



STEPPING & SERVO MOTOR CONTROLLER'S OPTION

MPL-34-02_{V3.00}/AL2W32

MPL-35-02_{V3.00}/AL2W64

取扱説明書 **応用機能編** (設計者用)

(AL-II シリーズ PCIマスター用Windowsデバイスドライバ)

MCC09ユニット編

USER'S MANUAL

既に本製品の別冊、取扱説明書を読まれていることを前提に、より多彩な機能・仕様を解説した応用機能編です。取扱説明書以上の内容についてはこの応用機能編を良く読んで十分に理解してください。
この応用機能編は、いつでも取り出して読めるように保管してください。

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、AL-II 対応シリーズのステッピングモータ、サーボモータ、および I/O システム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 における応用的な機能および仕様について説明しています。各コントローラの「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」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」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

- AL-Ⅱ対応コントローラ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。
また、本書では、*1の製品のことをコントローラドライバと呼称します。

製品名	1 軸目	2 軸目	3 軸目	4 軸目	5 軸目	6 軸目
2C-7760	X 軸	Y 軸	Z 軸	A 軸	—	—
2CD-7720/A5F31DE *1	X 軸	Y 軸	—	—	—	—
2CD-7721/A5F41DE *1	X 軸	Y 軸	—	—	—	—
2CD-7730/A5F31Q *1	X 軸	Y 軸	Z 軸	A 軸	—	—
2CD-7730G02/QM40X4E *1	X 軸	Y 軸	Z 軸	A 軸	—	—
2CD-7740G01/QM40X6 *1	X 軸	Y 軸	Z 軸	A 軸	B 軸	C 軸

以降、原則として X 軸についてのみ説明します。

- 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的な仕様については、別冊デバイスドライバ「取扱説明書」をご覧ください。

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

1. 概要

1-1. 特徴	7
---------	---

2. 関数リファレンス

2-1. MCC PORT アクセス関数の移行について	8
2-2. デバイス関数	9
2-2-1. MCC PORT アクセス関数	9
DRIVE DATA 32 ビット書き込み関数	9
DRIVE DATA1 PORT 書き込み関数	10
DRIVE DATA2 PORT 書き込み関数	11
DRIVE DATA 32 ビット読み出し関数	12
DRIVE DATA1 PORT 読み出し関数	13
DRIVE DATA2 PORT 読み出し関数	14
データ構造体	15
DRIVE COMMAND データ構造体書き込み関数	16
DRIVE DATA データ構造体書き込み関数	17
DRIVE DATA データ構造体読み出し関数	18
データセット関数	19
データゲット関数	20
2-2-2. コマンド予約機能関数	21
COMREG NOT FULL WAIT 関数	21
DRIVE COMMAND バッファ書き込み関数	22
2-2-3. 円弧補間の演算関数	23
円弧補間短軸 PULSE 数ゲット関数	23
2-2-4. その他の関数	24
16 ビット符号なし変換関数	24
16 ビット符号付き変換関数	25
32 ビット符号なし変換関数	26
32 ビット符号付き変換関数	27
2-3. ユニット MCM 関数	28
2-3-1. MCM SPEC の設定と読み出し	28
AXIS MCM SPEC 構造体	28
UNIT MCM SPEC 構造体	29
UNIT MCM SPEC0 SET 関数	30
UNIT MCM SPEC0 GET 関数	31
UNIT MCM SPEC2 SET 関数	32
UNIT MCM SPEC2 GET 関数	35
UNIT MCM SPEC3 SET 関数	36
UNIT MCM SPEC3 GET 関数	38
UNIT MCM SPEC4 SET 関数	39
UNIT MCM SPEC4 GET 関数	41
UNIT MCM SPEC5 SET 関数	42
UNIT MCM SPEC5 GET 関数	44

目 次

PAGE

3. コマンド仕様

3-1. ドライブコマンド	45
3-1-1. 入出力仕様の設定	45
(1) HARD INITIALIZE1	45
(2) HARD INITIALIZE2	47
(3) HARD INITIALIZE3	48
(4) HARD INITIALIZE7	49
(5) HARD INITIALIZE8	51
3-1-2. ドライブパラメータの設定	52
(1) FSPD SET	52
(2) HIGH SPEED SET	53
(3) LOW SPEED SET	54
(4) RATE SET	55
(5) SCAREA SET	56
(6) SHAREA SET	57
(7) DOWN PULSE ADJUST	58
3-1-3. ORIGIN ドライブの設定	59
(1) ORIGIN SPEC SET	59
3-1-4. 補間ドライブの設定	61
(1) CP SPEC SET	61
3-1-5. 直線補間ドライブの設定と実行	63
(1) LONG POSITION SET	68
(2) SHORT POSITION SET	69
(3) MAIN STRAIGHT CP	70
(4) SUB STRAIGHT CP	71
3-1-6. 円弧補間ドライブの設定と実行	72
(1) CIRCULAR XPOSITION SET	77
(2) CIRCULAR YPOSITION SET	78
(3) CIRCULAR PULSE SET	79
(4) MAIN CIRCULAR CP	80
(5) SUB CIRCULAR CP	81
3-1-7. SPEED CHANGE の設定と実行	82
(1) SPEED CHANGE SPEC SET	82
(2) SPEED RATE CHANGE	83
3-1-8. INDEX CHANGE の設定と実行	84
(1) INDEX CHANGE SPEC SET	84
(2) INC INDEX CHANGE	85
(3) ABS INDEX CHANGE	86
(4) PLS INDEX CHANGE	87
3-1-9. RSPD データの読み出し	88
(1) RSPD DATA READ	88
3-2. カウンタコマンド	89
3-2-1. アドレスカウンタの設定	89
(1) ADDRESS COUNTER MAX COUNT SET	89
3-2-2. パルスカウンタの設定	90
(1) PULSE COUNTER MAX COUNT SET	90
3-2-3. カウンタのラッチ・クリア機能の設定	91
(1) COUNT LATCH SPEC SET	91
3-2-4. カウントデータのラッチデータの読み出し	93
(1) ADDRESS LATCH DATA READ	93
(2) PULSE LATCH DATA READ	93
(3) DFL LATCH DATA READ	93

目 次

PAGE

4. 機能説明

4-1. ドライブ仕様	94
4-1-1. コマンド予約機能	94
4-1-2. 入出力仕様	96
(1) 入出力信号の論理切り替え機能	96
4-1-3. ドライブパラメータ	97
(1) 加減速パラメータ	97
(2) 加減速時定数	98
(3) 直線加減速ドライブ	100
(4) S 字加減速ドライブ	101
(5) 加速ドライブ	105
(6) 減速ドライブ	105
(7) 一定速ドライブ	105
(8) その他のドライブ	106
4-1-4. ORIGIN ドライブ (MCC09 チップの機械原点検出機能)	107
4-1-5. 補間ドライブ	109
(1) 任意軸補間ドライブ	109
(2) 直線補間ドライブ	112
(3) 円弧補間ドライブ	113
(4) 線速一定制御	116
4-1-6. ドライブ CHANGE 機能	117
(1) SPEED CHANGE 機能	117
(2) INDEX CHANGE 機能	119
4-1-7. 読み出し機能	124
(1) カウントデータのラッチデータ読み出し	124
4-1-8. RSPD 機能	125
4-2. カウンタ仕様	126
4-2-1. リングカウンタ機能	126
4-2-2. カウントデータのラッチ・クリア機能	127
4-3. I/O 信号割付機能	128
4-3-1. MCM の設定	128

5. 付録

5-1. 初期仕様一覧	133
(1) 応用設定	133
(2) 応用ドライブパラメータ	134
5-2. 関数一覧	135
5-3. コマンド一覧	139
(1) 汎用コマンド	139
(2) 特殊コマンド	141

本版で改訂された主な箇所

1. 概要

1-1. 特徴

- AL-II シリーズは、装置の分散化や補助軸の追加に柔軟且つ簡易に対応できるステッピングモータ、サーボモータ、デジタル I/O、アナログ I/O をコントロールする弊社オリジナルの高速シリアル通信システムです。
 - ・ AL-II シリーズは、20Mbps/50m または 10Mbps/100m の絶縁型高速シリアル通信です。
これにより、従来ボードコントローラに匹敵する性能(弊社比)でパソコンシステムの省配線化が図れます。
 - ・ Windows 用デバイスドライバ関数は、弊社製 PCI ボードコントローラ C-VX870 シリーズ(デバイス関数)、および USB シリーズ(デバイス関数とユニット関数)間で互いに移行が容易な仕様です。
- MPL-34-02v3.00/AL2W32、および MPL-35-02v3.00/AL2W64 は、Windows パソコン上で AL-II シリーズの PCI マスターまたは PCI Express マスターを介して、弊社製ステッピング& サーボモータコントローラ AL-II シリーズの製品を動作させるための DLL ベースのドライバ関数です。
当デバイスドライバは、MCC09 搭載した新たなスレーブコントローラドライバ 2CD-7730G02/QM40X4E、2CD-7740G01/QM40X6 に対応したバージョンアップ品ですが、MCC07 搭載製品仕様にも対応しています。
 - ・ Windows 32 ビット対応版が MPL-34-02v3.00/AL2W32 です。
 - ・ Windows 64 ビット対応版が MPL-35-02v3.00/AL2W64 です。
MPL-35-02v3.00/AL2W64 は、画像処理などの高速化を目的とした Windows 64 ビット環境のモーションおよび I/O システムを可能にします。

取扱説明書で解説している標準仕様の他に、この応用機能編では、主に以下を解説します。

- - ・ コマンド予約機能
 - ・ ドライブ CHANGE 機能
 - ・ 外部トリガ出力機能
 - ・ その他、従来関数を継承できるようにするための関数など

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
- ・ 標準的な関数、コマンド、機能の解説
- ・ 全体の実行シーケンス等

2. 関数リファレンス

当デバイスドライバ仕様は、MCC07 および MCC09 の搭載製品ならびにデジタル I/O やアナログ I/O を含む関数を網羅しています。

スレーブユニットに MCC09 および MCC07 搭載製品が混在したシステム構成も可能です。
原則、関数仕様は共通性がありますので、搭載チップ毎のスレーブユニットによって、ユーザアプリケーションを分ける必要はありません。

なお、MCC09 と MCC07 搭載の製品の差異を明らかにできるように、本書では MCC09 搭載製品を対象とした製品の仕様を示します。

2-1. MCC PORT アクセス関数の移行について

本書で説明するデバイス関数は、主に従来より対応している関数群です。

これらの関数は、下記に推奨している標準関数で代用することが可能です。

下記の従来関数から推奨関数に移行することで、特に、マルチスレッドプログラミングにおける、ユーザアプリケーション側での排他処理を不要とすることができます。

従来関数群
• DRIVE COMMAND データ構造体書き込み関数 (MC07_IWDrive) • DRIVE DATA データ構造体書き込み関数 (MC07_IWData) • DRIVE DATA 32 ビット書き込み関数 (MC07_LWData) • DRIVE DATA1 PORT 書き込み関数 (MC07_BWDriveData1) • DRIVE DATA2 PORT 書き込み関数 (MC07_BWDriveData2)



推奨関数
• DRIVE COMMAND 32 ビット書き込み関数 (MC07_LWDrive)

従来関数群
• DRIVE DATA データ構造体読み出し関数 (MC07_IRDrive) • DRIVE DATA 32 ビット読み出し関数 (MC07_LRDrive) • DRIVE DATA1 PORT 読み出し関数 (MC07_BRDriveData1) • DRIVE DATA2 PORT 読み出し関数 (MC07_BRDriveData2)



推奨関数
• DRIVE COMMAND 32 ビット書き込み関数／読み出し関数 (MC07_LWRDrive) • DRIVE COMMAND PORT 書き込み関数／読み出し関数 (MC07_BWRDrive)

*1

*1 MCC09 搭載製品のみ対応している関数です。

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

指定されたデバイスのDRIVE DATA1 PORTにデータを書き込みます。

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。

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

指定されたデバイスのDRIVE DATA2 PORTにデータを書き込みます。

C言語 `BOOL MC07_BWDriveData2 (DWORD hDev, WORD *pData, MC07_S_RESULT *psResult);`

VB.NET Function MC07_BWDData2(ByVal *hDev* As Integer, ByRef *pData* As Short, ByRef *psResult* As MC07_S_RESULT) As Boolean

```
C#.NET bool MC07.BWDriveData2(uint hDev, ref ushort pData, ref MC07_S_RESULT psResu/t);
```

<i>hDev</i>	… デバイスハンドルを指定します。
<i>pData</i>	… 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。
<i>psResult</i>	… この関数を実行した結果を格納するためのRESULT構造体のポインタを指定します。

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。

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

指定されたデバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTの内容を読み出します。

C言語 `BOOL MC07_LRDrive(DWORD hDev, DWORD *pData, MC07_S_RESULT *psResu/t);`

VB.NET Function MC07_LRDrive(ByVal *hDev* As Integer, ByRef *pData* As Integer, ByRef *psResult* As MC07_S_RESULT) As Boolean

```
C#.NET bool MC07.LRDrive(uint hDev, ref uint pData, ref MC07_S_RESULT psResult);
```

hDev ... デバイスハンドルを指定します。

<i>pData</i>	… 読み出した内容を格納するための変数のポインタを指定します。 ・ DRIVE DATA2 PORTの内容が変数の上位16ビットに格納されます。 ・ DRIVE DATA1 PORTの内容が変数の下位16ビットに格納されます。
<i>psResult</i>	… この関数を実行した結果を格納するためのRESULT構造体のポインタを指定します。

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。

AL2A

Table	Text
-------	------

書 式

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

指定されたデバイスのDRIVE DATA2 PORTの内容を読み出します。

C言語 `BOOL MC07_BRDriveData2(DWORD hDev, WORD *pData, MC07_S_RESULT *psResult);`

VB.NET Function MC07_BRDriveData2(ByVal *hDev* As Integer, ByRef *pData* As Short, ByRef *psResult* As MC07_S_RESULT) As Boolean

```
C#.NET bool MC07.BRDriveData2(uint hDev, ref ushort pData, ref MC07_S_RESULT psResul/t);
```

hDev ... デバイスハンドルを指定します。

pData ... 読み出した内容を格納するための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を格納するためのRESULT構造体のポインタを指定します。

