの基本パルス出力を停止します。
相関 2 軸直線補間ドライブでは、両軸のドライブを終了して、基本パルス出力を停止します。

・CPP STOP 機能でドライブを終了した場合は、メイン軸のエラーになります。
ERROR STATUS の CPP STOP ERROR = 1 にします。

・メイン軸は CPP STOP 機能でドライブを終了しますが、サブ軸はドライブを終了しません。
メイン軸が CPP STOP 機能でドライブを終了した場合は、すべてのサブ軸に停止指令を実行して、
ドライブを終了させてください。

● CPP MASK ENABLE
サブ軸直線補間ドライブで有効です。
サブ軸の CPPIN マスク機能を「有効にする／無効にする」を選択します。

【サブ軸の CPP STOP 機能】
サブ軸補間ドライブ実行中に機能します。

・サブ軸の CPP MASK ENABLE = 1 にすると、サブ軸がエラー（STATUS1 PORT の ERROR = 1）になると、CPPIN から入力するパルスをマスクします。

・CPPOUT SEL で CPPOUT 出力を「CPPIN から入力するパルス」に設定している場合は、CPPIN のマスクにより、CPPOUT 出力はハイレベル状態になります。

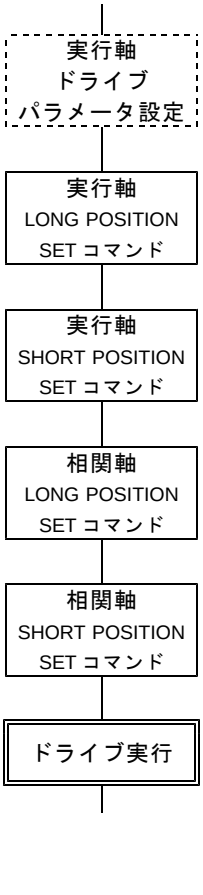
・CPPIN マスク機能で CPPIN をマスクした場合は、STATUS5 PORT の CPP MASK = 1 になります。
CPP MASK = 1 の間は、CPPIN のマスク状態を保持します。
CPP MASK = 1 は、STATUS1 PORT の ERROR = 1 → 0 で CPP MASK = 0 になります。

・C-VX870v1 シリーズでは、相関 2 軸の MCC07E 間同士を CPPIN と CPPOUT をデジーチェーン接続しています。
CPPIN と CPPOUT をデジーチェーン接続した多軸直線補間ドライブで、CPP STOP 機能と CPPIN マスク機能を有効にすると、サブ軸にエラーが発生した場合にすべての補間軸のパルス出力を停止させることができます。

・また、多軸直線補間ドライブの補間軸を、メイン軸補間ドライブまたはメインチップ２軸補間ドライブで構成して CPP STOP 機能を有効にすると、１軸が停止指令またはエラーにより補間ドライブを終了した場合に、すべての補間ドライブを終了させることができます。

■相関 2 軸直線補間ドライブの実行シーケンス

相関軸の 2 軸に実行する直線補間ドライブです。



①実行軸に加減速ドライブに必要なパラメータを設定します。
・第一パルス周期
・加減速パラメータ

②実行軸に、長軸の座標アドレスを設定します。

③実行軸に短軸の座標アドレスを設定します。

④相関軸に長軸の座標アドレスを設定します。
（設定するデータは実行軸と同じデータです。）

⑤相関軸に短軸の座標アドレスを設定します。
（設定するデータは実行軸と同じデータです。）

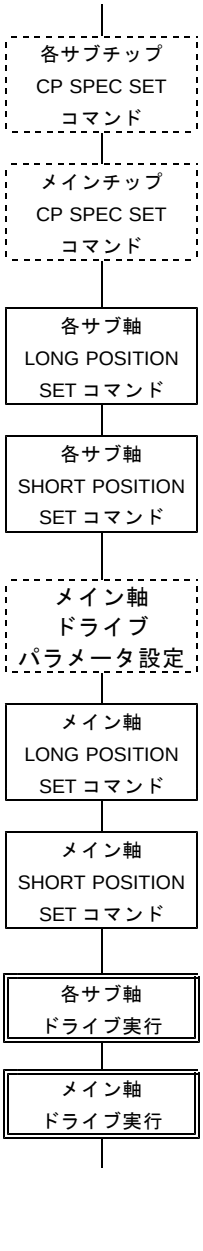
⑥実行軸にドライブ仕様を指定して、ドライブを実行します。

初期値および設定値に対して変更が必要な場合に設定します。

・相関 2 軸直線補間ドライブでは、CP SPEC SET による CPPOUT の設定は不要です。

■任意多軸直線補間ドライブの実行シーケンス

任意軸間、任意軸数で実行する直線補間ドライブです。



①各サブ軸のチップに、CPPOUT 出力を設定します。
CPPOUT SEL = "CPPIN から入力するパルス" に設定します。

②メイン軸のチップに、CPPOUT 出力を設定します。
CPPOUT SEL = "メイン軸の補間ドライブの基本パルス" に設定します。

③各サブ軸に、長軸の座標アドレスを設定します。

④各サブ軸に、短軸の座標アドレスを設定します。

⑤メイン軸に加減速ドライブに必要なパラメータを設定します。
・第一パルス周期
・加減速パラメータ

⑥メイン軸に、長軸の座標アドレスを設定します。

⑦メイン軸に、短軸の座標アドレスを設定します。

⑧各サブ軸に、サブ軸直線補間ドライブを実行します。

⑨メイン軸に、メイン軸直線補間ドライブを実行します。

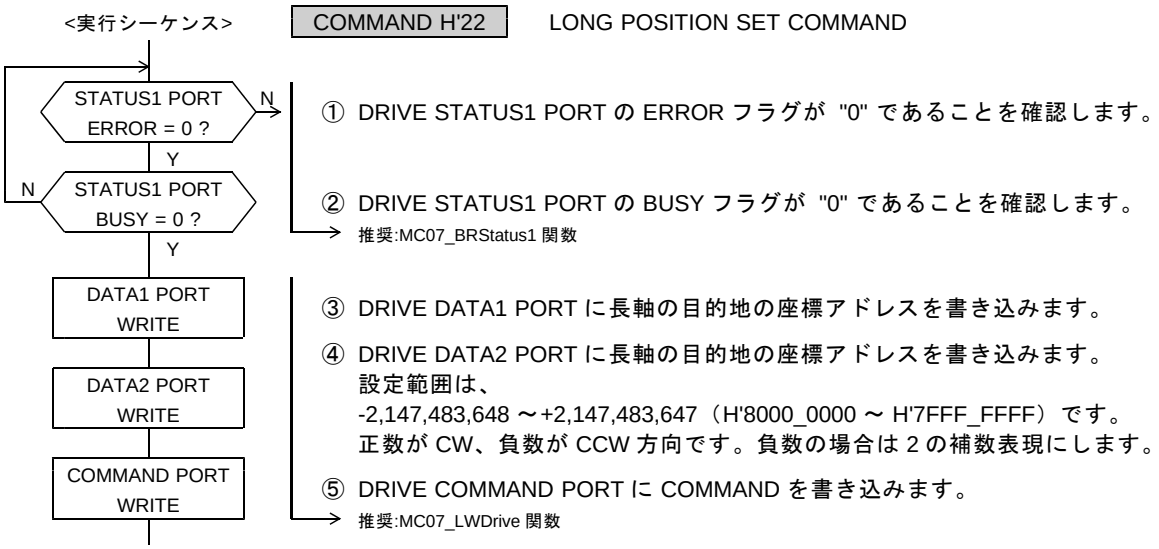
初期値および設定値に対して変更が必要な場合に設定します。

・サブ軸直線補間ドライブを実行すると、ドライブがスタンバイ状態になります。
メイン軸直線補間ドライブを実行すると、長軸パルスを出力して、ドライブを開始します。

・各サブ軸は CPPIN 端子から入力するパルスを長軸パルスとして、ドライブを開始します。

(1) LONG POSITION SET

直線補間ドライブの、長軸の座標アドレスを設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A15	←				長軸の目的地の座標アドレス									→	A0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A31	←				長軸の目的地の座標アドレス									→	A16

●リセット後の初期値は H'0000_0000 です。

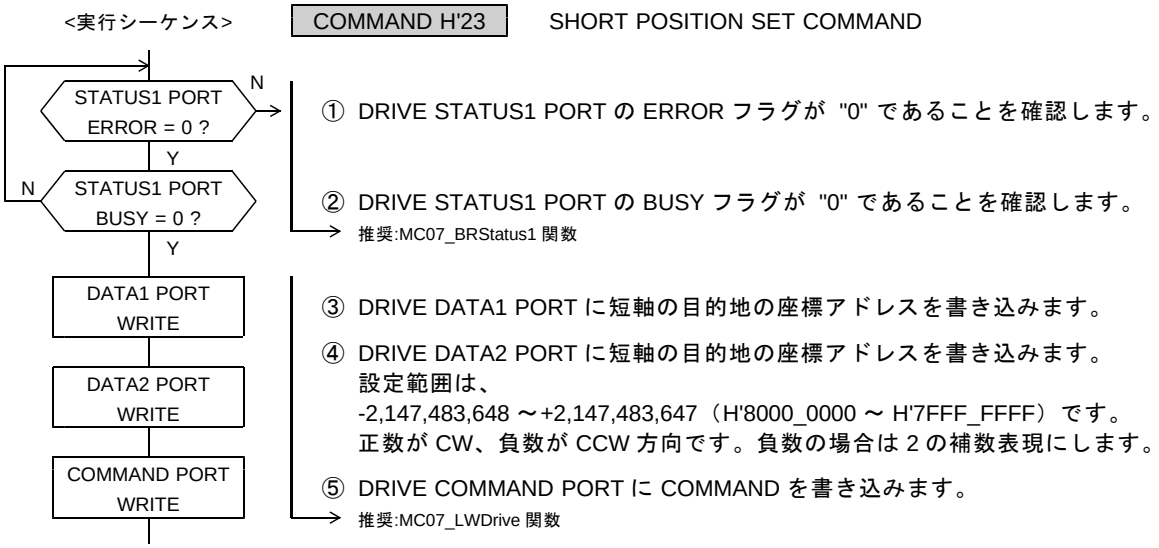
・指定する座標アドレスは、現在位置を座標中心（0, 0）とした相対アドレスです。

・「長軸の目的地の座標アドレス」には、補間軸の中で補間パルス数が大きい補間軸（長軸）の目的地を設定します。

・ドライブ実行コマンドの PULSE SEL で指定した軸の座標アドレスの符号が、出力する補間パルスの動作方向になります。

(2) SHORT POSITION SET

直線補間ドライブの、短軸の座標アドレスを設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A15	←				短軸の目的地の座標アドレス									→	A0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A31	←				短軸の目的地の座標アドレス									→	A16

●リセット後の初期値は H'0000_0000 です。

・指定する座標アドレスは、現在位置を座標中心（0, 0）とした相対アドレスです。

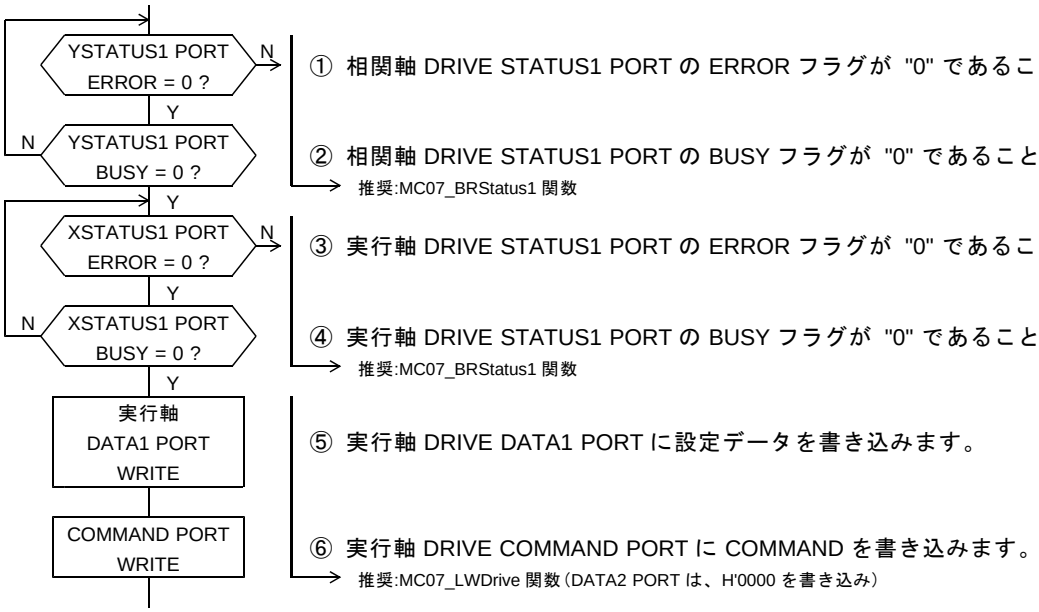
・「短軸の目的地の座標アドレス」には、短軸の目的地（符号付きアドレス）を設定します。
「短軸の目的地の座標アドレス」は「長軸の座標アドレスの絶対値≧短軸の座標アドレスの絶対値」となる様に設定してください。

・ドライブ実行コマンドの PULSE SEL で指定した軸の座標アドレスの符号が、出力する補間パルスの動作方向になります。

(3) MAIN XY STRAIGHT CP

2 軸相関コマンドです。相関軸両軸が BUSY = 0, ERROR = 0 のときにコマンドを実行します。
メインチップの相関 2 軸直線補間ドライブを実行します。相関軸のどちらの軸に実行しても有効です。
直線補間ドライブは実行軸の加減速パラメータで動作します。

<実行シーケンス> COMMAND H'32 MAIN XY STRAIGHT CP COMMAND



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	YPULSE SEL	XPULSE SEL	–	0	CONST CP ENABLE	DRIVE MODE

D0 : DRIVE MODE
直線補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。
0 : 連続ドライブにする (SCAN ドライブ)
1 : 位置決めドライブにする (INDEX ドライブ)

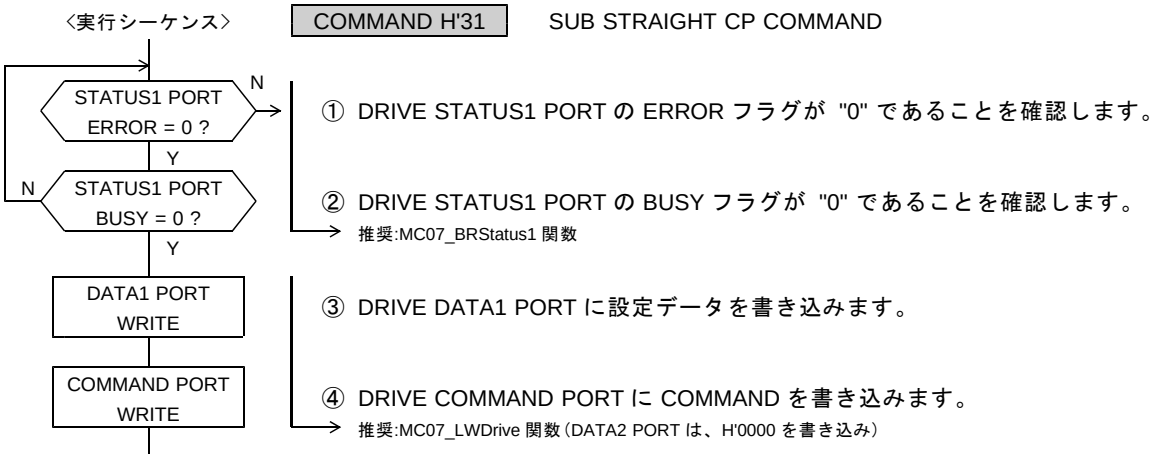
D1 : CONST CP ENABLE
線速一定制御を「無効にする／有効にする」を選択します。
0 : 線速一定制御を無効にする
1 : 線速一定制御を有効にする

D4 : XPULSE SEL
相関軸の内、Xn 軸(または Zn 軸,Bn 軸)に出力する補間パルスを選択します。
0 : X 軸に LONG POSITION (長軸) の補間パルスを出力する
1 : X 軸に SHORT POSITION (短軸) の補間パルスを出力する

D5 : YPULSE SEL
相関軸の内、Yn 軸(または An 軸,Cn 軸)に出力する補間パルスを選択します。
0 : Y 軸に LONG POSITION (長軸) の補間パルスを出力する
1 : Y 軸に SHORT POSITION (短軸) の補間パルスを出力する

(4) SUB STRAIGHT CP

1軸単位で直線補間ドライブを行うコマンドです。
任意軸間の直線補間、または複数軸で直線補間ドライブさせるときにサブ軸に実行します。
サブ軸の直線補間ドライブは CPPIN 入力パルスで動作します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	PULSE SEL	–	CPP MASK ENABLE	–	DRIVE MODE

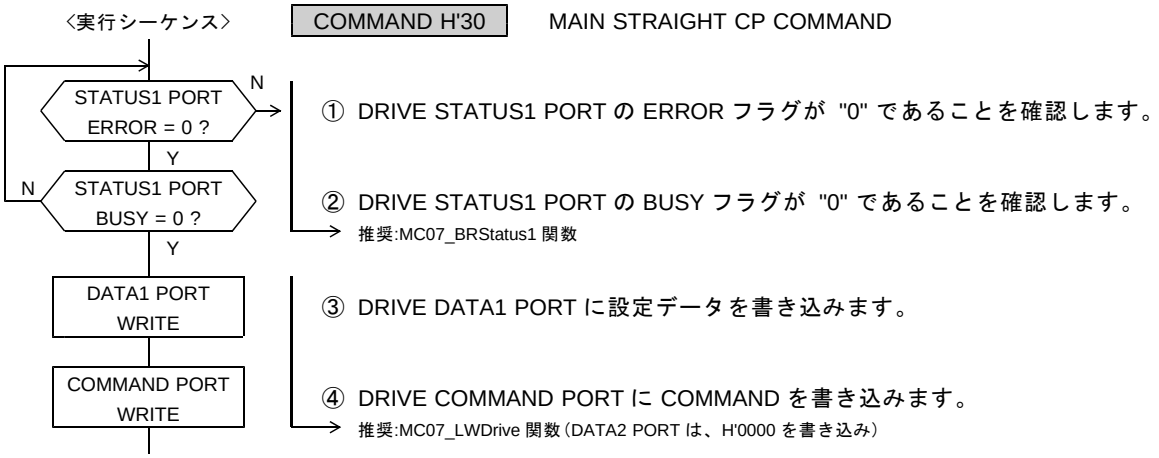
D0 ：DRIVE MODE
直線補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。
0 ：連続ドライブにする （SCAN ドライブ）
1 ：位置決めドライブにする （INDEX ドライブ）

D2 ：CPP MASK ENABLE
CPPIN マスク機能を「有効にする／無効にする」を選択します。
0 ：CPPIN マスク機能を無効にする
1 ：CPPIN マスク機能を有効にする

D4 ：PULSE SEL
出力する補間パルスを選択します。
0 ：LONG POSITION（長軸）の補間パルスを出力する
1 ：SHORT POSITION（短軸）の補間パルスを出力する

(5) MAIN STRAIGHT CP

1軸単位で直線補間ドライブを行うコマンドです。
任意軸間の直線補間、または複数軸で直線補間ドライブさせるときにメイン軸に実行します。
メイン軸の直線補間ドライブは実行軸の加減速パラメータで動作します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	PULSE SEL	–	CPP STOP ENABLE	CONST CP ENABLE	DRIVE MODE

D0 ：DRIVE MODE
直線補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。
0 ：連続ドライブにする （SCAN ドライブ）
1 ：位置決めドライブにする （INDEX ドライブ）

D1 ：CONST CP ENABLE
線速一定制御を「無効にする／有効にする」を選択します。
0 ：線速一定制御を無効にする
1 ：線速一定制御を有効にする

D2 ：CPP STOP ENABLE
CPP STOP 機能を「有効にする／無効にする」を選択します。
0 ：CPP STOP 機能を無効にする
1 ：CPP STOP 機能を有効にする

D4 ：PULSE SEL
出力する補間パルスを選択します。
0 ：LONG POSITION（長軸）の補間パルスを出力する
1 ：SHORT POSITION（短軸）の補間パルスを出力する

3-1-6. 円弧補間ドライブの設定と実行

現在位置の X-Y 座標アドレスと、目的地の短軸座標までの短軸パルス数と、ドライブ仕様を指定して、
相関 2 軸円弧補間ドライブ および 任意 2 軸直線補間ドライブを実行します。

・円弧補間ドライブは実行軸の加減速パラメータをドライブします。
・円弧補間ドライブでは、円弧の中心座標からみた短軸側の短軸パルスを補間ドライブの基本パルスとし、
長軸側は短軸パルスを補間演算して補間パルスを出力します。
・円弧の中心点座標を（0, 0）とした X 軸と Y 軸の相対アドレスを、X-Y 座標アドレスとします。
座標アドレスは、正数が ＋(CW)方向、負数が －(CCW)方向です。

※補間関数のメインチップ 2 軸相対アドレス円弧補間ドライブ関数は相関 2 軸円弧補間ドライブを
使用しています。

■円弧補間ドライブ仕様

● DRIVE MODE
円弧補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。

・円弧補間 SCAN ドライブ
各補間軸は、指定の円弧半径と回転方向で、補間パルス出力を続けます。
停止指令を検出すると、補間ドライブを終了します。

・円弧補間 INDEX ドライブ
各補間軸は、指定の円弧半径と回転方向で、補間パルス出力を続けます。
短軸パルスをカウントして、カウント数が指定の短軸パルス数になると、補間ドライブを終了します。

● CONST CP ENABLE
相関 2 軸円弧補間ドライブとメイン軸円弧補間ドライブで有効です。
線速一定制御を「無効にする／有効にする」を選択します。

・線速一定制御
補間ドライブしている２軸の合成速度を一定にする制御です。

コマンド実行軸が発生する補間ドライブの短軸パルスを線速一定制御します。
X 座標軸と Y 座標軸の２軸間で、２軸同時にパルス出力したときに、次の短軸パルスの出力周期を
1.414 倍にします。
線速一定で加減速ドライブを行うと、減速後の終了速度でのドライブが長くなります。

● CPP STOP ENABLE
2 軸相関円弧補間ドライブとメイン軸円弧補間ドライブで有効です。
メイン軸の CPP STOP 機能を「有効にする／無効にする」を選択します。

・CPP STOP ENABLE = 1 を選択した場合は、CPPIN 端子に入力できるパルス速度の最高値は、
2.5 MHz になります。

【メイン軸の CPP STOP 機能】
2 軸相関円弧補間ドライブとメイン軸円弧補間ドライブ実行中に機能します。

・メイン軸の CPP STOP ENABLE = 1 にすると、メイン軸が発生する補間ドライブの基本パルスと
CPPIN から入力するパルスを偏差カウントします。

・CPPIN のパルス数が、メイン軸の基本パルス数より２パルス分少なくなると、メイン軸のドライブ
を終了して、メイン軸が発生する補間ドライブの基本パルス出力を停止します。
２軸相関円弧補間ドライブでは、両軸のドライブを終了して、基本パルス出力を停止します。

・CPP STOP 機能でドライブを終了した場合は、メイン軸のエラーになります。
ERROR STATUS の CPP STOP ERROR = 1 にします。

・メイン軸は CPP STOP 機能でドライブを終了しますが、サブ軸はドライブを終了しません。
メイン軸が CPP STOP 機能でドライブを終了した場合は、すべてのサブ軸に停止指令を実行して、
ドライブを終了させてください。

● CPP MASK ENABLE
サブ軸円弧補間ドライブで有効です。
サブ軸の CPPIN マスク機能を「有効にする／無効にする」を選択します。

【サブ軸の CPP STOP 機能】
サブ軸補間ドライブ実行中に機能します。

・サブ軸の CPP MASK ENABLE = 1 にすると、サブ軸がエラー（STATUS1 PORT の ERROR = 1）になると、CPPIN から入力するパルスをマスクします。

・CPPOUT SEL で CPPOUT 出力を「CPPIN から入力するパルス」に設定している場合は、CPPIN のマスクにより、CPPOUT 出力はハイレベル状態になります。

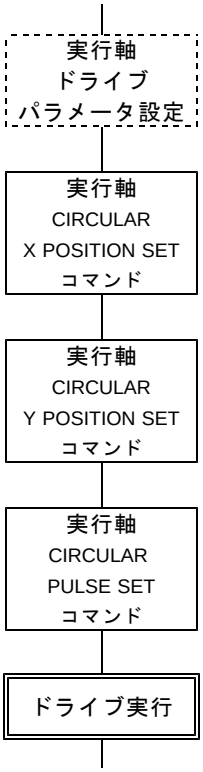
・CPPIN マスク機能で CPPIN をマスクした場合は、STATUS5 PORT の CPP MASK = 1 になります。
CPP MASK = 1 の間は、CPPIN のマスク状態を保持します。
CPP MASK = 1 は、STATUS1 PORT の ERROR = 1 → 0 で CPP MASK = 0 になります。

・C-VX870v1 シリーズでは、相関 2 軸の MCC07E 間同士を CPPIN と CPPOUT をデジーチェーン接続しています。
任意軸円弧補間ドライブで、CPP STOP 機能と CPPIN マスク機能を有効にすると、サブ軸にエラーが発生した場合にすべての補間軸のパルス出力を停止させることができます。

・また、任意軸円弧補間ドライブの補間軸を、メイン軸補間ドライブまたは２軸相関円弧補間ドライブで構成して CPP STOP 機能を有効にすると、１軸が停止指令またはエラーにより補間ドライブを終了した場合に、すべての補間ドライブを終了させることができます。

■相関 2 軸円弧補間ドライブの実行シーケンス

相関軸の 2 軸に実行する円弧補間ドライブです。



①実行軸にドライブに必要なパラメータを設定します。
・第一パルス周期
・加減速パラメータ

②実行軸に現在位置の X 座標アドレスを設定します。

③実行軸に現在位置の Y 座標アドレスを設定します。

④実行軸に目的地の短軸座標までのパルス数を設定します。

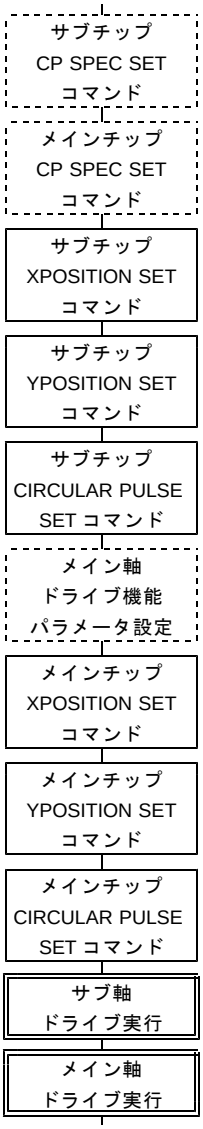
⑤実行軸にドライブ仕様を指定して、ドライブを実行します。

初期値および設定値に対して変更が必要な場合に設定します。

・相関 2 軸円弧補間ドライブでは、CP SPEC SET による CPPOUT の設定は不要です。

■任意 2 軸円弧補間ドライブの実行シーケンス

任意 2 軸間(2 チップ間)で実行する円弧補間ドライブです。



① サブ軸のチップに、CPPOUT 出力を設定します。
CPPOUT SEL = "CPPIN から入力するパルス" に設定します。

② メイン軸のチップに、CPPOUT 出力を設定します。
CPPOUT SEL = "メイン軸の補間ドライブの基本パルス" に設定します。

③ サブ軸のチップに、現在位置の X 座標アドレスを設定します。

④ サブ軸のチップに、現在位置の Y 座標アドレスを設定します。

⑤ サブ軸のチップに、目的地の短軸座標までのパルス数を設定します。

⑥ メイン軸に、加減速ドライブに必要なパラメータを設定します。

⑦ メイン軸のチップに、現在位置の X 座標アドレスを設定します。

⑧ メイン軸のチップに、現在位置の Y 座標アドレスを設定します。

⑨ メイン軸のチップに、目的地の短軸座標までのパルス数を設定します。

⑩ サブ軸に、サブ軸円弧補間ドライブを実行します。

⑪ メイン軸に、メイン軸円弧補間ドライブを実行します。

初期値および設定値に対して変更が必要な場合に設定します。

・サブ軸円弧補間ドライブを実行すると、ドライブがスタンバイ状態になります。

・メイン軸円弧補間ドライブを実行すると、短軸パルスを出力して、ドライブを開始します。
サブ軸は CPPIN 端子から入力するパルスを短軸パルスとして、ドライブを開始します。

【注意事項】
以下の手順でドライブを実行した場合、ドライブを実行していない軸からパルスが出力されます。
① Y 軸に円弧補間ドライブを実行する。
② Y 軸に最後に実行したドライブが円弧補間ドライブのとき、X 軸に「メイン軸円弧補間ドライブ」
または「サブ軸円弧補間ドライブ」を実行する。

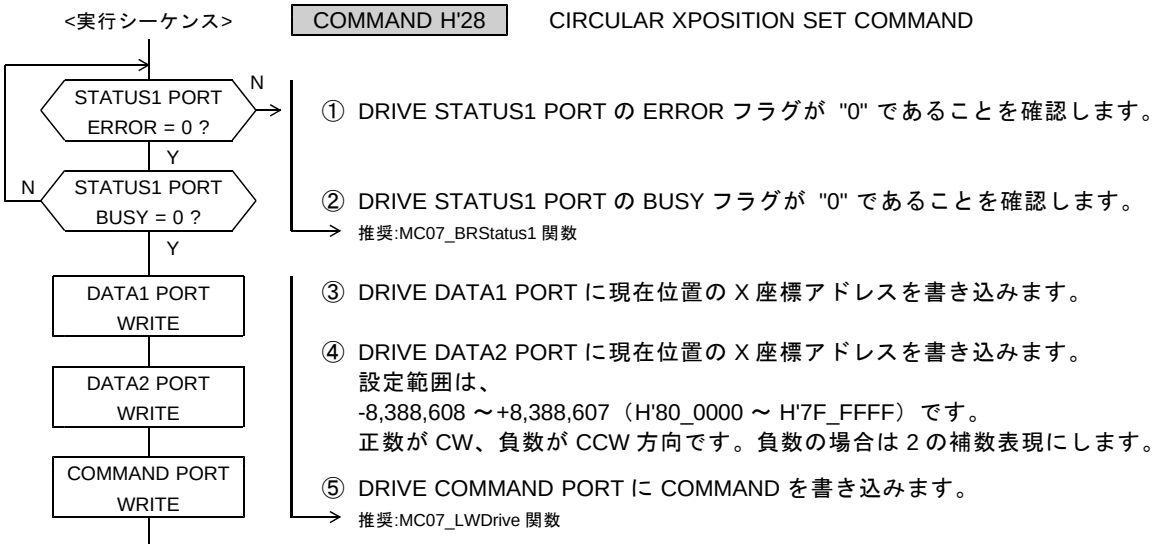
このとき X 軸の他に、ドライブを実行していない Y 軸からも円弧補間ドライブのパルスが出力されます。
これは X 軸と Y 軸が逆でも同様に発生します。また、各相関軸間で同様に発生します。

【対応方法】
「メイン軸円弧補間ドライブ」および「サブ軸円弧補間ドライブ」を実行する前に、ドライブを実行しない相関軸に対して、円弧補間ドライブ以外のドライブを実行してください。

例：① Y 軸に円弧補間ドライブを実行する。
② Y 軸に「移動量 0 の相対アドレス INDEX ドライブ」を実行する。
X 軸に「メイン軸円弧補間ドライブ」または「サブ軸円弧補間ドライブ」を実行する。

(1) CIRCULAR XPOSITION SET

円弧の中心点座標を（0, 0）とした現在位置の X 座標アドレスを設定します。
このコマンドの設定は相関軸で共有します。
実行軸のどちらの軸に設定しても有効です。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A15	←							現在位置の X 座標アドレス						→	A0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
−	−	−	−	−	−	−	−	A23	←	現在位置の X 座標アドレス				→	A16

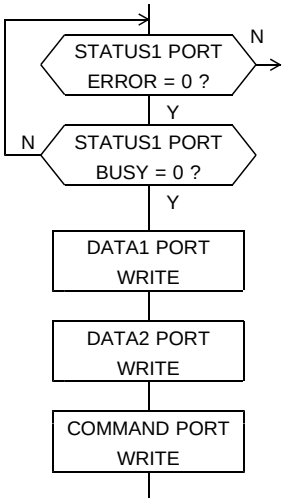
●リセット後の初期値は H'00_0000 です。

・指定する座標アドレスは、円弧の中心点座標を（0, 0）とした X 軸の相対アドレスです。

(2) CIRCULAR YPOSITION SET

円弧の中心点座標を（0, 0）とした現在位置の Y 座標アドレスを設定します。
このコマンドの設定は相関軸で共有します。
実行軸のどちらの軸に設定しても有効です。

<実行シーケンス>　COMMAND H'29　CIRCULAR YPOSITION SET COMMAND



① DRIVE STATUS1 PORT の ERROR フラグが "0" であることを確認します。

② DRIVE STATUS1 PORT の BUSY フラグが "0" であることを確認します。
推奨:MC07_BRStatus1 関数

③ DRIVE DATA1 PORT に現在位置の Y 座標アドレスを書き込みます。

④ DRIVE DATA2 PORT に現在位置の Y 座標アドレスを書き込みます。
設定範囲は、
-8,388,608 ～+8,388,607（H'80_0000 ～ H'7F_FFFF）です。
正数が CW、負数が CCW 方向です。負数の場合は 2 の補数表現にします。

⑤ DRIVE COMMAND PORT に COMMAND を書き込みます。
推奨:MC07_LWDrive 関数

DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A15	←					現在位置の Y 座標アドレス								→	A0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	A23	←	現在位置の Y 座標アドレス				→	A16