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。

データ構造体

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

機能

DRIVE DATA1 PORT、DRIVE DATA2 PORTに書き込むデータを格納します。

書式

C言語 typedef struct _MC07_S_DATA{
 WORD *MC07_Data*[4];
 } MC07_S_DATA

VB.NET Structure MC07_S_DATA
 <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> Public MC07_Data() As Short
 Public Sub Initialize()
 ReDim *MC07_Data*(3)
 End Sub
 End Structure

C#.NET struct MC07_S_DATA
 {
 [MarshalAs(UnmanagedType.ByValArray, SizeConst=4)] public ushort[] MC07_Data;
 MC07_S_DATA(ushort dummy)
 {
 MC07_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MC07_Data[0]～MC07_Data[3]は、Visual Basic.NETではMC07_Data(0)～MC07_Data(3)、C#.NETではMC07_Data[0]～MC07_Data[3]に対応します。

MC07_Data[0] ... DRIVE DATA1 PORTの内容を格納します。

MC07_Data[1] ... DRIVE DATA2 PORTの内容を格納します。

MC07_Data[2] ... 将来の拡張用です。

MC07_Data[3] ... 将来の拡張用です。

*VB.NETでは、配列を含む構造体は、Initializeメソッドにより配列を作成します。

*C#.NETでは、配列を含む構造体は、コンストラクタにより配列を作成します。

コンストラクタの引数dummyは何を指定しても無効です。

DRIVE COMMAND データ構造体書き込み関数

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

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んだ後、DRIVE COMMAND PORTにコマンドコードを書き込みます。

書 式

C言語 `BOOL MC07_IWDrive(DWORD hDev, WORD Cmd, MC07_S_DATA *psData, MC07_S_RESULT *psResult);`

VB.NET `Function MC07_IWDrive(ByVal hDev As Integer, ByVal Cmd As Short, ByRef psData As MC07_S_DATA, ByRef psResult As MC07_S_RESULT) As Boolean`

C#.NET `bool MC07_IWDrive(uint hDev, ushort Cmd, 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。
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を格納するための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。

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

指定されたデバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTにデータを書き込みます。

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

VB.NET `Function MC07_IWDData(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);
```

<i>hDev</i>	… デバイスハンドルを指定します。
<i>psData</i>	… 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。
<i>psResult</i>	… 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。

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。

AL2A

2C-7760 2CD-7720 2CD-7721 2CD-7730 2CD-7730G02 2CD-7740G01

指定されたデバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTの内容を読み出します。

C言語 `BOOL MC07_IRDrive(DWORD hDev, MC07_S_DATA *psData, MC07_S_RESULT *psResult);`

VB.NET Function MC07_IRDrive(ByVal *hDev* As Integer, ByRef *psData* As MC07_S_DATA, ByRef *psResult* As MC07_S_RESULT) As Boolean

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

<i>hDev</i>	… デバイスハンドルを指定します。
<i>psData</i>	… 読み出した内容を格納するためのデータ構造体のポインタを指定します。
<i>psResult</i>	… 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。

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。

データセット関数

AL2A

—

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。

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DATA1 PORTに対応]

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MC07_Data[0] (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 の上位 16 ビット (D31 ~ D16) → D16															

書 式

C言語 `VOID MC07_SetData(DWORD Data, MC07_S_DATA *psData);`

VB.NET `Sub MC07_SetData(ByVal Data As Integer, ByRef psData As MC07_S_DATA)`

C#.NET `void MC07.SetData(uint Data, ref MC07_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

AL2A

—

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
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DATA2 PORTに対応]															
D15	←													→	D0

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
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DATA1 PORTに対応]															
D15	←													→	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 `UInt 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。

2-2-2. コマンド予約機能関数

MCC09 の汎用コマンドは、コマンド予約機能によりコマンドを予約レジスタに書き込むことができます。

コマンド予約機能で汎用コマンドを書き込むときは、DRIVE STATUS1 PORT 内の ERROR=0 および

COMREG FL BIT=0 を確認してから汎用コマンドを書き込みます。

また、汎用コマンドをコマンド予約機能で書き込みした後に、予約コマンドの空き待ちを行います。

COMREG NOT FULL WAIT 関数は、この汎用コマンドを予約コマンドとして書き込みした後の終了待ち

するときに用いる関数です。

COMREG NOT FULL WAIT関数

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

機 能

指定されたデバイスがコマンドレジスタ予約可能な状態 (DRIVE 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。
- ・待機中にDRIVE STATUS1 PORTのERROR = 1 を検出したとき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を指定すると無限に待機します。

psResult … この関数を実行した結果を格納するための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 COMMAND バッファ書き込み関数

AL2A

2CD-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
----------	----------	----------	----------	-------------	-------------

機 能

コマンドデータ構造体の配列に格納されているコマンド、データを指定されたデバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORT、DRIVE COMMAND PORTに指定された個数書き込みます。

書 式

C言語 `BOOL MC07_LWDriveBuf(DWORD hDev, MC07_S_COMMAND_DATA CmdDataBuf[], WORD Count, MC07_S_RESULT *psResult);`

VB.NET `Function MC07_LWDriveBuf(ByVal hDev As Integer, ByVal CmdDataBuf() As MC07_S_COMMAND_DATA, ByVal Count As Short, ByRef psResult As MC07_S_RESULT) As Boolean`

C#.NET `bool MC07.LWDriveBuf(uint hDev, MC07_S_COMMAND_DATA[] CmdDataBuf, ushort Count, ref MC07_S_RESULT psResult);`

引 数

hDev ... デバイスハンドルを指定します。
CmdDataBuf ... コマンドデータ構造体の配列に指定します。
 CmdDataBuf[0] ... 1番目に書き込むコマンドコード構造体を格納します。
 ...
 CmdDataBuf[7] ... 8番目に書き込むコマンドコード構造体を格納します。
 CmdDataBuf[8]~*CmdDataBuf*[15] ... 将来の拡張用です。
Count ... コマンドコード構造体の配列に格納されている要素の個数(最大8個)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。

2-2-3. 円弧補間の演算関数

円弧補間短軸PULSE数ゲット関数

AL2A

2C-7760 2CD-7720 2CD-7721 2CD-7730 2CD-7730G02 2CD-7740G01

機 能

指定された円弧の中心点、目的地、回転方向から、目的地までの短軸パルス数を計算します。

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。

計算方法については、「4-1-5. 補間ドライブ (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, 648~+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 MC07_GetCirShortPulse(WORD Dir, MC07_S_XY_POSITION *psCenterPosition,
 MC07_S_XY_POSITION *psTargetPosition,
 LONG *pShortPulse, MC07_S_RESULT *psResult);`

VB.NET `Function MC07_GetCirShortPulse(ByVal Dir As Short,
 ByRef psCenterPosition As MC07_S_XY_POSITION,
 ByRef psTargetPosition As MC07_S_XY_POSITION, ByRef pShortPulse As Integer,
 ByRef psResult As MC07_S_RESULT) As Boolean`

C#.NET `bool MC07.GetCirShortPulse(ushort Dir, ref MC07_S_XY_POSITION psCenterPosition,
 ref MC07_S_XY_POSITION psTargetPosition, ref int pShortPulse,
 ref MC07_S_RESULT psResult);`

引 数

Dir ... 回転方向を指定します。

指定できる値	意味
MC07_CCW	-(CCW) 方向
MC07_CW	+(CW) 方向

psCenterPosition ... 中心点のX-Y相対座標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。
(指定できる値の範囲はX座標・Y座標ともに、-8,388,607~+8,388,607です。)
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。

psTargetPosition ... 目的地のX-Y相対座標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。
(指定できる値の範囲はX座標・Y座標ともに、-16,777,214~+16,777,214です。)
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 ... 目的地の短軸パルス数を格納するための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を格納するための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。

2-2-4. その他の関数

16ビット符号なし変換関数

AL2A

—

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 MC07_Unsigned16( SHORT Data );`

VB.NET `Function MC07_Unsigned16(ByVal Data As Short) As UShort`

C#.NET `ushort MC07.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

AL2A

—

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 MC07_Signed16( WORD Data );`

VB.NET `Function MC07_Signed16(ByVal Data As UShort) As Short`

C#.NET `short MC07.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

AL2A

—

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 MC07_Unsigned32( LONG Data );`

VB.NET `Function MC07_Unsigned32( ByVal Data As Integer ) As UInteger`

C#.NET `uint MC07.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

AL2A

—

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 MC07_Signed32( DWORD Data );`

VB.NET `Function MC07_Signed32( ByVal Data As UInteger ) As Integer`

C#.NET `int MC07.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。

2-3. ユニット MCM 関数

MCM の設定・制御を行います。

2-3-1. MCM SPEC の設定と読み出し

MCM の SPEC 設定と読み出しを行います。

設定する内容については、UNIT MCM SPEC SET1, 2, 3 関数を参照してください。

AXIS MCM SPEC構造体

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02	2CD-7740G01
---------	----------	----------	----------	-------------	-------------

機 能

軸の基本設定を格納します。

書 式

C言語 typedef struct _MC07_S_AXIS_MCM_SPEC {
 WORD *Spec*;
 } MC07_S_AXIS_MCM_SPEC;

VB.NET Structure MC07_S_AXIS_MCM_SPEC
 Public *Spec* As Short
 End Structure

C#.NET struct MC07_S_AXIS_MCM_SPEC
 {
 public ushort *Spec*;
 }

メンバ

Spec ... MCM SPEC2, 3, 4, 5の設定データを格納します。

UNIT MCM SPEC構造体 AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02
---------	----------	----------	----------	-------------

機 能
ユニットの基本設定を格納します。

書 式
C言語 typedef struct _MC07_S_UNIT_MCM_SPEC {
 MC07_S_AXIS_MCM_SPEC *X*;
 MC07_S_AXIS_MCM_SPEC *Y*;
 MC07_S_AXIS_MCM_SPEC *Z*;
 MC07_S_AXIS_MCM_SPEC *A*;
 } MC07_S_UNIT_MCM_SPEC;

VB.NET Structure MC07_S_UNIT_MCM_SPEC
 Public *X* As MC07_S_AXIS_SPEC
 Public *Y* As MC07_S_AXIS_SPEC
 Public *Z* As MC07_S_AXIS_SPEC
 Public *A* As MC07_S_AXIS_SPEC
 End Structure

C#.NET struct MC07_S_UNIT_MCM_SPEC
 {
 public MC07_S_AXIS_MCM_SPEC *X*;
 public MC07_S_AXIS_MCM_SPEC *Y*;
 public MC07_S_AXIS_MCM_SPEC *Z*;
 public MC07_S_AXIS_MCM_SPEC *A*;
 }

メンバ
X …… X軸(または $\overline{\text{SOUT0}}$ 、 $\overline{\text{OUT00}}$ 、 $\overline{\text{OUT04}}$ 信号)のAXIS MCM SPEC構造体の内容を格納します。
Y …… Y軸(または $\overline{\text{SOUT1}}$ 、 $\overline{\text{OUT01}}$ 、 $\overline{\text{OUT05}}$ 信号)のAXIS MCM SPEC構造体の内容を格納します。
Z …… Z軸(または $\overline{\text{SOUT2}}$ 、 $\overline{\text{OUT02}}$ 、 $\overline{\text{OUT06}}$ 信号)のAXIS MCM SPEC構造体の内容を格納します。
A …… A軸(または $\overline{\text{SOUT3}}$ 、 $\overline{\text{OUT03}}$ 、 $\overline{\text{OUT07}}$ 信号)のAXIS MCM SPEC構造体の内容を格納します。

AL2A

機能

この関数を実行する前に、STATUS1の全軸のERROR = 0, BUSY = 0を確認してください。

UNIT MCM SPEC0 SET関数は、ユニット全体の関数です。

C言語 `BOOL MC07_USetMcmSpec0( DWORD hUnit, WORD Spec, MC07_S_RESULT *psResult );`

VB.NET `Function MC07_USetMcmSpec0( ByVal hUnit As Integer, ByVal Spec As Short, ByRef psResult As MC07_S_RESULT ) As Boolean`

[illegible]

- ... ユニットハンドルを指定します。
- ... MCM SPEC0の設定データを指定します。
MCMの動作モードを設定します。ユニット単位の設定です。

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	SIGNAL OUT ENABLE	0

電源投入後の初期値は H' 0000（アンダーライン側）です。

OUTx信号の出力仕様として、「汎用出力／SIGNAL OUT機能」を選択します。

1: SIGNAL OUT機能 (MCM SPEC4、MCM SPEC5により、出力する信号を選択)

SIGNAL_OUT_ENABLEを1→0に切り替えた時、 $\overline{\text{OUTx}}$ 信号の出力状態は、全てノットアクティブにリセットされます。

psResult ... この関数を実行した結果を格納するためのRESULT構造体のポインタを指定します。

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。

AL2A

2CD-7730G02

機能

指定されたユニットに対し、MCM SPEC0の設定を読み出します。

書 式

C言語 `BOOL MC07_UGetMcmSpec0( DWORD hUnit, WORD *pSpec, MC07_S_RESULT *psResult );`

VB.NET Function MC07_UGetMcmSpec0(ByVal *hUnit* As Integer, ByRef *pSpec* As Short, ByRef *psResult* As MC07_S_RESULT) As Boolean

```
C#.NET bool MC07.UGetMcmSpec0( uint hUnit, ref ushort pSpec, ref MC07_S_RESULT psResult );
```

引 数

hUnit ... ユニットハンドルを指定します。

pSpec ... MCM SPEC0の設定データを格納するための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を格納するための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。

UNIT MCM SPEC2 SET関数

AL2A

2C-7760 2CD-7720 2CD-7721 2CD-7730 2CD-7730G02

機能

指定されたユニットにMCM SPEC2を設定します。

この関数を実行する前に、STATUS1の全軸のERROR = 0, BUSY = 0を確認してください。

UNIT MCM SPEC2 SET関数は、ユニット内の1軸に対する設定です。
停止センサSS0に入力する信号を設定します。

書式

C言語 BOOL MC07_USetMcmSpec2(DWORD *hUnit*, DWORD *AxisSel*,
MC07_S_UNIT_MCM_SPEC **psUnitMcmSpec*,
MC07_S_RESULT **psResult*);

VB.NET Function MC07_USetMcmSpec2(ByVal *hUnit* As Integer, ByVal *AxisSel* As Integer,
ByRef *psUnitMcmSpec* As MC07_S_UNIT_MCM_SPEC,
ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET bool MC07_USetMcmSpec2(uint *hUnit*, uint *AxisSel*,
ref MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec*,
ref MC07_S_RESULT *psResult*);

引数

hUnit ... ユニットハンドルを指定します。

AxisSel ... 軸（複数指定可）を指定します。

複数の軸を指定する場合は、論理和を指定します

指定できる値	軸
MC07_SEL_X	X軸
MC07_SEL_Y	Y軸
MC07_SEL_Z	Z軸
MC07_SEL_A	A軸

次の指定も組み合わせることができます

指定できる値	軸	指定できる値	軸
MC07_SEL_X_Y	X軸とY軸	MC07_SEL_X_Y_Z	X軸とY軸とZ軸
MC07_SEL_X_Z	X軸とZ軸	MC07_SEL_X_Y_A	X軸とY軸とA軸
MC07_SEL_X_A	X軸とA軸	MC07_SEL_X_Z_A	X軸とZ軸とA軸
MC07_SEL_Y_Z	Y軸とZ軸	MC07_SEL_Y_Z_A	Y軸とZ軸とA軸
MC07_SEL_Y_A	Y軸とA軸	MC07_SEL_X_Y_Z_A	X軸とY軸とZ軸とA軸
MC07_SEL_Z_A	Z軸とA軸		

psUnitMcmSpec ... 設定するデータが格納されているUNIT MCM SPEC構造体のポインタを指定します。

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	SS0 SEL3	SS0 SEL2	SS0 SEL1	SS0 SEL0
D7	D6	D5	D4	D3	D2	D1	D0
SIGNAL2 SEL				SIGNAL1 SEL			

電源投入後の初期値は次の通りです。

- 2C-7760、2CD-7720、2CD-7721、2CD-7730
 - ・ X軸: H' 0000 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, SS0 SEL = 0)
 - ・ Y軸: H' 0001 (SIGNAL1 SEL = 1, SIGNAL2 SEL = 0, SS0 SEL = 0)
 - ・ Z軸: H' 0400 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, SS0 SEL = 4)
 - ・ A軸: H' 0400 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, SS0 SEL = 4)
- 2CD-7730G02
 - ・ X軸: H' 0000 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, SS0 SEL = 0)
 - ・ Y軸: H' 0001 (SIGNAL1 SEL = 1, SIGNAL2 SEL = 0, SS0 SEL = 0)
 - ・ Z軸: H' 0002 (SIGNAL1 SEL = 2, SIGNAL2 SEL = 0, SS0 SEL = 0)
 - ・ A軸: H' 0003 (SIGNAL1 SEL = 3, SIGNAL2 SEL = 0, SS0 SEL = 0)

D3--0 : SIGNAL1 SEL

D7--4 : SIGNAL2 SEL

SIGNAL1, SIGNAL2に入力する信号を選択します。

- 2C-7760、2CD-7720、2CD-7721、2CD-7730

SEL3	SEL2	SEL1	SEL0	SIGNAL1, 2に入力する信号
0	0	0	0	IN0信号
0	0	0	1	IN1信号
0	0	1	0	設定禁止 (常時ローレベル)
0	0	1	1	設定禁止 (常時ローレベル)
0	1	0	0	SIN0信号
0	1	0	1	SIN1信号
0	1	1	0	SIN2信号
0	1	1	1	SIN3信号
1	-	-	-	設定禁止 (常時ローレベル)

- ・ SIGNAL1, SIGNAL2に入力する際に負論理信号は正論理信号になります。

- 2CD-7730G02

SEL3	SEL2	SEL1	SEL0	SIGNAL1, 2に入力する信号
0	0	0	0	IN00信号
0	0	0	1	IN01信号
0	0	1	0	IN02信号
0	0	1	1	IN03信号
0	1	0	0	IN04信号
0	1	0	1	IN05信号
0	1	1	0	IN06信号
0	1	1	1	IN07信号
1	0	0	0	SIN0信号
1	0	0	1	SIN1信号
1	0	1	0	SIN2信号
1	0	1	1	SIN3信号
1	-	-	-	設定禁止 (常時ローレベル)

- ・ SIGNAL1, SIGNAL2に入力する際に負論理信号は正論理信号になります。

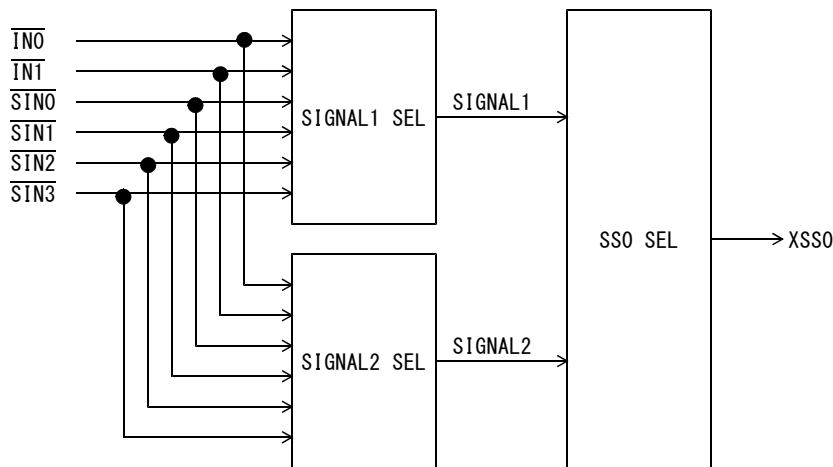
D10--8 : SS0 SEL

SS0に入力する信号を選択します。

SEL2	SEL1	SEL0	SS0に入力する信号
0	0	0	SIGNAL1のスルー出力
0	0	1	SIGNAL1 AND SIGNAL2のスルー出力
0	1	0	SIGNAL1 OR SIGNAL2のスルー出力
0	1	1	設定禁止 (常時ローレベル)
1	0	0	常時ローレベル
1	0	1	設定禁止 (常時ローレベル)
1	1	0	設定禁止 (常時ローレベル)
1	1	1	設定禁止

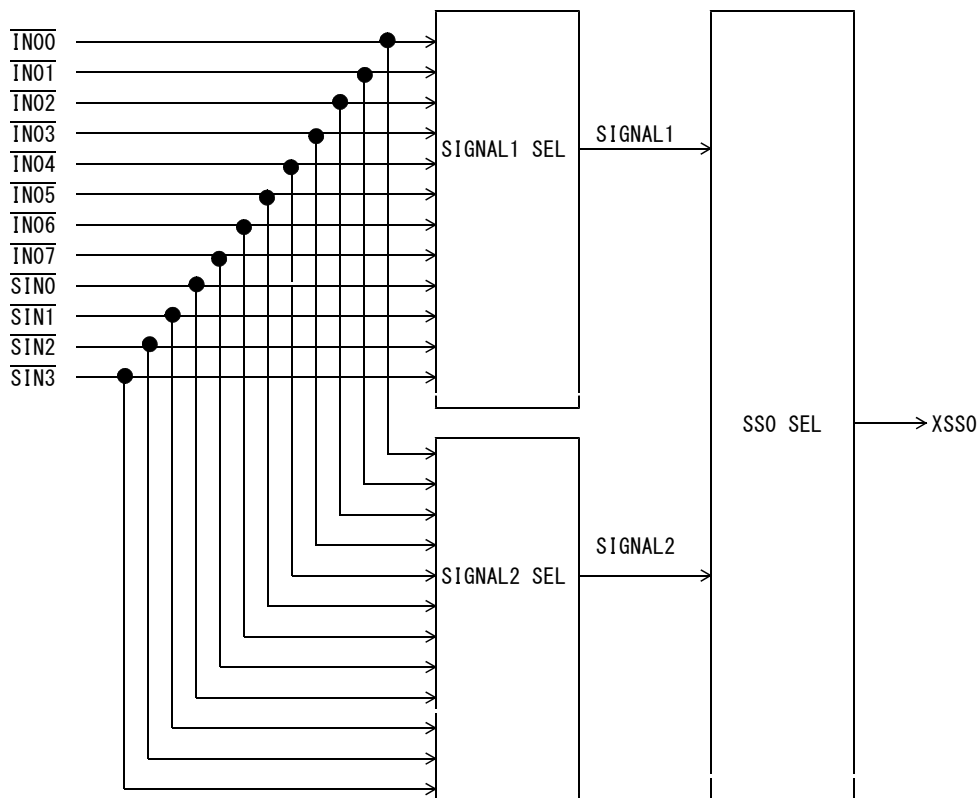
● 2C-7760、2CD-7720、2CD-7721、2CD-7730

【 停止センサ機能（X 軸の場合）】



● 2CD-7730G02

【 停止センサ機能（X 軸の場合）】



psResult ... この関数を実行した結果を格納するための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。

UNIT MCM SPEC2 GET関数

AL2A

2C-7760 2CD-7720 2CD-7721 2CD-7730 2CD-7730G02

機 能

指定されたユニットのMCM SPEC2の設定を読み出します。

書 式

C言語 `BOOL MC07_UGetMcmSpec2(DWORD hUnit, DWORD AxisSel,
 MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec,
 MC07_S_RESULT *psResult);`

VB.NET `Function MC07_UGetMcmSpec2(ByVal hUnit As Integer, ByVal AxisSel As Integer,
 ByRef psUnitMcmSpec As MC07_S_UNIT_MCM_SPEC,
 ByRef psResult As MC07_S_RESULT) As Boolean`

C#.NET `bool MC07.UGetMcmSpec2(uint hUnit, uint AxisSel,
 ref MC07_S_UNIT_MCM_SPEC psUnitMcmSpec,
 ref MC07_S_RESULT psResult);`

引 数

hUnit ... ユニットハンドルを指定します。
AxisSel ... 4軸ユニットはMC07_SEL_X_Y_Z_Aを指定します。
 2軸ユニットはMC07_SEL_X_Yを指定します。
psUnitMcmSpec ... 設定されているデータを格納するためのUNIT MCM SPEC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を格納するための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。

UNIT MCM SPEC3 SET関数

AL2A

2C-7760 2CD-7720 2CD-7721 2CD-7730 2CD-7730G02

機能

指定されたユニットにMCM SPEC3を設定します。
SOUT0信号～SOUT3信号のそれぞれに対する設定です。

この関数を実行する前に、STATUS1の全軸のERROR = 0, BUSY = 0を確認してください。

ステータス外部出力SOUTx信号に出力する信号を設定します。

書式

C言語 BOOL MC07_USetMcmSpec3(DWORD *hUnit*, DWORD *AxisSel*,
MC07_S_UNIT_MCM_SPEC **psUnitMcmSpec*,
MC07_S_RESULT **psResult*);

VB.NET Function MC07_USetMcmSpec3(ByVal *hUnit* As Integer, ByVal *AxisSel* As Integer,
ByRef *psUnitMcmSpec* As MC07_S_UNIT_MCM_SPEC,
ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET bool MC07_USetMcmSpec3(uint *hUnit*, uint *AxisSel*,
ref MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec*,
ref MC07_S_RESULT *psResult*);

引数

hUnit … ユニットハンドルを指定します。
AxisSel … 信号（複数指定可）を指定します。

複数の信号を指定する場合は、論理和を指定します

指定できる値	信号
MC07_SEL_X	SOUT0信号
MC07_SEL_Y	SOUT1信号
MC07_SEL_Z	SOUT2信号
MC07_SEL_A	SOUT3信号

psUnitMcmSpec … 設定するデータが格納されているUNIT MCM SPEC構造体のポインタを指定します。

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	SOUT SEL2	SOUT SEL1	SOUT SEL0