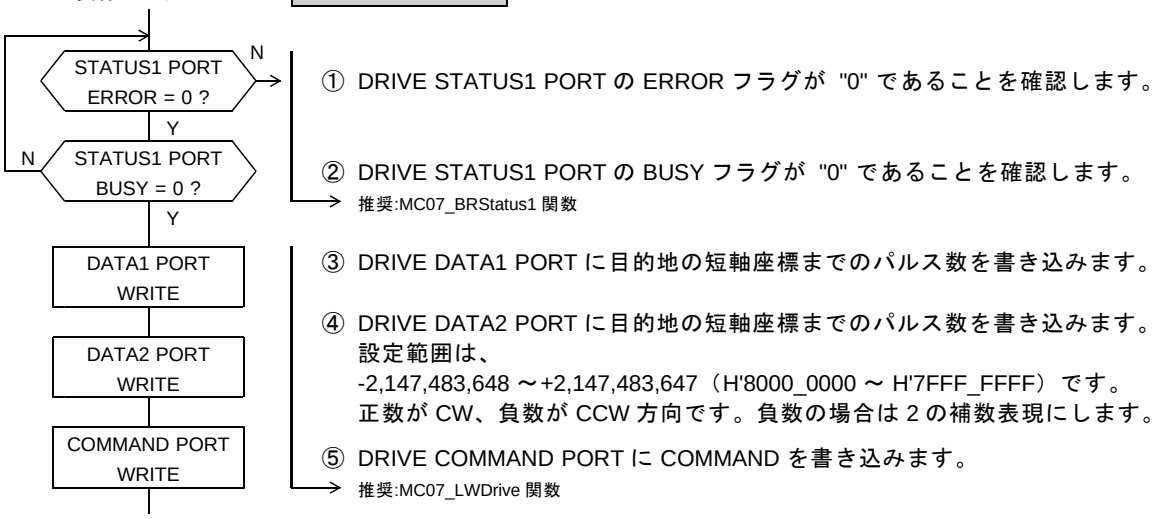
●リセット後の初期値は H'00_0000 です。

・指定する座標アドレスは、円弧の中心点座標を（0, 0）とした Y 軸の相対アドレスです。

(3) CIRCULAR PULSE SET

現在位置の X-Y 座標アドレスから目的地の短軸座標までの短軸パルス数を設定します。
このコマンドの設定は相関軸で共有します。
実行軸のどちらの軸に設定しても有効です。

<実行シーケンス>　COMMAND H'2A　CIRCULAR PULSE SET COMMAND



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←					目的地の短軸座標までのパルス数									→	D0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D31 ←					目的地の短軸座標までのパルス数									→	D16

●リセット後の初期値は H'0000_0000（0 パルス）です。

・指定するパルス数は、目的地の短軸座標に到達するまでに経由する、各象限の短軸の合計パルス数です。

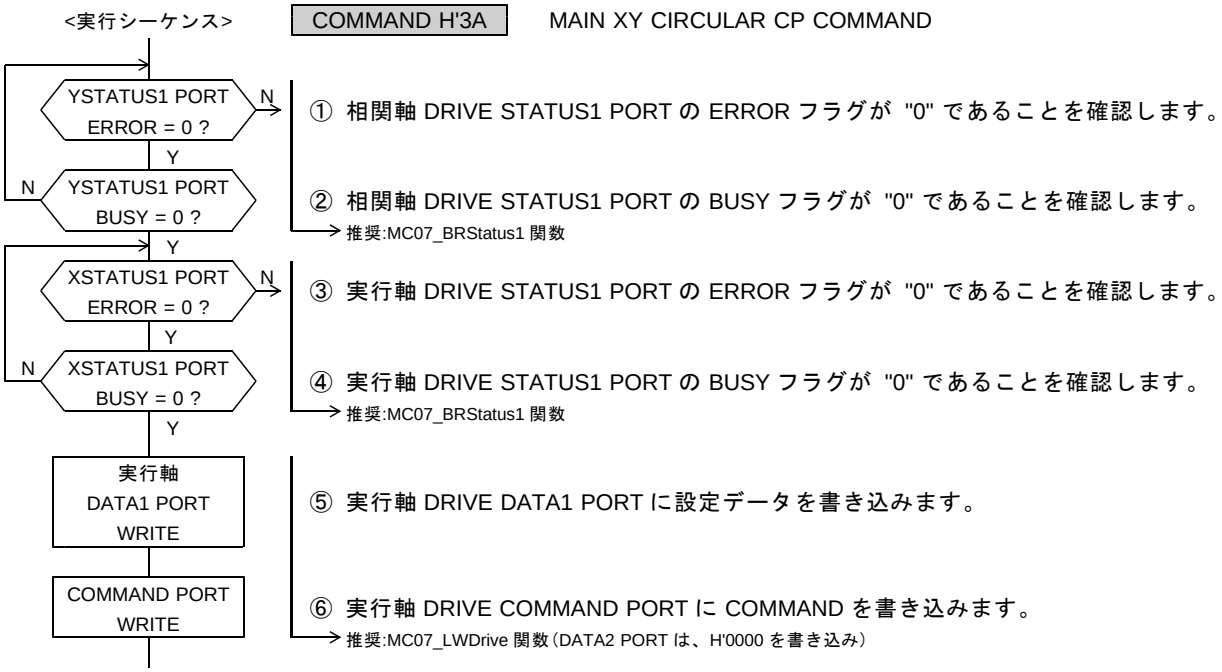
・円弧を描く回転方向は、パルス数の符号で指定します。
　正数を指定すると CW 方向に回転します。
　負数を指定すると CCW 方向に回転します。

・短軸パルス数の計算式は、「4-1-5.(4)円弧補間ドライブ」をご覧ください。

・デバイスドライバの円弧補間短軸 PULSE 数ゲット関数を使用すると
指定された円弧の中心点相対アドレス、目的地相対アドレス、回転方向をもとに
目的地の短軸座標までのパルス数を算出します。

(4) MAIN XY CIRCULAR CP

2 軸相関コマンドです。相関軸両軸が BUSY = 0, ERROR = 0 のときにコマンドを実行します。
メインチップの相関 2 軸円弧補間ドライブを実行します。相関軸のどちらの軸に実行しても有効です。
円弧補間ドライブは実行軸の加減速パラメータで動作します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	YPULSE SEL	XPULSE SEL	–	0	CONST CP ENABLE	DRIVE MODE

D0 ：DRIVE MODE
円弧補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。
0 ：連続ドライブにする (SCAN ドライブ)
1 ：位置決めドライブにする (INDEX ドライブ)

D1 ：CONST CP ENABLE
線速一定制御を「無効にする／有効にする」を選択します。
0 ：線速一定制御を無効にする
1 ：線速一定制御を有効にする

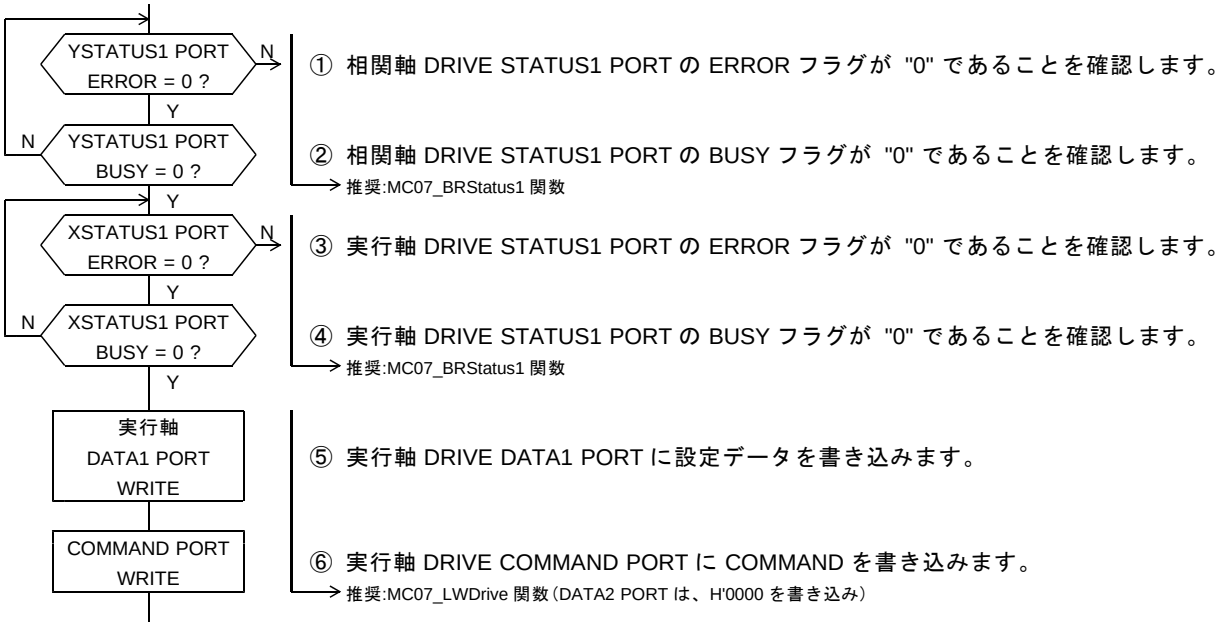
D4 ：XPULSE SEL
相関軸の内、Xn 軸(または Zn 軸,Bn 軸)に出力する補間パルスを選択します。
0 ：X 軸に円弧補間演算の X 座標アドレスの補間パルス（XCP）を出力する
1 ：X 軸に円弧補間演算の Y 座標アドレスの補間パルス（YCP）を出力する

D5 ：YPULSE SEL
相関軸の内、Yn 軸(または An 軸,Cn 軸)に出力する補間パルスを選択します。
0 ：Y 軸に円弧補間演算の X 座標アドレスの補間パルス（XCP）を出力する
1 ：Y 軸に円弧補間演算の Y 座標アドレスの補間パルス（YCP）を出力する

(5) SUB CIRCULAR CP

2 軸相関コマンドです。X, Y 軸が BUSY = 0, ERROR = 0 のときにコマンドを実行します。
1 軸単位で円弧補間ドライブを行うコマンドです。
任意軸間の円弧補間ドライブさせるときにサブ軸に実行します。
サブ軸の円弧補間ドライブは CPPIN 入力パルスで動作します。

〈実行シーケンス〉 COMMAND H'39 SUB CIRCULAR CP COMMAND



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	PULSE SEL	–	CPP MASK ENABLE	–	DRIVE MODE

【注意】
ドライブ実行時に注意事項があります。
3-1-6.項の、「■任意 2 軸円弧補間ドライブの実行シーケンス」をご覧ください。

D0 ：DRIVE MODE
円弧補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。
0 ：連続ドライブにする (SCAN ドライブ)
1 ：位置決めドライブにする (INDEX ドライブ)

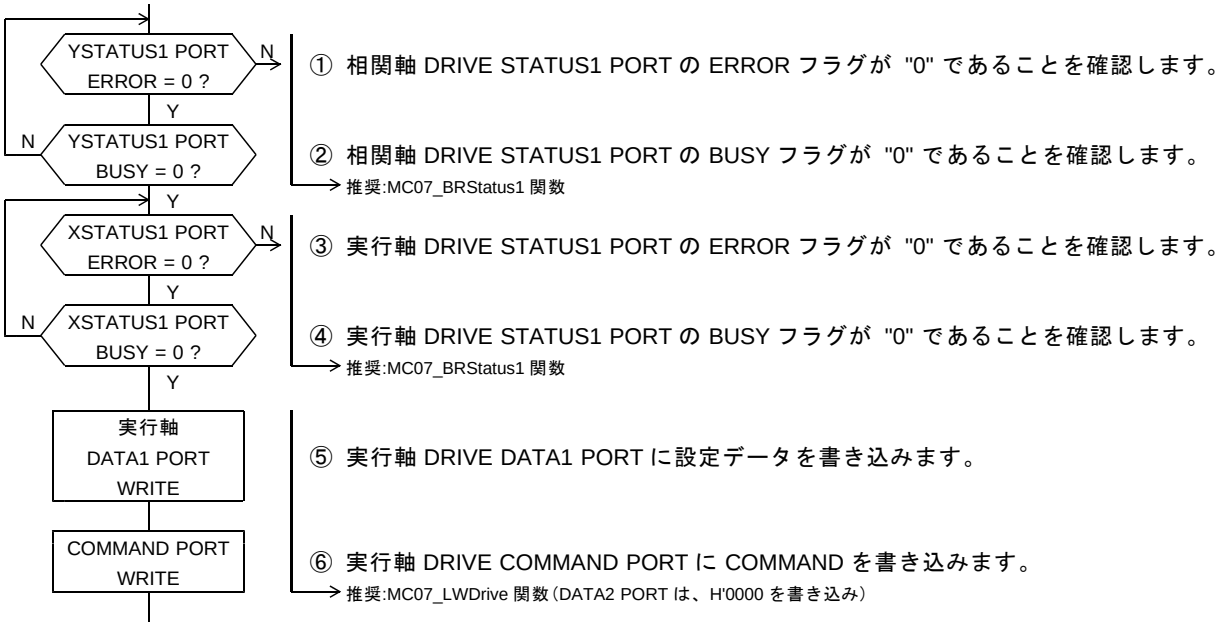
D2 ：CPP MASK ENABLE
CPPIN マスク機能を「有効にする／無効にする」を選択します。
0 ：CPPIN マスク機能を無効にする
1 ：CPPIN マスク機能を有効にする

D4 ：PULSE SEL
出力する補間パルスを選択します。
0 ：円弧補間演算の X 座標アドレスの補間パルス（XCP）を出力する
1 ：円弧補間演算の Y 座標アドレスの補間パルス（YCP）を出力する

(6) MAIN CIRCULAR CP

2 軸相関コマンドです。X, Y 軸が BUSY = 0, ERROR = 0 のときにコマンドを実行します。
1 軸単位で円弧補間ドライブを行うコマンドです。
任意軸間の円弧補間ドライブさせるときにメイン軸に実行します。
メイン軸の円弧補間ドライブは実行軸の加減速パラメータで動作します。

〈実行シーケンス〉 COMMAND H'38 MAIN CIRCULAR CP COMMAND



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	PULSE SEL	–	CPP STOP ENABLE	CONST CP ENABLE	DRIVE MODE

【注意】
ドライブ実行時に注意事項があります。
3-1-6.項の、「■任意 2 軸円弧補間ドライブの実行シーケンス」をご覧ください。

D0 ：DRIVE MODE
円弧補間ドライブを「連続ドライブにする／位置決めドライブにする」を選択します。
0 ：連続ドライブにする （SCAN ドライブ）
1 ：位置決めドライブにする （INDEX ドライブ）

D1 ：CONST CP ENABLE
線速一定制御を「無効にする／有効にする」を選択します。
0 ：線速一定制御を無効にする
1 ：線速一定制御を有効にする

D2 ：CPP STOP ENABLE
CPP STOP 機能を「有効にする／無効にする」を選択します。
0 ：CPP STOP 機能を無効にする
1 ：CPP STOP 機能を有効にする

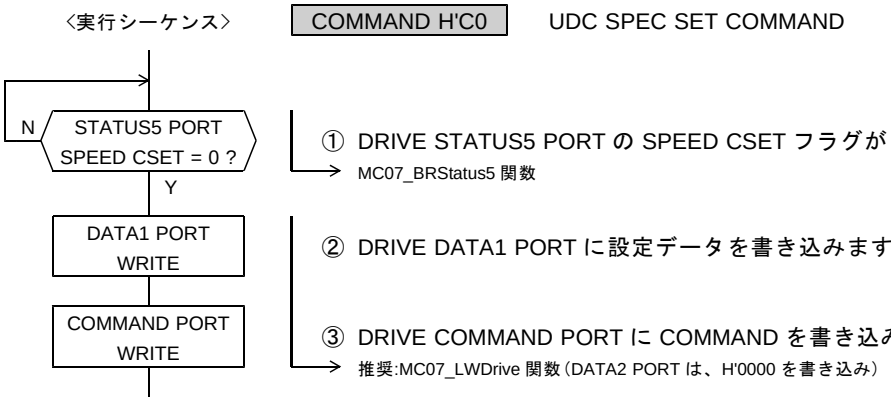
D4 ：PULSE SEL
出力する補間パルスを選択します。
0 ：円弧補間演算の X 座標アドレスの補間パルス（XCP）を出力する
1 ：円弧補間演算の Y 座標アドレスの補間パルス（YCP）を出力する

3-1-7. UP/DOWN/CONST ドライブ CHANGE の設定と実行

変更動作点を設定して、UP/DOWN/CONST のドライブ CHANGE を実行します。
変更動作点の設定は、変更動作点の変更が必要な場合に設定します。

(1) UDC SPEC SET

UP/DOWN/CONST のドライブ CHANGE 指令を実行する変更動作点を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	CONST TYPE2	CONST TYPE1	CONST TYPE0

D7	D6	D5	D4	D3	D2	D1	D0
–	DOWN TYPE2	DOWN TYPE1	DOWN TYPE0	–	UP TYPE2	UP TYPE1	UP TYPE0

●リセット後の初期値は H'000（アンダーライン側）です。

D2--D0 ：UP TYPE2--0
UP DRIVE コマンドのドライブ CHANGE 指令を実行する変更動作点を選択します。

D6--D4 ：DOWN TYPE2--0
DOWN DRIVE コマンドのドライブ CHANGE 指令を実行する変更動作点を選択します。

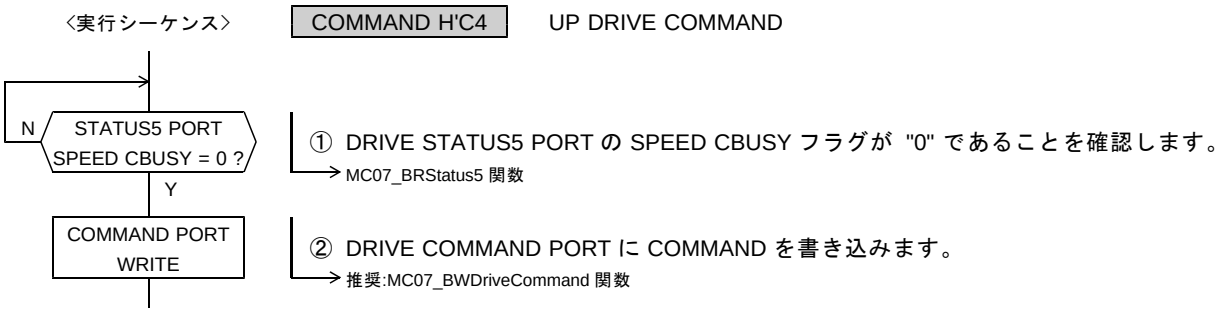
D10--D8 ：CONST TYPE2--0
CONST DRIVE コマンドのドライブ CHANGE 指令を実行する変更動作点を選択します。

TYPE2	TYPE1	TYPE0	ドライブ CHANGE を実行する変更動作点〈レベル検出〉
0	0	0	ドライブ CHANGE コマンドの書き込みで実行する
0	0	1	設定禁止
0	1	0	STATUS5 PORT の SS0 = 1 で実行する（注）
0	1	1	STATUS5 PORT の SS1 = 1 で実行する（注）
1	0	0	設定禁止
1	0	1	設定禁止
1	1	0	設定禁止
1	1	1	設定禁止

（注）SS0, SS1 信号は SPEC INITIALIZE2 コマンドで「汎用入力」に設定している場合に有効です。

(2) UP DRIVE

実行中のパルス出力を、最高速度まで加速または減速します。



(3) DOWN DRIVE

実行中のパルス出力を、終了速度まで加速または減速します。

(4) CONST DRIVE

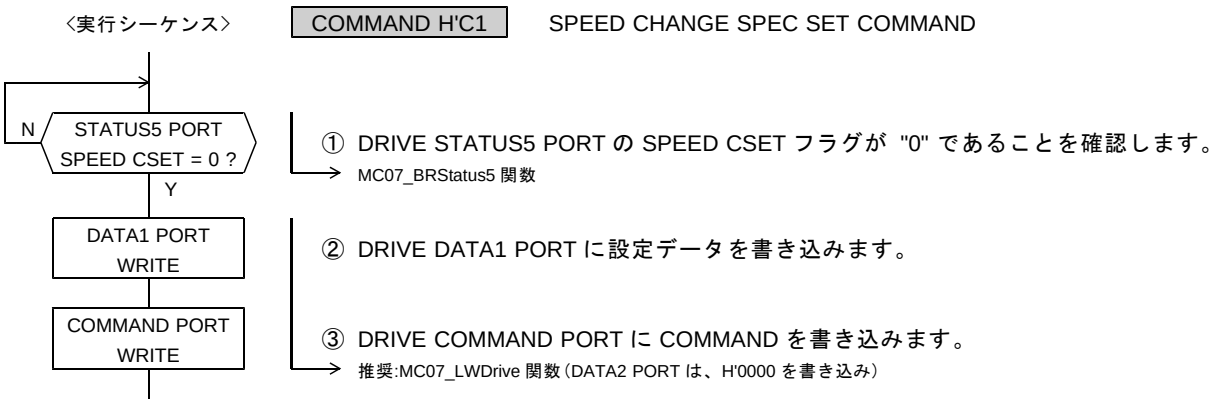
実行中の加速または減速を終了して、パルス出力を一定速にします。

3-1-8. SPEED CHANGE の設定と実行

変更動作点を設定して、SPEED CHANGE を実行します。
変更動作点の設定は、変更動作点の変更が必要な場合に設定します。

(1) SPEED CHANGE SPEC SET

SPEED CHANGE 指令を実行する変更動作点を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	SPEED CHANGE TYPE2	SPEED CHANGE TYPE1	SPEED CHANGE TYPE0

●リセット後の初期値は H'0（アンダーライン側）です。

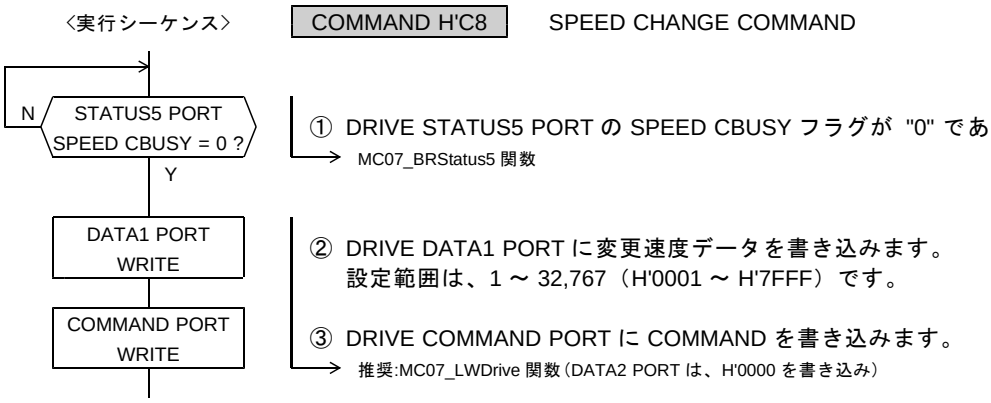
D0 ：SPEED CHANGE TYPE0
D1 ：SPEED CHANGE TYPE1
D2 ：SPEED CHANGE TYPE2
SPEED CHANGE 指令を実行する変更動作点を選択します。

TYPE2	TYPE1	TYPE0	SPEED CHANGE を実行する変更動作点〈レベル検出〉
0	0	0	SPEED CHANGE コマンドの書き込みで実行する
0	0	1	設定禁止
0	1	0	STATUS5 PORT の SS0 = 1 で実行する（注）
0	1	1	STATUS5 PORT の SS1 = 1 で実行する（注）
1	0	0	設定禁止
1	0	1	設定禁止
1	1	0	設定禁止
1	1	1	設定禁止

（注）SS0, SS1 信号は SPEC INITIALIZE2 コマンドで「汎用入力」に設定している場合に有効です。

(2) SPEED CHANGE

実行中のパルス出力を、指定したドライブパルス速度まで加速または減速します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
－	D14 ←					SPEED CHANGE データ								→	D0

・SPEED CHANGE データの設定値が "0" の場合は、"1" に補正します。
SPEED CHANGE の速度（Hz）＝ SPEED CHANGE データ x RESOL

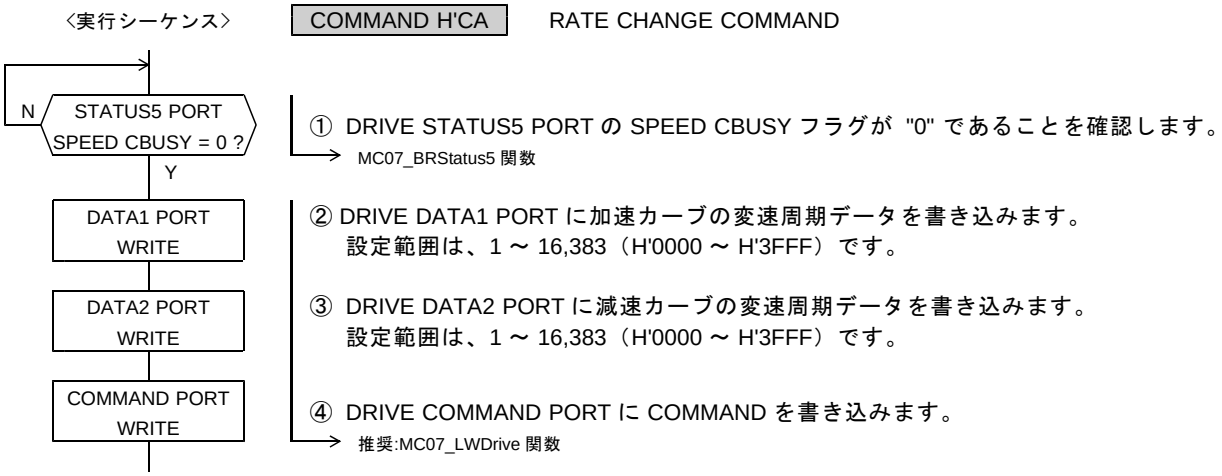
・SPEED CHANGE コマンドを実行しても、速度パラメータの設定は変わりません。

3-1-9. RATE CHANGE の設定と実行

RATE CHANGE 指令は、スピード系のドライブ CHANGE 指令の検出と同時に実行します。

(1) RATE CHANGE

ドライブ CHANGE 動作時の変更周期データを、指定したデータに変更します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	D13 ←					UCYCLE							→	D0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	D13 ←					DCYCLE							→	D0

・RATE CHANGE データの設定値が "0" の場合は、"1" に補正します。
　加速カーブの変速周期（μs）＝ UCYCLE x 0.5 μs　：0.5 μs ～ 8.1915 ms
　減速カーブの変速周期（μs）＝ DCYCLE x 0.5 μs　：0.5 μs ～ 8.1915 ms

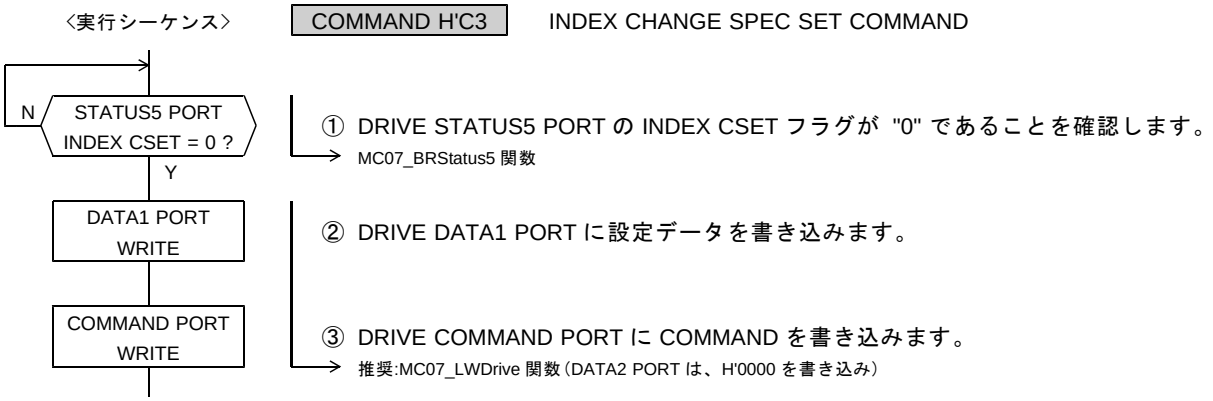
・RATE CHANGE コマンドを実行しても、速度パラメータの設定は変わりません。

3-1-10. INDEX CHANGE の設定と実行

変更動作点を設定して、INDEX CHANGE を実行します。
変更動作点の設定は、変更動作点の変更が必要な場合に設定します。

(1) INDEX CHANGE SPEC SET

INDEX CHANGE 指令を実行する変更動作点を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	INDEX CHANGE TYPE2	INDEX CHANGE TYPE1	INDEX CHANGE TYPE0

●リセット後の初期値は H'0（アンダーライン側）です。

D0 ：INDEX CHANGE TYPE0
D1 ：INDEX CHANGE TYPE1
D2 ：INDEX CHANGE TYPE2

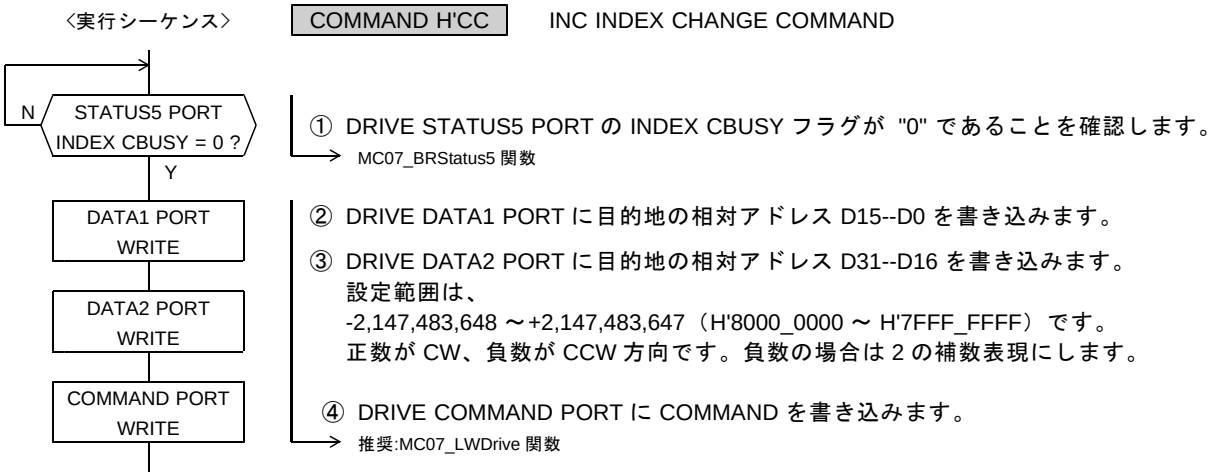
INDEX CHANGE 指令を実行する変更動作点を選択します。

TYPE2	TYPE1	TYPE0	INDEX CHANGE を実行する変更動作点〈レベル検出〉
0	0	0	INDEX CHANGE コマンドの書き込みで実行する
0	0	1	設定禁止
0	1	0	STATUS5 PORT の SS0 = 1 で実行する（注）
0	1	1	STATUS5 PORT の SS1 = 1 で実行する（注）
1	0	0	設定禁止
1	0	1	設定禁止
1	1	0	設定禁止
1	1	1	設定禁止

（注）SS0, SS1 信号は SPEC INITIALIZE2 コマンドで「汎用入力」に設定している場合に有効です。

(2) INC INDEX CHANGE

指定したデータを、起動位置を原点とする相対アドレスの停止位置に設定して、
INC INDEX ドライブを行います。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←						INDEX CHANGE データ									→ D0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D31 ←						INDEX CHANGE データ									→ D16

・指定する相対アドレスは、起動位置から停止位置までのパルス数を、起動位置を原点として
符号付きで表現した値です。

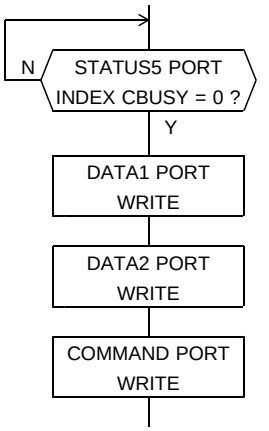
【注意】
INC INDEX CHANGE コマンドで指定する相対アドレスは、実行中のドライブ方向と同じ符号にしてください。
・実行中のドライブが ＋(CW)方向の場合は、H'0000_0000 ～ H'7FFF_FFFF を設定範囲にします。
・実行中のドライブが －(CCW)方向の場合は、H'8000_0000 ～ H'FFFF_FFFF を設定範囲にします。

INC INDEX CHANGE コマンドの指定アドレスを、実行中のドライブ方向と反対の符号で指定すると、
「反転動作が必要な INDEX CHANGE 指令のエラー」が検出されず、そのまま反対符号の指定アドレスを
停止位置に設定してしまいます。
この状態では、内部の 32 ビット INDEX COUNTER がオーバーフローを超えて回転して、指定した反対符号
のアドレスになるまで停止しません。

(3) ABS INDEX CHANGE

指定したデータを、アドレスカウンタで管理している絶対アドレスの停止位置に設定して、
ABS INDEX ドライブを行います。

〈実行シーケンス〉 COMMAND H'CD ABS INDEX CHANGE COMMAND



① DRIVE STATUS5 PORT の INDEX CBUSY フラグが "0" であることを確認します。
→ MC07_BRStatus5 関数

② DRIVE DATA1 PORT に目的地の絶対アドレス A15--A0 を書き込みます。

③ DRIVE DATA2 PORT に目的地の絶対アドレス A31--A16 を書き込みます。
設定範囲は、
-2,147,483,648 ～+2,147,483,647（H'8000_0000 ～ H'7FFF_FFFF）です。
負数の場合は 2 の補数表現にします。

④ DRIVE COMMAND PORT に COMMAND を書き込みます。
→ 推奨:MC07_LWDrive 関数

DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A15	←					INDEX CHANGE データ								→	A0

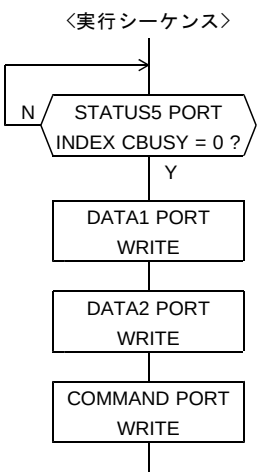
DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A31	←					INDEX CHANGE データ								→	A16