D7	D6	D5	D4	D3	D2	D1	D0
SIGNAL2 SEL				SIGNAL1 SEL			

電源投入後の初期値は次の通りです。

4 軸ユニット

- ・ SOUT0 : H' 0000 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, SOUT SEL = 0)
- ・ SOUT1 : H' 0001 (SIGNAL1 SEL = 1, SIGNAL2 SEL = 0, SOUT SEL = 0)
- ・ SOUT2 : H' 0002 (SIGNAL1 SEL = 2, SIGNAL2 SEL = 0, SOUT SEL = 0)
- ・ SOUT3 : H' 0003 (SIGNAL1 SEL = 3, SIGNAL2 SEL = 0, SOUT SEL = 0)

2 軸ユニット

- ・ SOUT0 : H' 0000 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, SOUT SEL = 0)
- ・ SOUT1 : H' 0001 (SIGNAL1 SEL = 1, SIGNAL2 SEL = 0, SOUT SEL = 0)
- ・ SOUT2 : H' 0000 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, SOUT SEL = 0)
- ・ SOUT3 : H' 0001 (SIGNAL1 SEL = 1, SIGNAL2 SEL = 0, SOUT SEL = 0)

D3--0 : SIGNAL1 SEL
D7--4 : SIGNAL2 SEL

SIGNAL1, SIGNAL2に出力する信号を選択します。

SEL3	SEL2	SEL1	SEL0	SIGNAL1, 2に出力する信号	
				4軸ユニット	2 軸ユニット
0	0	0	0	XOUT0	XOUT0
0	0	0	1	YOUT0	YOUT0
0	0	1	0	ZOUT0	設定禁止
0	0	1	1	AOUT0	設定禁止
0	1	0	0	XOUT1	XOUT1
0	1	0	1	YOUT1	YOUT1
0	1	1	0	ZOUT1	設定禁止
0	1	1	1	AOUT1	設定禁止
1	-	-	-	設定禁止 (常時ローレベル)	

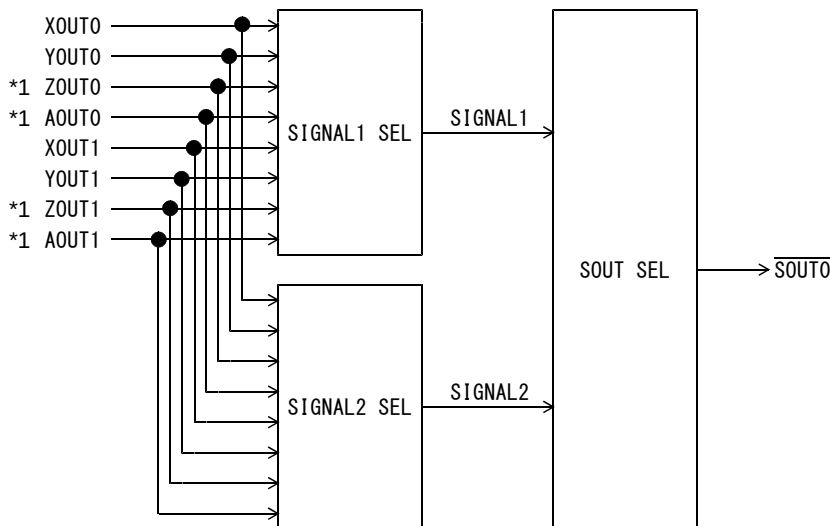
D10--8 : SOUT SEL

SOUTx信号に出力する信号を選択します。

SEL2	SEL1	SEL0	SOUTn信号に出力する信号
0	0	0	SIGNAL1のスルー出力
0	0	1	SIGNAL1 AND SIGNAL2のスルー出力
0	1	0	SIGNAL1 OR SIGNAL2のスルー出力
0	1	1	設定禁止 (常時ハイレベル)
1	0	0	設定禁止 (常時ハイレベル)
1	0	1	設定禁止 (常時ハイレベル)
1	1	0	設定禁止 (常時ハイレベル)
1	1	1	設定禁止 (常時ハイレベル)

・ $\overline{\text{SOUTx}}$ 信号に出力する際に正論理信号は負論理信号になります。

【 ステータス外部出力機能 ($\overline{\text{SOUT0}}$ 信号の場合) 】



*1 ZOUT0,AOUT0,ZOUT1,AOUT1 は、4 軸ユニットのみ選択が出来ます。

psResult ... この関数を実行した結果を格納するための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。

UNIT MCM SPEC3 GET関数

AL2A

2C-7760	2CD-7720	2CD-7721	2CD-7730	2CD-7730G02
---------	----------	----------	----------	-------------

機 能

指定されたユニットのMCM SPEC3の設定を読み出します。

書 式

C言語 `BOOL MC07_UGetMcmSpec3(DWORD hUnit, DWORD AxisSel,
 MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec,
 MC07_S_RESULT *psResult);`

VB.NET `Function MC07_UGetMcmSpec3(ByVal hUnit As Integer, ByVal AxisSel As Integer,
 ByRef psUnitMcmSpec As MC07_S_UNIT_MCM_SPEC,
 ByRef psResult As MC07_S_RESULT) As Boolean`

C#.NET `bool MC07.UGetMcmSpec3(uint hUnit, uint AxisSel,
 ref MC07_S_UNIT_MCM_SPEC psUnitMcmSpec,
 ref MC07_S_RESULT psResult);`

引 数

hUnit … ユニットハンドルを指定します。
AxisSel … MC07_SEL_X_Y_Z_Aを指定します。
psUnitMcmSpec … 設定されているデータを格納するためのUNIT MCM SPEC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を格納するための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。

UNIT MCM SPEC4 SET関数

AL2A

2CD-7730G02

機能

指定されたユニットにMCM SPEC4を設定します。
OUT00信号～OUT03信号のそれぞれに対する設定です。

この関数を実行する前に、STATUS1の全軸のERROR = 0, BUSY = 0を確認してください。

ステータス外部出力OUT0x信号に出力する信号を設定します。

書式

C言語 BOOL MC07_USetMcmSpec4(DWORD *hUnit*, DWORD *AxisSel*,
MC07_S_UNIT_MCM_SPEC **psUnitMcmSpec*,
MC07_S_RESULT **psResult*);

VB.NET Function MC07_USetMcmSpec4(ByVal *hUnit* As Integer, ByVal *AxisSel* As Integer,
ByRef *psUnitMcmSpec* As MC07_S_UNIT_MCM_SPEC,
ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET bool MC07_USetMcmSpec4(uint *hUnit*, uint *AxisSel*,
ref MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec*,
ref MC07_S_RESULT *psResult*);

引数

hUnit … ユニットハンドルを指定します。
AxisSel … 信号（複数指定可）を指定します。

複数の信号を指定する場合は、論理和を指定します

指定できる値	信号
MC07_SEL_X	OUT00信号
MC07_SEL_Y	OUT01信号
MC07_SEL_Z	OUT02信号
MC07_SEL_A	OUT03信号

psUnitMcmSpec … 設定するデータが格納されているUNIT MCM SPEC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を格納するためのRESULT構造体のポインタを指定します。

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	OUT SEL2	OUT SEL1	OUT SEL0
D7	D6	D5	D4	D3	D2	D1	D0
SIGNAL2 SEL				SIGNAL1 SEL			

電源投入後の初期値は次の通りです。

- ・ $\overline{\text{OUT00}}$: H'0000 (SIGNAL1 SEL = 0, SIGNAL2 SEL = 0, OUT SEL = 0)
- ・ $\overline{\text{OUT01}}$: H'0001 (SIGNAL1 SEL = 1, SIGNAL2 SEL = 0, OUT SEL = 0)
- ・ $\overline{\text{OUT02}}$: H'0002 (SIGNAL1 SEL = 2, SIGNAL2 SEL = 0, OUT SEL = 0)
- ・ $\overline{\text{OUT03}}$: H'0003 (SIGNAL1 SEL = 3, SIGNAL2 SEL = 0, OUT SEL = 0)

D3--0 : SIGNAL1 SEL

D7--4 : SIGNAL2 SEL

SIGNAL1, SIGNAL2に出力する信号を選択します。

SEL3	SEL2	SEL1	SEL0	SIGNAL1, 2に出力する信号
0	0	0	0	XOUT0
0	0	0	1	YOUT0
0	0	1	0	ZOUT0
0	0	1	1	AOUT0
0	1	0	0	XOUT1
0	1	0	1	YOUT1
0	1	1	0	ZOUT1
0	1	1	1	AOUT1
1	-	-	-	設定禁止 (常時ローレベル)

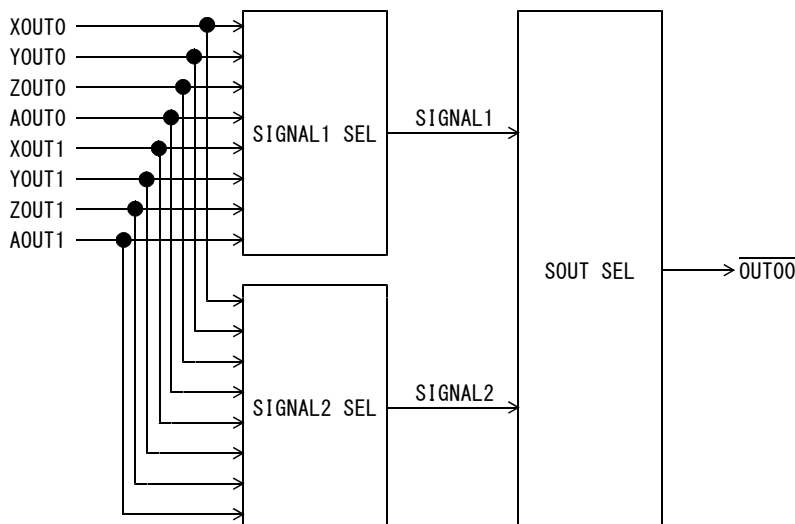
D10--8 : OUT SEL

$\overline{\text{OUT0x}}$ 信号に出力する信号を選択します。

SEL2	SEL1	SEL0	$\overline{\text{OUT0x}}$ 信号に出力する信号
0	0	0	SIGNAL1のスルー出力
0	0	1	SIGNAL1 AND SIGNAL2のスルー出力
0	1	0	SIGNAL1 OR SIGNAL2のスルー出力
0	1	1	設定禁止 (常時ハイレベル)
1	0	0	設定禁止 (常時ハイレベル)
1	0	1	設定禁止 (常時ハイレベル)
1	1	0	設定禁止 (常時ハイレベル)
1	1	1	設定禁止 (常時ハイレベル)

- ・ $\overline{\text{OUT0x}}$ 信号に出力する際に正論理信号は負論理信号になります。

【ステータス外部出力機能 ($\overline{\text{OUT00}}$ 信号の場合)】



psResult ... この関数を実行した結果を格納するための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。

UNIT MCM SPEC4 GET関数

AL2A

2CD-7730G02

機 能

指定されたユニットのMCM SPEC4の設定を読み出します。

書 式

C言語 `BOOL MC07_UGetMcmSpec4(DWORD hUnit, DWORD AxisSel,
 MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec,
 MC07_S_RESULT *psResult);`

VB.NET `Function MC07_UGetMcmSpec4(ByVal hUnit As Integer, ByVal AxisSel As Integer,
 ByRef psUnitMcmSpec As MC07_S_UNIT_MCM_SPEC,
 ByRef psResult As MC07_S_RESULT) As Boolean`

C#.NET `bool MC07.UGetMcmSpec4(uint hUnit, uint AxisSel,
 ref MC07_S_UNIT_MCM_SPEC psUnitMcmSpec,
 ref MC07_S_RESULT psResult);`

引 数

hUnit … ユニットハンドルを指定します。
AxisSel … MC07_SEL_X_Y_Z_Aを指定します。
psUnitMcmSpec … 設定されているデータを格納するためのUNIT MCM SPEC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を格納するための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。

UNIT MCM SPEC5 SET関数

AL2A

2CD-7730G02

機能

指定されたユニットにMCM SPEC5を設定します。
OUT04信号～OUT07信号のそれぞれに対する設定です。

この関数を実行する前に、STATUS1の全軸のERROR = 0, BUSY = 0を確認してください。

ステータス外部出力OUT0x信号に出力する信号を設定します。

書式

C言語 BOOL MC07_USetMcmSpec5(DWORD *hUnit*, DWORD *AxisSel*,
MC07_S_UNIT_MCM_SPEC **psUnitMcmSpec*,
MC07_S_RESULT **psResult*);

VB.NET Function MC07_USetMcmSpec5(ByVal *hUnit* As Integer, ByVal *AxisSel* As Integer,
ByRef *psUnitMcmSpec* As MC07_S_UNIT_MCM_SPEC,
ByRef *psResult* As MC07_S_RESULT) As Boolean

C#.NET bool MC07_USetMcmSpec5(uint *hUnit*, uint *AxisSel*,
ref MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec*,
ref MC07_S_RESULT *psResult*);

引数

hUnit … ユニットハンドルを指定します。
AxisSel … 信号（複数指定可）を指定します。

複数の信号を指定する場合は、論理和を指定します

指定できる値	信号
MC07_SEL_X	OUT04信号
MC07_SEL_Y	OUT05信号
MC07_SEL_Z	OUT06信号
MC07_SEL_A	OUT07信号

psUnitMcmSpec … 設定するデータが格納されているUNIT MCM SPEC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を格納するためのRESULT構造体のポインタを指定します。

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	OUT SEL2	OUT SEL1	OUT SEL0
D7	D6	D5	D4	D3	D2	D1	D0
SIGNAL2 SEL				SIGNAL1 SEL			

電源投入後の初期値は次の通りです。

- ・ $\overline{\text{OUT04}}$: H'0004 (SIGNAL1 SEL = 4, SIGNAL2 SEL = 0, OUT SEL = 0)
- ・ $\overline{\text{OUT05}}$: H'0005 (SIGNAL1 SEL = 5, SIGNAL2 SEL = 0, OUT SEL = 0)
- ・ $\overline{\text{OUT06}}$: H'0006 (SIGNAL1 SEL = 6, SIGNAL2 SEL = 0, OUT SEL = 0)
- ・ $\overline{\text{OUT07}}$: H'0007 (SIGNAL1 SEL = 7, SIGNAL2 SEL = 0, OUT SEL = 0)

D3--0 : SIGNAL1 SEL

D7--4 : SIGNAL2 SEL

SIGNAL1, SIGNAL2に出力する信号を選択します。

SEL3	SEL2	SEL1	SEL0	SIGNAL1, 2に出力する信号
0	0	0	0	XOUT0
0	0	0	1	YOUT0
0	0	1	0	ZOUT0
0	0	1	1	AOUT0
0	1	0	0	XOUT1
0	1	0	1	YOUT1
0	1	1	0	ZOUT1
0	1	1	1	AOUT1
1	-	-	-	設定禁止 (常時ローレベル)

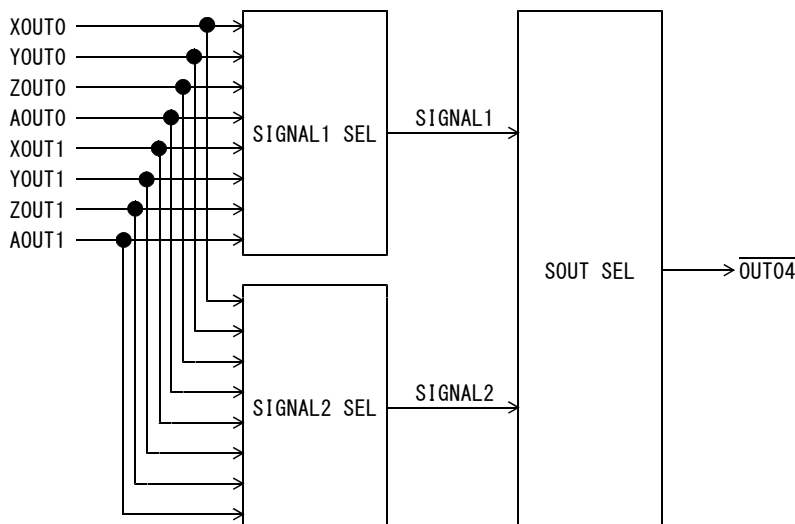
D10--8 : OUT SEL

$\overline{\text{OUT0x}}$ 信号に出力する信号を選択します。

SEL2	SEL1	SEL0	$\overline{\text{OUT0x}}$ 信号に出力する信号
0	0	0	SIGNAL1のスルー出力
0	0	1	SIGNAL1 AND SIGNAL2のスルー出力
0	1	0	SIGNAL1 OR SIGNAL2のスルー出力
0	1	1	設定禁止 (常時ハイレベル)
1	0	0	設定禁止 (常時ハイレベル)
1	0	1	設定禁止 (常時ハイレベル)
1	1	0	設定禁止 (常時ハイレベル)
1	1	1	設定禁止 (常時ハイレベル)

- ・ $\overline{\text{OUT0x}}$ 信号に出力する際に正論理信号は負論理信号になります。

【ステータス外部出力機能 ($\overline{\text{OUT04}}$ 信号の場合)】



psResult ... この関数を実行した結果を格納するための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。

UNIT MCM SPEC5 GET関数

AL2A

2CD-7730G02

機 能

指定されたユニットのMCM SPEC5の設定を読み出します。

書 式

C言語 `BOOL MC07_UGetMcmSpec5(DWORD hUnit, DWORD AxisSel,
 MC07_S_UNIT_MCM_SPEC *psUnitMcmSpec,
 MC07_S_RESULT *psResult);`

VB.NET `Function MC07_UGetMcmSpec5(ByVal hUnit As Integer, ByVal AxisSel As Integer,
 ByRef psUnitMcmSpec As MC07_S_UNIT_MCM_SPEC,
 ByRef psResult As MC07_S_RESULT) As Boolean`

C#.NET `bool MC07.UGetMcmSpec5(uint hUnit, uint AxisSel,
 ref MC07_S_UNIT_MCM_SPEC psUnitMcmSpec,
 ref MC07_S_RESULT psResult);`

引 数

hUnit ... ユニットハンドルを指定します。
AxisSel ... MC07_SEL_X_Y_Z_Aを指定します。
psUnitMcmSpec ... 設定されているデータを格納するためのUNIT MCM SPEC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を格納するための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。

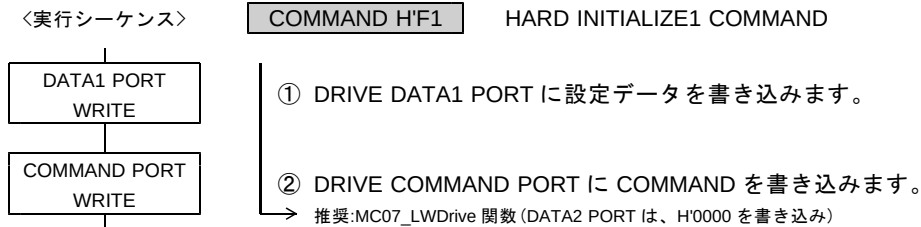
3. コマンド仕様

3-1. ドライブコマンド

3-1-1. 入出力仕様の設定

(1) HARD INITIALIZE1

外部ステータス信号 (SOUT 信号) に出力する機能、および MCM (RAM) 動作の実行タイミング条件を設定します。
このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
OUT3	OUT3	OUT3	OUT3	OUT2	OUT2	OUT2	OUT2
TYPE3	TYPE2	TYPE1	TYPE0	TYPE3	TYPE2	TYPE1	TYPE0

D7	D6	D5	D4	D3	D2	D1	D0
OUT1	OUT1	OUT1	OUT1	OUT0	OUT0	OUT0	OUT0
TYPE3	TYPE2	TYPE1	TYPE0	TYPE3	TYPE2	TYPE1	TYPE0

● 電源投入後の初期値は H'EE31 (アンダーライン側) です。

D3--D0 : OUT0 TYPE3--0 初期値 = CNTINT (外部信号出力可能)
 D7--D4 : OUT1 TYPE3--0 初期値 = RDYINT (外部信号出力可能)
 D11--D8 : OUT2 TYPE3--0 初期値 = 汎用出力 (ステータスを内部フラグに利用可能)
 D15--D12 : OUT3 TYPE3--0 初期値 = 汎用出力 (ステータスを内部フラグに利用可能)

OUT0, 1 信号および OUT2, 3 信号の出力機能を選択します。

TYPE3	TYPE2	TYPE1	TYPE0	OUT0,1,2,3 の出力機能	
0	0	0	0	ADRINT	カウンタ割り込み要求の ADRINT 出力
0	0	0	1	CNTINT	カウンタ割り込み要求の CNTINT 出力
0	0	1	0	DFLINT	カウンタ割り込み要求の DFLINT 出力
0	0	1	1	RDYINT	コマンド終了割り込み要求の RDYINT 出力
0	1	0	0	STBY	STATUS1 の STBY フラグ出力
0	1	0	1	DRIVE	STATUS1 の DRIVE フラグ出力
0	1	1	0	nSPEED FL	STATUS5 の SPEED FL フラグの反転出力
0	1	1	1	nINDEX FL	STATUS5 の INDEX FL フラグの反転出力
1	0	0	0	UP	STATUS1 の UP フラグ出力
1	0	0	1	DOWN	STATUS1 の DOWN フラグ出力
1	0	1	0	CONST	STATUS1 の CONST フラグ出力
1	0	1	1	EXT PULSE	STATUS1 の EXT PULSE フラグ出力
1	1	0	0	nPULSE MASK	STATUS2 の PULSE MASK フラグの反転出力
1	1	0	1	ORG SIGNAL	STATUS2 の ORG SIGNAL フラグ出力
1	1	1	0	汎用出力	汎用出力として使用する
1	1	1	1	設定禁止	—

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

SIGNAL I/O コネクタから外部出力できる信号は、初期設定は下記の割り付けになっています。
 UNIT MCM SPEC3 SET 関数で任意軸や OUT1 信号の割り付けも可能です。

外部出力信号	軸と出力(割付信号名)	
	4 軸ユニット	2 軸ユニット
SOUT0 信号	XOUT0 (XCNTINT)	XOUT0 (XCNTINT)
SOUT1 信号	YOUT0 (YCNTINT)	YOUT0 (YCNTINT)
SOUT2 信号	ZOUT0 (ZCNTINT)	XOUT0 (XCNTINT)
SOUT3 信号	AOUT0 (ACNTINT)	YOUT0 (YCNTINT)

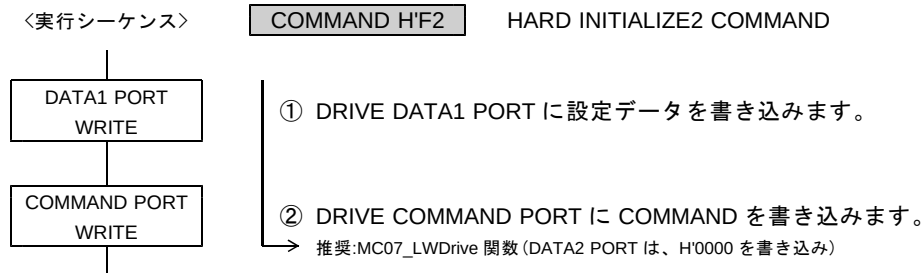
OUT 2,3 信号は、ステータスを利用して、次の機能の条件に応用することができます。

- ・ PAUSE 機能の STBY 解除条件
- ・ SPEED RATE CHANGE の変更動作点
- ・ INDEX CHANGE の変更動作点
- ・ パルス偏差カウンタのパルスカウント開始タイミング
- ・ 各カウンタのラッチタイミング
- ・ RAM 機能の JUMP 条件
- ・ RAM 機能のメモリーデータの NO OPERATION コマンド実行条件

(2) HARD INITIALIZE2

GPIO 2 信号入出力の入出力機能を設定します。

このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。



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