・指定する絶対アドレスは、アドレスカウンタで管理している絶対アドレスです。

(4) PLS INDEX CHANGE

指定したデータを、変更動作点の検出位置を原点とする相対アドレスの停止位置に設定して、
INC INDEX ドライブを行います。



COMMAND H'CE　PLS INDEX CHANGE COMMAND

① DRIVE STATUS5 PORT の INDEX CBUSY フラグが "0" であることを確認します。
→ MC07_BRStatus5 関数

② DRIVE DATA1 PORT に目的地の相対アドレス D15--D0 を書き込みます。

③ DRIVE DATA2 PORT に目的地の相対アドレス D31--D16 を書き込みます。
設定範囲は、
-2,147,483,648 ～+2,147,483,647（H'8000_0000 ～ H'7FFF_FFFF）です。
正数が CW、負数が CCW 方向です。負数の場合は 2 の補数表現にします。

④ DRIVE COMMAND PORT に COMMAND を書き込みます。
→ 推奨:MC07_LWDrive 関数

DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←						INDEX CHANGE データ								→	D0

DRIVE DATA2 PORT の設定データ

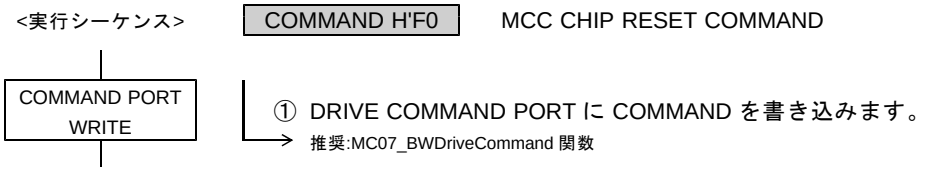
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D31 ←						INDEX CHANGE データ								→	D16

・指定する相対アドレスは、変更動作点の検出位置から停止位置までのパルス数を、変更動作点の
検出位置を原点として符号付きで表現した値です。

3-1-11. その他

(1) MCC CHIP RESET

相関軸両軸のどちらに実行しても有効です。このコマンドの実行は常時可能です。
MCC07E 内部の両軸の全てのデータを初期化して、リセット入力後と同じ状態にします。



* 当コマンドを実行すると、MCC07E の全てのデータがリセット直後の設定に戻ります。
当コマンドを実行した後は、アプリケーションの最初から起動する、または MPL リセット関数を実行して、デバイスドライバも併せて初期化し直してください。

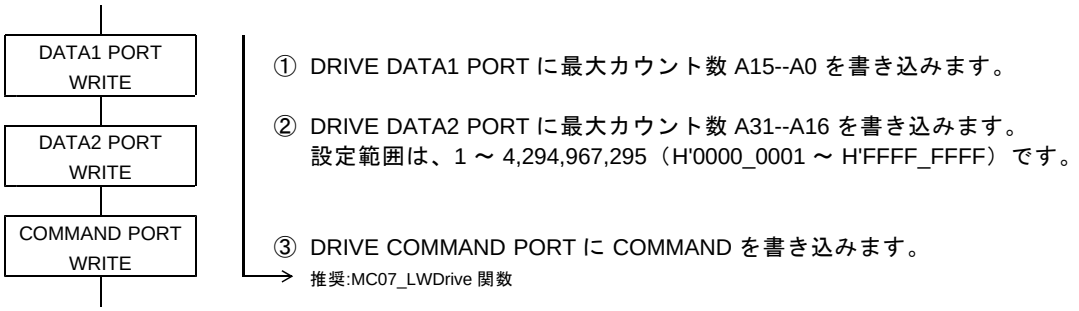
3-2. カウンタコマンド

3-2-1. アドレスカウンタの設定

(1) ADDRESS COUNTER MAX COUNT SET

アドレスカウンタの最大カウント数を設定します。
このコマンドは常時実行可能です。

<実行シーケンス>　COMMAND H'87　ADDRESS COUNTER MAX COUNT SET COMMAND



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A15	←					最大カウント数								→	A0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
A31	←					最大カウント数								→	A16

●リセット後の初期値は H'FFFF_FFFF です。

・カウント数が設定値の 1/2 に達すると、STATUS4 PORT の ADDRESS OVF = 1 になります。

・最大カウント数を設定しても、現在のアドレスカウンタの値は変わりません。
アドレスカウンタの値が、最大カウント数の範囲内になったときから、設定が有効になります。

・アドレスカウンタの最大カウント数を H'FFFF_FFFF 以外に設定した場合、ABS INDEX および ABS INDEX CHANGE コマンドを実行しないで下さい。

■最大カウント数

設定値をカウンタの最大値として、リングカウントします。
STATUS4 PORT の ADDRESS OVF フラグを無視すれば、回転系のアドレス管理ができます。

・最大カウント数＝ 1,999 の場合（2,000 カウントで 1 回転）
＋方向のカウント：0 → 1 →…→ 999 → 1000（ADDRESS OVF = 1）→ 1001 →…→ 1999 → 0
－方向のカウント：0 → 1999 →…→ 1001 → 1000（ADDRESS OVF = 1）→ 999 →…→ 1 → 0

・最大カウント数＝ 2,000 の場合（2,001 カウントで 1 回転）
＋方向のカウント：0 → 1 →…→ 1000 → 1001（1001 になると ADDRESS OVF = 1）→…→ 2000 → 0
－方向のカウント：0 → 2000 →…→ 1001 → 1000（1000 になると ADDRESS OVF = 1）→…→ 1 → 0

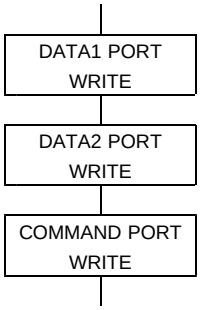
3-2-2. パルスカウンタの設定

(1) PULSE COUNTER MAX COUNT SET

パルスカウンタの最大カウント数を設定します。このコマンドは常時実行可能です。

<実行シーケンス>

COMMAND H'97　PULSE COUNTER MAX COUNT SET COMMAND



① DRIVE DATA1 PORT に最大カウント数 D15--D0 を書き込みます。

② DRIVE DATA2 PORT に最大カウント数 D31--D16 を書き込みます。
設定範囲は、1 ～ 4,294,967,295（H'0000_0001 ～ H'FFFF_FFFF）です。

③ DRIVE COMMAND PORT に COMMAND を書き込みます。
→ 推奨:MC07_LWDrive 関数

DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←						最大カウント数								→	D0

DRIVE DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D31 ←						最大カウント数								→	D16

●リセット後の初期値は H'FFFF FFFF です。

・カウント数が設定値の 1/2 に達すると、STATUS4 PORT の PULSE OVF = 1 になります。

・最大カウント数を設定しても、現在のパルスカウンタの値は変わりません。
パルスカウンタの値が、最大カウント数の範囲内になったときから、設定が有効になります。

■最大カウント数

設定値をカウンタの最大値として、リングカウントします。
STATUS4 PORT の PULSE OVF フラグを無視すれば、回転系のアドレス管理ができます。

・最大カウント数＝ 1,999 の場合（2,000 カウントで 1 回転）
＋方向のカウント：0 → 1 →…→ 999 → 1000（ADDRESS OVF = 1）→ 1001 →…→ 1999 → 0
－方向のカウント：0 → 1999 →…→ 1001 → 1000（ADDRESS OVF = 1）→ 999 →…→ 1 → 0

・最大カウント数＝ 2,000 の場合（2,001 カウントで 1 回転）
＋方向のカウント：0 → 1 →…→ 1000 → 1001（1001 になると ADDRESS OVF = 1）→…→ 2000 → 0
－方向のカウント：0 → 2000 →…→ 1001 → 1000（1000 になると ADDRESS OVF = 1）→…→ 1 → 0

3-2-3. カウンタのラッチ・クリア機能の設定

設定したラッチタイミングのアクティブエッジで、カウンタのカウントデータをラッチします。
ラッチしたデータは、次のラッチタイミングのアクティブエッジが入力するまで保存します。
ラッチデータは、DRIVE DATA1, 2 PORT（READ）から読み出します。

各カウンタには、ラッチタイミングによるカウンタのクリア機能があります。

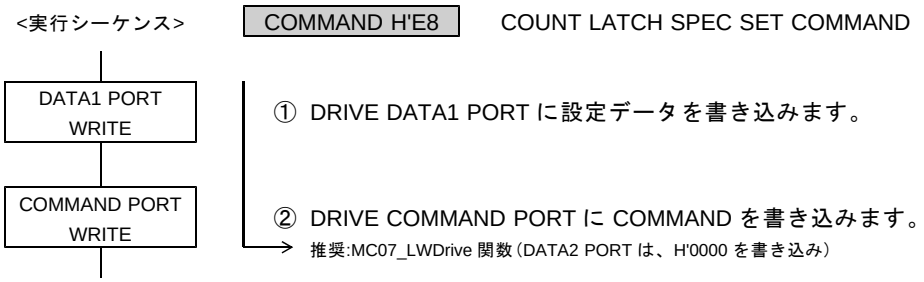
■カウンタのクリア機能

カウントデータのラッチと同時に、カウンタのデータを "0" にクリアします。
カウンタのカウントとクリアのタイミングが同時に発生した場合は、クリアを優先します。

(1) COUNT LATCH SPEC SET

各種カウンタのカウントデータをラッチするタイミングとクリア機能を設定します。
このコマンドの実行は常時可能です。

<実行シーケンス>



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
−	−	−	−	DFL CLR ENABLE	DFL LATCH TYPE2	DFL LATCH TYPE1	DFL LATCH TYPE0

D7	D6	D5	D4	D3	D2	D1	D0
PULSE CLR ENABLE	PULSE LATCH TYPE2	PULSE LATCH TYPE1	PULSE LATCH TYPE0	ADDRESS CLR ENABLE	ADDRESS LATCH TYPE2	ADDRESS LATCH TYPE1	ADDRESS LATCH TYPE0

●リセット後の初期値は H'00（アンダーライン側）です。

D0 ：ADDRESS LATCH TYPE0
D1 ：ADDRESS LATCH TYPE1
D2 ：ADDRESS LATCH TYPE2
アドレスカウンタのカウントデータをラッチするタイミングを選択します。

TYPE2	TYPE1	TYPE0	ラッチタイミング <エッジ検出>
0	0	0	ADDRESS LATCH DATA READ コマンドの実行でラッチする
0	0	1	設定禁止
0	1	0	STATUS5 PORT の SS0 = 0 → 1 でラッチする
0	1	1	ORIGIN SPEC SET コマンドの ORG 検出信号の検出エッジでラッチする
1	0	0	設定禁止
1	0	1	設定禁止
1	1	0	設定禁止
1	1	1	設定禁止

D3 ：ADDRESS CLR ENABLE
カウンタのクリア機能で、アドレスカウンタを「クリアする／クリアしない」を選択します。
0 ：クリアしない
1 ：クリアする

D4 ：PULSE LATCH TYPE0
D5 ：PULSE LATCH TYPE1
D6 ：PULSE LATCH TYPE2
パルスカウンタのカウントデータをラッチするタイミングを選択します。

TYPE2	TYPE1	TYPE0	ラッチタイミング <エッジ検出>
0	0	0	PULSE LATCH DATA READ コマンドの実行でラッチする
0	0	1	設定禁止
0	1	0	STATUS5 PORT の SS0 = 0 → 1 でラッチする
0	1	1	ORIGIN SPEC SET コマンドの ORG 検出信号の検出エッジでラッチする
1	0	0	設定禁止
1	0	1	設定禁止
1	1	0	設定禁止
1	1	1	設定禁止

D7 ：PULSE CLR ENABLE
カウンタのクリア機能で、パルスカウンタを「クリアする／クリアしない」を選択します。
0 ：クリアしない
1 ：クリアする

D8 ：DFL LATCH TYPE0
D9 ：DFL LATCH TYPE1
D10 ：DFL LATCH TYPE2
パルス偏差カウンタのカウントデータをラッチするタイミングを選択します。

TYPE2	TYPE1	TYPE0	ラッチタイミング <エッジ検出>
0	0	0	DFL LATCH DATA READ コマンドの実行でラッチする
0	0	1	設定禁止
0	1	0	STATUS5 PORT の SS0 = 0 → 1 でラッチする
0	1	1	ORIGIN SPEC SET コマンドの ORG 検出信号の検出エッジでラッチする
1	0	0	設定禁止
1	0	1	設定禁止
1	1	0	設定禁止
1	1	1	設定禁止

D11 ：DFL CLR ENABLE
カウンタのクリア機能で、パルス偏差カウンタを「クリアする／クリアしない」を選択します。
0 ：クリアしない
1 ：クリアする

3-2-4. カウントデータのラッチデータの読み出し

■ラッチデータの読み出しシーケンス

DRIVE DATA1 PORT の読み出しデータ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←							ラッチデータ								→ D0

DRIVE DATA2 PORT の読み出しデータ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D31 ←							ラッチデータ								→ D16

・ADDRESS LATCH DATA READ コマンドまたは PULSE LATCH DATA READ コマンドを実行すると、カウンタのラッチデータを DRIVE DATA1, 2 PORT（READ）にセットします。

・DFL LATCH DATA READ コマンドを実行すると、カウンタのラッチデータを DRIVE DATA1 PORT（READ）にセットします。

（1）ADDRESS LATCH DATA READ

アドレスカウンタのラッチデータを読み出します。このコマンドの実行は常時可能です。

COMMAND H'DC　ADDRESS LATCH DATA READ COMMAND

（2）PULSE LATCH DATA READ

パルスカウンタのラッチデータを読み出します。このコマンドの実行は常時可能です。

COMMAND H'DD　PULSE LATCH DATA READ COMMAND

（3）DFL LATCH DATA READ

パルス偏差カウンタのラッチデータを読み出します。このコマンドの実行は常時可能です。

COMMAND H'DE　DFL LATCH DATA READ COMMAND

3-3. HARD CONFIG コマンド

3-3-1. 入出力仕様の設定

(1) MAN MASK

特殊 I/O コネクタ(J3)の $\overline{\text{MAN}}$ 信号(MANUAL モード切り替え信号)の入力をマスクします。
このコマンドの実行は常時可能です。

CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
–	–	–	–	–	–	–	–

D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	MAN MASK	–	–	–	–

●リセット後の初期値はマスクしないです。

D4：MAN MASK
0 ：マスクしない
1 ：マスクする

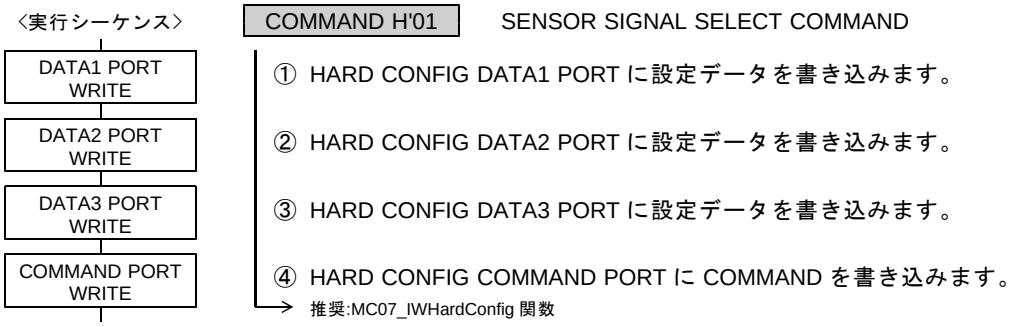
特殊 I/O コネクタの $\overline{\text{MAN}}$ 信号による MANUAL モード切り替えをアプリケーションから禁止することができます。

・「マスクする」に設定すると、MANUAL モードの切り替えが $\overline{\text{MAN}}$ 信号から操作禁止の状態になります。
このとき、HARD CONFIG STATUS1 PORT の $\overline{\text{MAN RDY}}$=0、特殊 I/O コネクタの $\overline{\text{MAN RDY}}$=HIGH にします。

・「マスクしない」に設定すると、MANUAL モードの切り替えが $\overline{\text{MAN}}$ 信号から可能な状態になります。
このとき、HARD CONFIG STATUS1 PORT の $\overline{\text{MAN RDY}}$=1、特殊 I/O コネクタの $\overline{\text{MAN RDY}}$=LOW にします。

(2) SENSOR SIGNAL SELECT

各軸 MCC07E の多用途センサ信号(SS0,SS1 信号)に接続する信号を設定します。



HARD CONFIG DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	–	← AXIS SEL →				

HARD CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	–	–	–	–	SIGNAL GATE TYPE	

HARD CONFIG DATA3 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	← SIGNAL2 SEL →					–	–	–	← SIGNAL1 SEL →				

■ HARD CONFIG DATA1 PORT の内容

D4-0 : AXIS SEL

設定の対象多用途センサを選択します。

AXIS SEL	対象多用途センサ			
	C-VX870(E)v1	C-VX871(E)v1	C-VX872(E)v1	C-VX873(E)v1
H'00	X 軸 SS0	X 軸 SS0	X1 軸 SS0	X1 軸 SS0
H'01	X 軸 SS1	X 軸 SS1	X1 軸 SS1	X1 軸 SS1
H'02	Y 軸 SS0	Y 軸 SS0	Y1 軸 SS0	Y1 軸 SS0
H'03	Y 軸 SS1	Y 軸 SS1	Y1 軸 SS1	Y1 軸 SS1
H'04	Z 軸 SS0	Z 軸 SS0	Z1 軸 SS0	Z1 軸 SS0
H'05	Z 軸 SS1	Z 軸 SS1	Z1 軸 SS1	Z1 軸 SS1
H'06	A 軸 SS0	A 軸 SS0	A1 軸 SS0	A1 軸 SS0
H'07	A 軸 SS1	A 軸 SS1	A1 軸 SS1	A1 軸 SS1
H'08	設定禁止(無効)	B 軸 SS0	設定禁止(無効)	B1 軸 SS0
H'09	設定禁止(無効)	B 軸 SS1	設定禁止(無効)	B1 軸 SS1
H'0A	設定禁止(無効)	C 軸 SS0	設定禁止(無効)	C1 軸 SS0
H'0B	設定禁止(無効)	C 軸 SS1	設定禁止(無効)	C1 軸 SS1
H'0C-H'0F	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)
H'10	設定禁止(無効)	設定禁止(無効)	X2 軸 SS0	X2 軸 SS0
H'11	設定禁止(無効)	設定禁止(無効)	X2 軸 SS1	X2 軸 SS1
H'12	設定禁止(無効)	設定禁止(無効)	Y2 軸 SS0	Y2 軸 SS0
H'13	設定禁止(無効)	設定禁止(無効)	Y2 軸 SS1	Y2 軸 SS1
H'14	設定禁止(無効)	設定禁止(無効)	Z2 軸 SS0	Z2 軸 SS0
H'15	設定禁止(無効)	設定禁止(無効)	Z2 軸 SS1	Z2 軸 SS1
H'16	設定禁止(無効)	設定禁止(無効)	A2 軸 SS0	A2 軸 SS0
H'17	設定禁止(無効)	設定禁止(無効)	A2 軸 SS1	A2 軸 SS1
H'18	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	B2 軸 SS0
H'19	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	B2 軸 SS1
H'1A	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	C2 軸 SS0
H'1B	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	C2 軸 SS1
H'1C-H'1F	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)

■ HARD CONFIG DATA2 PORT の内容

D1-0：SIGNAL GATE TYPE
"なし"を選択した場合は、SIGNAL1 SEL で選択した信号をそのまま接続します。
"AND"または"OR"を選択した場合は SIGNAL1 SEL と SIGNAL2 SEL で選択した信号の指定合成論理信号を接続します。

SIGNAL GATE TYPE	合成論理
H'0	なし（SIGNAL1 SEL 選択信号を使用）
H'1	AND
H'2	OR
H'3	設定禁止

●リセット後の初期値はアンダーライン側です。

■ HARD CONFIG DATA3 PORT の内容

D4-0 ：SIGNAL1 SEL
D12-8 ：SIGNAL2 SEL
合成論理に使用する信号を選択します。

SIGNAL SEL	合成論理に使用する信号			
	C-VX870(E)v1	C-VX871(E)v1	C-VX872(E)v1 *1	C-VX873(E)v1 *1
H'00	X 軸 OUTA	X 軸 OUTA	Xn 軸 OUTA	Xn 軸 OUTA
H'01	X 軸 OUTB	X 軸 OUTB	Xn 軸 OUTB	Xn 軸 OUTB
H'02	Y 軸 OUTA	Y 軸 OUTA	Yn 軸 OUTA	Yn 軸 OUTA
H'03	Y 軸 OUTB	Y 軸 OUTB	Yn 軸 OUTB	Yn 軸 OUTB
H'04	Z 軸 OUTA	Z 軸 SUTA	Zn 軸 OUTA	Zn 軸 OUTA
H'05	Z 軸 OUTB	Z 軸 SUTB	Zn 軸 OUTB	Zn 軸 OUTB
H'06	A 軸 OUTA	A 軸 OUTA	An 軸 OUTA	An 軸 OUTA
H'07	A 軸 OUTB	A 軸 OUTB	An 軸 OUTB	An 軸 OUTB
H'08	なし（LOW レベル固定）	B 軸 OUTA	なし（LOW レベル固定）	Bn 軸 OUTA
H'09	なし（LOW レベル固定）	B 軸 OUTB	なし（LOW レベル固定）	Bn 軸 OUTB
H'0A	なし（LOW レベル固定）	C 軸 OUTA	なし（LOW レベル固定）	Cn 軸 OUTA
H'0B	なし（LOW レベル固定）	C 軸 OUTB	なし（LOW レベル固定）	Cn 軸 OUTB
H'0C	$\overline{\text{SENSOR0}}$ 入力信号	$\overline{\text{SENSOR0}}$ 入力信号	$\overline{\text{SENSORn0}}$ 入力信号	$\overline{\text{SENSORn0}}$ 入力信号
H'0D	$\overline{\text{SENSOR1}}$ 入力信号	$\overline{\text{SENSOR1}}$ 入力信号	$\overline{\text{SENSORn1}}$ 入力信号	$\overline{\text{SENSORn1}}$ 入力信号
H'0E	$\overline{\text{SIGNAL IN0}}$ 入力信号	$\overline{\text{SIGNAL IN0}}$ 入力信号	$\overline{\text{SIGNAL INn0}}$ 入力信号	$\overline{\text{SIGNAL INn0}}$ 入力信号
H'0F	$\overline{\text{SIGNAL IN1}}$ 入力信号	$\overline{\text{SIGNAL IN1}}$ 入力信号	$\overline{\text{SIGNAL INn1}}$ 入力信号	$\overline{\text{SIGNAL INn1}}$ 入力信号
H'1x	なし（LOW レベル固定）	なし（LOW レベル固定）	なし（LOW レベル固定）	なし（LOW レベル固定）

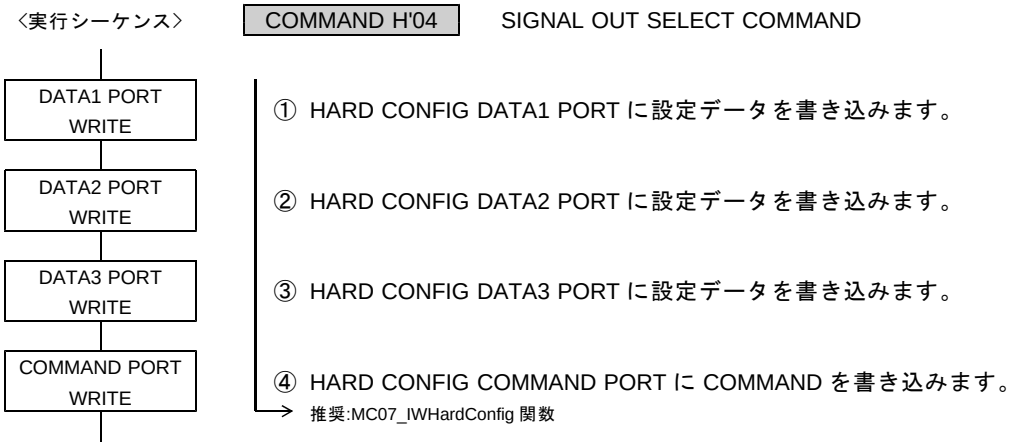
●リセット後の初期値は以下の通りです。

対象多用途センサ	SIGNAL1 SEL	SIGNAL2 SEL
Zn 軸 SS0	$\overline{\text{SENSORn0}}$ 入力信号	なし（LOW レベル固定）
An 軸 SS0	$\overline{\text{SENSORn1}}$ 入力信号	なし（LOW レベル固定）
その他の軸 SS0,SS1	なし（LOW レベル固定）	なし（LOW レベル固定）

*1：C-VX872(E)v1, C-VX873(E)v1 の場合、使用できる信号は設定の対象と同じ系列軸内の信号になります。
例） X2 軸 SS0 の設定の場合、n=2

(3) SIGNAL OUT SELECT

特殊 I/O コネクタの SIGNAL OUTnx 信号(ステータス出力信号)に出力する信号を設定します。



HARD CONFIG DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	–	–	–	–	SIGNAL OUT SEL	

HARD CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	SIGNAL OUT TYPE		–	–	SIGNAL GATE TYPE	

HARD CONFIG DATA3 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–		← SIGNAL2 SEL →				–	–			← SIGNAL1 SEL →			

■ HARD CONFIG DATA1 PORT の内容

D1-0 ：SIGNAL OUT SEL

設定の対象ステータス出力信号を選択します。

SIGNAL OUT SEL	対象ステータス出力信号	
	C-VX870(E)v1,C-VX871(E)v1	C-VX872(E)v1,C-VX873(E)v1
H'0	$\overline{\text{SIGNAL OUT0}}$	$\overline{\text{SIGNAL OUT10}}$
H'1	$\overline{\text{SIGNAL OUT1}}$	$\overline{\text{SIGNAL OUT11}}$
H'2	設定禁止(無効)	$\overline{\text{SIGNAL OUT20}}$
H'3	設定禁止(無効)	$\overline{\text{SIGNAL OUT21}}$

■ HARD CONFIG DATA2 PORT の内容

D1-0：SIGNAL GATE TYPE1-0
"なし"を選択した場合は、SIGNAL1 SEL で選択した信号を出力方式に応じて出力します。
"AND"または"OR"を選択した場合は SIGNAL1 SEL と SIGNAL2 SEL で選択した信号の指定合成論理信号を出力方式に応じて出力します。

SIGNAL GATE TYPE	合成論理
H'0	なし(SIGNAL1 SEL 選択信号を使用)
H'1	AND
H'2	OR
H'3	設定禁止

●リセット後の初期値はアンダーライン側です。

D5-4：SIGNAL OUT TYPE
出力方式を設定します。

SIGNAL OUT TYPE	出力方式
H'0	合成論理信号をスルー出力
H'1	合成論理信号のアクティブエッジでワンショット出力
H'2	合成論理信号のノットアクティブエッジでワンショット出力
H'3	設定禁止(合成論理信号をスルー出力)

●リセット後の初期値はアンダーライン側です。

・ワンショット出力に設定した場合、合成論理信号の指定されたエッジを検出すると
SIGNAL OUT TIMER SET コマンドで設定された時間、アクティブレベルを出力します。

■ HARD CONFIG DATA3 PORT の内容

D3-0 ：SIGNAL1 SEL
D11-8 ：SIGNAL2 SEL
合成論理に使用する信号を選択します。

SIGNAL SEL	合成論理に使用する信号			
	C-VX870(E)v1	C-VX871(E)v1	C-VX872(E)v1 *1	C-VX873(E)v1 *1
H'0	X 軸 OUTA	X 軸 OUTA	Xn 軸 OUTA	Xn 軸 OUTA
H'1	X 軸 OUTB	X 軸 OUTB	Xn 軸 OUTB	Xn 軸 OUTB
H'2	Y 軸 OUTA	Y 軸 OUTA	Yn 軸 OUTA	Yn 軸 OUTA
H'3	Y 軸 OUTB	Y 軸 OUTB	Yn 軸 OUTB	Yn 軸 OUTB
H'4	Z 軸 OUTA	Z 軸 SUTA	Zn 軸 OUTA	Zn 軸 OUTA
H'5	Z 軸 OUTB	Z 軸 SUTB	Zn 軸 OUTB	Zn 軸 OUTB
H'6	A 軸 OUTA	A 軸 OUTA	An 軸 OUTA	An 軸 OUTA
H'7	A 軸 OUTB	A 軸 OUTB	An 軸 OUTB	An 軸 OUTB
H'8	なし(ノットアクティブ固定)	B 軸 OUTA	なし(ノットアクティブ固定)	Bn 軸 OUTA
H'9	なし(ノットアクティブ固定)	B 軸 OUTB	なし(ノットアクティブ固定)	Bn 軸 OUTB
H'A	なし(ノットアクティブ固定)	C 軸 OUTA	なし(ノットアクティブ固定)	Cn 軸 OUTA
H'B	なし(ノットアクティブ固定)	C 軸 OUTB	なし(ノットアクティブ固定)	Cn 軸 OUTB
H'C	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)
H'D	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)
H'E	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)
H'F	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)	なし(ノットアクティブ固定)

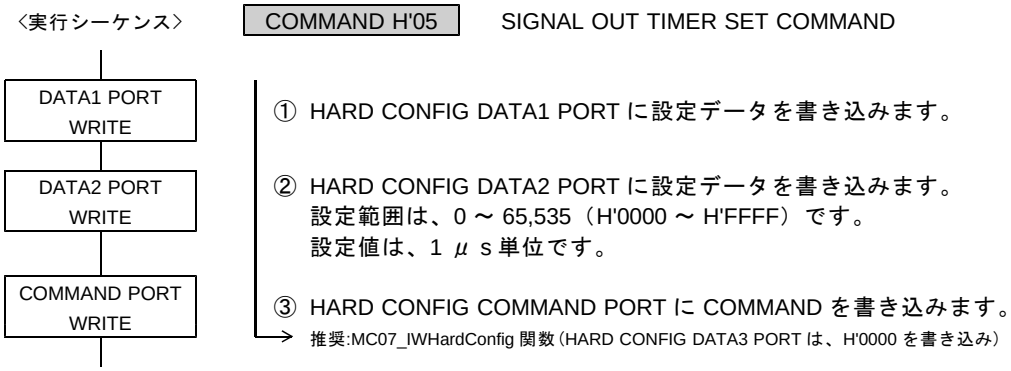
●リセット後の初期値は以下の通りです。

対象ステータス出力	SIGNAL1 SEL	SIGNAL2 SEL
$\overline{\text{SIGNAL OUTn0}}$	Xn 軸 OUTA	Xn 軸 OUTA
$\overline{\text{SIGNAL OUTn1}}$	Yn 軸 OUTA	Xn 軸 OUTA

*1：C-VX872(E)v1, C-VX873(E)v1 の場合、使用できる信号は設定の対象と同じ系列軸内の信号になります。
例） $\overline{\text{SIGNAL OUT20}}$ の設定の場合、n=2

(4) SIGNAL OUT TIMER SET

特殊 I/O コネクタの SIGNAL OUTnx 信号(ステータス出力信号)の出力方式をワンショット出力とした場合の出力時間を設定します。



HARD CONFIG DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	–	–	–	–	SIGNAL OUT SEL	

HARD CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
D15 ←						ワンショット出力時間								→	D0

●リセット後の初期値は H'0001(1 μ s)です。

・H'0000 を設定した場合、出力時間は 1 μ s に補正して出力します。

D1-0 ：SIGNAL OUT SEL 1-0

設定の対象ステータス出力信号を選択します。

SIGNAL OUT SEL	対象ステータス出力信号	
	C-VX870(E)v1,C-VX871(E)v1	C-VX872(E)v1,C-VX873(E)v1
H'0	SIGNAL OUT0	SIGNAL OUT10
H'1	SIGNAL OUT1	SIGNAL OUT11
H'2	設定禁止(無効)	SIGNAL OUT20
H'3	設定禁止(無効)	SIGNAL OUT21

(5) SIGNAL OUT LATCH STATUS CLR

HARD CONFIG STATUS4 PORT の SIGNAL OUTnx LATCH フラグを個別にクリアします。
このコマンドの実行は常時可能です。

HARD CONFIG STATUS4 PORT の SIGNAL OUTnx LATCH フラグは対応する出力信号の
アクティブエッジを検出をラッチします。当コマンドはこのフラグのクリアに使用します。

<実行シーケンス>

COMMAND H'06 SIGNAL OUT STATUS CLR COMMAND

DATA2 PORT WRITE

① CONFIG DATA2 PORT にクリアデータを書き込みます。
推奨:MC07_BWHardConfigData2 関数

COMMAND PORT WRITE

② CONFIG COMMAND PORT に COMMAND を書き込みます。
推奨:MC07_BWHardConfigCommand 関数

HARD CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	
–	–	–	–	SIGNAL OUT21 LATCH CLR	SIGNAL OUT20 LATCH CLR	–	–	C-VX872(E)v1 C-VX873(E)v1
				–	–			C-VX870(E)v1 C-VX871(E)v1

D7	D6	D5	D4	D3	D2	D1	D0	
–	–	–	–	SIGNAL OU11 LATCH CLR	SIGNAL OUT10 LATCH CLR	–	–	C-VX872(E)v1 C-VX873(E)v1
				SIGNAL OUT1 LATCH CLR	SIGNAL OUT0 LATCH CLR			C-VX870(E)v1 C-VX871(E)v1

D3 - D2 ：クリアデータ
D11-D10：クリアデータ
SIGNAL OUT LATCH のクリアデータを選択します。
0 ：クリアしない
1 ：クリアする

・コマンドの実行で、SIGNAL OUT LATCH フラグをクリアします。
このコマンドのデータは、コマンド実行時のみ有効です。（トリガ入力）

3-3-2. 同期スタート機能の設定と実行

(1) PAUSE SET SPEC

同期スタートさせる軸の PAUSE 信号を ON にする条件を設定します。
このコマンドの実行は常時可能です。

〈実行シーケンス〉 COMMAND H'11 PAUSE SET SPEC COMMAND

HARD CONFIG DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	–	← AXIS SEL →				

HARD CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	SIGNAL GATE TYPE		–	–	PAUSE SET TYPE	

HARD CONFIG DATA3 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	← SIGNAL2 SEL →				–	–	–	–	← SIGNAL1 SEL →			