GPIO2 TYPE3	GPIO2 TYPE2	GPIO2 TYPE1	GPIO2 TYPE0	1	1	1	1

● 電源投入後の初期値は H'00FF (アンダーライン側) です。

D3--D0 : 1 にしてください

D7--D4 : GPIO2 TYPE3--0 初期値 = 汎用入力 (ステータスを内部フラグに利用可能)

GPIO 2 の入出力機能を選択します。

TYPE3	TYPE2	TYPE1	TYPE0	GPIO2 の入出力機能	
0	0	0	0	ADRINT COMP1	STATUS4 の ADRINT COMP1 フラグ出力
0	0	0	1	ADRINT COMP2	STATUS4 の ADRINT COMP2 フラグ出力
0	0	1	0	ADRINT COMP3	STATUS4 の ADRINT COMP3 フラグ出力
0	0	1	1	XADRINT AND YADRINT	X 軸と Y 軸の ADRINT の AND (論理積) 出力
0	1	0	0	XADRINT OR YADRINT	X 軸と Y 軸の ADRINT の OR (論理和) 出力
0	1	0	1	DFLINT COMP1	STATUS4 の DFLINT COMP1 フラグ出力
0	1	1	0	DFLINT COMP2	STATUS4 の DFLINT COMP2 フラグ出力
0	1	1	1	DFLINT COMP3	STATUS4 の DFLINT COMP3 フラグ出力
1	0	0	0	ERROR	STATUS1 の ERROR フラグ出力
1	0	0	1	LSEND	STATUS1 の LSEND フラグ出力
1	0	1	0	SSEND	STATUS1 の SSEND フラグ出力
1	0	1	1	FSEND	STATUS1 の FSEND フラグ出力
1	1	0	0	SS0	STATUS5 の SS0 フラグ出力
1	1	0	1	DRVEND	STATUS1 の DRVEND フラグ出力
1	1	1	0	汎用出力	汎用出力として使用する
1	1	1	1	汎用入力	汎用入力として使用する

・「汎用出力」に設定した場合は、SIGNAL OUT コマンドで出力レベルが操作できます。

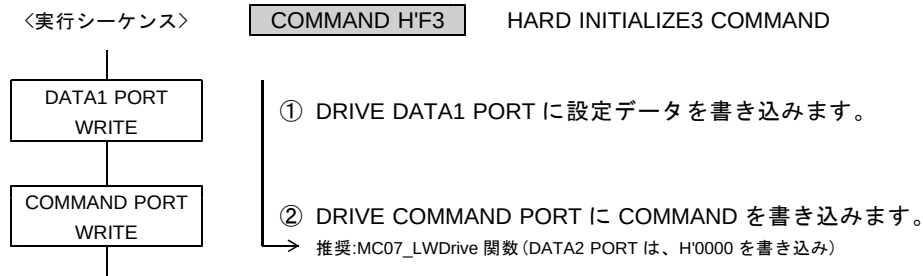
GPIO 2 信号のステータスを利用して、次の機能の条件に応用することができます。

- ・ PAUSE 機能の STBY 解除条件
- ・ SPEED RATE CHANGE の変更動作点
- ・ INDEX CHANGE の変更動作点
- ・ 各カウンタのラッチタイミング
- ・ RAM 機能の JUMP 条件
- ・ RAM 機能のメモリーデータの NO OPERATION コマンド実行条件

(3) HARD INITIALIZE3

GPIO 3 信号の入力機能を設定します。

このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。



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
GPIO3 TYPE3	GPIO3 TYPE2	GPIO3 TYPE1	GPIO3 TYPE0	1	1	1	1

- 電源投入後の初期値は H'FF (アンダーライン側) です。

D3--D0 : 1 にしてください

D7--D4 : GPIO3 TYPE3--0 初期値 = 汎用入力 (ステータスを内部フラグに利用可能)

GPIO 3 の入出力機能を選択します。

TYPE3	TYPE2	TYPE1	TYPE0	GPIO 3 の入出力機能	
0	0	0	0	CNTINT COMP1	STATUS4 の CNTINT COMP1 フラグ出力
0	0	0	1	CNTINT COMP2	STATUS4 の CNTINT COMP2 フラグ出力
0	0	1	0	CNTINT COMP3	STATUS4 の CNTINT COMP3 フラグ出力
0	0	1	1	XCNTINT AND YCNTINT	X 軸と Y 軸の CNTINT の AND (論理積) 出力
0	1	0	0	XCNTINT OR YCNTINT	X 軸と Y 軸の CNTINT の OR (論理和) 出力
0	1	0	1	DFLINT COMP1	STATUS4 の DFLINT COMP1 フラグ出力
0	1	1	0	DFLINT COMP2	STATUS4 の DFLINT COMP2 フラグ出力
0	1	1	1	DFLINT COMP3	STATUS4 の DFLINT COMP3 フラグ出力
1	0	0	0	ERRINT	ERROR STATUS 出力の ERRINT 出力
1	0	0	1	ORGEND	STATUS2 の ORGEND フラグ出力
1	0	1	0	COMREG EP	STATUS1 の COMREG EP フラグ出力
1	0	1	1	COMREG FL	STATUS1 の COMREG FL フラグ出力
1	1	0	0	SS1	STATUS5 の SS1 フラグ出力
1	1	0	1	nBUSY	STATUS1 の BUSY フラグの反転出力
1	1	1	0	汎用出力	汎用出力として使用する
1	1	1	1	汎用入力	汎用入力として使用する

「汎用出力」に設定した場合は、SIGNAL OUT コマンドで出力レベルが操作できます。

GPIO 3 信号のステータスを利用して、次の機能の条件に応用することができます。

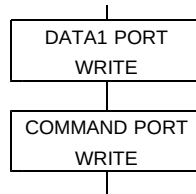
- ・ RAM 機能の JUMP 条件
- ・ RAM 機能のメモリーデータの NO OPERATION コマンド実行条件

(4) HARD INITIALIZE7

各入力信号のアクティブ論理を設定します。
このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。

注意 設定を誤ると製品が正しく機能しなくなります。
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(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。

＜実行シーケンス＞



COMMAND H'F7

HARD INITIALIZE7 COMMAND

- ① DRIVE DATA1 PORT にアクティブデータを書き込みます。
- ② DRIVE COMMAND PORT に COMMAND を書き込みます。
→ 推奨:MC07_LWDrive 関数 (DATA2 PORT は、H'0000 を書き込み)

DRIVE DATA1 PORT の設定データ

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

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

- 電源投入後の初期値は H'F7FE (すべてハイアクティブ) です。
上記の値で、入力アクティブ論理は以下の初期値になっています。

D1 : FSSTOP ACTIVE

FSSTOP 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)

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)

上記の信号は、初期値 B 接点入力になっています。

D9 : 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 (ハイクティブ)

1 : 負論理 (ローアクティブ)

D15 : PAUSE ACTIVE

PAUSE 信号のアクティブ論理を選択します。

0 : PAUSE=1 にして STBY にする

1 : PAUSE=0 にして STBY を解除する

- ・ D15 の PAUSE 信号のアクティブ論理を操作することで、コマンド予約機能による連続ドライブの設定と実行ができます。
- ・ 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、変更した信号の製品内部デジタルフィルタ時定数経過後に、アクティブ論理の変更が確定します。

(5) HARD INITIALIZE8

パルス出力信号のアクティブ論理を設定します。
このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。

**注意**

設定を誤ると製品が正しく機能しなくなります。
出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(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
CCWP ACTIVE (0)	CWP ACTIVE (0)	—	—	1	1	1	1

D7	D6	D5	D4	D3	D2	D1	D0
—	—	1	1	1	1	1	1

● 電源投入後の初期値は H'0F3F です。

D15 : CCWP ACTIVE
D14 : CWP ACTIVE

パルス出力信号のアクティブ論理を選択します。
0 : ローアクティブ
1 : ハイアクティブ

CWP, CCWP 信号の電源投入後の初期状態は、「ローアクティブ（負論理出力）」です。
この出力論理の切替は、方向指定型の場合にも有効です。

コントローラドライバ製品では、当コマンドは実行禁止です。

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

- 電源投入後の初期値は H'00 1388 (5,000 Hz : 1 周期 200 μ s) です。

*FSPD は、設定された値の半周期を経過してから第 1 パルス目のパルス出力を開始します。
初期値 200 μ s は、この周期が実際に第 1 パルス目の出力(立ち上がりエッジ)までの遅延時間になります。
よって、起動指令後に速くパルス出力を開始したいときは、ドライバ側の入力応答周波数に注意して、この FSPD の値を設定してください。

FSPD の設定値が "0" の場合は、FSPD を $FSPD = RSPD \times RESOL$ に補正します。

- ・ RSPD : RSPD は、HSPD, LSPD, ELSPD と同様の 15 ビットのパルス速度データです。
DRIVE = 1 $\rightarrow$ 0 になると、最終出力のパルス速度データを RSPD に記憶します。
ただし、最終出力のパルス速度が FSPD, RFSPD と JSPD の場合は、RSPD を書き換えません。
RSPD のリセット後の初期値は、H'012C (300) です。

- ・ 第 1 パルスのパルス周期の計算式

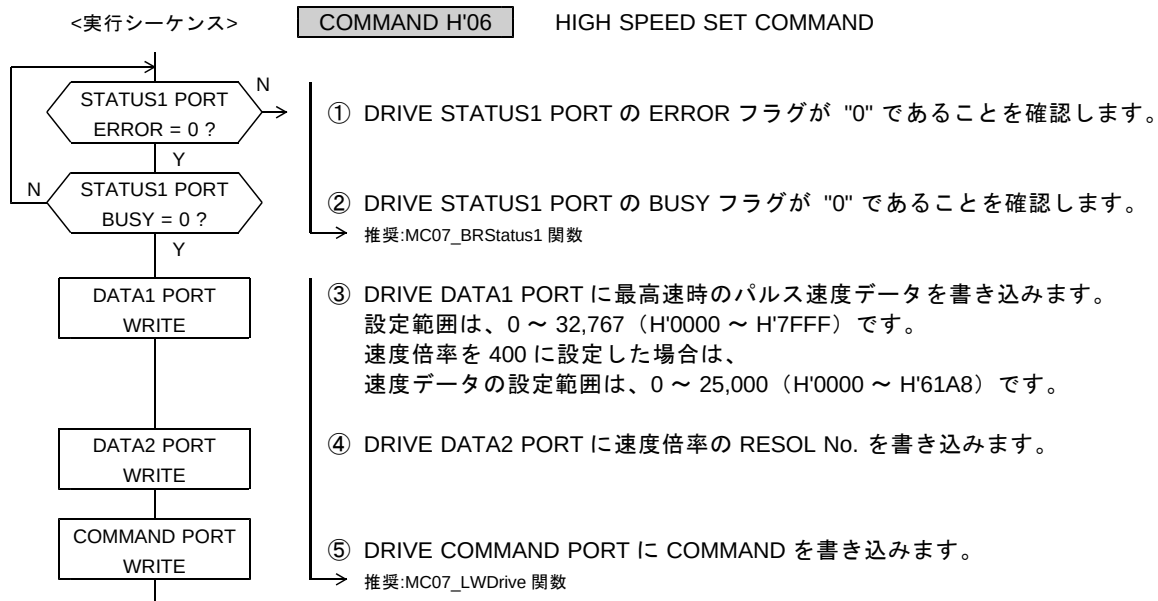
$10,000,000 / FSPD = \text{商 A} + \text{余り B}$	$\rightarrow$ OFF 周期	= 商 A x 50 ns
$(10,000,000 + \text{余り B}) / FSPD = \text{商 C} + \text{余り D}$	$\rightarrow$ ON 周期	= 商 C x 50 ns

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	$\rightarrow$ OFF 周期 = 50 ns	ON 周期 = 50 ns (10,000,000 Hz)
6,666,666 ~ 5,000,001 Hz	$\rightarrow$ OFF 周期 = 50 ns	ON 周期 = 100 ns (6,666,666 Hz)
5,000,000 ~ 4,000,001 Hz	$\rightarrow$ OFF 周期 = 100 ns	ON 周期 = 100 ns (5,000,000 Hz)