■ HARD CONFIG DATA1 PORT の内容

D4-0 : AXIS SEL
設定の対象軸を選択します。

AXIS SEL	設定の対象軸			
	C-VX870(E)v1	C-VX871(E)v1	C-VX872(E)v1	C-VX873(E)v1
H'00	X 軸	X 軸	X1 軸	X1 軸
H'01	Y 軸	Y 軸	Y1 軸	Y1 軸
H'02	Z 軸	Z 軸	Z1 軸	Z1 軸
H'03	A 軸	A 軸	A1 軸	A1 軸
H'04	設定禁止(無効)	B 軸	設定禁止(無効)	B1 軸
H'05	設定禁止(無効)	C 軸	設定禁止(無効)	C1 軸
H'06-H'0F	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)
H'10	設定禁止(無効)	設定禁止(無効)	X2 軸	X2 軸
H'11	設定禁止(無効)	設定禁止(無効)	Y2 軸	Y2 軸
H'12	設定禁止(無効)	設定禁止(無効)	Z2 軸	Z2 軸
H'13	設定禁止(無効)	設定禁止(無効)	A2 軸	A2 軸
H'14	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	B2 軸
H'15	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	C2 軸
H'16-H'1F	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)

■ HARD CONFIG DATA2 PORT の内容

D1-0 ：PAUSE SET TYPE
PAUSE SET 条件を選択します。

PAUSE SET TYPE	PAUSE SET 条件
H'0	PAUSE コマンド実行で PAUSE SET
H'1	合成論理信号のアクティブエッジ検出で PAUSE SET
H'2	合成論理信号のノットアクティブエッジ検出で PAUSE SET
H'3	設定禁止（PAUSE コマンド実行で PAUSE SET）

●リセット後の初期値はアンダーライン側です。

D5-4：SIGNAL GATE TYPE
PAUSE SET 条件を「合成論理信号のアクティブエッジ／ノットアクティブエッジ検出」とした場合は、合成論理を設定します。
"なし"を選択した場合は、SIGNAL1 SEL で選択した信号のエッジ検出を PAUSE SET 条件にします。
"AND"または"OR"を選択した場合は SIGNAL1 SEL と SIGNAL2 SEL で選択した信号の指定合成論理信号のエッジ検出を PAUSE SET 条件にします。

SIGNAL GATE TYPE	合成論理
H'0	なし(SIGNAL1 SEL 選択信号を使用)
H'1	AND
H'2	OR
H'3	設定禁止

●リセット後の初期値はアンダーライン側です。

■ HARD CONFIG DATA3 PORT の内容

D3-0 ：SIGNAL1 SEL
D11-8 ：SIGNAL2 SEL
合成論理に使用する信号を選択します。

SIGNAL SEL	合成論理に使用する信号			
	C-VX870（E）v1	C-VX871（E）v1	C-VX872（E）v1 *1	C-VX873（E）v1 *1
H'00	X 軸 OUTA	X 軸 OUTA	Xn 軸 OUTA	Xn 軸 OUTA
H'01	X 軸 OUTB	X 軸 OUTB	Xn 軸 OUTB	Xn 軸 OUTB
H'02	Y 軸 OUTA	Y 軸 OUTA	Yn 軸 OUTA	Yn 軸 OUTA
H'03	Y 軸 OUTB	Y 軸 OUTB	Yn 軸 OUTB	Yn 軸 OUTB
H'04	Z 軸 OUTA	Z 軸 SUTA	Zn 軸 OUTA	Zn 軸 OUTA
H'05	Z 軸 OUTB	Z 軸 SUTB	Zn 軸 OUTB	Zn 軸 OUTB
H'06	A 軸 OUTA	A 軸 OUTA	An 軸 OUTA	An 軸 OUTA
H'07	A 軸 OUTB	A 軸 OUTB	An 軸 OUTB	An 軸 OUTB
H'08	なし（LOW レベル固定）	B 軸 OUTA	なし（LOW レベル固定）	Bn 軸 OUTA
H'09	なし（LOW レベル固定）	B 軸 OUTB	なし（LOW レベル固定）	Bn 軸 OUTB
H'0A	なし（LOW レベル固定）	C 軸 OUTA	なし（LOW レベル固定）	Cn 軸 OUTA
H'0B	なし（LOW レベル固定）	C 軸 OUTB	なし（LOW レベル固定）	Cn 軸 OUTB
H'0C	$\overline{\text{SENSOR0}}$ 入力信号	$\overline{\text{SENSOR0}}$ 入力信号	$\overline{\text{SENSORn0}}$ 入力信号	$\overline{\text{SENSORn0}}$ 入力信号
H'0D	$\overline{\text{SENSOR1}}$ 入力信号	$\overline{\text{SENSOR1}}$ 入力信号	$\overline{\text{SENSORn1}}$ 入力信号	$\overline{\text{SENSORn1}}$ 入力信号
H'0E	$\overline{\text{SIGNAL IN0}}$ 入力信号	$\overline{\text{SIGNAL IN0}}$ 入力信号	$\overline{\text{SIGNAL INn0}}$ 入力信号	$\overline{\text{SIGNAL INn0}}$ 入力信号
H'0F	$\overline{\text{SIGNAL IN1}}$ 入力信号	$\overline{\text{SIGNAL IN1}}$ 入力信号	$\overline{\text{SIGNAL INn1}}$ 入力信号	$\overline{\text{SIGNAL INn1}}$ 入力信号

●リセット後の初期値はアンダーライン側です。

*1：C-VX872（E）v1, C-VX873（E）v1 の場合、使用できる信号は設定の対象軸と同じ系列軸内の信号になります。
例）X2 軸の設定の場合、n=2

・対象軸の PAUSE SET 条件と PAUSE CLR 条件が同じ場合は、PAUSE 信号は ON しません。
（PAUSE コマンドの場合は除く）

(2) PAUSE CLR SPEC

同期スタートさせる軸の PAUSE 信号を OFF にする条件を設定します。
このコマンドの実行は常時可能です。

〈実行シーケンス〉 COMMAND H'12 PAUSE CLR SPEC COMMAND

HARD CONFIG DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	–	← AXIS SEL →				

HARD CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	–	–	–	–	–	–	SIGNAL GATE TYPE		–	–	PAUSE CLR TYPE	

HARD CONFIG DATA3 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
–	–	–	–	← SIGNAL2 SEL			→	–	–	–	–	← SIGNAL1 SEL			→

■ HARD CONFIG DATA1 PORT の内容

D4-0 : AXIS SEL
設定の対象軸を選択します。

AXIS SEL	設定の対象軸			
	C-VX870(E)v1	C-VX871(E)v1	C-VX872(E)v1	C-VX873(E)v1
H'00	X 軸	X 軸	X1 軸	X1 軸
H'01	Y 軸	Y 軸	Y1 軸	Y1 軸
H'02	Z 軸	Z 軸	Z1 軸	Z1 軸
H'03	A 軸	A 軸	A1 軸	A1 軸
H'04	設定禁止(無効)	B 軸	設定禁止(無効)	B1 軸
H'05	設定禁止(無効)	C 軸	設定禁止(無効)	C1 軸
H'06-H'0F	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)
H'10	設定禁止(無効)	設定禁止(無効)	X2 軸	X2 軸
H'11	設定禁止(無効)	設定禁止(無効)	Y2 軸	Y2 軸
H'12	設定禁止(無効)	設定禁止(無効)	Z2 軸	Z2 軸
H'13	設定禁止(無効)	設定禁止(無効)	A2 軸	A2 軸
H'14	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	B2 軸
H'15	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	C2 軸
H'16-H'1F	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)	設定禁止(無効)

■ HARD CONFIG DATA2 PORT の内容

D1-0 ：PAUSE CLR TYPE
PAUSE CLR 条件を選択します。

PAUSE CLR TYPE	PAUSE CLR 条件
H'0	PAUSE コマンド実行で PAUSE CLR
H'1	合成論理信号のアクティブエッジ検出で PAUSE CLR
H'2	合成論理信号のノットアクティブエッジ検出で PAUSE CLR
H'3	設定禁止（PAUSE コマンド実行で PAUSE CLR）

●リセット後の初期値はアンダーライン側です。

D5-4：SIGNAL GATE TYPE
PAUSE CLR 条件を「合成論理信号のアクティブエッジ／ノットアクティブエッジ検出」とした場合は、合成論理を設定します。
"なし"を選択した場合は、SIGNAL1 SEL で選択した信号のエッジ検出を PAUSE CLR 条件にします。
"AND"または"OR"を選択した場合は SIGNAL1 SEL と SIGNAL2 SEL で選択した信号の指定合成論理信号のエッジ検出を PAUSE CLR 条件にします。

SIGNAL GATE TYPE	合成論理
H'0	なし(SIGNAL1 SEL 選択信号を使用)
H'1	AND
H'2	OR
H'3	設定禁止

●リセット後の初期値はアンダーライン側です。

■ HARD CONFIG DATA3 PORT の内容

D3-0 ：SIGNAL1 SEL
D11-8 ：SIGNAL2 SEL
合成論理に使用する信号を選択します。

SIGNAL SEL1,2	合成論理に使用する信号			
	C-VX870（E）v1	C-VX871（E）v1	C-VX872（E）v1 *1	C-VX873（E）v1 *1
H'00	X 軸 OUTA	X 軸 OUTA	Xn 軸 OUTA	Xn 軸 OUTA
H'01	X 軸 OUTB	X 軸 OUTB	Xn 軸 OUTB	Xn 軸 OUTB
H'02	Y 軸 OUTA	Y 軸 OUTA	Yn 軸 OUTA	Yn 軸 OUTA
H'03	Y 軸 OUTB	Y 軸 OUTB	Yn 軸 OUTB	Yn 軸 OUTB
H'04	Z 軸 OUTA	Z 軸 SUTA	Zn 軸 OUTA	Zn 軸 OUTA
H'05	Z 軸 OUTB	Z 軸 SUTB	Zn 軸 OUTB	Zn 軸 OUTB
H'06	A 軸 OUTA	A 軸 OUTA	An 軸 OUTA	An 軸 OUTA
H'07	A 軸 OUTB	A 軸 OUTB	An 軸 OUTB	An 軸 OUTB
H'08	なし（LOW レベル固定）	B 軸 OUTA	なし（LOW レベル固定）	Bn 軸 OUTA
H'09	なし（LOW レベル固定）	B 軸 OUTB	なし（LOW レベル固定）	Bn 軸 OUTB
H'0A	なし（LOW レベル固定）	C 軸 OUTA	なし（LOW レベル固定）	Cn 軸 OUTA
H'0B	なし（LOW レベル固定）	C 軸 OUTB	なし（LOW レベル固定）	Cn 軸 OUTB
H'0C	$\overline{\text{SENSOR0}}$ 入力信号	$\overline{\text{SENSOR0}}$ 入力信号	$\overline{\text{SENSORn0}}$ 入力信号	$\overline{\text{SENSORn0}}$ 入力信号
H'0D	$\overline{\text{SENSOR1}}$ 入力信号	$\overline{\text{SENSOR1}}$ 入力信号	$\overline{\text{SENSORn1}}$ 入力信号	$\overline{\text{SENSORn1}}$ 入力信号
H'0E	$\overline{\text{SIGNAL IN0}}$ 入力信号	$\overline{\text{SIGNAL IN0}}$ 入力信号	$\overline{\text{SIGNAL INn0}}$ 入力信号	$\overline{\text{SIGNAL INn0}}$ 入力信号
H'0F	$\overline{\text{SIGNAL IN1}}$ 入力信号	$\overline{\text{SIGNAL IN1}}$ 入力信号	$\overline{\text{SIGNAL INn1}}$ 入力信号	$\overline{\text{SIGNAL INn1}}$ 入力信号

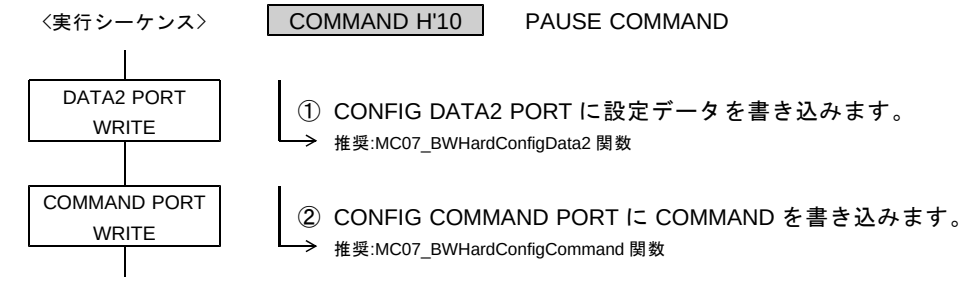
●リセット後の初期値はアンダーライン側です。

*1：C-VX872（E）v1, C-VX873（E）v1 の場合、使用できる信号は設定の対象と同じ系列軸内の信号になります。
例）X2 軸の設定の場合、n=2

・対象軸の PAUSE CLR 条件と PAUSE CLR 条件が同じ場合は、PAUSE 信号は ON しません。
（PAUSE コマンドの場合は除く）

(3) PAUSE

各軸の PAUSE 信号をコマンドで操作します。
このコマンドの実行は常時可能です。



HARD CONFIG DATA2 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8	
–	–	C2 PAUSE	B2 PAUSE	A2 PAUSE	Z2 PAUSE	Y2 PAUSE	X2 PAUSE	C-VX872(E)v1 C-VX873(E)v1
		–	–	–	–	–	–	C-VX870(E)v1 C-VX871(E)v1

D7	D6	D5	D4	D3	D2	D1	D0	
–	–	C1 PAUSE	B1 PAUSE	A1 PAUSE	Z1 PAUSE	Y1 PAUSE	X1 PAUSE	C-VX872(E)v1 C-VX873(E)v1
		C PAUSE	B PAUSE	A PAUSE	Z PAUSE	Y PAUSE	X PAUSE	C-VX870(E)v1 C-VX871(E)v1

C-VX870(E)v1,C-VX872(E)v1
C-VX871(E)v1,C-VX873(E)v1

D3 -D0：PAUSE データ
D11-D8：PAUSE データ
各軸の PAUSE 信号の状態を設定します。
0 ：PAUSE OFF
1 ：PAUSE ON

・当コマンドで PAUSE 信号を ON にするには、対象軸の PAUSE SET 条件が「PAUSE コマンド実行」に設定されている必要があります。
PAUSE SET 条件は PAUSE SET SPEC コマンドで設定します。

・当コマンドで PAUSE 信号を OFF にするには、対象軸の PAUSE CLR 条件が「PAUSE コマンド実行」に設定されている必要があります。
PAUSE CLR 条件は PAUSE CLR SPEC コマンドで設定します。

3-3-3. 設定データの読み出し

(1) HARD CONFIG SET DATA READ

HARD CONFIGURATION に設定したデータを読み出します。
このコマンドの実行は常時可能です。

<実行シーケンス>

COMMAND H'20　HARD CONFIG SET DATA READ COMMAND

DATA1 PORT WRITE
① HARD CONFIG DATA1 PORT に読み出すデータの対象軸または対象信号を書き込みます。
（読み出す COMMAND CODE の実行時の対象指定方法と同じです）

DATA2 PORT WRITE
② HARD CONFIG DATA2 PORT に読み出すデータの COMMAND CODE を書き込みます。

COMMAND PORT WRITE
③ HARD CONFIG COMMAND PORT に COMMAND を書き込みます。
→ 推奨:MC07_IWHardConfig 関数

DATA2 PORT READ
④ HARD CONFIG DATA2 PORT から指定した COMMAND CODE の DATA2 PORT の設定データを読み出します。
→ MC07_BRHardConfigData2 関数

DATA3 PORT READ
⑤ HARD CONFIG DATA3 PORT から指定した COMMAND CODE の DATA3 PORT の設定データを読み出します。
→ MC07_BRHardConfigData3 関数

・リセット後は、各機能の設定データの初期値が読み出されます。

・HARD CONFIG SET DATA READ コマンドを実行すると指定したコマンドの設定データを DRIVE DATA2, 3 PORT（READ）にセットします。
コマンドで書き込みが不要な DATA PORT のデータは、"0" になります。

●読み出しできる設定データ

COMMAND CODE	コマンド名称
H'01	SENSOR SIGNAL SELECT
H'04	SIGNAL OUT SELECT
H'05	SIGNAL OUT TIMER SET
H'11	PAUSE SET SPEC
H'12	PAUSE CLR SPEC

3-3-4. その他

(1) HARD CONFIG RESET

HARD CONFIGURATION の全てのデータを初期化して、リセット入力後と同じ状態にします。
このコマンドの実行は常時可能です。

<実行シーケンス>

COMMAND H'F0　HARD CONFIG RESET COMMAND

COMMAND PORT WRITE
① CONFIG COMMAND PORT に COMMAND を書き込みます。
→ 推奨:MC07_BWHardConfigCommand 関数

4．機能説明

4-1. ドライブ仕様

4-1-1. コマンド予約機能

各軸には、10 命令分のデータ・コマンドを格納する予約レジスタがあります。
予約レジスタには DRIVE COMMAND の汎用コマンドを予約することができます。
ドライブ起動後に次のドライブコマンドを予約することで連続したドライブを行うことができます。

予約レジスタの状態は、STATUS1 PORT の COMREG EP と COMREG FL フラグで確認します。
BUSY = 1 で、COMREG FL = 0 のときに、DRIVE COMMAND PORT に汎用コマンドを書き込むと DRIVE DATA 1, 2 PORT のデータと汎用コマンドの 1 命令分を、予約レジスタに格納します。

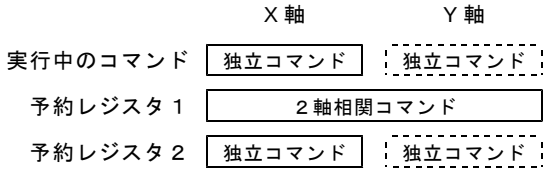
予約レジスタは FIFO 構成になっています。
実行中のコマンド処理が終了すると、予約レジスタに格納したコマンドを順次実行します。

DRIVE COMMAND の特殊コマンドは予約できません。

● 2 軸相関コマンドの予約

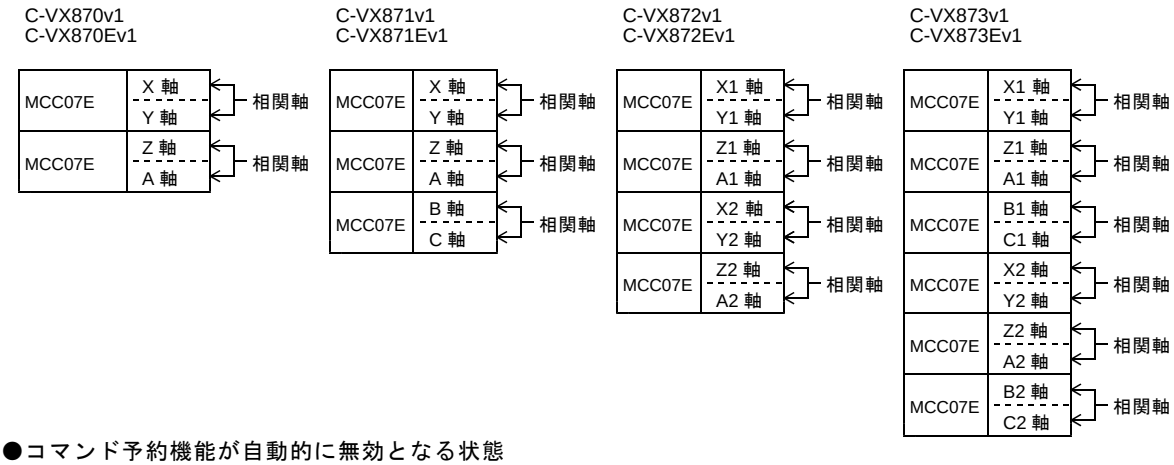
2 軸相関コマンドは、相関軸 2 軸の COMREG FL = 0 のときに予約することができます。
2 軸相関コマンドの予約は、以下のように実行します。

・2 軸相関コマンドの命令は、相関軸両軸の予約レジスタに格納します。
・片方の軸の次に実行する予約コマンドが 2 軸相関コマンドになっても、もう片方の軸が別のコマンド処理中の場合は、BUSY = 1 のまま、もう片方の軸のコマンド処理終了を待ちます。
・両軸の次に実行する予約コマンドが 2 軸相関コマンドになると、2 軸相関コマンドを実行します。



以下の 2 軸相関コマンド実行中は、相関軸のどちらの軸で停止指令が発生しても有効になります。
・MAIN XY STRAIGHT CP コマンド
・MAIN XY CIRCULAR CP コマンド

当製品での相関軸は以下の通りです。



●コマンド予約機能が自動的に無効となる状態

STATUS1 PORT の ERROR = 1 になると、予約コマンドをすべてクリアします。
また、実行待ちの予約コマンドをクリアした場合は、ERROR STATUS の COMREG CLR ERROR = 1 にします。

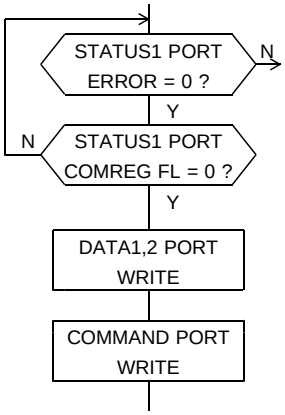
2 軸相関コマンドを予約している場合は、ERROR = 1 で、X, Y 軸の予約コマンドをすべてクリアします。
エラーが発生していない軸も、COMREG CLR ERROR = 1 により ERROR = 1 になります。

ERROR = 1 の間は、COMREG FL = 1、COMREG EP = 1 になり、予約コマンド（汎用コマンド）の書き込みが無効になります。
ERROR = 0 にクリアすると、COMREG FL = 0 になり、予約コマンドの書き込みが有効になります。

■コマンド予約機能の実行例

コマンドを予約する場合は、BUSY = 0 の代わりに COMREG FL = 0 を確認します。
予約シーケンス実行中に BUSY = 0 になった場合は、通常のコマンド実行と同様になります。

●コマンド予約シーケンス



② STATUS1 PORT の ERROR フラグが "0" であることを確認します。

③ STATUS1 PORT の COMREG FL フラグが "0" であることを確認します。

① COMMAND に必要なデータを DRIVE DATA1,2 PORT に書き込みます。

④ DRIVE COMMAND PORT に COMMAND を書き込みます。

● INDEX ドライブ → JOG ドライブの連続実行例

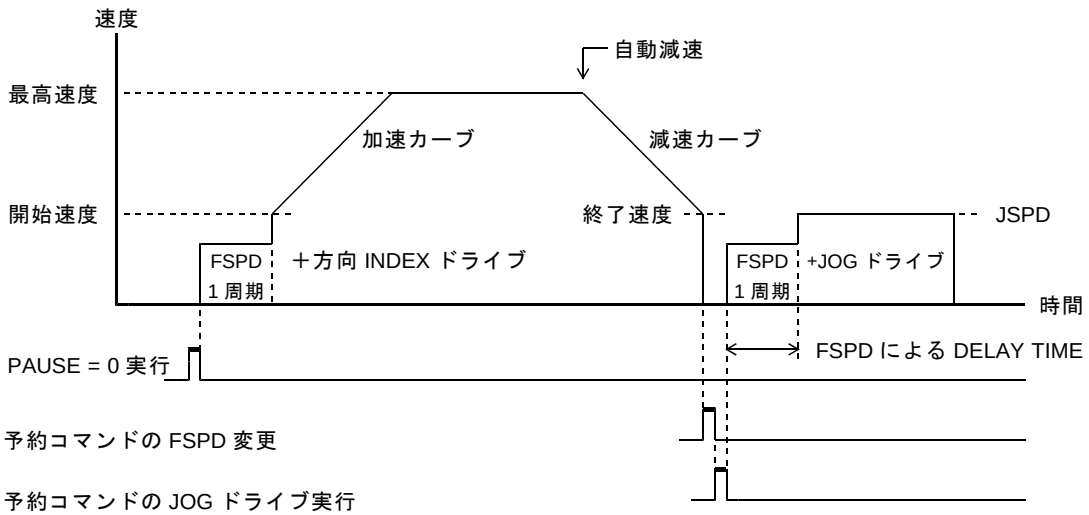
① PAUSE コマンド(HARD CONFIG コマンド)で対象軸の PAUSE を 1 にします。

② ＋方向 INDEX ドライブを実行します。
STBY = 1 のままドライブが保留になります。

③ 予約コマンドで、FSPD による DELAY TIME を設定します。

④ 予約コマンドで、＋方向 JOG ドライブを実行します。

⑤ PAUSE コマンド(HARD CONFIG コマンド)で対象軸の PAUSE を 0 にします。
STBY = 0 になり、INDEX ドライブ → JOG ドライブを開始します。



4-1-2. 入出力仕様

(1) 入力信号のデジタルフィルタ機能

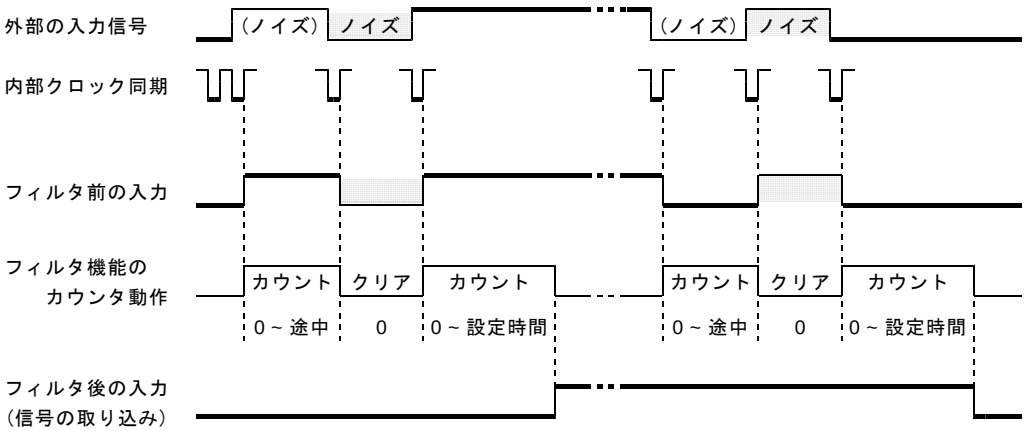
ボードコントローラ製品の入力回路には、ハード的なノイズフィルタが入っており、原則設定不要です。
環境下によって不要な信号を拾うようなとき、デジタルフィルタを設定することができます。
デジタルフィルタを設定すると設定時定数により、信号の応答性は低下します。

以下の入力信号に対してデジタルフィルタを設定できます。

入力信号	設定可能範囲	対応コマンド
CWLM, CCWLM 信号 $\overline{\text{DEND}}$/$\overline{\text{PO}}$, DALM($\overline{\text{INnx}}$)信号	0 ～ 10ms	HARD INITIALIZE4 コマンド
$\overline{\text{ORG}}$,$\overline{\text{NORG}}$ 信号	0 ～ 10ms	HARD INITIALIZE5 コマンド
± ZORG 信号	0 ～ 1ms	HARD INITIALIZE5 コマンド
± EA,± EB 信号	0 ～ 12.75 μs	HARD INITIALIZE6 コマンド

●リセット後の初期値はアンダーライン側です。

■デジタルフィルタ機能



・入力信号がＬ→Ｈ、またはＨ→Ｌに変化すると、フィルタ機能のカウントを開始して
入力信号のレベルを計測します。
フィルタ機能の設定時間分のカウントが終了すると、入力信号のレベルを取り込みます。

・計測の途中で、レベルが変化（ノイズが入力）すると、フィルタ機能のカウンタを
クリアして計測を中止します。この場合は、入力信号のレベルを取り込みません。

(2) 入力信号の論理切り替え機能

下記の入力信号のアクティブ論理を初期値から切り替えることができます。

入力信号	初期値	対応コマンド
CWLM 信号	正論理入力	HARD INITIALIZE7 コマンド
CCWLM 信号	正論理入力	
DALM($\overline{\text{INnx}}$)信号	負論理入力	
$\overline{\text{ORG}}$ 信号	負論理入力	
$\overline{\text{NORG}}$ 信号	負論理入力	

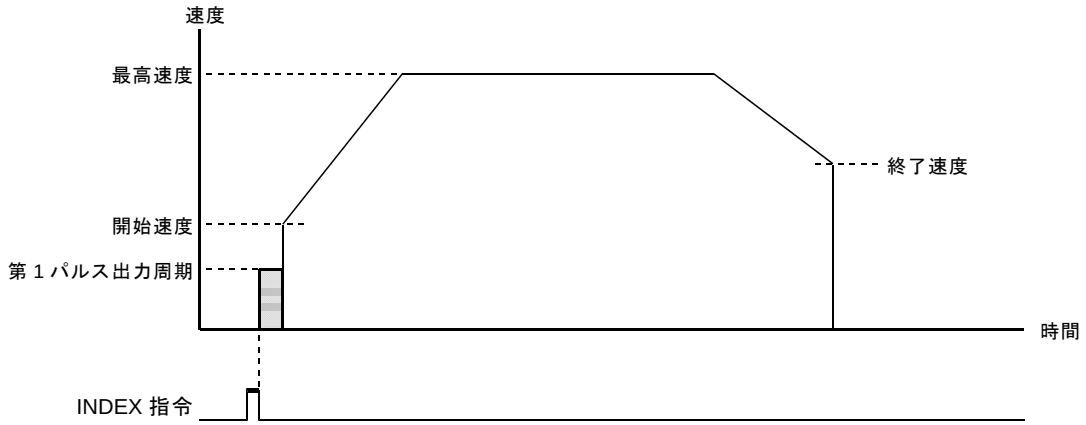
・アクティブ論理を変更すると、変更した信号のデジタルフィルタ機能が動作します。
デジタルフィルタ機能の時定数経過後に、アクティブ論理の変更が確定します。

4-1-3. ドライブパラメータ

(1) 加減速パラメータ

■最高速度、開始速度、終了速度

加減速ドライブの最高速度、開始速度、終了速度を設定できます。
設定範囲は 0.1 ～ 6,553,400 Hz です。



各速度は以下の様に設定します。

最高速度(Hz)	＝	最高速時のパルス速度データ	(HSPD)	×	速度倍率	(RESOL)
開始速度(Hz)	＝	加速開始時のパルス速度データ	(LSPD)	×	速度倍率	(RESOL)
終了速度(Hz)	＝	減速終了時のパルス速度データ	(ELSPD)	×	速度倍率	(RESOL)

速度倍率(RESOL)は RESOL No.を選択して設定します。

RESOL No.	速度倍率 (RESOL)
H'0	0.1
H'1	0.2
H'2	0.5
H'3	1

RESOL No.	速度倍率 (RESOL)
H'4	2
H'5	5
H'6	10
H'7	20

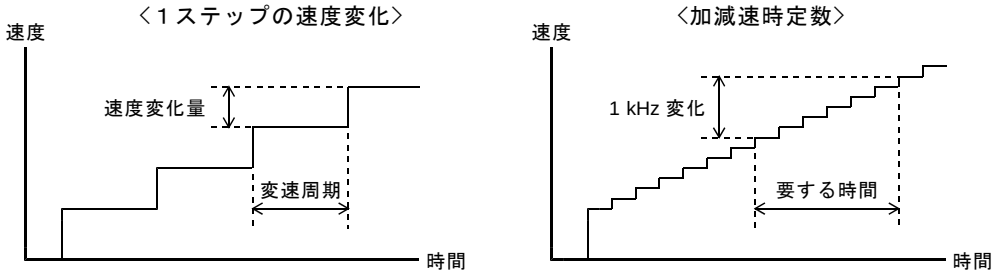
RESOL No.	速度倍率 (RESOL)
H'8	50
H'9	100
H'A	200
H'B	200

RESOL No.	速度倍率 (RESOL)
H'C	200
H'D	200
H'E	200
H'F	200

・RESOL と HSPD は HIGH SPEED SET コマンドで設定します。
・LSPD と ELSPD は LOW SPEED SET コマンドで設定します。

■加減速時定数

加減速時定数は、ドライブ速度を 1kHz 変化させるのに要する時間（ms/kHz）です。
本書ではこの時定数を RATE と呼称しています。
加速および減速は速度変化量を変速周期毎に加算および減算することで行います。



各 RATE は以下の様に設定します。

$$\text{直線加速 RATE(ms/kHz)} = \frac{\text{加速カーブの変速周期(ms)}}{\text{直線加速カーブの速度変化量(kHz)}} = \frac{\text{UCYCLE}}{\text{RESOL} \times 2}$$

・加速カーブの変速周期(ms) ＝ UCYCLE × 0.0005
・直線加速カーブの速度変化量(kHz) ＝ RESOL × 0.001

$$\text{直線減速 RATE(ms/kHz)} = \frac{\text{減速カーブの変速周期(ms)}}{\text{直線減速カーブの速度変化量(kHz)}} = \frac{\text{DCYCLE}}{\text{RESOL} \times 2}$$

・減速カーブの変速周期(ms) ＝ DCYCLE × 0.0005
・直線減速カーブの速度変化量(kHz) ＝ RESOL × 0.001

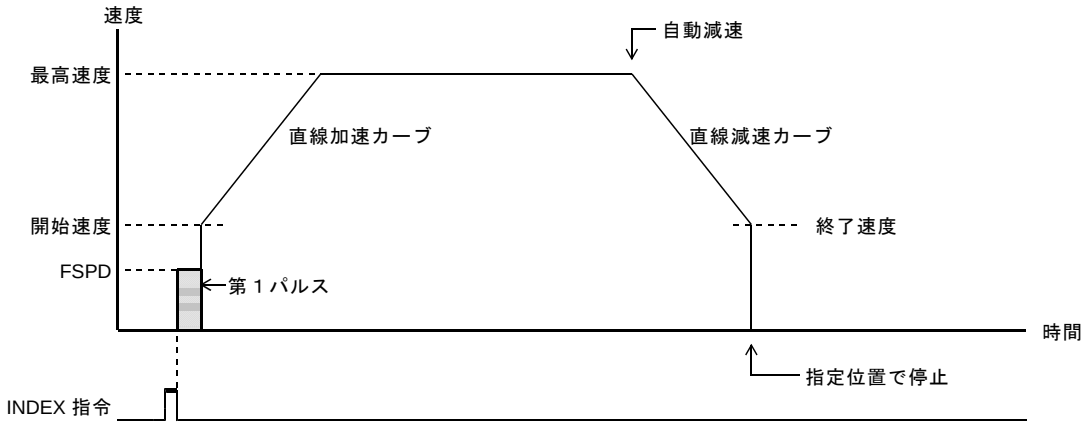
・UCYCLE と DCYCLE は RATE SET コマンドで設定します。
・RESOL で速度変化量が決定します。RESOL を小さくすると加減速が滑らかになります。

■ RATE DATA TABLE(参考)

RATE (ms/kHz)	RESOL = 1 U/D CYCLE	RESOL = 5 U/D CYCLE	RESOL = 10 U/D CYCLE	RESOL = 20 U/D CYCLE	RESOL = 50 U/D CYCLE	RESOL = 200 U/D CYCLE
5,000	10,000					
3,000	6,000					
2,000	4,000					
1,000	2,000	10,000				
500	1,000	5,000	10,000			
300	600	3,000	6,000	12,000		
200	400	2,000	4,000	8,000		
100	200	1,000	2,000	4,000	10,000	
50	100	500	1,000	2,000	5,000	
30	60	300	600	1,200	3,000	12,000
20	40	200	400	800	2,000	8,000
10	20	100	200	400	1,000	4,000
5	10	50	100	200	500	2,000
3	6	30	60	120	300	1,200
2	4	20	40	80	200	800
1	2	10	20	40	100	400
0.5	1	5	10	20	50	200
0.3		3	6	12	30	120
0.2		2	4	8	20	80
0.1		1	2	4	10	40
0.05			1	2	5	20
0.03					3	12
0.02					2	8
0.01					1	4
0.005						2
0.0025						1

(2) 直線加減速ドライブ

直線加減速ドライブは、S 字加速の変速領域を "0" に設定した加速カーブと、S 字減速の変速領域を "0" に設定した減速カーブで加減速を行うドライブです。
開始速度から最高速度まで、S 字変速領域がない直線加速カーブで加速し、最高速度から終了速度まで、S 字変速領域がない直線減速カーブで減速します。



・直線加速、直線減速カーブとするときは、SCAREA SET コマンドで S 字変速領域(SUAREA ／ SDAREA)を"0"に設定します。

●直線加減速ドライブの加速時間と減速時間

直線加速カーブの加速時間（ms）
＝（UCYCLE x 0.0005）×（HSPD − LSPD+1）＋ 第 1 パルスの周期(ms)＋ TU
：0 ≦ TU ＜ 最高速度の 1 周期

直線減速カーブの減速時間（ms）
＝（DCYCLE x 0.0005）×（HSPD − ELSPD+1)＋ TD ：0 ≦ TD ＜ 終了速度の 1 周期

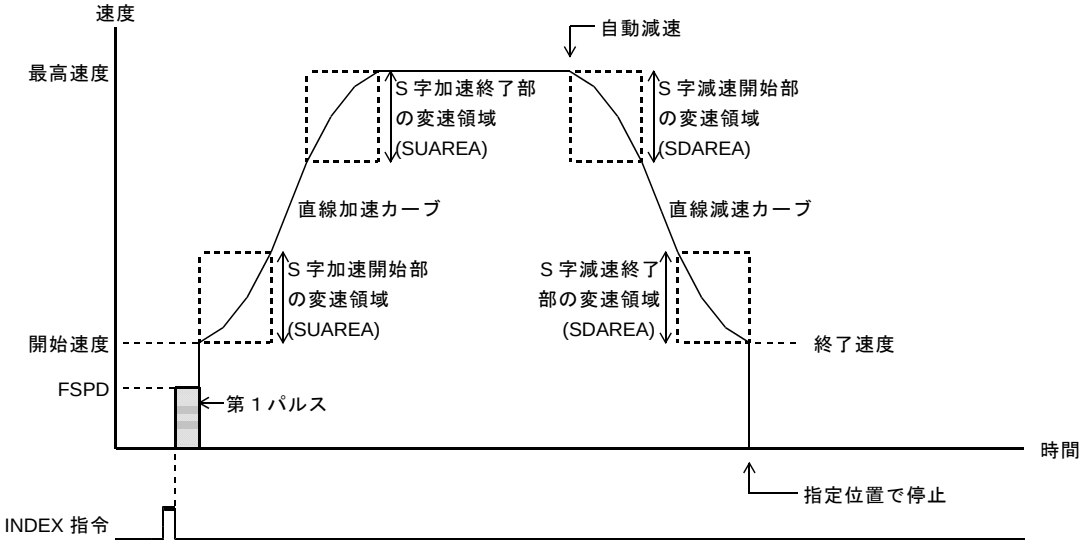
・減速停止指令で減速停止する場合の減速時間です。
・INDEX ドライブの自動減速停止時には、オフセットパルス数分の増減があります。

(3) S 字加減速ドライブ

S 字加減速ドライブは、S 字加速の変速領域を設定した加速カーブと S 字減速の変速領域を設定した減速カーブで加減速を行うドライブです。
加速開始時の S 字変速領域と加速終了時の S 字変速領域を、放物線に近似した S 字加速カーブで加速し、減速開始時の S 字変速領域と減速終了時の S 字変速領域を、放物線に近似した S 字減速カーブで減速します。

● S 字加速カーブ
SCAREA SET コマンドの SUAREA で S 字加速の変速領域を設定します。
SUAREA で設定した変速領域が、加速開始時の S 字変速領域と加速終了時の S 字変速領域になり、S 字加速カーブを形成します。残りの速度領域は、UCYCLE の直線加速カーブで加速します。

● S 字減速カーブ
SCAREA SET コマンドの SDAREA で S 字減速の変速領域を設定します。
SDAREA で設定した変速領域が、減速開始時の S 字変速領域と減速終了時の S 字変速領域になり、S 字減速カーブを形成します。残りの速度領域は、DCYCLE の直線減速カーブで減速します。



・SUAREA, SDAREA は SCAREA SET コマンドで設定します。

● S 字加減速ドライブの加速時間と減速時間

S 字加速カーブの加速時間（ms）
＝（UCYCLE x 0.0005）×（HSPD − LSPD ＋ 1 ＋ SUAREA x 2）＋ 第 1 パルスの周期(ms)＋ TU
: 0 ≦ TU ＜ 最高速度の 1 周期

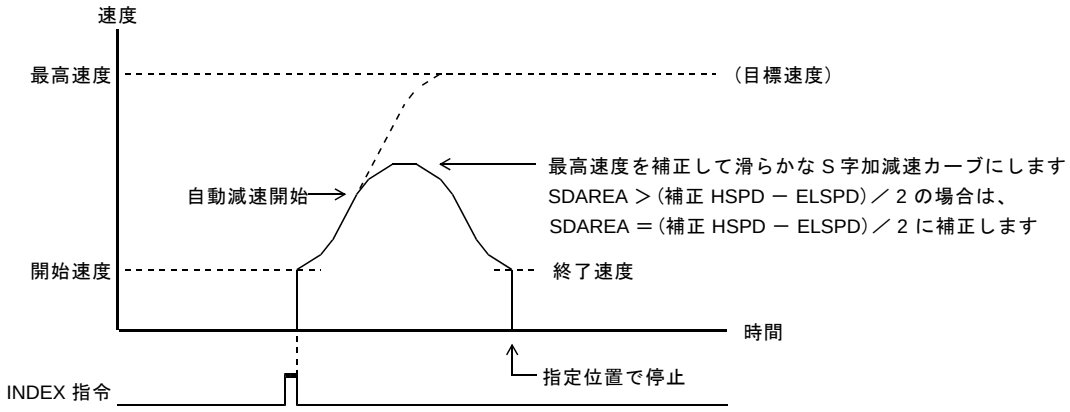
・SUAREA ＜(HSPD − LSPD)／2 で、加速する場合の加速時間です。

S 字減速カーブの減速時間（ms）
＝（DCYCLE x 0.0005）×（HSPD − ELSPD ＋ 1 ＋ SDAREA x 2）＋ TD
: 0 ≦ TD ＜ 終了速度の 1 周期

・SDAREA ＜(HSPD − ELSPD)／2 で、減速停止指令で減速停止する場合の減速時間です。
・INDEX ドライブの自動減速停止時には、オフセットパルス数分の増減があります。

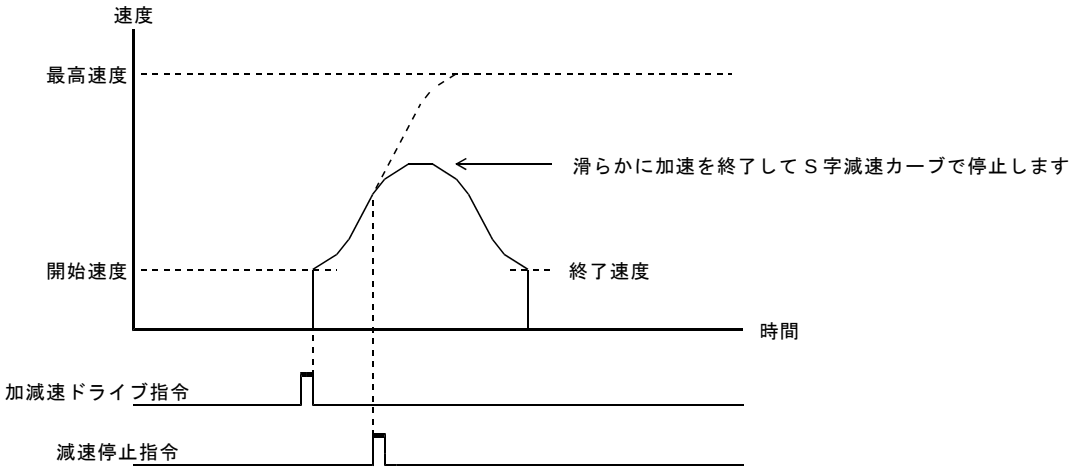
■ S 字加減速 INDEX ドライブの三角駆動回避動作

S 字加減速の INDEX ドライブで、停止位置までのパルス数が少なくて最高速度（目標速度）に達しない場合は自動的に最高速度を引き下げて、滑らかな S 字加減速カーブで INDEX ドライブを停止します。
この機能は常時有効です。



■減速停止指令検出時の三角駆動回避動作

S 字加速中に減速停止指令を検出した場合は、SUAREA の S 字加速終了カーブで滑らかに加速を終了し、S 字減速カーブで減速停止します。
この機能は常時有効です。

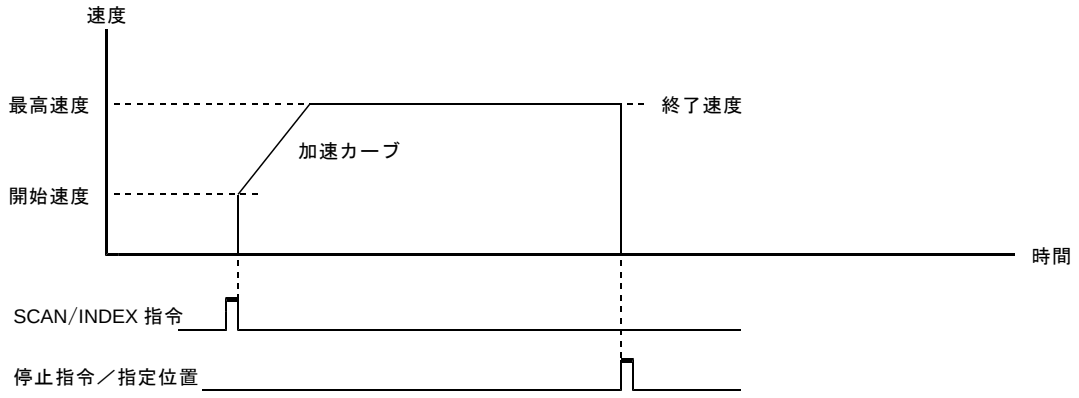


(4) その他のドライブ

以下のような加速ドライブ、一定速ドライブ、減速ドライブを、コマンド予約機能を用いて繋ぎ合わせ、連続したドライブパターンを作ることができます。

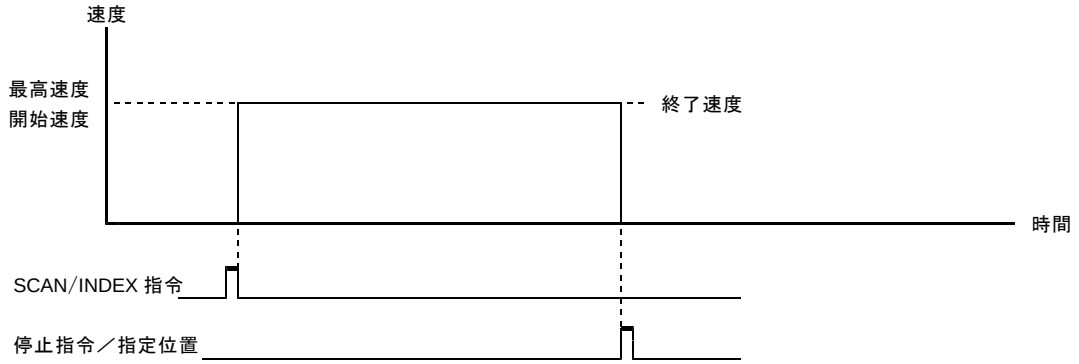
■加速ドライブ

「開始速度＜最高速度」および「最高速度＝終了速度」に設定すると、開始速度と最高速度による加速ドライブを行います。



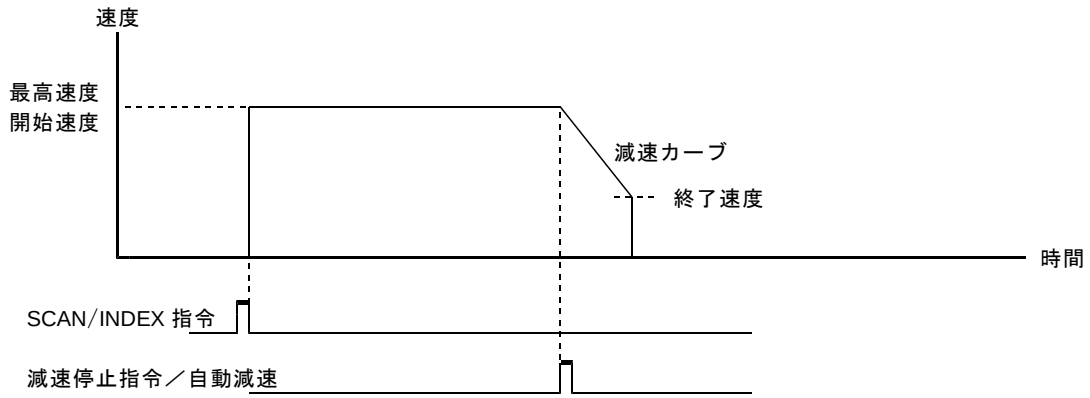
■一定速ドライブ

「開始速度＝最高速度＝終了速度」または「最高速度＝終了速度＝０」に設定すると、開始速度での一定速ドライブを行います。



■減速ドライブ

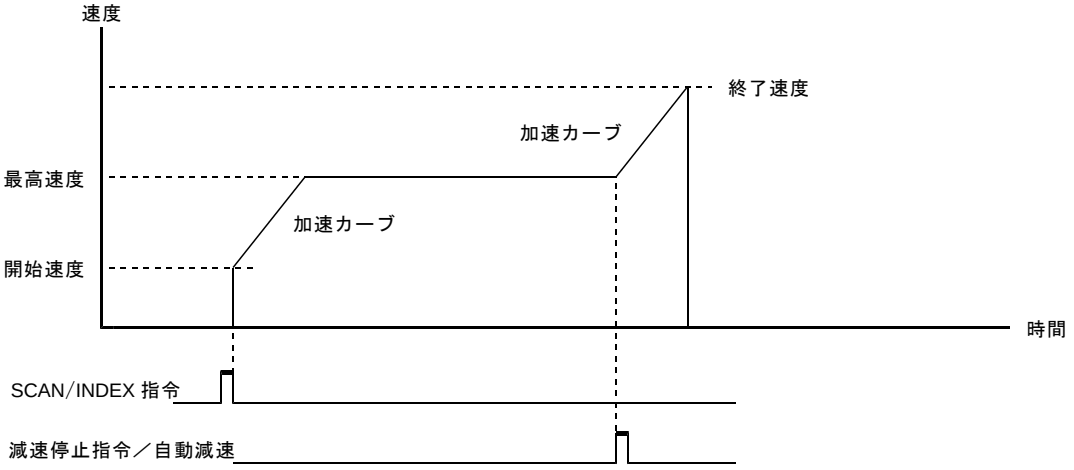
「開始速度＝最高速度」および「最高速度＞終了速度」に設定すると、最高速度と終了速度による減速ドライブを行います。



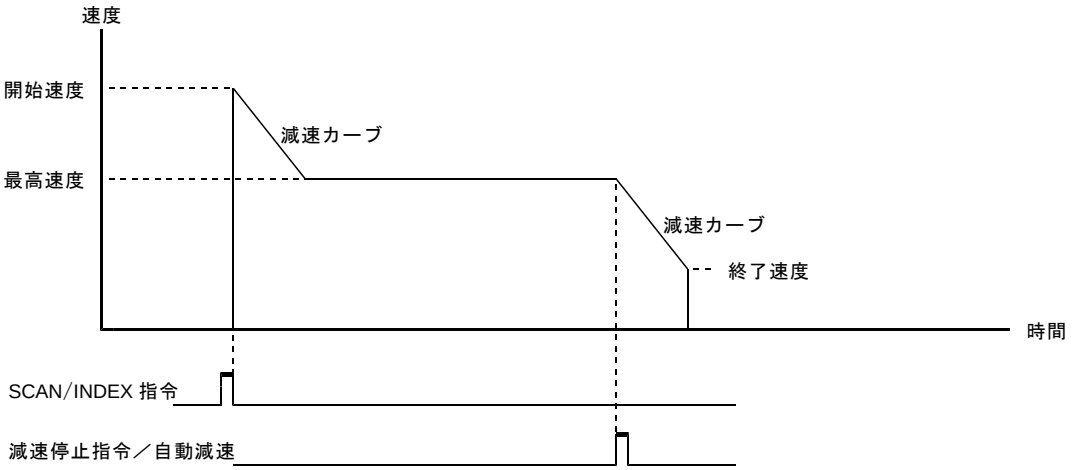
■その他のドライブ

また、開始速度、最高速度、終了速度を組み合わせると、色々なドライブパターンが形成できます。

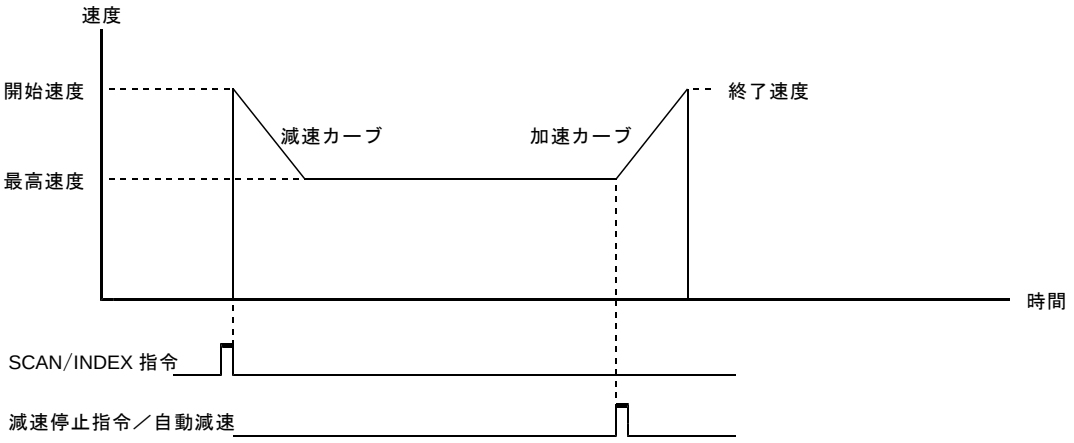
●「開始速度＜最高速度＜終了速度」に設定すると、以下の加速ドライブを行います。



●「開始速度＞最高速度＞終了速度」に設定すると、以下の減速ドライブを行います。



●「開始速度＞最高速度」および「最高速度＜終了速度」に設定すると、以下の加減速ドライブを行います。



4-1-4. ORIGIN ドライブ

（1）ORIGIN ドライブ

ドライブ工程を指定して ORIGIN ドライブコマンドを実行すると、ORG 検出信号の指定エッジを検出してドライブを終了します。
検出する ORG 検出信号は、$\overline{\text{ORG}}$ 信号, ± ZORG 信号, $\overline{\text{DEND}}$/$\overline{\text{PO}}$ 信号, $\overline{\text{NORG}}$ 信号の合成信号から選択します。

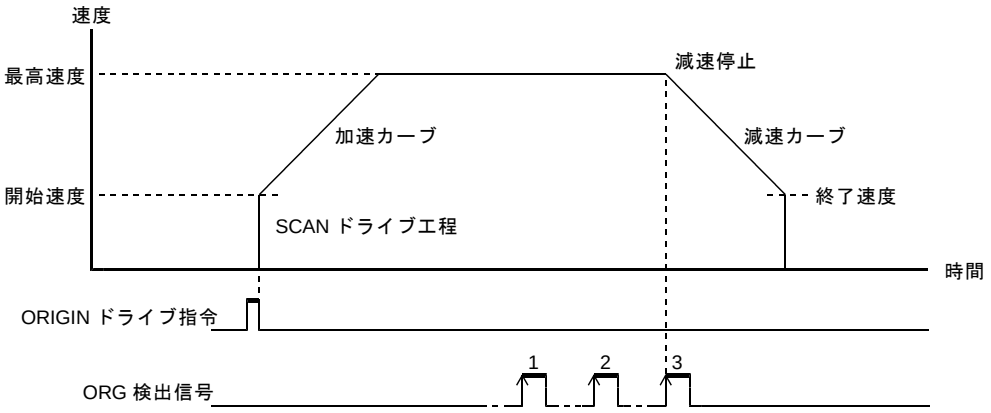
ORIGIN ドライブには、以下のドライブパラメータの設定が必要です。
・加減速ドライブのパラメータ
・JSPD　：JOG ドライブのパルス速度

■ ORIGIN ドライブ工程

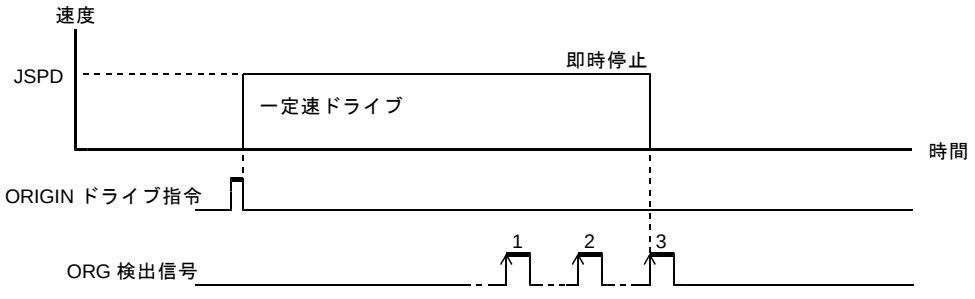
ドライブ工程は、ORIGIN SCAN ドライブと ORIGIN CONSTANT SCAN ドライブが選択できます。

＜ドライブ工程の実行例＞
ORG 検出信号の３カウント目のアクティブエッジ検出で、停止機能を動作させます。
・ORG DETECT EDGE = 0　　：ORG 検出信号の 0 → 1（アクティブ）エッジを検出する
・ORIGIN COUNT D3--D0 = H'2：３カウント目のエッジ検出で、停止機能を動作させる

● ORIGIN SCAN ドライブ
加減速ドライブのパラメータで、SCAN ドライブを行います。
ORG 検出信号の指定エッジを検出すると減速停止します。



● ORIGIN CONSTANT SCAN ドライブ
JOG ドライブのパルス速度（JSPD）で、一定速ドライブを行います。
ORG 検出信号の指定エッジを検出すると即時停止します。

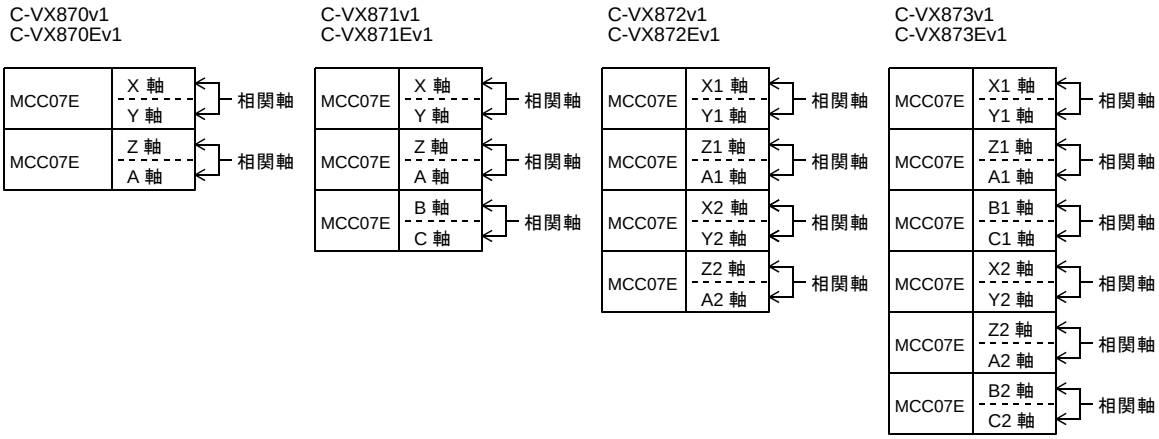


4-1-5. 補間ドライブ

補間ドライブには、相関軸で実行する相関 2 軸補間ドライブと、任意軸間で実行する任意軸補間ドライブがあります。

(1) 相関 2 軸補間ドライブ

相関軸で行う補間ドライブです。当製品における相関軸は以下の通りです。



相関軸で 2 軸直線補間ドライブ、および 2 軸円弧補間ドライブが行えます。
相関 2 軸直線補間ドライブ … MAIN XY STRAIGHT CP コマンド
相関 2 軸円弧補間ドライブ … MAIN XY CIRCULAR CP コマンド

・相関 2 軸補間ドライブは、コマンド実行軸の加減速パラメータで、補間ドライブの基本パルスを発生します。
補間制御部は、発生した基本パルスを補間演算して補間パルスを出力します。

・相関 2 軸補間ドライブのコマンドは、相関軸のどちらの軸に実行しても有効です。
コマンドの実行で 2 軸のドライブを開始します。

・同期スタート機能は、コマンド実行軸で有効です。他軸の同期スタート機能は無効になります。
コマンド実行軸の STBY = 0 で、他軸も STBY = 0 になります。

・LIMIT 停止指令、減速停止指令、即時停止指令は、相関軸両軸のどちらで発生しても有効です。

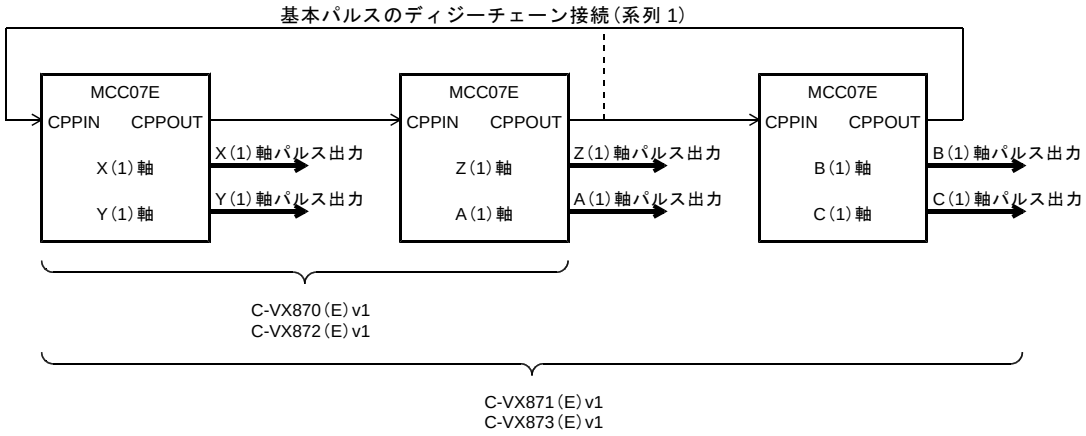
・エラーが発生した場合は、エラー該当軸が ERROR = 1 になります。
但し、エラーによる停止機能は、X, Y 軸のどちらでエラーが発生しても有効です。

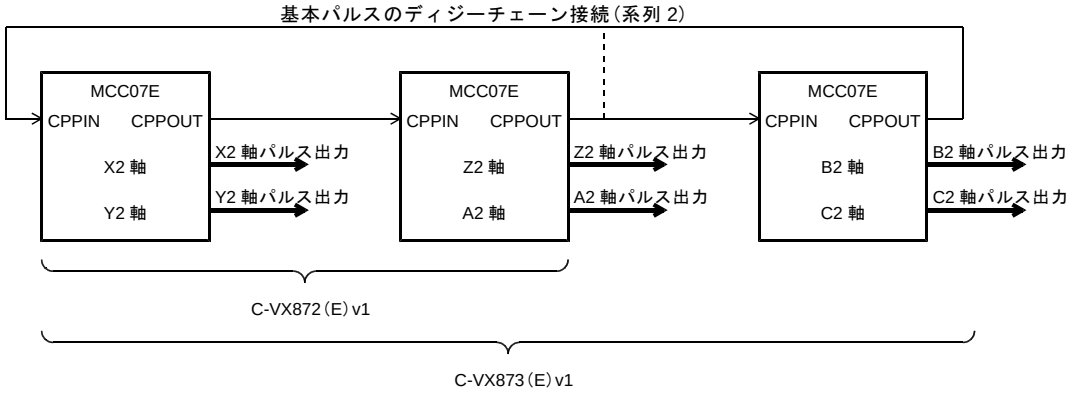
・両軸のドライブが終了すると、両軸が BUSY = 0 になります。
DEND/PO 信号または DRST 信号を〈サーボ対応〉に設定している場合は、両軸の〈サーボ対応〉が終了した後に、両軸が BUSY = 0 になります。

(2) 任意軸補間ドライブ

基本パルスのディジーチェーン接続を使用して任意軸で行う補間ドライブです。
各軸 MCC07E の CPPOUT 端子と CPPIN 端子はディジーチェーン接続で繋がっています。
任意軸補間ドライブではメイン軸のチップが CPPOUT 端子に補間ドライブの基本パルスを出力します。
サブ軸のチップは基本パルスを CPPIN 端子から入力して CPPOUT 端子に出力します。

・C-VX872(E)v1, C-VX873(E)v1 にはディジーチェーン接続が 2 系列あります。
　任意軸補間ドライブはディジーチェーン接続の系列内の軸間で行います。





任意軸で多軸直線補間ドライブ、および 2 軸円弧補間ドライブが行えます。

任意多軸直線補間ドライブ	…	MAIN STRAIGHT CP コマンド	(メイン軸)
		SUB STRAIGHT CP コマンド	(サブ軸)
任意 2 軸円弧補間ドライブ	…	MAIN CIRCULAR CP コマンド	(メイン軸)
		SUB CIRCULAR CP コマンド	(サブ軸)

【注意】
任意 2 軸円弧補間ドライブ実行時に注意事項があります。
3-1-6.章「任意 2 軸円弧補間ドライブの実行シーケンス」をご覧ください。

・任意軸補間ドライブは、メイン軸の加減速パラメータで、補間ドライブの基本パルスを発生します。
　補間制御部は、発生した基本パルスを補間演算して補間パルスを出力します。

・サブ軸は、CPPIN 端子から入力するパルスを補間ドライブの基本パルスにします。
　補間制御部は、CPPIN 端子の入力パルスを補間演算して補間パルスを出力します。

・メイン軸補間ドライブは、各サブ軸にサブ軸補間ドライブを実行した後に、メイン軸に実行してください。
　メイン軸はコマンドの実行で任意軸補間ドライブを開始します。

・同期スタート機能は、各軸で有効です。

■任意軸補間ドライブの停止機能

●メイン軸で停止指令が発生した場合

・減速停止指令を検出した場合は、メイン軸の基本パルスを減速停止して、ドライブを終了します。

・即時停止指令を検出した場合は、メイン軸の基本パルスがハイレベルのときに、パルス出力を停止して、ドライブを終了します。

・停止指令の発生したメイン軸はパルス停止後にドライブを終了しますが、サブ軸はドライブを終了しません。（但し、基本パルスが停止するのでサブ軸のパルス出力は停止します。）
メイン軸が停止指令により補間ドライブを終了した場合は、サブ軸すべてに停止指令を実行して、ドライブを終了させてください。

●サブ軸で停止指令が発生した場合

・減速停止指令を検出した場合は、出力中の補間パルスがハイレベルのときに、ドライブを終了します。

・即時停止指令を検出した場合は、出力中の補間パルスがハイレベルのときに、ドライブを終了します。

・停止指令の発生したサブ軸はパルス停止後にドライブを終了しますが、他の補間軸はドライブを終了しません。（メイン軸の基本パルスは停止していないので、メイン軸および他のサブ軸のパルス出力も停止しません。）
サブ軸が停止指令により補間ドライブを終了した場合は、他のすべての補間軸に停止指令を実行して、ドライブを終了させてください。

● CPP STOP 機能と CPPIN マスク機能

メイン軸の CPP STOP 機能とサブ軸の CPPIN マスク機能を有効にして、任意軸補間ドライブを実行すると、サブ軸にエラーが発生した場合に、すべての補間軸のパルス出力を停止させることができます。

・エラーが発生したサブ軸は、CPPIN マスク機能で CPPOUT 出力を停止します。

・メイン軸は、CPP STOP 機能でドライブを終了し、CPPOUT 出力を終了します。

・他のサブ軸は、メイン軸の CPPOUT 出力終了でパルス出力を停止します。

(3) 直線補間ドライブ

補間軸は任意の長軸と短軸の座標を構成し、指定軸のパルスを出力して直線補間します。
補間ドライブの最高速度は、5MHz(任意軸補間ドライブは 4MHz)です。指定直線に対する位置誤差は、
± 0.5 LSB です。
座標指定できる相対アドレス範囲は、−2,147,483,648 ～ +2,147,483,647（32 ビット）です。
長軸のパルス出力が INDEX ドライブと同様の加減速ドライブとなります。