4,000,000 ~ 3,333,334 Hz	$\rightarrow$ OFF 周期 = 100 ns	ON 周期 = 150 ns (4,000,000 Hz)
3,333,333 ~ 2,857,143 Hz	$\rightarrow$ 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）を設定します。
加減速ドライブの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

- 電源投入後の初期値は 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
—	—	—	—	—	—	—	—	—	—	—	—	D3	RESOL No.		D0

- 電源投入後の初期値は H'3 (速度倍率 = 1) です。

HSPD の設定値が "0" の場合は、HSPD を HSPD = RSPD 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。

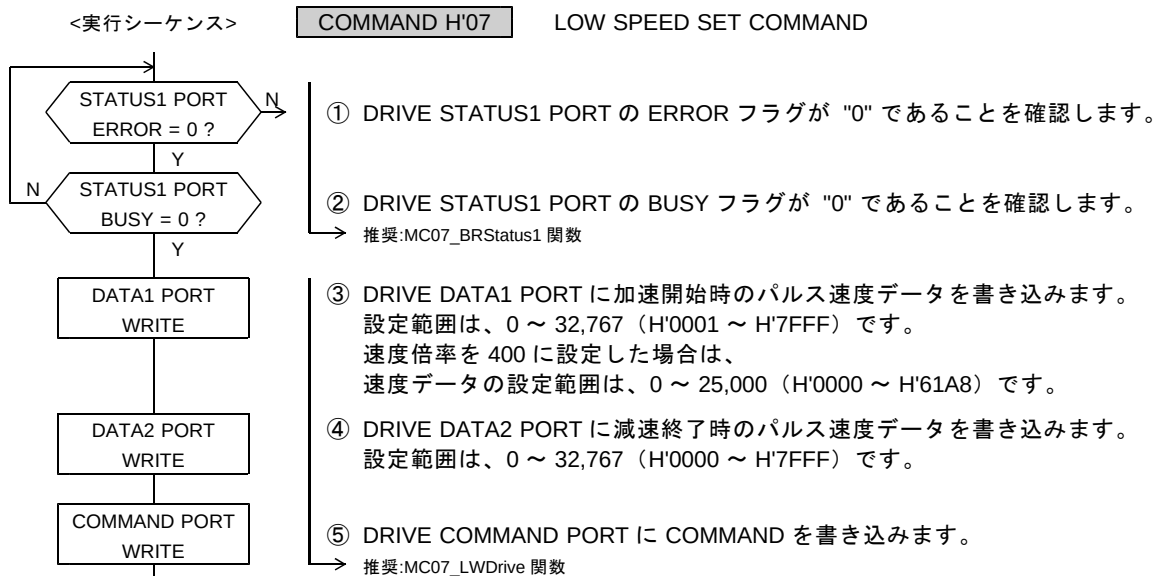
RESOL No.を選択して、速度倍率（RESOL）を設定します。

RESOL No.	速度倍率 (RESOL)	RESOL No.	速度倍率 (RESOL)	RESOL No.	速度倍率 (RESOL)	RESOL No.	速度倍率 (RESOL)
H'0	1	H'4	2	H'8	50	H'C	400
H'1	1	H'5	5	H'9	100	H'D	400
H'2	1	H'6	10	H'A	200	H'E	400
H'3	1	H'7	20	H'B	400	H'F	400

- ・ 速度設定値 = 速度データ x 速度倍率 (RESOL) : 1 ～ 10,000,000 Hz
- ・ 速度倍率を 400 に設定した場合は、速度データの設定範囲は、0 ～ 25,000 (H'0000 ～ H'61A8) です。

(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) を設定します。
加減速ドライブの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

- 電源投入後の初期値は 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

- 電源投入後の初期値は H'0000 (LSPD と同じ) です。

LSPD の設定値が "0" の場合は、LSPD を $LSPD = RSPD$ に補正します。

- ・加速開始時の速度 (Hz) = $LSPD \times RESOL$

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) = $ELSPD \times RESOL$

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。

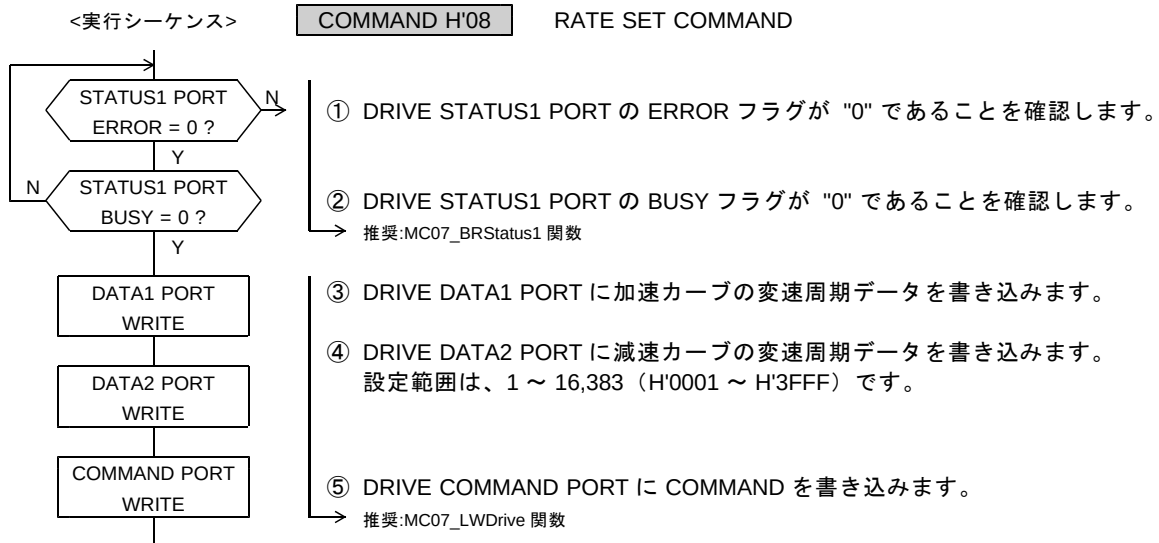
2 パルス目から $LSPD \times RESOL$ の速度になります。

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 $HSPD \times RESOL$ 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) を設定します。
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

- 電源投入後の初期値は 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

- 電源投入後の初期値は H'0000 (UCYCLE と同じ) 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、DCYCLE を DCYCLE = UCYCLE 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) を設定します。

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
—	D14	← SUAREA →													D0

- 電源投入後の初期値は 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
—	D14	← SDAREA →													D0

- 電源投入後の初期値は H'0000 (0 : SDAREA の変速領域なし) です。

SUAREA (および SUH) 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。

- ・ 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 (または SUH) × RESOL

SDAREA (および SDH) 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。

- ・ S 字減速開始部の変速領域 (Hz) = SDAREA (または SDH) × RESOL
- ・ S 字減速終了部の変速領域 (Hz) = SDAREA × RESOL

- SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合

変速領域データは以下のように補正します。

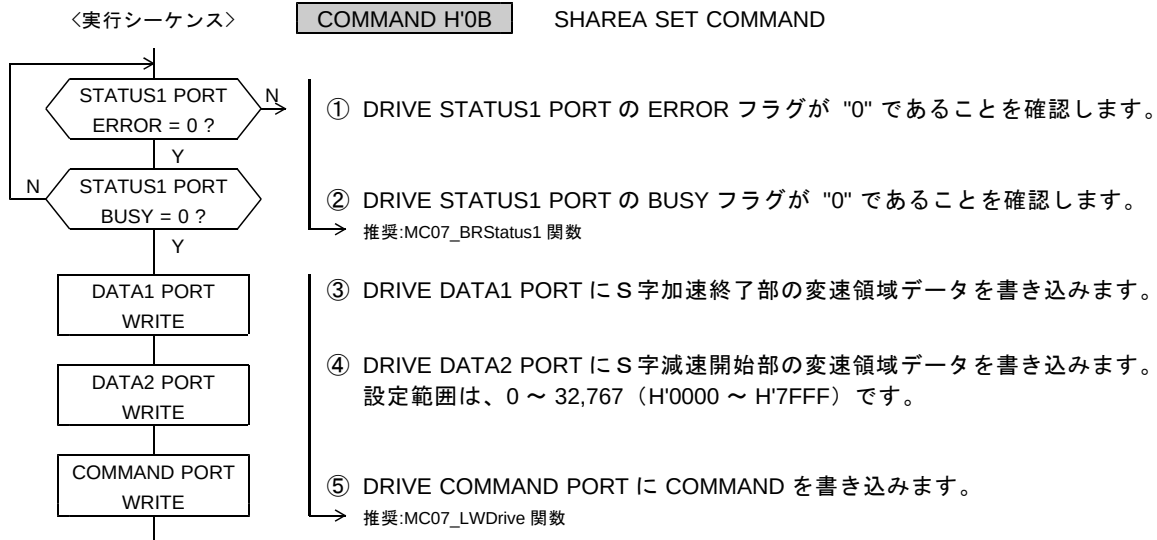
- ・ (HSPD - LSPD) < (2 * SUAREA) のときは、重複した変速領域を重ねて加速します。
- ・ (HSPD - ELSPD) < (2 * SDAREA) のときは、SDAREA = (HSPD - ELSPD) * 1/2 に補正します。

(6) SHAREA SET

SPEC INITIALIZE3 コマンドの SCAREA MODE = 1 に設定している場合に有効です。

加速カーブの S 字加速終了部の変速領域データ (SUH) を設定します。

減速カーブの S 字減速開始部の変速領域データ (SDH) 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	SUH													D0

- 電源投入後の初期値は H'0000 (0 : SUH 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
—	D14	SDH													D0

- 電源投入後の初期値は H'0000 (0 : SDH の変速領域なし) です。

- SPEC INITIALIZE3 コマンドの SCAREA MODE = 1 の場合

SUAREA および SUH 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。

- ・ S 字加速開始部の変速領域 (Hz) = SUAREA x RESOL
- ・ S 字加速終了部の変速領域 (Hz) = SUH x RESOL

SDAREA および SDH 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。

- ・ S 字減速開始部の変速領域 (Hz) = SDH x RESOL
- ・ S 字減速終了部の変速領域 (Hz) = SDAREA x RESOL

変速領域データは以下のように補正します。

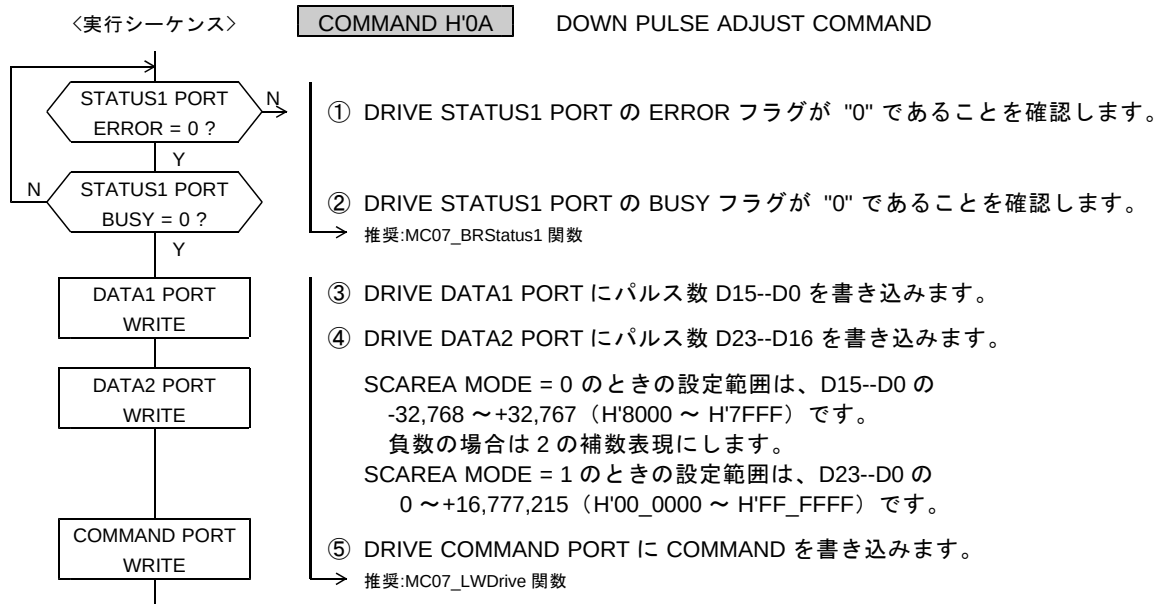
- ・ (HSPD - LSPD) < (SUAREA) のときは、SUAREA = (HSPD - LSPD) に補正します。
- ・ (HSPD - ELSPD) < (SDAREA) のときは、SDAREA = (HSPD - ELSPD) に補正します。
- ・ (HSPD - LSPD) < (SUH + SUAREA) のときは、SUH = (HSPD - LSPD - SUAREA) に補正します。
- ・ (HSPD - ELSPD) < (SDH + SDAREA) のときは、SDH = (HSPD - ELSPD - SDAREA) に補正します。
- ・ HSPD < LSPD or HSPD < ELSPD のときは、SUAREA = SDAREA = SUH = SDH = 0 に補正します。

(7) DOWN PULSE ADJUST

INDEX ドライブの停止位置への減速停止動作時に有効です。

MCC09 が自動検出する減速パルス数に加算するオフセットパルス数を設定します。

または、停止位置への減速停止動作を開始するための減速パルス数をマニュアル設定します。



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／減速パルス数 → 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
—	—	—	—	—	—	—	—	D23	← 減速パルス数 →						

- 電源投入後の初期値は H'00 0001 (+1 パルス) です。

このコマンドは、SPEC INITIALIZE3 コマンドの SCAREA MODE の選択により、設定データの機能が変わります。

- SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合

DRIVE DATA1 PORT の D15--D0 のみ有効です。

INDEX ドライブのオフセットパルス数を設定します。

設定したオフセットパルス数は、MCC09 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。

- SPEC INITIALIZE3 コマンドの SCAREA MODE = 1 の場合

DRIVE DATA1, 2 PORT の D23--D0 が有効です。

INDEX ドライブの減速停止動作を開始するための減速パルス数をマニュアル設定します。

設定した減速パルス数と INDEX ドライブの残アドレス（残パルス数）が同じパルス数になると、INDEX ドライブの停止位置への減速停止動作を開始します。

- ・ 減速パルス数の計算式は、4-1-3.章「S 字加減速ドライブ」に記載しています。

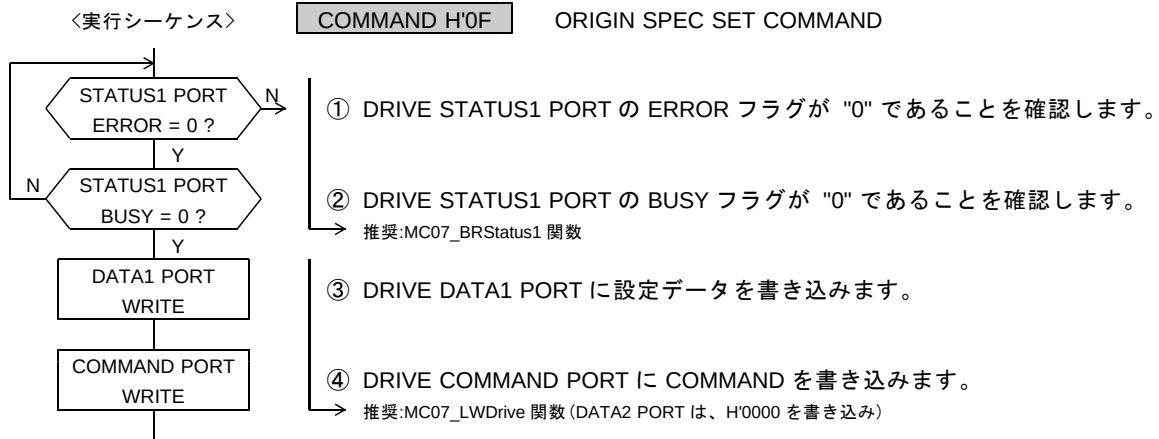
3-1-3. ORIGIN ドライブの設定

ORIGIN ドライブの動作仕様を設定します。

※ 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。

(1) ORIGIN SPEC SET

ORIGIN 停止機能を設定します。



DRIVE DATA1 PORT の設定データ

D15	D14	D13	D12	D11	D10	D9	D8
ORG DIVISION D7	ORG DIVISION D6	ORG DIVISION D5	ORG DIVISION D4	ORG DIVISION D3	ORG DIVISION D2	ORG DIVISION D1	ORG DIVISION D0

D7	D6	D5	D4	D3	D2	D1	D0
AUTO DRST ENABLE	ORG STOP TYPE1	ORG STOP TYPE0	ORG DETECT EDGE	ORG SIGNAL TYPE3	ORG SIGNAL TYPE2	ORG SIGNAL TYPE1	ORG SIGNAL TYPE0

- 電源投入後の初期値は H'0003 (アンダーライン側) です。

D3--0 : ORG SIGNAL TYPE3--0