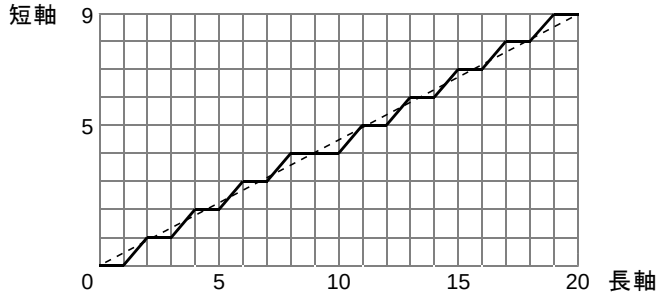
直線補間ドライブの基本パルスは、以下のようになります。
・相関 2 軸直線補間ドライブは、コマンド実行軸から基本パルスを発生します。
　相関 2 軸とも実行軸の基本パルスを補間演算して補間パルスを出力します。
・任意軸補間ドライブのメイン軸直線補間ドライブは、コマンド実行軸から基本パルスを発生します。
・任意軸補間ドライブのサブ軸直線補間ドライブは、CPPIN 端子から入力するパルスを基本パルスとします。

相関 2 軸直線補間ドライブには、以下のドライブパラメータの設定が必要です。
・実行軸の加減速ドライブのパラメータ
・XLONG POSITION、XSHORT POSITION：X 補間軸の長軸と短軸の座標アドレス
・YLONG POSITION、YSHORT POSITION：Y 補間軸の長軸と短軸の座標アドレス

任意軸補間ドライブのメイン軸直線補間ドライブには、以下のドライブパラメータの設定が必要です。
・実行軸の加減速ドライブのパラメータ
・CP SPEC　　：CPPOUT 出力
・LONG POSITION、SHORT POSITION　　：補間軸の長軸と短軸の座標アドレス

任意軸補間ドライブのサブ軸直線補間ドライブには、以下のドライブパラメータの設定が必要です。
・CP SPEC　　：CPPOUT 出力
・LONG POSITION、SHORT POSITION　　：補間軸の長軸と短軸の座標アドレス

■直線補間ドライブの軌跡(長軸 20:短軸 9 の例)



・直線補間ドライブの軌跡は、現在位置と目的地を結ぶ直線に沿います。
・直線補間 SCAN ドライブの場合は、停止指令を検出するまで目的地の指定方向にパルス出力を続けます。
・直線補間 INDEX ドライブの場合は、長軸のパルス数が目的地のパルス数になるとドライブを終了します。

●直線補間の長軸と短軸
補間パルス数が大きい方の軸が長軸、小さい方の軸が短軸になります。

(4) 円弧補間ドライブ

現在の座標と中心点で形成する円弧曲線上を、指定の短軸パルス数に達するまで円弧補間します。
補間ドライブの最高速度は、5MHz(任意軸補間ドライブは 4MHz)です。指定円弧曲線に対する位置誤差は、±1 LSB です。
座標指定できる相対アドレス範囲は、-8,388,608 ～ +8,388,607（24 ビット）です。
短軸パルス数の設定範囲は、-2,147,483,648 ～ +2,147,483,647（32 ビット）です。
短軸パルス出力が INDEX ドライブと同様の加減速ドライブとなります。

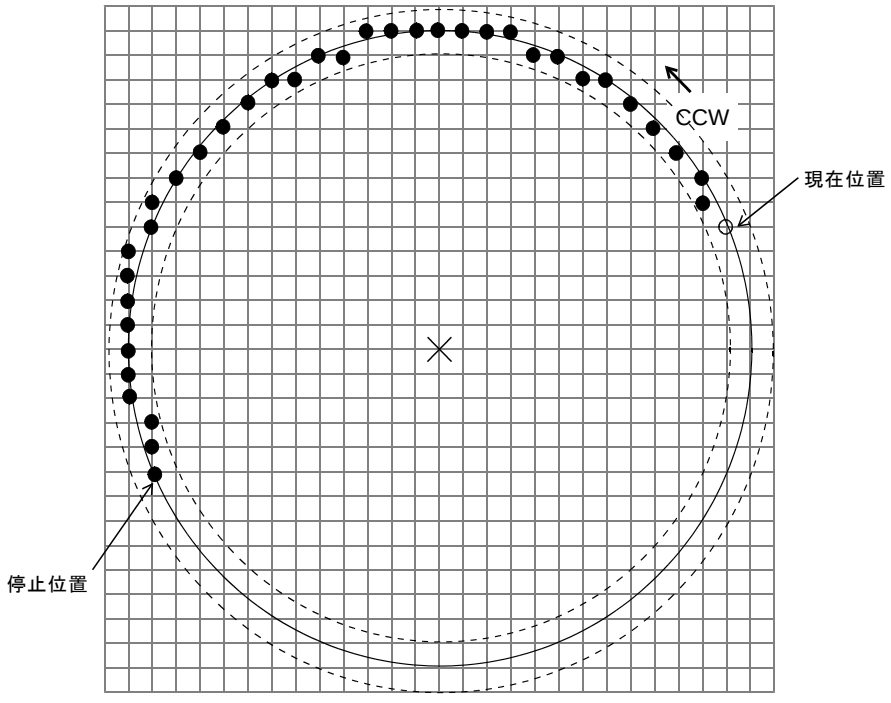
円弧補間ドライブの基本パルスは、以下のようになります。
・相関 2 軸円弧補間ドライブは、コマンド実行軸から基本パルスを発生します。
　相関 2 軸は実行軸の基本パルスを補間演算して XCP, YCP の補間パルスを出力します。
・任意軸補間ドライブのメイン軸円弧補間ドライブは、コマンド実行軸から基本パルスを発生します。
・任意軸補間ドライブのサブ軸円弧補間ドライブは、CPPIN 端子から入力するパルスを基本パルスとします。

相関 2 軸円弧補間ドライブには、以下のドライブパラメータの設定が必要です。
・実行軸の加減速ドライブのパラメータ
・CIRCULAR XPOSITION　：現在位置の X 座標アドレス
・CIRCULAR YPOSITION　：現在位置の Y 座標アドレス
・CIRCULAR PULSE　：目的地の短軸座標までの短軸パルス数

任意軸補間ドライブのメイン軸円弧補間ドライブには、以下のドライブパラメータの設定が必要です。
・実行軸の加減速ドライブのパラメータ
・CP SPEC　：CPPOUT 出力
・CIRCULAR XPOSITION　：現在位置の X 座標アドレス
・CIRCULAR YPOSITION　：現在位置の Y 座標アドレス
・CIRCULAR PULSE　：目的地の短軸座標までの短軸パルス数

任意軸補間ドライブのサブ軸円弧補間ドライブには、以下のドライブパラメータの設定が必要です。
・CP SPEC　：CPPOUT 出力
・CIRCULAR XPOSITION　：現在位置の X 座標アドレス
・CIRCULAR YPOSITION　：現在位置の Y 座標アドレス
・CIRCULAR PULSE　：目的地の短軸座標までの短軸パルス数

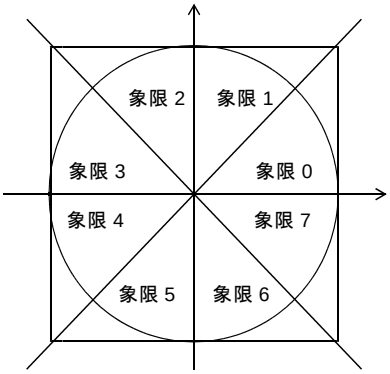
■円弧補間ドライブの軌跡(CCW 回転の例)



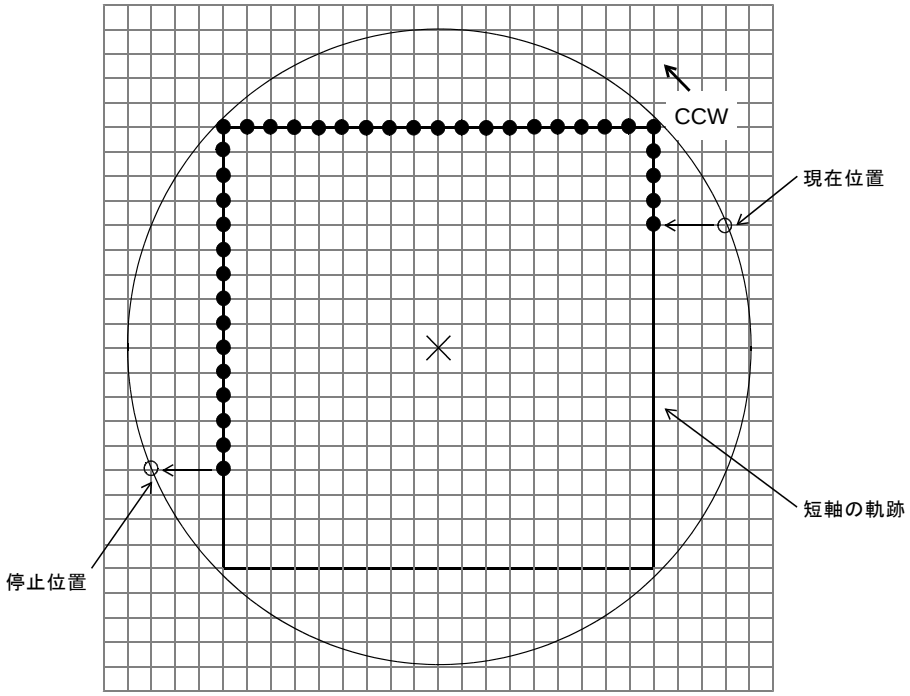
・円弧補間ドライブの軌跡は、現在位置と円弧の中心点の距離を半径とした円周に沿います。
・円弧補間 SCAN ドライブの場合は、停止指令を検出するまで指定の円弧半径と回転方向で
　パルス出力を続けます。
　円弧補間 INDEX ドライブの場合は、短軸パルス数が指定の短軸パルス数になるとドライブを終了します。

●円弧補間の短軸

円弧の中心点を（0, 0）とした円周上の X, Y 座標において座標（X, Y）の絶対値が小さい方の軸が短軸になります。右図の 1, 2, 5, 6 象限は X 軸が短軸、0, 3, 4, 7 象限は Y 軸が短軸です。



■円弧補間ドライブ短軸パルスの軌跡（CCW 回転の例）



円弧補間ドライブは、円弧の中心座標からみた短軸側が補間ドライブの基本パルス(短軸パルス)を常に出力し、長軸側は基本パルス（短軸パルス）を補間演算して補間パルスを出力します。

●短軸パルス数の計算式
半径 R の円における 1 象限当たりの短軸パルス数 Ps は、以下の条件式で算出します。

K ＝ int(R ／√ 2)　：int() は小数点以下を切り捨てた整数

(1) $R^2 \leqq K^2 + (K+1)^2$ のとき

条件式： $|K^2 + (K+1)^2 - R^2| > |K^2 + K^2 - R^2|$ のときは、Ps ＝ K
$|K^2 + (K+1)^2 - R^2| \leqq |K^2 + K^2 - R^2|$ のときは、Ps ＝ K ＋ 1/2

(2) $R^2 > K^2 + (K+1)^2$ のとき

条件式： $|K^2 + (K+1)^2 - R^2| > |(K+1)^2 + (K+1)^2 - R^2|$ のときは、Ps ＝ K ＋ 1
$|K^2 + (K+1)^2 - R^2| \leqq |(K+1)^2 + (K+1)^2 - R^2|$ のときは、Ps ＝ K ＋ 1/2

短軸パルス数 P ＝ int(各象限の短軸パルス数の合計)

1 象限当たりの短軸パルス数の 8 倍（Ps x 8）が、1 回転のパルス数になります。

●短軸パルス数の計算例 1（CCW 回転）
中心点（0, 0）に対して、現在位置を（10, 5）、目的地を（-10, -5）として CCW 回転させる場合の目的地の短軸座標までの短軸パルス数 P は、以下のようになります。

R ＝ $\sqrt{(10^2 + 5^2)}$ ＝ $\sqrt{125}$,　K ＝ int(R ／√ 2) ＝ int(7.9) ＝ 7

R^2 ＝ 125,　$K^2 + (K+1)^2$ ＝ 113,　なので　$R^2 > K^2 + (K+1)^2$

$|K^2 + (K+1)^2 - R^2|$ ＝ 12,　$|(K+1)^2 + (K+1)^2 - R^2|$ ＝ 3, なので

$|K^2 + (K+1)^2 - R^2| > |(K+1)^2 + (K+1)^2 - R^2|$ のときは、Ps ＝ K ＋ 1 ＝ 8

短軸パルス数 P ＝ int(象限 0 のパルス数 ＋ 象限 1, 2, 3 のパルス数 ＋ 象限 4 のパルス数)
＝（8 － 5）＋（8 ＋ 8 ＋ 8）＋ 5 ＝ 32

CCW 回転は負数で指定するので、CIRCULAR PULSE ＝ -32 ＝ H'FFFF_FFE0

CCW 回転時の現在位置の象限の短軸パルス数は、以下のようになります。
・現在位置が象限 0, 2, 4, 6 の短軸パルス数： Ps －(現在位置の短軸座標の絶対値)
・現在位置が象限 1, 3, 5, 7 の短軸パルス数： 現在位置の短軸座標の絶対値

CCW 回転時の目的位置の象限の短軸パルス数は、以下のようになります。
・目的位置が象限 0, 2, 4, 6 の短軸パルス数： 目的位置の短軸座標の絶対値
・目的位置が象限 1, 3, 5, 7 の短軸パルス数： Ps －(目的位置の短軸座標の絶対値)

●短軸パルス数の計算例 2（CW 回転）
中心点（0, 0）に対して、現在位置を（20, 5）、目的地を（-20, -5）として CW 回転させる場合の目的地の短軸座標までの短軸パルス数 P は、以下のようになります。

R ＝ $\sqrt{(20^2 + 5^2)}$ ＝ $\sqrt{425}$,　K ＝ int(R ／√ 2) ＝ int(14.6) ＝ 14

R^2 ＝ 425,　$K^2 + (K+1)^2$ ＝ 421,　なので　$R^2 > K^2 + (K+1)^2$

$|K^2 + (K+1)^2 - R^2|$ ＝ 4,　$|(K+1)^2 + (K+1)^2 - R^2|$ ＝ 25,　なので

$|K^2 + (K+1)^2 - R^2| \leqq |(K+1)^2 + (K+1)^2 - R^2|$ のときは、Ps ＝ K ＋ 1/2 ＝ 14.5

短軸パルス数 P ＝ int(象限 0 のパルス数 ＋ 象限 7, 6, 5 のパルス数 ＋ 象限 4 のパルス数)
＝ 5 ＋（14.5 ＋ 14.5 ＋ 14.5）＋（14.5 － 5）＝ 58

CW 回転は正数で指定するので、CIRCULAR PULSE ＝ 58 ＝ H'0000_003A

CW 回転時の現在位置の象限の短軸パルス数は、以下のようになります。
・現在位置が象限 0, 2, 4, 6 の短軸パルス数： 現在位置の短軸座標の絶対値
・現在位置が象限 1, 3, 5, 7 の短軸パルス数： Ps －(現在位置の短軸座標の絶対値)

CW 回転時の目的位置の象限の短軸パルス数は、以下のようになります。
・目的位置が象限 0, 2, 4, 6 の短軸パルス数： Ps －(目的位置の短軸座標の絶対値)
・目的位置が象限 1, 3, 5, 7 の短軸パルス数： 目的位置の短軸座標の絶対値

(5) 線速一定制御

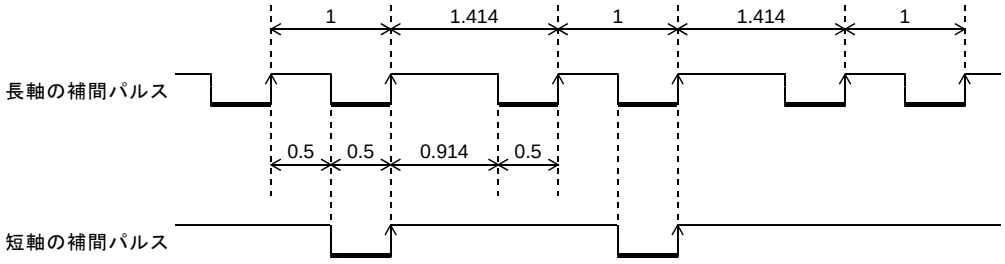
補間ドライブしている 2 軸の合成速度を一定にする制御です。

コマンド実行軸が発生する補間ドライブの基本パルスを線速一定制御します。
2 軸同時にパルス出力したときに、次の基本パルスの出力周期を 1.414 倍にします。
・直線補間ドライブでは、コマンド実行軸の長軸と短軸の２軸間で、線速一定制御します。
・円弧補間ドライブでは、X 座標軸と Y 座標軸の２軸間で、線速一定制御します。

線速一定で加減速ドライブを行うと、減速後の終了速度でのドライブが長くなります。

■線速一定の補間パルス出力（２軸直線補間ドライブの例）

ローレベルの幅はそのままで、ハイレベルの幅が長くなります。



・線速一定制御は各補間ドライブの実行コマンドで設定します。

【注意】
線速一定制御有効の円弧補間ドライブが、2 軸同時にパルス出力した位置で終了した場合に、
以降に実行する線速一定制御有効の直線補間ドライブのパルス出力が、
常に設定値の 1.414 倍の周期（常に線速一定制御される）になります。
・1 軸のみパルス出力する位置（例：0°,90°,180°,270°）で終了した場合は正常です。
・2 軸同時にパルス出力する位置（例：45°,135°,225°,315°）で終了した場合に不具合が発生します。

線速一定制御有効の円弧補間ドライブ終了後は、
以下の円弧補間ドライブ（0 パルス、終了位置 0°）を実行して、正常終了にしてください。
・CIRCULAR XPOSITION SET コマンド（H'28） ：H'00_0000 に設定
・CIRCULAR YPOSITION SET コマンド（H'29） ：H'00_0000 に設定
・CIRCULAR PULSE SET コマンド（H'2A） ： H'0000_0000 に設定
・MAIN CIRCULAR CP コマンド（H'38） ：DATA1=H'0001 で実行

4-1-6. ドライブ CHANGE 機能

ドライブ実行中に、各種ドライブ CHANGE 指令を実行することができます。

（1）UP/DOWN/CONST ドライブ CHANGE 機能

UP/DOWN/CONST のドライブ CHANGE 指令は、STBY = 1 から有効になります。
変更動作点の検出で、UP/DOWN/CONST のドライブ CHANGE 指令を実行します。

停止指令またはエラーを検出すると、ドライブ CHANGE 指令は無効になります。
INDEX ドライブの減速地点を検出すると、ドライブ CHANGE 指令は無効になります。

UP DRIVE 指令を検出すると、最高速度まで加速または減速します。

DOWN DRIVE 指令を検出すると、終了速度まで加速または減速します。

CONST DRIVE 指令を検出すると、加速または減速を終了して、一定速にします。
・直線加減速ドライブの加減速中の場合は、CHANGE 指令の検出で加速または減速を終了します。
・S 字加減速ドライブの加減速中の場合は、S 字カーブで滑らかに加速または減速を終了します。

UP/DOWN/CONST のドライブ CHANGE 指令には、以下のドライブパラメータの設定が必要です。
・UDC SPEC ：UP/DOWN/CONST のドライブ CHANGE 指令を実行する変更動作点

■ UP/DOWN/CONST ドライブ指令が有効となるコマンド

ORIGIN ドライブと MANUAL ドライブでは、加減速の SCAN ドライブ実行時に有効です。
補間ドライブでは、実行軸の加減速パラメータに対して CHANGE 指令を実行します。

COMMAND CODE	汎用コマンド名称		機能
H'12	+SCAN	*P	＋方向 SCAN ドライブの実行
H'13	−SCAN	*P	－方向 SCAN ドライブの実行
H'14	INC INDEX	*P	相対アドレス INDEX ドライブの実行
H'15	ABS INDEX	*P	絶対アドレス INDEX ドライブの実行
H'18	ORIGIN SCAN	*P	ORIGIN SCAN ドライブの実行
H'32	MAIN XY STRAIGHT CP	*P	相関 2 軸直線補間ドライブの実行
H'3A	MAIN XY CIRCULAR CP	*P	相関 2 軸円弧補間ドライブの実行
H'30	MAIN STRAIGHT CP	*P	任意軸補間のメイン軸直線補間ドライブの実行
H'38	MAIN CIRCULAR CP	*P	任意軸補間のメイン軸円弧補間ドライブの実行
−	−	*P	MANUAL ドライブの SCAN ドライブの実行

・上記のドライブを実行すると、STBY = 1 で SPEED CBUSY = 0 になります。
　上記のドライブ以外の実行では、SPEED CBUSY = 1 のままです。

・ドライブ CHANGE 指令の書き込みで、SPEED CBUSY = 1、SPEED CSET = 1 になります。
・ドライブ CHANGE 指令の実行終了で、SPEED CBUSY = 0、SPEED CSET = 0 になります。

・以下の状態を検出すると、SPEED CBUSY = 1、SPEED CSET = 0（クリア）になります。
・INDEX ドライブ、ORIGIN SCAN ドライブ、MANUAL SCAN ドライブの減速地点の検出
・LSEND = 1、SSEND = 1、FSEND = 1、ERROR = 1 の検出
　SPEED CSET をクリアした場合は、実行待ちのドライブ CHANGE 指令は無効になります。

■ UP/DOWN/CONST ドライブ CHANGE 信号

SS0, SS1 信号の操作で、UP/DOWN/CONST のドライブ CHANGE ができます。
SS0, SS1 信号のドライブ CHANGE では、変更動作点の設定は無効になります。

SS0, SS1 信号のドライブ CHANGE 機能は、SPEC INITIALIZE2 コマンドで設定します。
・SS0, SS1 信号で、UP DRIVE, DOWN DRIVE, CONST DRIVE のドライブ CHANGE 指令が実行できます。
・SS0 信号のみを使用すると、UP DRIVE のドライブ CHANGE 指令が実行できます。
・SS1 信号のみを使用すると、DOWN DRIVE のドライブ CHANGE 指令が実行できます。

SS0 または SS1 信号のレベル変化の検出で、ドライブ CHANGE 指令を実行します。

● SS0, SS1 信号によるドライブ CHANGE 動作
ORIGIN ドライブと MANUAL ドライブでは、加減速の SCAN ドライブ実行時に有効です。
補間ドライブでは、メイン軸の加減速パラメータに対して CHANGE 指令を実行します。

・信号のレベル変化の検出は、SPEED CBUSY = 0 となるドライブの STBY = 1 から有効になります。
STBY = 1 で検出した信号のレベルがドライブ CHANGE 指令の場合は、STBY = 1 からドライブ CHANGE 指令を開始します。

・ドライブ CHANGE 指令を検出すると、SPEED CBUSY = 1、SPEED CSET = 1 になります。
SPEED CBUSY = 1 の間は、信号のレベル変化の検出は保留になります。

・ドライブ CHANGE 指令の実行終了で、SPEED CBUSY = 0、SPEED CSET = 0 になります。
同時に、ドライブ CHANGE 信号のレベル変化の検出が有効になります。

・ドライブ CHANGE 指令実行後に、実行前と同じ信号レベル（または機能なし状態）を検出した場合は、SPEED CBUSY = 0、SPEED CSET = 0 のままです。
ドライブ CHANGE 指令実行後に、実行前と異なる信号レベル（異なるドライブ CHANGE 指令）を検出した場合は、SPEED CBUSY = 1、SPEED CSET = 1 になり、次のドライブ CHANGE 指令を実行します。

・以下の状態を検出すると、SPEED CBUSY = 1、SPEED CSET = 0（クリア）になります。
　・INDEX ドライブ、ORIGIN SCAN ドライブ、MANUAL SCAN ドライブの減速地点の検出
　・LSEND = 1、SSEND = 1、FSEND = 1、ERROR = 1 の検出
SPEED CSET をクリアした場合は、ドライブ CHANGE 指令は無効になります。

(2) SPEED CHANGE 機能

SPEED CHANGE 指令は、STBY = 1 から有効になります。
変更動作点の検出で、SPEED CHANGE 指令を実行します。

停止指令またはエラーを検出すると、SPEED CHANGE 指令は無効になります。
INDEX ドライブの減速地点を検出すると、SPEED CHANGE 指令は無効になります。

SPEED CHANGE 指令を検出すると、指定したドライブパルス速度まで加速または減速します。
指定する速度は、最高速度以上および開始速度／終了速度以下にできます。

- STBY = 1 で SPEED CHANGE 指令を検出した場合は、最初の加速の目標速度を SPEED CHANGE 指令の指定速度にします（開始速度＜最高速度、開始速度＜ SPEED CHANGE 指定速度の場合）。
- 直線加減速ドライブの加減速中に SPEED CHANGE 指令を検出した場合は、指令の検出で指定速度まで加速または減速します。
- S 字加減速ドライブの加減速中に SPEED CHANGE 指令を検出した場合は、現在の加減速状態を S 字カーブで滑らかに終了させてから、指定速度まで加速または減速します。

SPEED CHANGE 指令によるドライブパルス速度の変更は、現在のドライブ中のみの変更です。
SPEED CHANGE 指令を実行しても、速度パラメータの設定は変わりません。

SPEED CHANGE 指令には、以下のドライブパラメータの設定が必要です。

- SPEED CHANGE SPEC ：SPEED CHANGE 指令を実行する変更動作点

■ SPEED CHANGE 指令が有効となるコマンド

ORIGIN ドライブと MANUAL ドライブでは、加減速の SCAN ドライブ実行時に有効です。
補間ドライブでは、実行軸の加減速パラメータに対して CHANGE 指令を実行します。

COMMAND CODE	汎用コマンド名称		機能
H'12	+SCAN	*P	＋方向 SCAN ドライブの実行
H'13	-SCAN	*P	－方向 SCAN ドライブの実行
H'14	INC INDEX	*P	相対アドレス INDEX ドライブの実行
H'15	ABS INDEX	*P	絶対アドレス INDEX ドライブの実行
H'18	ORIGIN SCAN	*P	ORIGIN SCAN ドライブの実行
H'32	MAIN XY STRAIGHT CP	*P	相関 2 軸直線補間ドライブの実行
H'3A	MAIN XY CIRCULAR CP	*P	相関 2 軸円弧補間ドライブの実行
H'30	MAIN STRAIGHT CP	*P	任意軸補間のメイン軸直線補間ドライブの実行
H'38	MAIN CIRCULAR CP	*P	任意軸補間のメイン軸円弧補間ドライブの実行
－	－	*P	MANUAL ドライブの SCAN ドライブの実行

- 上記のドライブを実行すると、STBY = 1 で SPEED CBUSY = 0 になります。
  上記のドライブ以外の実行では、SPEED CBUSY = 1 のままです。

- SPEED CHANGE 指令の書き込みで、SPEED CBUSY = 1、SPEED CSET = 1 になります。
- SPEED CHANGE 指令の実行終了で、SPEED CBUSY = 0、SPEED CSET = 0 になります。

- 以下の状態を検出すると、SPEED CBUSY = 1、SPEED CSET = 0（クリア）になります。
  - INDEX ドライブ、ORIGIN SCAN ドライブ、MANUAL SCAN ドライブの減速地点の検出
  - LSEND = 1、SSEND = 1、FSEND = 1、ERROR = 1 の検出

  SPEED CSET をクリアした場合は、実行待ちのドライブ CHANGE 指令は無効になります。

(3) RATE CHANGE 機能

RATE CHANGE 指令は、STBY = 1 から有効になります。
RATE CHANGE 指令は、以下のドライブ CHANGE 指令の検出と同時に実行します。
・UP DRIVE、DOWN DRIVE、CONST DRIVE、SPEED CHANGE

停止指令またはエラーを検出すると、RATE CHANGE 指令は無効になります。
INDEX ドライブの減速地点を検出すると、RATE CHANGE 指令は無効になります。

RATE CHANGE 指令は、ドライブ CHANGE 動作時の変速周期データの変更です。
RATE CHANGE 指令を検出すると、ドライブ CHANGE 動作時の加速カーブと減速カーブの変速周期データを、指定したデータに変更します。
・RATE CHANGE 指令を設定すると、他のドライブ CHANGE 指令の検出と同時に RATE CHANGE 指令を検出し、変速周期データを変更します。
・変更した変速周期データは、次のドライブ CHANGE 動作時も有効です。
現在のドライブが終了すると、変更した変速周期データは無効になります。

・減速停止動作時は、RATE SET コマンドで設定した DCYCLE の変速周期で減速停止します。

RATE CHANGE 指令による変速周期データの変更は、ドライブ CHANGE 動作時のみの変更です。
RATE CHANGE 指令を実行しても、速度パラメータの設定は変わりません。

■ RATE CHANGE 指令が有効となるコマンド

ORIGIN ドライブと MANUAL ドライブでは、加減速の SCAN ドライブ実行時に有効です。
補間ドライブでは、実行軸の加減速パラメータに対して CHANGE 指令を実行します。

COMMAND CODE	汎用コマンド名称		機能
H'12	+SCAN	*P	＋方向 SCAN ドライブの実行
H'13	−SCAN	*P	－方向 SCAN ドライブの実行
H'14	INC INDEX	*P	相対アドレス INDEX ドライブの実行
H'15	ABS INDEX	*P	絶対アドレス INDEX ドライブの実行
H'18	ORIGIN SCAN	*P	ORIGIN SCAN ドライブの実行
H'32	MAIN XY STRAIGHT CP	*P	相関 2 軸直線補間ドライブの実行
H'3A	MAIN XY CIRCULAR CP	*P	相関 2 軸円弧補間ドライブの実行
H'30	MAIN STRAIGHT CP	*P	任意軸補間のメイン軸直線補間ドライブの実行
H'38	MAIN CIRCULAR CP	*P	任意軸補間のメイン軸円弧補間ドライブの実行
−	−	*P	MANUAL ドライブの SCAN ドライブの実行

・上記のドライブを実行すると、STBY = 1 で SPEED CBUSY = 0 になります。
上記のドライブ以外の実行では、SPEED CBUSY = 1 のままです。

・RATE CHANGE 指令の書き込みで、RATE CSET = 1 になります。
RATE CSET = 1 でも、RATE CHANGE 指令の書き込みは可能です（上書きします）。
（他のドライブ CHANGE 指令の書き込みで、SPEED CBUSY = 1、SPEED CSET = 1 になります）

・他のドライブ CHANGE 指令の実行と同時に RATE CHANGE 指令を実行し、RATE CSET = 0 になります。
（他のドライブ CHANGE 指令の実行終了で、SPEED CBUSY = 0、SPEED CSET = 0 になります）

・以下の状態を検出すると、SPEED CBUSY = 1、RATE CSET = 0（クリア）になります。
　・INDEX ドライブ、ORIGIN SCAN ドライブ、MANUAL SCAN ドライブの減速地点の検出
　・LSEND = 1、SSEND = 1、FSEND = 1、ERROR = 1 の検出
RATE CSET をクリアした場合は、実行待ちの RATE CHANGE 指令は無効になります。

(4) INDEX CHANGE 機能

INDEX CHANGE 指令は、STBY = 1 から有効になります。
変更動作点の検出で、INDEX CHANGE 指令を実行します。

停止指令またはエラーを検出すると、INDEX CHANGE 指令は無効になります。
・実行待ちの INDEX CHANGE 指令が無効になった場合は、エラーになります。
ERROR STATUS の CHANGE CLR ERROR = 1 にします。

・反転動作が必要な INDEX CHANGE 指令を検出した場合は、エラーになります。
実行中のドライブを減速停止して、ERROR STATUS の INDEX CHANGE ERROR = 1 にします。

INDEX CHANGE 指令は、INC/ABS/PLS INDEX CHANGE の 3 種類あります。
・INC INDEX CHANGE 指令を検出すると、指定したデータを、起動位置を原点とする相対アドレスの停止位置に設定して、INC INDEX ドライブを行います。
＊ INC INDEX CHANGE コマンドの指定データは、実行中のドライブ方向と同じ符号にしてください。
詳しくは、「INC INDEX CHANGE コマンド」の頁をご覧ください。

・ABS INDEX CHANGE 指令を検出すると、指定したデータを、アドレスカウンタで管理している絶対アドレスの停止位置に設定して、ABS INDEX ドライブを行います。

・PLS INDEX CHANGE 指令を検出すると、指定したデータを、変更動作点の検出位置を原点とする相対アドレスの停止位置に設定して、INC INDEX ドライブを行います。

通常の INDEX ドライブでは、自動減速停止動作開始後に、停止位置を検出した時点で停止します。
INDEX CHANGE 指令を検出した場合は、自動減速停止動作開始後に、終了速度に達してから停止位置を検出して停止します。

INDEX CHANGE 指令には、以下のドライブパラメータの設定が必要です。
・INDEX CHANGE SPEC ：INDEX CHANGE 指令を実行する変更動作点

■ INC/ABS/PLS INDEX CHANGE 指令が有効となるコマンド

COMMAND CODE	汎用コマンド名称		機能
H'12	+SCAN	*P	＋方向 SCAN ドライブの実行
H'13	-SCAN	*P	－方向 SCAN ドライブの実行
H'14	INC INDEX	*P	相対アドレス INDEX ドライブの実行
H'15	ABS INDEX	*P	絶対アドレス INDEX ドライブの実行

・上記のドライブを実行すると、STBY = 1 で INDEX CBUSY = 0 になります。
上記のドライブ以外の実行では、INDEX CBUSY = 1 のままです。

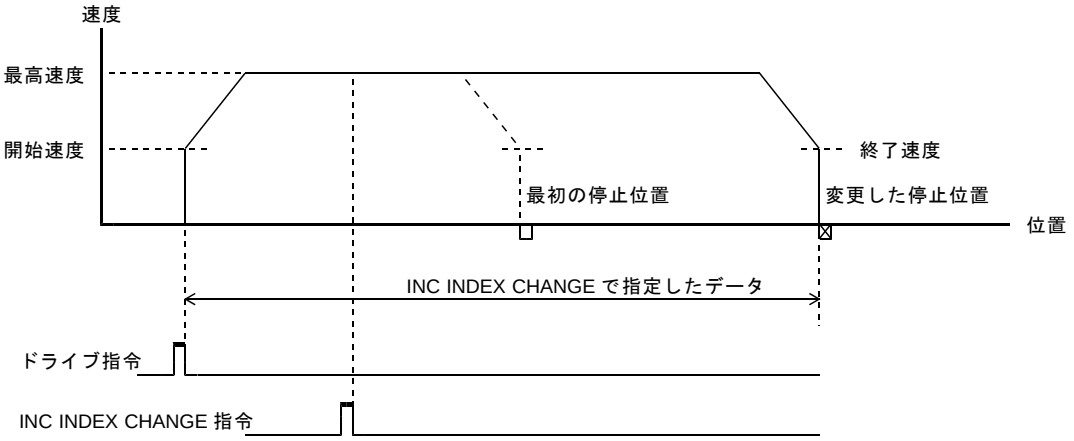
・INDEX CHANGE 指令の書き込みで、INDEX CBUSY = 1、INDEX CSET = 1 になります。
・INDEX CHANGE 指令の実行終了で、INDEX CBUSY = 0、INDEX CSET = 0 になります。
・ドライブが終了すると、DRIVE = 0 で INDEX CBUSY = 1、INDEX CSET = 0 になります。

・以下の状態を検出すると、INDEX CBUSY = 1、INDEX CSET = 0（クリア）になります。
・LSEND = 1、SSEND = 1、FSEND = 1、ERROR = 1 の検出
INDEX CSET をクリアした場合は、実行待ちの INDEX CHANGE 指令は無効になります。

・実行待ちの INDEX CHANGE 指令が無効になった場合は、ERROR = 1 になります。

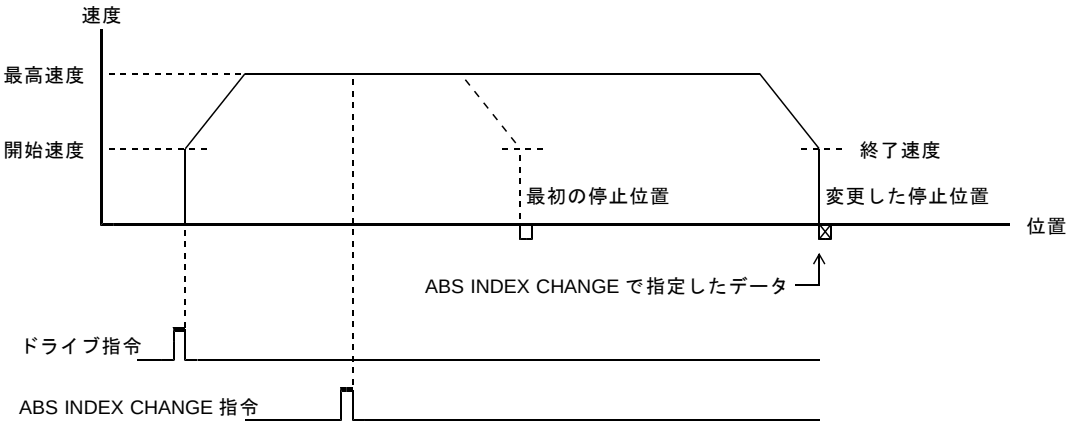
■ INC INDEX CHANGE の動作

指定したデータを、起動位置を原点とする相対アドレスの停止位置にします。



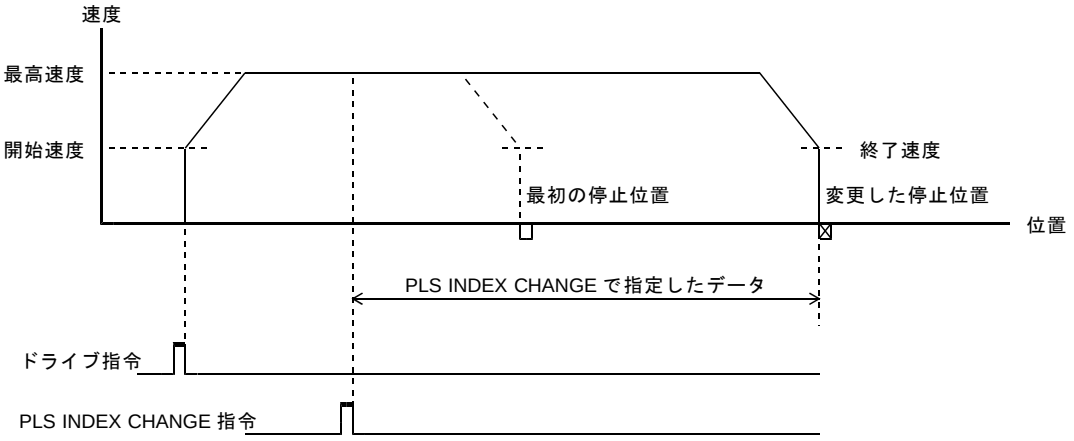
■ ABS INDEX CHANGE の動作

指定したデータを、アドレスカウンタで管理している絶対アドレスの停止位置にします。



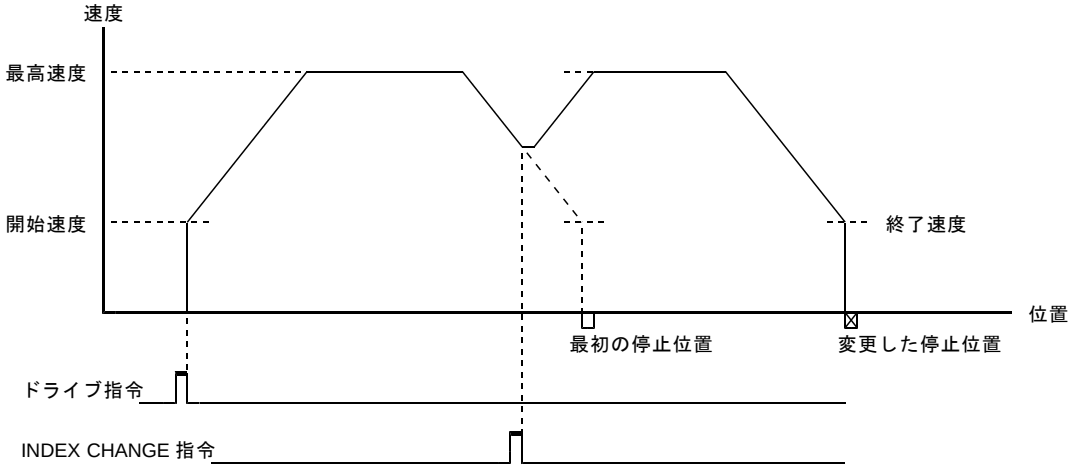
■ PLS INDEX CHANGE の動作

指定したデータを、変更動作点の検出位置を原点とする相対アドレスの停止位置にします。

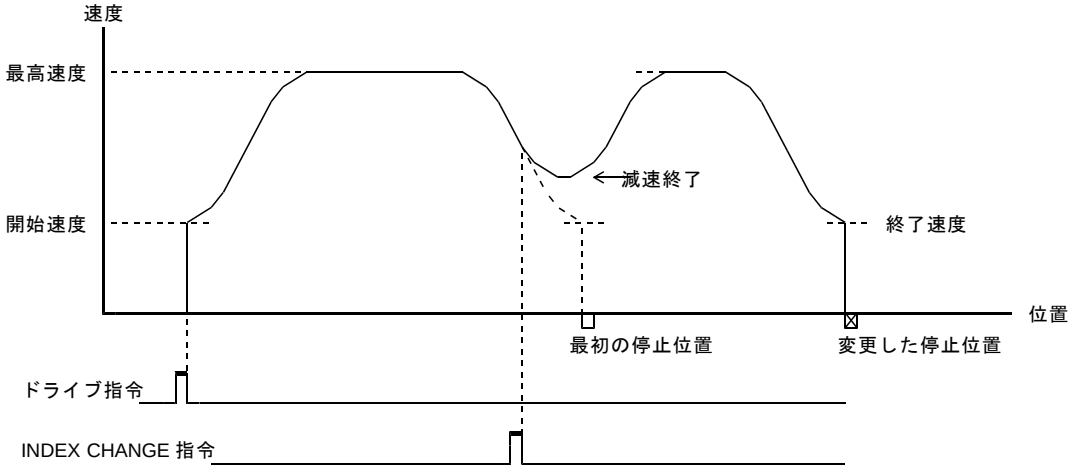


■減速中の INDEX CHANGE 動作

●直線減速中の INDEX CHANGE
直線加減速ドライブでは、停止位置への減速中に、加速が必要な INDEX CHANGE 指令を検出した場合は、減速の途中から再加速して、変更した停止位置までドライブします。



● S 字減速中の INDEX CHANGE
S 字加減速ドライブでは、停止位置への減速中に、加速が必要な INDEX CHANGE 指令を検出した場合は、S 字減速カーブで滑らかに減速を終了させてから、S 字加速カーブで再加速します。



4-1-7. MANUAL ドライブ

■ MANUAL モード

特殊 I/O コネクタの MAN 信号入力を LOW にすると全軸が MANUAL モードになります。
MANUAL モードでは、特殊 I/O コネクタの SEL A ～ SEL D 信号と CWMS, CCWMS 信号入力の操作で
指定軸をドライブさせることができます。

・MAN 信号入力が HIGH(オープン)のときはバスインターフェイスによりドライブを実行するモードです。
これを BUS モードと呼称します。

・MANUAL モードでは全軸 STATUS1 PORT の MAN = 1、BUSY = 1 となります。
MANUAL モードへの切り替えは、全軸 BUSY = 0 のときに行ってください。

・MAN 信号入力による MANUAL モードへの切り替えをアプリケーションから禁止することができます。
MAN 信号入力の禁止は MAN MASK コマンドで行います。
MANUAL モードの切り替え可能な状態では特殊 I/O コネクタの MAN RDY 出力が LOW になります。
切り替え禁止の状態では特殊 I/O コネクタの MAN RDY 出力が HIGH になります。

■ MANUAL ドライブ

MANUAL ドライブには MANUAL SCAN ドライブと MANUAL JOG ドライブがあります。

・MANUAL ドライブの選択(SCAN / JOG)は SPEC INITIALIZE1 コマンドで行います。

● MANUAL ドライブの軸指定

MANUAL モード時に SEL A, SEL B, SEL C, SEL D 信号の操作で、MANUAL ドライブする軸を指定します。

SEL D	SEL C	SEL B	SEL A	C-VX870(E)v1	C-VX871(E)v1	C-VX872(E)v1	C-VX873(E)v1
LOW	**LOW**	**LOW**	**LOW**	**X 軸**	**X 軸**	**X1 軸**	**X1 軸**
LOW	**LOW**	**LOW**	HIGH	**Y 軸**	**Y 軸**	**Y1 軸**	**Y1 軸**
LOW	**LOW**	HIGH	**LOW**	**Z 軸**	**Z 軸**	**Z1 軸**	**Z1 軸**
LOW	**LOW**	HIGH	HIGH	**A 軸**	**A 軸**	**A1 軸**	**A1 軸**
LOW	HIGH	**LOW**	**LOW**	無効	**B 軸**	無効	**B1 軸**
LOW	HIGH	**LOW**	HIGH	無効	**C 軸**	無効	**C1 軸**
LOW	HIGH	HIGH	**LOW**	無効	無効	無効	無効
LOW	HIGH	HIGH	HIGH	無効	無効	無効	無効
HIGH	**LOW**	**LOW**	**LOW**	無効	無効	**X2 軸**	**X2 軸**
HIGH	**LOW**	**LOW**	HIGH	無効	無効	**Y2 軸**	**Y2 軸**
HIGH	**LOW**	HIGH	**LOW**	無効	無効	**Z2 軸**	**Z2 軸**
HIGH	**LOW**	HIGH	HIGH	無効	無効	**A2 軸**	**A2 軸**
HIGH	HIGH	**LOW**	**LOW**	無効	無効	無効	**B2 軸**
HIGH	HIGH	**LOW**	HIGH	無効	無効	無効	**C2 軸**
HIGH	HIGH	HIGH	**LOW**	無効	無効	無効	無効
HIGH	HIGH	HIGH	HIGH	無効	無効	無効	無効

● MANUAL ドライブの実行

MANUAL モード時に CWMS 信号または CCWMS 信号を LOW レベルにすると、SEL A ～ SEL D 信号で
指定した軸がドライブを起動します。
CWMS 信号で＋方向に、CCWMS 信号で－方向にドライブします。

・MANUAL ドライブのドライブパラメータは、各軸現在の設定値です。
・MANUAL SCAN ドライブは CWMS 信号,または CCWMS 信号を HIGH にすると減速停止します。

● MANUAL SCAN ドライブの速度変更

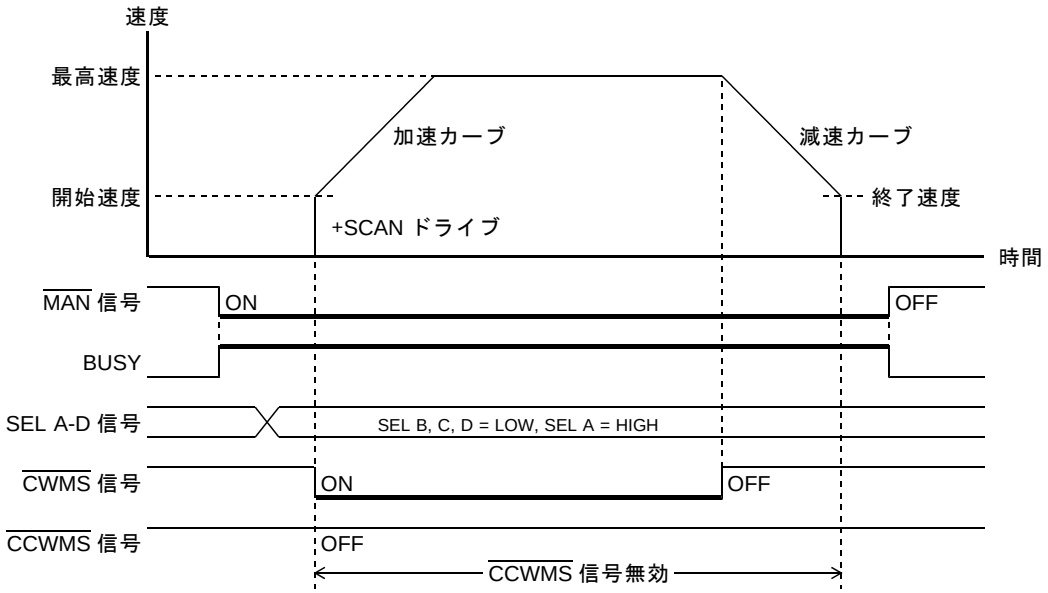
MANUAL モード時、特殊 I/O コネクタの SS0,SS1 信号は SEL A ～ SEL D 信号で指定した軸の
多用途センサ信号(SS0,SS1)に割り付けられます。
多用途センサ信号の入力機能を「UP/DOWN/CONST ドライブ CHANGE 指令入力」に設定した後、
MANUAL SCAN ドライブを起動すると、SS0, SS1 信号の操作によって速度を変更することができます。

・汎用センサ信号の入力機能は SPEC INITIALIZE2 コマンドで設定します。
・UP/DOWN/CONST ドライブ CHANGE 指令

SS1	SS0	ドライブ CHANGE 動作
HIGH	HIGH	CONST DRIVE
HIGH	**LOW**	UP DRIVE
LOW	HIGH	DOWN DRIVE
LOW	**LOW**	CONST DRIVE

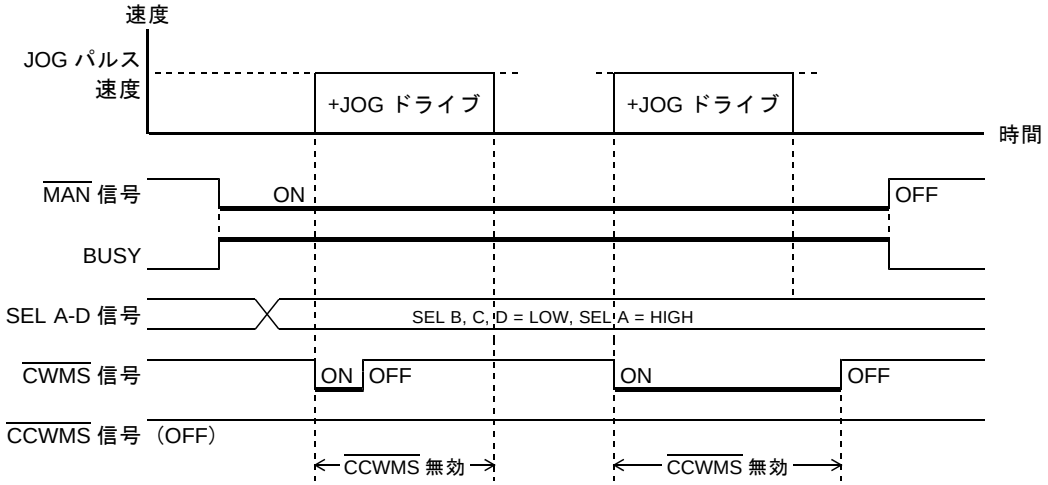
・UP/DOWN/CONST ドライブ CHANGE 機能をご覧ください。

■ MANUAL SCAN ドライブ動作例(Y 軸、＋方向)



① 全軸 BUSY = 0 のときに、MAN 信号を LOW レベルにします。 (MANUAL モードになります。)
② SEL A-D 信号で Y 軸を指定します。
③ CWMS 信号を LOW レベルにします。 (+方向の SCAN ドライブを開始します。)
④ CWMS 信号を HIGH レベルにします。 (+方向の SCAN ドライブを減速停止します。)
⑤ MAN 信号を HIGH レベルにします。 (BUS モードに戻ります。)

■ MANUAL JOG ドライブ動作例(Y 軸、＋方向)



① 全軸 BUSY = 0 のときに、MAN 信号を LOW レベルにします。 (MANUAL モードになります。)
② SEL A-D 信号で Y 軸を指定します。
③ CWMS 信号を LOW レベルにします。 (+方向の SCAN ドライブを開始します。)
(JOG パルス数分のパルスを出力するとドライブを終了します。)
④ JOG ドライブ開始後に CWMS 信号を HIGH レベルにします。
⑤ MAN 信号を HIGH レベルにします。 (BUS モードに戻ります。)

4-1-8. 読み出し機能

(1) カウントデータのラッチデータ読み出し

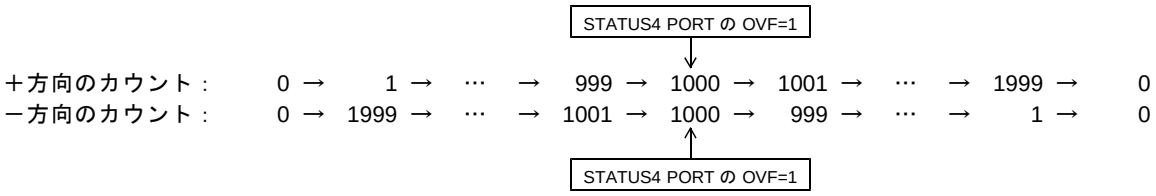
MCC07E に各カウンタ LATCH DATA READ コマンドを実行するとラッチしたカウントデータを読み出すことができます。

4-2. カウンタ仕様

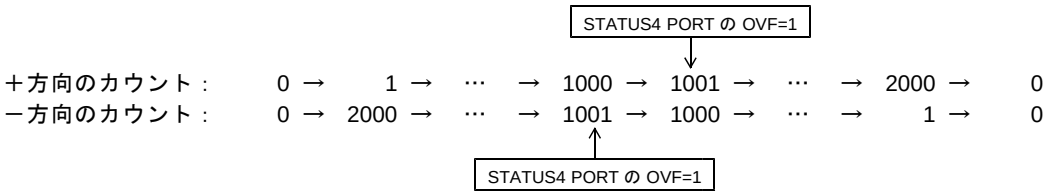
4-2-1. リングカウンタ機能

アドレスカウンタ、およびパルスカウンタは、カウント最大値を設定すると、設定値をカウンタの最大値としてリングカウントします。
回転系のアドレス管理に便利です。

●最大カウント数＝ 1,999 の場合（2,000 カウントで 1 回転）



●最大カウント数＝ 2,000 の場合（2,001 カウントで 1 回転）



・パルス偏差カウンタにはリングカウンタ機能はありません。

・カウント数が設定値の 1/2 に達すると、STATUS4 PORT の各カウンタの OVF=1 になります。

・カウント最大値は各カウンタの COUNTER MAX COUNT SET コマンドで設定します。

・アドレスカウンタにカウント最大値を設定して、ABS INDEX ドライブでカウント最大値以上のアドレスを指定した場合、アドレスカウンタが OVF した時点でドライブは停止し、エラーとなります。

4-2-2. カウントデータのラッチ・クリア機能

■カウンタのラッチ機能

設定したラッチタイミングのアクティブエッジで、カウンタのカウントデータをラッチします。
ラッチしたデータは次のラッチタイミングのアクティブエッジを検出するまで保存します。
ラッチしたデータは各カウンタ LATCH DATA READ コマンドで読み出すことができます。

●設定できるラッチタイミング

ラッチタイミング〈エッジ検出〉
・各カウンタ LATCH DATA READ コマンドの実行でラッチする ・多用途センサ信号：SS0 = 0 → 1 でラッチする ・ORIGIN SPEC SET コマンドの ORG 検出信号の検出エッジでラッチする

・各カウンタのラッチタイミングは COUNT LATCH SPEC SET コマンドで設定します。

■カウンタのクリア機能

カウントデータのラッチと同時にカウントデータを "0" にクリアします。
カウンタのカウントタイミングとクリアタイミングが同時に発生した場合はクリアを優先します。

・各カウンタのラッチクリア機能は COUNT LATCH SPEC SET コマンドで設定します。

4-3. HARD CONFIG 仕様

4-3-1. 入出力仕様

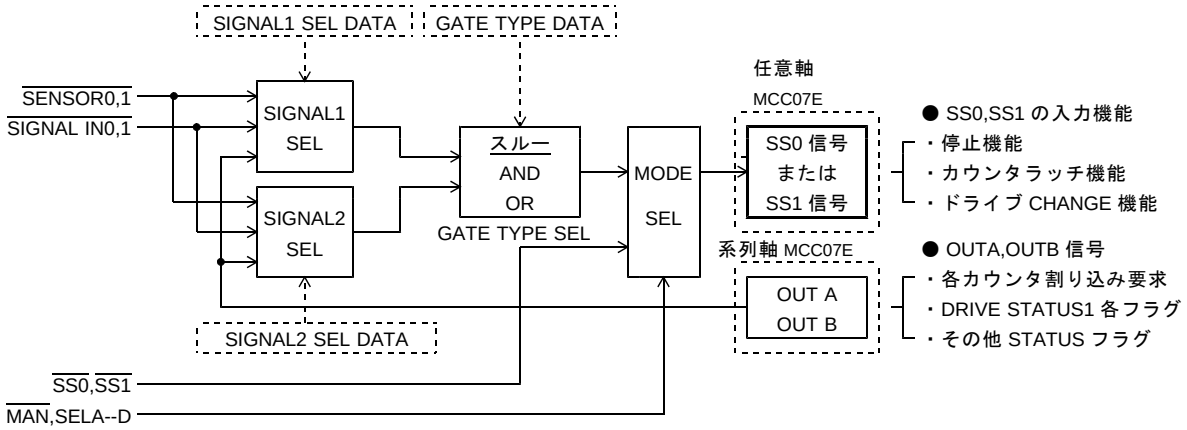
(1) 多用途センサ機能

各軸には停止機能、カウンタのラッチ機能、ドライブ CHANGE 機能のトリガとして使用可能な
多用途センサ信号 SS0, SS1 があります。
これら SS0,SS1 信号に、下記のうち任意な信号を接続することができます。

多用途センサ信号	接続できる信号	
SS0,SS1	ユーザ I/O	$\overline{\text{SENSOR0}}$ 信号 $\overline{\text{SENSOR1}}$ 信号
	特殊 I/O	$\overline{\text{SIGNAL IN0}}$ 信号 $\overline{\text{SIGNAL IN1}}$ 信号
	MCC07E ステータス信号	任意軸 OUT A 信号 任意軸 OUT B 信号

・C-VX872(E)v1, C-VX873(E)v1 の場合、SS0,SS1 信号に接続できる信号は同じ系列軸内の信号になります。

■多用途センサ信号の接続構成



・接続の設定は、HARD CONFIGURATION に SENSOR SIGNAL SELECT コマンドにより設定します。
・SIGNAL2 SEL と GATE TYPE SEL を使用すると任意 2 信号の合成論理信号を多用途センサに接続できます。
・MANUAL モード時はユーザ接続設定が無効となり、MANUAL ドライブ指定軸の多用途センサ SS0,SS1 に
特殊 I/O の $\overline{\text{SS0}}$,$\overline{\text{SS1}}$ 信号が接続されます。
BUS モードに復帰するとユーザ接続設定が有効になります。

■多用途センサ信号の入力機能

多用途センサ信号 SS0, SS1 の入力機能をそれぞれ以下の内から選択することができます。
・減速停止信号として使用する。
・即時停止信号として使用する。
・カウンタラッチ機能(SS0 のみ)、ドライブ CHANGE 機能のトリガ信号として使用する。
・ドライブ CHANGE の UP/DOWN/CONS 指令入力として使用する。

・SS0,SS1 機能の設定は、MCC07E の SPEC INITIALIZE2 コマンドにより設定します。

■ OUT A,B 信号

MCC07E の OUT A,B 信号は以下のステータスを出力することができます。

・カウンタ割り込み要求の ARDINT	・STATUS1 の UP フラグ
・カウンタ割り込み要求の CNTINT	・STATUS1 の DOWN フラグ
・カウンタ割り込み要求の DFLINT	・STATUS1 の CONST フラグ
・コマンド終了割り込みの RDYINT	・STATUS1 の EXT PULSE フラグ
・STATUS1 の STBY フラグ	・STATUS2 の PULSE MASK フラグの反転
・STATUS1 の DRIVE フラグの反転	・STATUS2 の ORG SIGNAL フラグ
・STATUS5 の SPEED CBUSY の反転	・汎用出力としての出力状態
・STATUS5 の INDEX CBUSY の反転	・STATUS4 の PULSE OVF フラグ

・OUTA,OUTB に出力する信号の設定は、MCC07E の HARD INITIALIZE1 コマンドにより設定します。

(2) ステータス外部出力機能

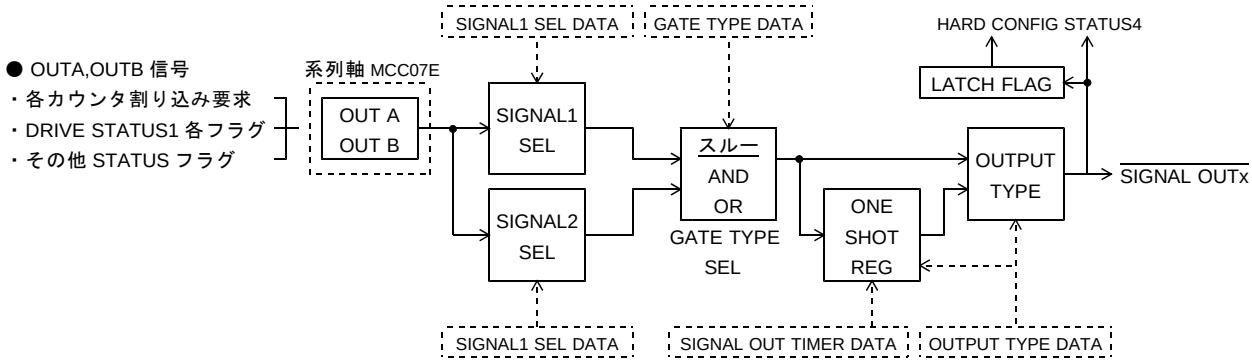
任意な軸のステータス信号(OUT A,B 信号)を、特殊 I/O コネクタ(J3)の $\overline{\text{SIGNAL OUTx}}$ 信号に接続し、外部にステータス信号を出力することができます。

外部出力信号	接続できる信号	
$\overline{\text{SIGNAL OUT0,1}}$	MCC07E ステータス信号	任意軸 OUT A 信号 任意軸 OUT B 信号

・C-VX872(E)v1, C-VX873(E)v1 の場合、$\overline{\text{SIGNAL OUTn0,n1}}$ 信号に接続できる信号は同じ系列軸内の信号になります。

$\overline{\text{SIGNAL OUTx}}$ 信号の状態は、HARD CONFIG STATUS4 PORT から読み出すことができます。
また、$\overline{\text{SIGNAL OUTx}}$ 信号のアクティブ出力検出を HARD CONFIG STATUS4 PORT のラッチフラグで確認することができます。
ラッチフラグのクリアは、SIGNAL OUT LATCH CLR コマンドで行います。

■ステータス外部出力信号の接続構成



・接続の設定は、HARD CONFIGURATION に SIGNAL OUT SELECT コマンドにより設定します。
・SIGNAL2 SEL と GATE TYPE SEL を使用すると任意 2 信号の合成論理信号を外部出力することができます。
・OUTPUT TYPE を設定すると任意信号の任意エッジ検出でワンショット出力することができます。
ワンショット出力時間は 1us ～ 65.535ms の範囲で 1us 単位で設定します。
・ワンショット出力時間の設定は、HARD CONFIGURATION に SIGNAL OUT TIMER SET コマンドで設定します。

■ OUT A,B 信号

MCC07E の OUT A,B 信号は以下のステータスを出力することができます。

・カウンタ割り込み要求の ARDINT
・カウンタ割り込み要求の CNTINT
・カウンタ割り込み要求の DFLINT
・コマンド終了割り込みの RDYINT
・STATUS1 の STBY フラグ
・STATUS1 の DRIVE フラグの反転
・STATUS5 の SPEED CBUSY の反転
・STATUS5 の INDEX CBUSY の反転
・STATUS1 の UP フラグ
・STATUS1 の DOWN フラグ
・STATUS1 の CONST フラグ
・STATUS1 の EXT PULSE フラグ
・STATUS2 の PULSE MASK フラグの反転
・STATUS2 の ORG SIGNAL フラグ
・汎用出力としての出力状態
・STATUS4 の PULSE OVF フラグ

・OUTA,OUTB に出力する信号の設定は、MCC07E の HARD INITIALIZE1 コマンドにより設定します。

4-3-2. 同期スタート機能

任意複数軸のドライブ開始を同時にスタートさせることができます。
各軸は STBY 解除条件(PAUSE=0)を検出するまでドライブパルス出力の開始を保留します。
各軸の PAUSE SET および PAUSE CLR 条件を設定することで、同時パルス出力開始や他軸のドライブに連動したパルス出力開始を行うことができます。

■ STBY フラグ

各軸はドライブパルス出力の準備(データ処理)が完了すると、STBY 状態(STBY=1)となります。
STBY 状態で STBY 解除条件(PAUSE=0)を検出すると、STBY 状態を解除(STBY=0)し、ドライブパルス出力を開始します。

・各軸 MCC07E の OUT A 信号に STBY フラグを出力すると、HARD CONFIG STATUS3 PORT で一括して各軸の STBY 状態を確認することができます。

■ PAUSE 信号

各軸は PAUSE SET 条件を検出すると PAUSE 信号=1 となり、PAUSE CLR 条件を検出すると PAUSE 信号=0 となります。
PAUSE 信号=1 のときは、STBY 状態(STBY=1)を保持して、ドライブパルス出力の開始を保留します。
PAUSE 信号=0 で、STBY 状態を解除(STBY=0)し、ドライブパルス出力を開始します。

各軸の PAUSE SET 条件および PAUSE CLR 条件は以下からそれぞれ選択することができます。

PAUSE SET 条件
・PAUSE コマンドの実行で SET する。
・選択信号のアクティブエッジ検出で SET する。
・選択信号のノットアクティブエッジ検出で SET する。

PAUSE CLR 条件
・PAUSE コマンドの実行で CLR する。
・選択信号のアクティブエッジ検出で CLR する。
・選択信号のノットアクティブエッジ検出で CLR する。

・各軸の PAUSE SET 条件は、HARD CONFIGURATION に PAUSE SET SPEC コマンドで設定します。
・各軸の PAUSE CLR 条件は、HARD CONFIGURATION に PAUSE CLR SPEC コマンドで設定します。

● PAUSE コマンドの実行で SET または CLR する
HARD CONFIG DATA2 PORT の対象軸のビットに PAUSE データを設定し、PAUSE コマンドで実行します。
PAUSE データは"1"で PAUSE 信号=1(SET)、"0"で PAUSE 信号=0(CLR)となります。

・PAUSE データ="1"での PAUSE コマンド実行は、対象軸の PAUSE SET 条件が「PAUSE コマンドの実行で SET する」に設定されている場合にのみ有効です。
・PAUSE データ="0"での PAUSE コマンド実行は、対象軸の PAUSE CLR 条件が「PAUSE コマンドの実行で CLR する」に設定されている場合にのみ有効です。

●選択信号のアクティブエッジ／ノットアクティブエッジ検出で SET または CLR する
以下の信号のうち、任意な信号のエッジ検出で PAUSE 信号を 1(SET)または 0(CLR)にします。

エッジ検出に使用できる信号	
ユーザ I/O	$\overline{\text{SENSOR0}}$ 信号 $\overline{\text{SENSOR1}}$ 信号
特殊 I/O	$\overline{\text{SIGNAL IN0}}$ 信号 $\overline{\text{SIGNAL IN1}}$ 信号
MCC07E ステータス信号	任意軸 OUT A 信号 任意軸 OUT B 信号

・C-VX872(E)v1, C-VX873(E)v1 の場合、エッジ検出に使用できる信号は同じ系列軸内の信号になります。
・PAUSE SET 条件におけるエッジ検出信号の選択は PAUSE SET SPEC コマンドで設定します。
・PAUSE CLR 条件におけるエッジ検出信号の選択は PAUSE CLR SPEC コマンドで設定します。
・エッジ検出信号の選択は任意 2 信号の合成論理信号とすることもできます。

■ OUT A,B 信号

MCC07E の OUT A,B 信号は以下のステータスを出力することができます。

- ・カウンタ割り込み要求の ARDINT
- ・カウンタ割り込み要求の CNTINT
- ・カウンタ割り込み要求の DFLINT
- ・コマンド終了割り込みの RDYINT
- ・STATUS1 の STBY フラグ
- ・STATUS1 の DRIVE フラグの反転
- ・STATUS5 の SPEED CBUSY の反転
- ・STATUS5 の INDEX CBUSY の反転
- ・STATUS1 の UP フラグ
- ・STATUS1 の DOWN フラグ
- ・STATUS1 の CONST フラグ
- ・STATUS1 の EXT PULSE フラグ
- ・STATUS2 の PULSE MASK フラグの反転
- ・STATUS2 の ORG SIGNAL フラグ
- ・汎用出力としての出力状態
- ・STATUS4 の PULSE OVF フラグ

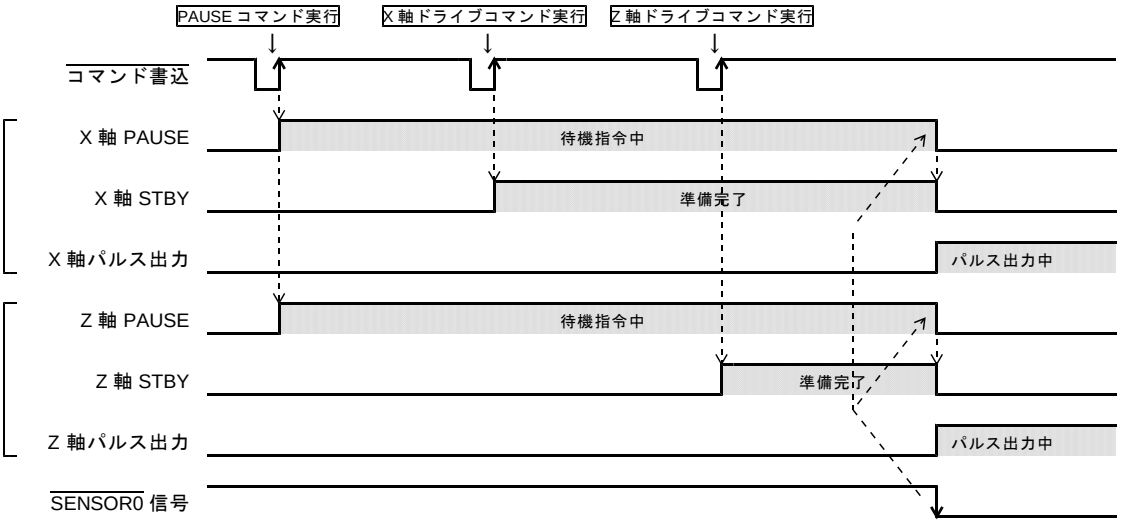
・OUTA,OUTB に出力する信号の設定は、MCC07E の HARD INITIALIZE1 コマンドにより設定します。

■同期スタート動作例（X 軸,Z 軸が $\overline{\text{SENSOR0}}$ 信号で同時スタート）

●設定

- ・X 軸 STBY 解除条件　：PAUSE=0
- ・X 軸 PAUSE SET 条件　：PAUSE コマンド実行
- ・X 軸 PAUSE CLR 条件　：$\overline{\text{SENSOR0}}$ 入力信号のアクティブエッジ
- ・Z 軸 STBY 解除条件　：PAUSE=0
- ・Z 軸 PAUSE SET 条件　：PAUSE コマンド実行
- ・Z 軸 PAUSE CLR 条件　：$\overline{\text{SENSOR0}}$ 入力信号のアクティブエッジ

●動作



4-3-3. 読み出し機能

(1) ステータス読み出し

各 HARD CONFIG STATUS PORT 読み出し関数にて、HARD CONFIGURATION の STATUS PORT を読み出すことで、MAN RDY 信号の状態、SIGNAL OUTx, SIGNAL INx 信号の状態、各軸 PAUSE 信号の状態などが読み出せます。

(2) 設定データ読み出し

HARD CONFIGURATION に HARD CONFIG SET DATA READ コマンドを実行すると、設定したデータが読み出せます。

５．付録

5-1. 初期仕様一覧

(1) 応用設定

項　目	初期仕様	対応関数/コマンド
■入力信号のデジタルフィルタ機能		
CWLM, CCWLM 信号	0 ～ 50ns	HARD INITIALIZE4 コマンド
$\overline{\text{DEND}}$/$\overline{\text{PO}}$, DALM(I$\overline{\text{Nnx}}$) 信号	0 ～ 50ns	
$\overline{\text{ORG}}$, $\overline{\text{NORG}}$ 信号	0 ～ 50ns	HARD INITIALIZE5 コマンド
± ZORG 信号	0 ～ 50ns	
± EA, ± EB 信号	0 ～ 50ns	HARD INITIALIZE6 コマンド
■入力信号のアクティブ論理		
CWLM 信号	正論理	HARD INITIALIZE7 コマンド
CCWLM 信号	正論理	
DALM(I$\overline{\text{Nnx}}$) 信号	負論理	
$\overline{\text{ORG}}$ 信号	負論理	
$\overline{\text{NORG}}$ 信号	負論理	
■アドレスカウンタ		
最大カウント数	H'FFFF_FFFF	ADDRESS COUNTER MAX COUNT SET コマンド
■パルスカウンタ		
最大カウント数	H'FFFF_FFFF	PULSE COUNTER MAX COUNT SET コマンド
■カウンタのラッチ・クリア機能		
アドレスカウンタラッチタイミング	ADDRESS LATCH DATA READ コマンドの実行	COUNT LATCH SPEC SET コマンド
ADDRESS CLR ENABLE	クリアしない	
パルスカウンタラッチタイミング	PULSE LATCH DATA READ コマンドの実行	
PULSE CLR ENALBE	クリアしない	
パルス偏差カウンタラッチタイミング	DFL LATCH DATA READ コマンドの実行	
DFL CLR ENABLE	クリアしない	
■多用途センサ機能		
各軸 SS0 入力機能	汎用入力、各種機能のトリガ入力	SPEC INITIALIZE2 コマンド
各軸 SS1 入力機能	汎用入力、各種機能のトリガ入力	
Xn 軸 SS0 接続信号	なし(LOW 固定)	SENSOR SELECT コマンド (HARD CONFIG コマンド)
Xn 軸 SS1 接続信号	なし(LOW 固定)	
Yn 軸 SS0 接続信号	なし(LOW 固定)	
Yn 軸 SS1 接続信号	なし(LOW 固定)	
Zn 軸 SS0 接続信号	$\overline{\text{SENSORn0}}$ 信号	
Zn 軸 SS1 接続信号	なし(LOW 固定)	
An 軸 SS0 接続信号	$\overline{\text{SENSORn1}}$ 信号	
An 軸 SS1 接続信号	なし(LOW 固定)	
Bn 軸 SS0 接続信号　*6 軸,12 軸のみ	なし(LOW 固定)	
Bn 軸 SS1 接続信号　*6 軸,12 軸のみ	なし(LOW 固定)	
Cn 軸 SS0 接続信号　*6 軸,12 軸のみ	なし(LOW 固定)	
Cn 軸 SS1 接続信号　*6 軸,12 軸のみ	なし(LOW 固定)	
■ステータス外部出力機能		
各軸 OUT A 出力ステータス	CNTINT	HARD INITIALIZE1 コマンド
各軸 OUT B 出力ステータス	DFLINT	
$\overline{\text{SIGNAL OUTn0}}$ 出力信号	Xn 軸 OUT A	SIGNAL OUT SELECT コマンド (HARD CONFIG コマンド)
SIGNAL OUTn1 出力信号	Yn 軸 OUT A	
■同期スタート機能		
各軸 STBY 解除条件	PAUSE=0 で解除	SPEC INITIALIZE3 コマンド
各軸 PAUSE SET 条件	PAUSE コマンドの実行	PAUSE SET SPEC コマンド (HARD CONFIG コマンド)
各軸 PAUSE CLR 条件	PAUSE コマンドの実行	PAUSE CLR SPEC コマンド (HARD CONFIG コマンド)

(2) 応用ドライブパラメータ

項　目	初期仕様	対応関数/コマンド
■第 1 パルス出力周期　(FSPD)	5,000Hz	FSPD SET コマンド
■加減速パラメータ		
速度倍率　(RESOL)	1 (No.3)	HIGH SPEED SET コマンド
最高速時の速度データ　(HSPD)	3,000	
加速開始時の速度データ　(LSPD)	300	LOW SPEED SET コマンド
減速終了時の速度データ　(ELSPD)	300	
加速カーブ変速周期データ　(UCYCLE)	200	RATE SET コマンド
減速カーブ変速周期データ　(DCYCLE)	200	
加速カーブ S 字変速領域データ　(SUAREA)	0(変速領域なし)	SCAREA SET コマンド
減速カーブ S 字変速領域データ　(SDAREA)	0(変速領域なし)	
減速パルス数のオフセットパルス数	1 パルス	DOWN PULSE ADJUST コマンド
■ ORIGIN ドライブパラメータ		
ORG 検出信号	$\overline{\text{ORG}}$ 信号と± ZORG 信号の OR(論理和)	ORIGIN SPEC SET コマンド
ORG 検出信号の検出エッジ	ORG 検出信号の 0 → 1(アクティブ)エッジ	
ORG ドライブ起動方向	－(CCW)方向	
検出エッジのカウント数	1 カウント目のエッジ検出で停止	
停止時 DRST 出力	出力しない	
■補間ドライブパラメータ		
CPPOUT 端子から出力するパルス	CPPIN 端子から入力するパルス	CP SPEC SET コマンド
直線補間 長軸の目的地の座標アドレス	H'0000_0000	LONG POSITION SET コマンド
直線補間 短軸の目的地の座標アドレス	H'0000_0000	SHORT POSITION SET コマンド
円弧補間 現在位置の X 座標アドレス	H'00_0000	CIRCULAR XPOSITION SET コマンド
円弧補間 現在位置の Y 座標アドレス	H'00_0000	CIRCULAR YPOSITION SET コマンド
円弧補間 目的地までの短軸パルス数	H'0000_0000	CIRCULAR PULSE SET コマンド
■ドライブ CHANGE パラメータ		
UP DRIVE 変更動作点	UP DRIVE CHANGE コマンドの実行	UDC SPEC SET コマンド
DOWN DRIVE 変更動作点	DOWN DRIVE CHANGE コマンドの実行	
CONST DRIVE 変更動作点	CONST DRIVE CHANGE コマンド実行	
SPEED CHANGE 変更動作点	SPEED CHANGE コマンドの実行	SPEED CHANGE SPEC SET コマンド
INDEX CHANGE 変更動作点	INDEX CHANGE コマンドの実行	INDEX CHANGE SPEC SET コマンド

5-2. 関数一覧

種別	名称 説明	記号	C-VX870(E)v1 C-VX872(E)v1	C-VX871(E)v1 C-VX873(E)v1	ページ
デバイス関数	デバイス割り込みオープン関数 指定されたデバイスの割り込みをオープンします。	MCO7_OpenDevInt	○	○	11
	デバイス割り込みクローズ関数 指定されたデバイスの割り込みをクローズします。	MCO7_CloseDevInt	○	○	12
	割り込みコールバック設定関数 指定されたデバイスで指定された割り込みが発生した場合の、コールバック関数を設定します。	MCO7_SetIntCallBack	○	○	13
	割り込みメッセージ設定関数 指定されたデバイスで指定された割り込みが発生した場合にポストするメッセージを設定します。	MCO7_SetIntMessage	○	○	14
	割り込みイベント設定関数 指定されたデバイスで指定された割り込みが発生した場合、シグナル状態になるイベントを設定します。	MCO7_SetIntEvent	○	○	15
	割り込みステータス読み出し関数 割り込みステータスを読み出します。	MCO7_ReadIntStatus	○	○	16
	割り込み設定クリア関数 各割り込み設定関数の設定をクリアします。	MCO7_ClearIntSet	○	○	17
	HARD CONFIG COMMAND一括書き込み関数 指定ボードのHARD CONFIG DATA1、2、3 PORTにデータ構造体のデータを書込後、COMMAND PORTにｺﾏﾝﾄﾞを書込みます。	MCO7_IWHardConfig	○	○	19
	HARD CONFIG COMMAND PORT書き込み関数 指定ボードのHARD CONFIG COMMAND PORTにコマンドを書込みます。	MCO7_BWHardConfigCommand	○	○	20
	HARD CONFIG DATA1 PORT書き込み関数 指定ボードのHARD CONFIG DATA1 PORTにデータを書込みます。	MCO7_BWHardConfigData1	○	○	21
	HARD CONFIG DATA2 PORT書き込み関数 指定ボードのHARD CONFIG DATA2 PORTにデータを書込みます。	MCO7_BWHardConfigData2	○	○	22
	HARD CONFIG DATA3 PORT書き込み関数 指定ボードのHARD CONFIG DATA3 PORTにデータを書込みます。	MCO7_BWHardConfigData3	○	○	23
	HARD CONFIG STATUS1 PORT読み出し関数 指定ボードのHARD CONFIG STATUS1 PORTにデータを読み出します。	MCO7_BRHardConfigStatus	○	○	24
	HARD CONFIG STATUS2 PORT読み出し関数 指定ボードのHARD CONFIG STATUS2 PORTにデータを書込みます。	MCO7_BRHardConfigPauseStatus	○	○	25
	HARD CONFIG STATUS3 PORT読み出し関数 指定ボードのHARD CONFIG STATUS3 PORTにデータを読み出します。	MCO7_BRHardConfigStbyStatus	○	○	26
	HARD CONFIG STATUS4 PORT読み出し関数 指定ボードのHARD CONFIG STATUS4 PORTにデータを読み出します。	MCO7_BRHardConfigSigStatus	○	○	27
	HARD CONFIG DATA1 PORT読み出し関数 指定ボードのHARD CONFIG DATA1 PORTにデータを読み出します。	MCO7_BRHardConfigData1	○	○	28
	HARD CONFIG DATA2 PORT読み出し関数 指定ボードのHARD CONFIG DATA2 PORTにデータを読み出します。	MCO7_BRHardConfigData2	○	○	29
	HARD CONFIG DATA3 PORT読み出し関数 指定ボードのHARD CONFIG DATA PORTにデータを読み出します。	MCO7_BRHardConfigData3	○	○	30
	COMREG NOT FULL WAIT関数 指定デバイスのコマンド予約レジスタが予約可能な状態になるまで待機します。	MCO7_BWaitComregNotFull	○	○	31
	円弧補間短軸PULSE数ゲット関数 MCCO7Eコマンドによる円弧補間ドライブを行う際に必要となる短軸パルス数を算出します。	MCO7_GetCirShortPulse	○	○	32
	データ構造体 データを一括で読み書きするときに使用します。	MCO7_TAG_S_DATA	○	○	33
	データセット関数 32ビットデータをデータ構造体に格納します。	MCO7_SetData	○	○	34
	データゲット関数 データ構造体の内容を、32ビットデータに変換し返します。	MCO7_GetData	○	○	35
	DRIVE COMMAND データ構造体書き込み関数 指定デバイスのDRIVE DATA1,2 PORTにデータ構造体のデータを書込後、コマンド書き込みます。	MCO7_IWDrive	○	○	36
	DRIVE DATA データ構造体書き込み関数 指定デバイスのDRIVE DATA1,2 PORTにデータ構造体の内容を書き込みます。	MCO7_IWData	○	○	37
	DRIVE DATA データ構造体読み出し関数 指定デバイスのDRIVE DATA1,2 PORTを読み出し、データ構造体に格納します。	MCO7_ IRDrive	○	○	38

種別	名称 / 説明	記号	C-VX870(E)v1 C-VX872(E)v1	C-VX871(E)v1 C-VX873(E)v1	ページ
関数 デバイス	MPLリセット関数 指定されたボードのMPL内部の状態を初期状態に戻します。	MCO7_Reset	○	○	39
	16ビット符号なし変換関数 指定された16ビット符号付きデータを16ビット符号なしデータに変換します。	MCO7_Unsigned16	○	○	40
	16ビット符号付き変換関数 指定された16ビット符号なしデータを16ビット符号つきデータに変換します。	MCO7_Signed16	○	○	41
	32ビット符号なし変換関数 指定された32ビット符号付きデータを32ビット符号なしデータに変換します。	MCO7_Unsigned32	○	○	42
	32ビット符号付き変換関数 指定された32ビット符号なしデータを32ビット符号つきデータに変換します。	MCO7_Signed32	○	○	43
	ボードタイプ読み出し関数 指定されたボードのボードタイプ及びバージョンを読み出します。	MCO7_ReadBoardType	○	○	44

5-3. ドライブコマンド一覧

デバイスドライバで対応している MCC07E ドライブコマンドの一覧を示します。

●汎用コマンド ---------- 各軸 DRIVE STATUS1 PORT BUSY=0 および ERROR=0 のときに DRIVE COMMAND PORT に書き込み可能なコマンドです。

●特殊コマンド ---------- 各軸 DRIVE COMMAND PORT に常時書き込み可能なコマンドです。カウンタコマンドもドライブ特殊コマンドの中に含まれます。

(1) 汎用コマンド

COMMAND CODE	コマンド名称	機能	実行時間 (起動時間)	PAGE 基本	PAGE 応用
H'00	NO OPERATION	機能なし	225 ns	95	−
H'01	SPEC INITIALIZE1	ドライブパルスの出力仕様の設定	225 ns	76	−
H'02	SPEC INITIALIZE2	CWLM, CCWLM, RDYINT, SS0, SS1 の設定	225 ns	77	−
H'03	SPEC INITIALIZE3	$\overline{\text{DRST}}$, $\overline{\text{DEND/PO}}$, $\overline{\text{DALM}}$, STBY の設定	225 ns	79	−
H'04	−	−	−	−	
H'05	FSPD SET	第 1 パルスのパルス周期の設定	225 ns	−	50
H'06	HIGH SPEED SET	加減速ドライブの速度倍率と最高速度の設定	225 ns	−	51
H'07	LOW SPEED SET	加減速ドライブの開始速度と終了速度の設定	225 ns	−	52
H'08	RATE SET	加減速カーブの変速周期の設定	225 ns	−	53
H'09	SCAREA SET	加減速カーブの S 字変速領域の設定	225 ns	−	54
H'0A	DOWN PULSE ADJUST	減速パルス数のオフセット設定	225 ns	−	55
H'0B	−	−	−	−	
H'0C	JSPD SET	JOG ドライブのパルス速度の設定	225 ns	81	−
H'0D	JOG PULSE SET	JOG ドライブのパルス数の設定	225 ns	82	−
H'0E	−	−	−	−	
H'0F	ORIGIN SPEC SET	ORIGIN ドライブの動作仕様の設定	225 ns	−	57
H'10	+JOG *P	＋方向 JOG ドライブの実行	375 ns	83	−
H'11	-JOG *P	－方向 JOG ドライブの実行	375 ns	83	−
H'12	+SCAN *P	＋方向 SCAN ドライブの実行	375 ns	84	−
H'13	-SCAN *P	－方向 SCAN ドライブの実行	375 ns	84	−
H'14	INC INDEX *P	相対アドレス INDEX ドライブの実行	375ns	85	−
H'15	ABS INDEX *P	絶対アドレス INDEX ドライブの実行	375 ns	86	−
H'16	−	−	−	−	
H'17	−	−	−	−	
H'18	ORIGIN SCAN *P	ORIGIN SCAN ドライブの実行	375 ns	−	59
H'19	ORIGIN CONSTANT SCAN *P	ORIGIN CONSTANT SCAN ドライブの実行	375 ns	−	59
H'1A	−	−	−	−	
H'1B	−	−	−	−	
H'1C	−	−	−	−	
H'1D	−	−	−	−	
H'1E	−	−	−	−	
H'1F	−	−	−	−	
H'20	CP SPEC SET *1	CPPOUT 出力の設定	225 ns	−	61
H'21	−	−	−	−	
H'22	LONG POSITION SET	直線補間ドライブの長軸アドレスの設定	225 ns	−	66
H'23	SHORT POSITION SET	直線補間ドライブの短軸アドレスの設定	225 ns	−	67
H'24	−	−	−	−	
H'25	−	−	−	−	
H'26	−	−	−	−	
H'27	−	−	−	−	
H'28	CIRCULAR XPOSITION SET	円弧補間ドライブの X 座標アドレスの設定	225 ns	−	75
H'29	CIRCULAR YPOSITION SET	円弧補間ドライブの Y 座標アドレスの設定	225 ns	−	76
H'2A	CIRCULAR PULSE SET	円弧補間ドライブの短軸パルス数の設定	225 ns	−	77
H'2B	−	−	−	−	
H'2C	−	−	−	−	
H'2D	−	−	−	−	
H'2E	−	−	−	−	
H'2F	−	−	−	−	

COMMAND CODE	コマンド名称		機能	実行時間（起動時間）	PAGE 基本	PAGE 応用
H'30	MAIN STRAIGHT CP	*P	メイン軸直線補間ドライブの実行	375 ns	―	70
H'31	SUB STRAIGHT CP	*P	サブ軸直線補間ドライブの実行	375 ns	―	69
H'32	MAIN XY STRAIGHT CP	*1 *P	メインチップ２軸直線補間ドライブの実行	375 ns	―	68
H'33	―		―	―	―	
H'34	―		―	―	―	
H'35	―		―	―	―	
H'36	―		―	―	―	
H'37	―		―	―	―	
H'38	MAIN CIRCULAR CP	*1 *P	メイン軸円弧補間ドライブの実行	375 ns	―	80
H'39	SUB CIRCULAR CP	*1 *P	サブ軸円弧補間ドライブの実行	375 ns	―	79
H'3A	MAIN XY CIRCULAR CP	*1 *P	メインチップ２軸円弧補間ドライブの実行	375 ns	―	78
H'3B	―		―	―	―	
H'3C	―		―	―	―	
H'3D	―		―	―	―	
H'3E	―		―	―	―	
H'3F	―		―	―	―	

*1 ：2 軸相関コマンド
*P ：パルス出力を伴うコマンド

(2) 特殊コマンド

COMMAND CODE	コマンド名称	機能	実行時間 (起動時間)	PAGE 基本	PAGE 応用
H'80	ADDRESS COUNTER PRESET	アドレスカウンタの現在位置の設定	225 ns	101	−
H'81	ADDRESS COUNTER INITIALIZE1	アドレスカウンタの各機能の設定	225 ns	96	−
H'82	ADDRESS COUNTER INITIALIZE2	アドレスカウンタの各機能の設定	225 ns	99	−
H'83	−	−	−	−	
H'84	−	−	−	−	
H'85	−	−	−	−	
H'86	−	−	−	−	
H'87	ADDRESS COUNTER MAX COUNT SET	アドレスカウンタの最大カウント数の設定	225 ns	−	91
H'88	ADRINT COMPARE REGISTER1 SET	ADRINT のコンペアレジスタ１の設定	225 ns	102	−
H'89	ADRINT COMPARE REGISTER2 SET	ADRINT のコンペアレジスタ２の設定	225 ns	102	−
H'8A	ADRINT COMPARE REGISTER3 SET	ADRINT のコンペアレジスタ３の設定	225 ns	102	−
H'8B	−	−	−	−	
H'8C	ADRINT COMP1 ADD DATA SET	ADRINT の COMP1 ADD データの設定	225 ns	103	−
H'8C	−	−	−	−	
H'8E	−	−	−	−	
H'8F	−	−	−	−	
H'90	PULSE COUNTER PRESET	パルスカウンタのカウント初期値の設定	225 ns	109	−
H'91	PULSE COUNTER INITIALIZE1	パルスカウンタの各機能の設定	225 ns	104	−
H'92	PULSE COUNTER INITIALIZE2	パルスカウンタの各機能の設定	175 ns	107	−
H'93	−	−	−	−	
H'94	−	−	−	−	
H'95	−	−	−	−	
H'96	−	−	−	−	
H'97	PULSE COUNTER MAX COUNT SET	パルスカウンタの最大カウント数の設定	225 ns	−	92
H'98	CNTINT COMPARE REGISTER1 SET	CNTINT のコンペアレジスタ１の設定	225 ns	110	−
H'99	CNTINT COMPARE REGISTER2 SET	CNTINT のコンペアレジスタ２の設定	225 ns	110	−
H'9A	CNTINT COMPARE REGISTER3 SET	CNTINT のコンペアレジスタ３の設定	225 ns	110	−
H'9B	−	−	−	−	
H'9C	CNTINT COMP1 ADD DATA SET	CNTINT の COMP1 ADD データの設定	225 ns	111	−
H'9D	−	−	−	−	
H'9E	−	−	−	−	
H'9F	−	−	−	−	
H'A0	DFL COUNTER PRESET	パルス偏差カウンタのカウント初期値の設定	225 ns	118	−
H'A1	DFL COUNTER INITIALIZE1	パルス偏差カウンタの各機能の設定	225 ns	112	−
H'A2	DFL COUNTER INITIALIZE2	パルス偏差カウンタの各機能の設定	225 ns	115	−
H'A3	DFL COUNTER INITIALIZE3	パルス偏差カウンタの各機能の設定	225 ns	117	−
H'A4	−	−	−	−	
H'A5	−	−	−	−	
H'A6	−	−	−	−	
H'A7	−	−	−	−	
H'A8	DFLINT COMPARE REGISTER1 SET	DFLINT のコンペアレジスタ１の設定	225 ns	119	−
H'A9	DFLINT COMPARE REGISTER2 SET	DFLINT のコンペアレジスタ２の設定	225 ns	119	−
H'AA	DFLINT COMPARE REGISTER3 SET	DFLINT のコンペアレジスタ３の設定	225 ns	119	−
H'AB	−	−	−	−	
H'AC	DFLINT COMP1 ADD DATA SET	DFLINT の COMP1 ADD データの設定	225 ns	120	−
H'AD	−	−	−	−	
H'AE	−	−	−	−	
H'AF	−	−	−	−	
H'B0--H'BF	−	−	−	−	
H'C0	UDC SPEC SET	UP/DOWN/CONST の変更動作点の設定	225 ns	−	81
H'C1	SPEED CHANGE SPEC SET	SPEED CHANGE の変更動作点の設定	225 ns	−	83
H'C2	−	−	−	−	
H'C3	INDEX CHANGE SPEC SET	INDEX CHANGE の変更動作点の設定	225 ns	−	86
H'C4	UP DRIVE	UP DRIVE の実行	225 ns	−	82
H'C5	DOWN DRIVE	DOWN DRIVE の実行	225 ns	−	82
H'C6	CONST DRIVE	CONST DRIVE の実行	225 ns	−	82
H'C7	−	−	−	−	

COMMAND CODE	コマンド名称	機能	実行時間（起動時間）	PAGE 基本	PAGE 応用
H'C8	SPEED CHANGE	SPEED CHANGE の実行	225 ns	－	84
H'C9	－	－	－	－	
H'CA	RATE CHANGE	RATE CHANGE の設定	225 ns	－	85
H'CB	－	－	－	－	
H'CC	INC INDEX CHANGE	INC INDEX CHANGE の実行	225 ns	－	87
H'CD	ABS INDEX CHANGE	ABS INDEX CHANGE の実行	225 ns	－	88
H'CE	PLS INDEX CHANGE	PLS INDEX CHANGE の実行	225 ns	－	89
H'CF	－	－	－	－	
H'D0	－	－	－	－	
H'D1	ERROR STATUS READ	ERROR STATUS の読み出し	225 ns	90	－
H'D2	－	－	－	－	
H'D3	－	－	－	－	
H'D4	MCC SPEED READ	ドライブパルス速度の読み出し	225 ns	92	－
H'D5	MCC SET DATA READ	設定データの読み出し	225 ns	93	－
H'D6	－	－	－	－	
H'D7	－	－	－	－	
H'D8	ADDRESS COUNTER READ	アドレスカウンタの読み出し	225 ns	121	－
H'D9	PULSE COUNTER READ	パルスカウンタの読み出し	225 ns	121	－
H'DA	DFL COUNTER READ	パルス偏差カウンタの読み出し	225 ns	121	－
H'DB	－	－	－	－	
H'DC	ADDRESS LATCH DATA READ	アドレスカウンタのラッチデータの読み出し	225 ns	－	95
H'DD	PULSE LATCH DATA READ	パルスカウンタのラッチデータの読み出し	225 ns	－	95
H'DE	DFL LATCH DATA READ	パルス偏差カウンタのラッチデータの読み出し	225 ns	－	95
H'DF	－	－	－	－	
H'E0	－	－	－	－	
H'E1	－	－	－	－	
H'E2	－	－	－	－	
H'E3	－	－	－	－	
H'E4	－	－	－	－	
H'E5	ERROR STATUS MASK	ERROR に出力する ERROR STATUS のマスク	225 ns	89	－
H'E6	－	－	－	－	
H'E7	－	－	－	－	
H'E8	COUNT LATCH SPEC SET	カウントデータのラッチタイミングの設定	225 ns	－	93
H'E9	－	－	－	－	
H'EA	－	－	－	－	
H'EB	－	－	－	－	
H'EC	－	－	－	－	
H'ED	－	－	－	－	
H'EE	－	－	－	－	
H'EF	－	－	－	－	
H'F0	MCC CHIP RESET	MCC07E の初期化の実行	475 ns	－	90
H'F1	HARD INITIALIZE1	OUT A,B 出力機能の設定	225 ns	－	45
H'F2	－	－	－	－	
H'F3	－	－	－	－	
H'F4	HARD INITIALIZE4	軸制御部のデジタルフィルタの設定	225 ns	－	46
H'F5	HARD INITIALIZE5	軸制御部のデジタルフィルタの設定	225 ns	－	47
H'F6	HARD INITIALIZE6	外部パルスのデジタルフィルタの設定	225 ns	－	48
H'F7	HARD INITIALIZE7	入力信号のアクティブ論理の選択	225 ns	－	49
H'F8	－	－	－	－	
H'F9	－	－	－	－	
H'FA	－	－	－	－	
H'FB	－	－	－	－	
H'FC	SIGNAL OUT	汎用出力信号の操作	225 ns	88	－
H'FE	SLOW STOP	減速停止の実行	225 ns	87	－
H'FF	FAST STOP	即時停止の実行	225 ns	87	－

5-4. HARD CONFIG コマンド一覧表

COMMAND CODE	コマンド名称	機能	実行時間（起動時間）	PAGE 基本	PAGE 応用
H'00	MAN MASK	MAN 信号入力のマスク	200ns	－	96
H'01	SENSOR SIGNAL SELECT	各軸汎用センサ信号（SS0,SS1）接続信号の設定	200ns	－	97
H'02	－	－	－	－	
H'03	－	－	－	－	
H'04	SIGNAL OUT SELECT	SIGNAL OUT 出力接続信号の設定	200ns	－	99
H'05	SIGNAL OUT TIMER SET	SIGNAL OUT 出力時間の設定	200ns	－	101
H'06	SIGNAL OUT LATCH STATUS CLR	SIGNAL STATUS のラッチフラグのクリア	200ns	－	102
H'07	－	－	－	－	
H'08	－	－	－	－	
H'09	－	－	－	－	
H'0A	－	－	－	－	
H'0B	－	－	－	－	
H'0C	－	－	－	－	
H'0D	－	－	－	－	
H'0E	－	－	－	－	
H'0F	－	－	－	－	
H'10	PAUSE	各軸 PAUSE の SET/CLR 操作	200ns	－	107
H'11	PAUSE SET SPEC	各軸 PAUSE の SET 条件設定	200ns	－	103
H'12	PAUSE CLR SPEC	各軸 PAUSE の CLR 条件設定	200ns	－	105
H'13	－	－	－	－	
H'14	－	－	－	－	
H'15	－	－	－	－	
H'16	－	－	－	－	
H'17	－	－	－	－	
H'18	－	－	－	－	
H'19	－	－	－	－	
H'1A	－	－	－	－	
H'1B	－	－	－	－	
H'1C	－	－	－	－	
H'1D	－	－	－	－	
H'1E	－	－	－	－	
H'1F	－	－	－	－	
H'20	HARD CONFIG SET DATA READ	HARD CONFIG 設定データの読み出し	200ns	－	108
H'21	－	－	－	－	
H'22	－	－	－	－	
H'23	－	－	－	－	
H'24	－	－	－	－	
H'25	－	－	－	－	
H'26	－	－	－	－	
H'27	－	－	－	－	
H'28	－	－	－	－	
H'29	－	－	－	－	
H'2A	－	－	－	－	
H'2B	－	－	－	－	
H'2C	－	－	－	－	
H'2D	－	－	－	－	
H'2E	－	－	－	－	
H'2F	－	－	－	－	
H'30-EF	－	－	－	－	
H'F0	HARD CONFIG RESET	HARD CONFIGURATION の初期化の実行	200ns	－	108
H'F1-FF	－	－	－	－	

本版で改訂された主な箇所

箇　所	内　容

■製品保証

保証期間と保証範囲について

● 納入品の保証期間は、納入後１ヶ年と致します。
● 上記保証期間中に当社の責により故障を生じた場合は、その修理を当社の責任において行います。
(日本国内のみ)
ただし、次に該当する場合は、この保証対象範囲から除外させて頂きます。
(1) お客様の不適当な取り扱い、ならびに使用による場合。
(2) 故障の原因が、当製品以外からの事由による場合。
(3) お客さまの改造、修理による場合。
(4) 製品出荷当時の科学・技術水準では予見が不可能だった事由による場合。
(5) その他、天災、災害等、当社の責にない場合。
(注1)ここでいう保証は、納入品単体の保証を意味するもので、納入品の故障により誘発される損害はご容赦頂きます。
(注2)当社において修理済みの製品に関しましては、保証外とさせて頂きます。

技術相談のお問い合わせ
販売に関するお問い合わせ

TEL.(042)664-5384　FAX.(042)666-2031
E-mail　s-support@melec-inc.com

株式会社メレック
〒193-0834 東京都八王子市東浅川町516-10
www.melec-inc.com