ORIGIN 停止機能で検出する信号 (ORG 検出信号) を選択します。

TYPE3	TYPE2	TYPE1	TYPE0	ORG 検出信号	
0	0	0	0	ORG 信号	
0	0	0	1	ZPO 信号	
0	0	1	0	ORG 信号と ZPO 信号の AND (論理積)	*
<u>0</u>	<u>0</u>	<u>1</u>	<u>1</u>	<u>ORG 信号と ZPO 信号の OR (論理和)</u>	*

0	1	0	0	ORG 信号	
0	1	0	1	DEND 信号	
0	1	1	0	ORG 信号と DEND 信号の AND (論理積)	*
0	1	1	1	ORG 信号と DEND 信号の OR (論理和)	*

1	0	0	0	NORG 信号	
1	0	0	1	設定禁止	
1	0	1	0	SS0	
1	0	1	1	設定禁止	

1	1	0	0	設定禁止	
1	1	0	1	設定禁止	

1	1	1	0	設定禁止	
1	1	1	1	設定禁止	

* : 各信号のアクティブレベルを論理合成して、ORG 検出信号にします。

DEND 信号を ORG 検出信号に選択した場合は、
各信号の入力機能と ORIGIN 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 : ORG STOP TYPE0

D6 : ORG STOP TYPE1

検出エッジの検出条件と ORG エッジ信号の停止機能を選択します。

TYPE1	TYPE0	検出エッジの検出条件	ORG エッジ信号の停止機能	備考
<u>0</u>	<u>0</u>	常時検出する	停止しない	—
0	1	DRIVE = 1 のときに検出する	パルス出力を減速停止する	ORGEND = 1 になる SSEND フラグは変化しない
1	0	DRIVE = 1 のときに検出する	パルス出力を即時停止する	ORGEND = 1 になる FSEND フラグは変化しない
1	1	DRIVE = 1 のときに検出する	1 パルス停止する	ORGEND = 1 になる FSEND フラグは変化しない

1 パルス停止 :

- ・ 1 パルス停止機能は即時停止機能とほぼ同様です。1 パルス停止機能を検出すると、実行中のドライブパルス出力を完全に 1 周期出力してから、ドライブパルス出力を停止します。
- ・ 1 パルス停止の後の連続ドライブでは、ドライブパルス周期の連続性を保つことが容易になります。

"01, 10, 11" に設定した場合は、検出と同時に STATUS2 PORT の ORGEND = 1 にします。

D7 : AUTO DRST ENABLE

SPEC INITIALIZE3 コマンドの DRST TYPE を<サーボ対応>に設定している場合に有効です。

ORG エッジ信号の停止機能が動作 (ORGEND = 1) して、ドライブパルス出力を終了した時に、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)

D15--D8 : ORG DIVISION D7--D0

ORG STOP TYPE の検出条件で検出したエッジ信号の分周数を選択します。

分周した信号が、最終検出の ORG エッジ信号になります。

D7--D0	H'FF	H'FE	H'FD	~	H'03	H'02	H'01	H'00
分周数	256	255	254	~	4	3	2	1 (分周なし)

ORG DIVISION = H'00 : 1 カウント毎のエッジ信号 (ORG エッジ信号) を出力する

ORG DIVISION = H'01 : 2 カウント毎のエッジ信号 (ORG エッジ信号) を出力する

:

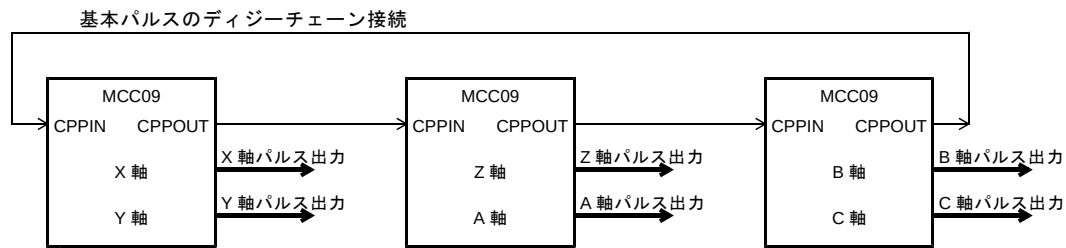
ORG DIVISION = H'FF : 256 カウント毎のエッジ信号 (ORG エッジ信号) を出力する

以下の場合、分周カウント値をクリアします。

- ・ ORIGIN SPEC SET コマンドの実行
- ・ ORG STOP TYPE = "01, 10, 11" に設定したときの DRIVE = 0 の状態

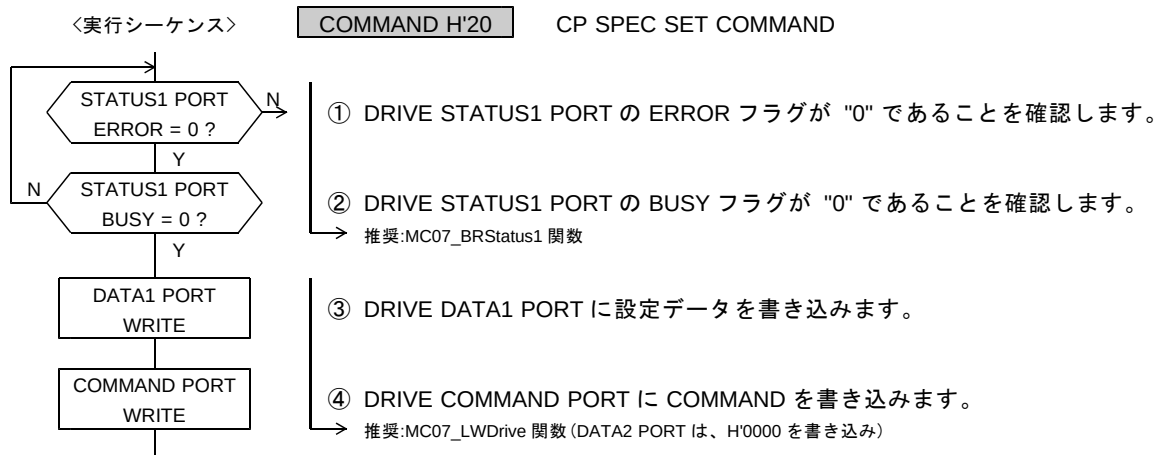
3-1-4. 補間ドライブの設定

各軸 MCC09 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。



(1) CP SPEC SET

補間ドライブの補間パルスの入出力機能を設定します。
このコマンドの設定は X(Z) (B), Y(A) (C) 軸で共有します。X(Z) (B), Y(A) (C) のど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
—	—	—	CPPIN SEL	—	CPPOUT SEL2	CPPOUT SEL1	CPPOUT SEL0

- 電源投入後の初期値は H'01 (アンダーライン側) です。

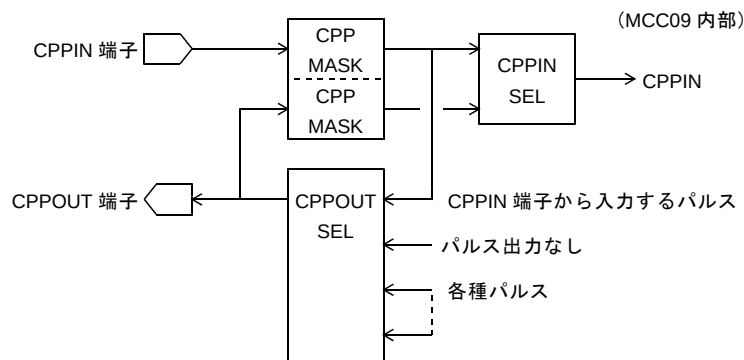
D0 : CPPOUT SEL0

D1 : CPPOUT SEL1

D2 : CPPOUT SEL2

CPPOUT 端子から出力する補間パルスを選択します。

SEL2	SEL1	SEL0	CPPOUT 端子から出力するパルス
0	0	0	CPPIN 端子から入力するパルス
<u>0</u>	<u>0</u>	<u>1</u>	<u>パルス出力なし（ハイレベル出力）</u>
0	1	0	X(Z)(B)軸の出力パルス（XOP）
0	1	1	Y(A)(C)軸の出力パルス（YOP）
1	0	0	EA0, EB0 信号を変換したパルス *2
1	0	1	EA1, EB1 信号を変換したパルス *3
1	1	0	X(Z)(B)軸が発生する補間ドライブの基本パルス *1
1	1	1	Y(A)(C)軸が発生する補間ドライブの基本パルス *1



*1 : メイン軸直線補間ドライブを実行すると、直線補間ドライブの基本パルスを出します。
 メイン軸円弧補間ドライブを実行すると、円弧補間ドライブの基本パルスを出します。
 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）にします。

*2 : EA0, EB0 端子から入力する信号を、X 軸の外部パルス出力仕様の設定でパルス出力します。
 外部パルス出力仕様は、X 軸の ADDRESS COUNTER INITIALIZE1 コマンドで設定します。
 ただし、X 軸の COUNT PULSE SEL を EA0, EB0 または EA1, EB1 に設定している場合は、
 X 軸の出力パルス（XOP : EA0, EB0 または EA1, EB1）を CPPOUT 端子から出力します。

*3 : EA1, EB1 端子から入力する信号を、Y 軸の外部パルス出力仕様の設定でパルス出力します。
 外部パルス出力仕様は、Y 軸の ADDRESS COUNTER INITIALIZE1 コマンドで設定します。
 ただし、Y 軸の COUNT PULSE SEL を EA0, EB0 または EA1, EB1 に設定している場合は、
 Y 軸の出力パルス（YOP : EA0, EB0 または EA1, EB1）を CPPOUT 端子から出力します。

D4 : CPPIN SEL

MCC09 内部の補間パルス入力（CPPIN）に入力する補間パルスを選択します。

0 : CPPIN 端子から入力するパルス

1 : CPPOUT 端子から出力するパルス

- 電源投入後の初期値は 1 です。

3-1-5. 直線補間ドライブの設定と実行

※デバイスドライバによる補間関数は、これらのコマンドを使用して、座標指定による補間ドライブを実現しています。

メイン軸直線補間ドライブの実行軸には、加減速ドライブのパラメータを設定します。

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、短軸側は基本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。

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

■ 直線補間ドライブのドライブ仕様

ドライブ仕様は、直線補間ドライブのコマンド実行時に指定します。

- ・メイン軸は、長軸と短軸のパルス比で基本パルスと補間パルスを発生し、指定した座標アドレスの補間パルスを出力します。
- ・サブ軸は、長軸と短軸のパルス比で補間パルスを発生し、指定した座標アドレスの補間パルスを出力します。

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。

● 直線補間 SCAN ドライブ

停止指令を検出するまで、連続して補間ドライブを行います。

停止指令を検出すると、メイン軸は、基本パルスと補間パルス出力を停止します。

サブ軸は、補間パルス出力を即時停止します。

● 直線補間 INDEX ドライブ

基本パルスをカウントして、カウント値が長軸の座標アドレスのパルス数と一致すると、補間ドライブを終了します。

D1 : CONST CP ENABLE

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。

メイン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。

メイン軸の長軸と短軸の 2 軸間で、2 軸同時にパルス出力したときに、次の基本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。

D2 : CPP STOP ENABLE

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。

● メイン軸の CPP STOP 機能

メイン軸補間ドライブ実行中に機能します。

- ・メイン軸の CPP STOP 機能を有効にすると、補間ドライブの基本パルスと CPPIN に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、補間ドライブの基本パルス出力を停止して、メイン軸のドライブを終了します。
- ・CPP STOP 機能でドライブを終了した場合は、メイン軸の 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。

D2 : 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 CPPIN マスク機能

サブ軸補間ドライブ実行中に機能します。

- ・サブ軸の CPPIN マスク機能を有効にすると、

ERROR = 1 になると、MCC09 内部の CPPIN に入力するパルスをマスクします。

- ・出力中の補間パルスが OFF レベルのときにマスクします。

出力中の補間パルスのアクティブレベルが続く場合は、

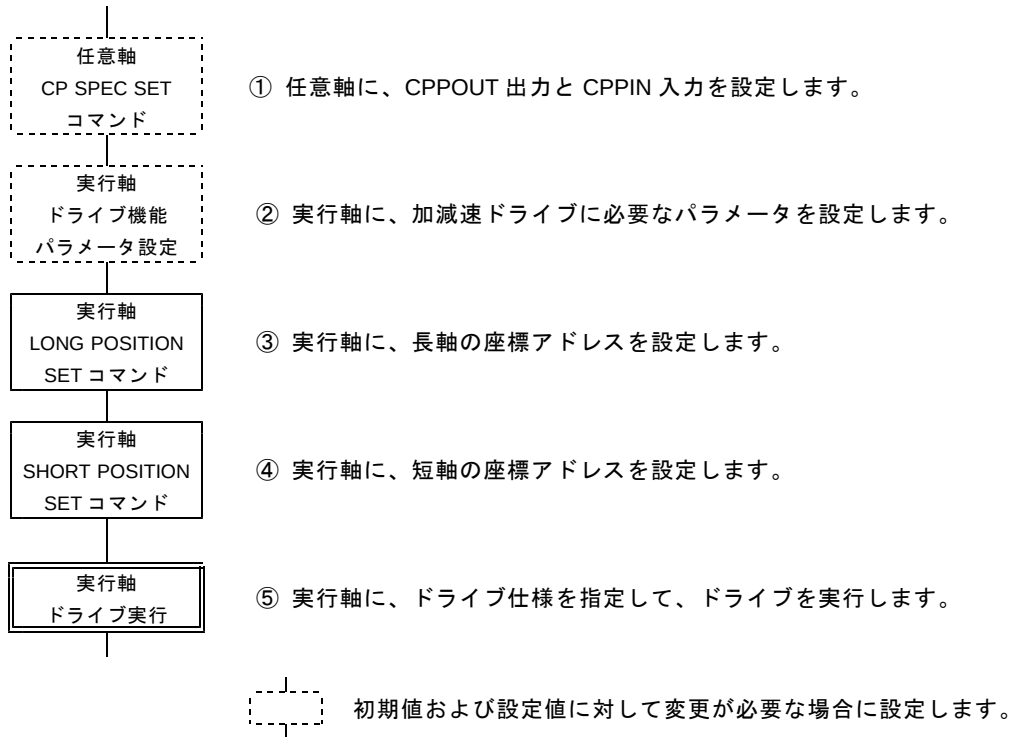
ERROR = 1 から 100 μ s 後に、補間パルス出力を OFF レベルにしてマスクします。

- ・CPPOUT SEL で CPPOUT 出力を「CPPIN 端子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。

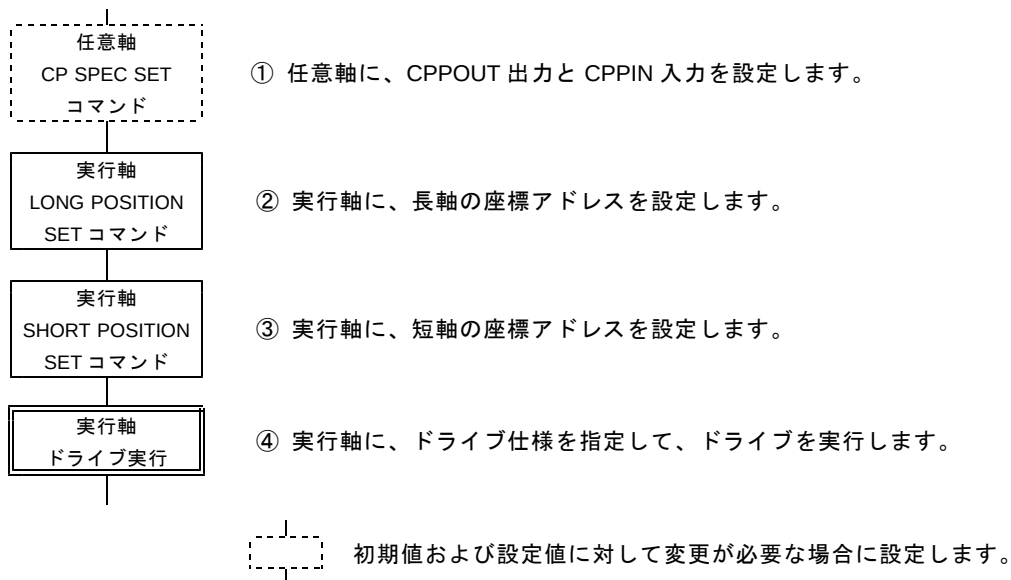
- ・CPPIN マスク機能で CPPIN をマスク中は、STATUS5 PORT の CPP MASK = 1 にします。

ERROR = 0 にクリアすると、CPP MASK = 0 にします。

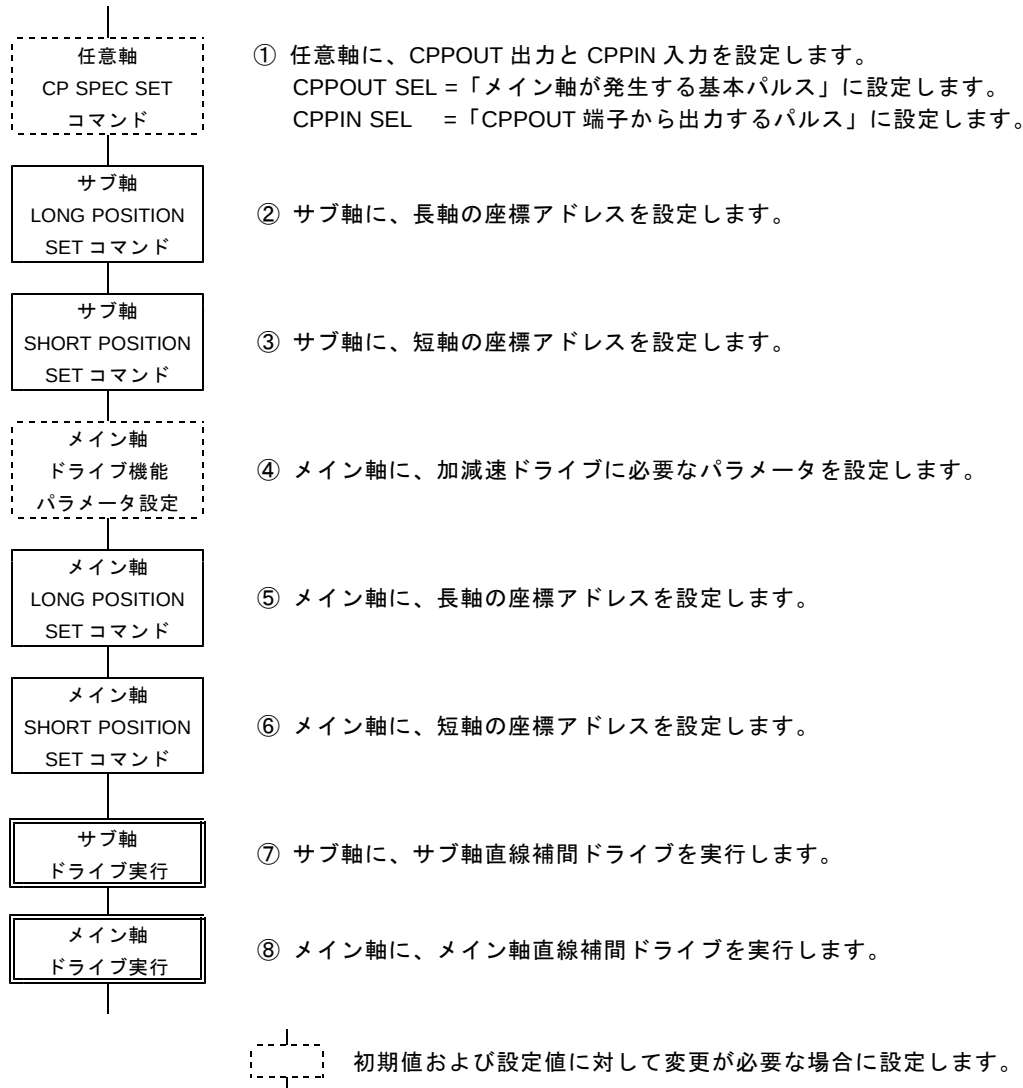
■ メイン軸直線補間ドライブの実行シーケンス



■ サブ軸直線補間ドライブの実行シーケンス

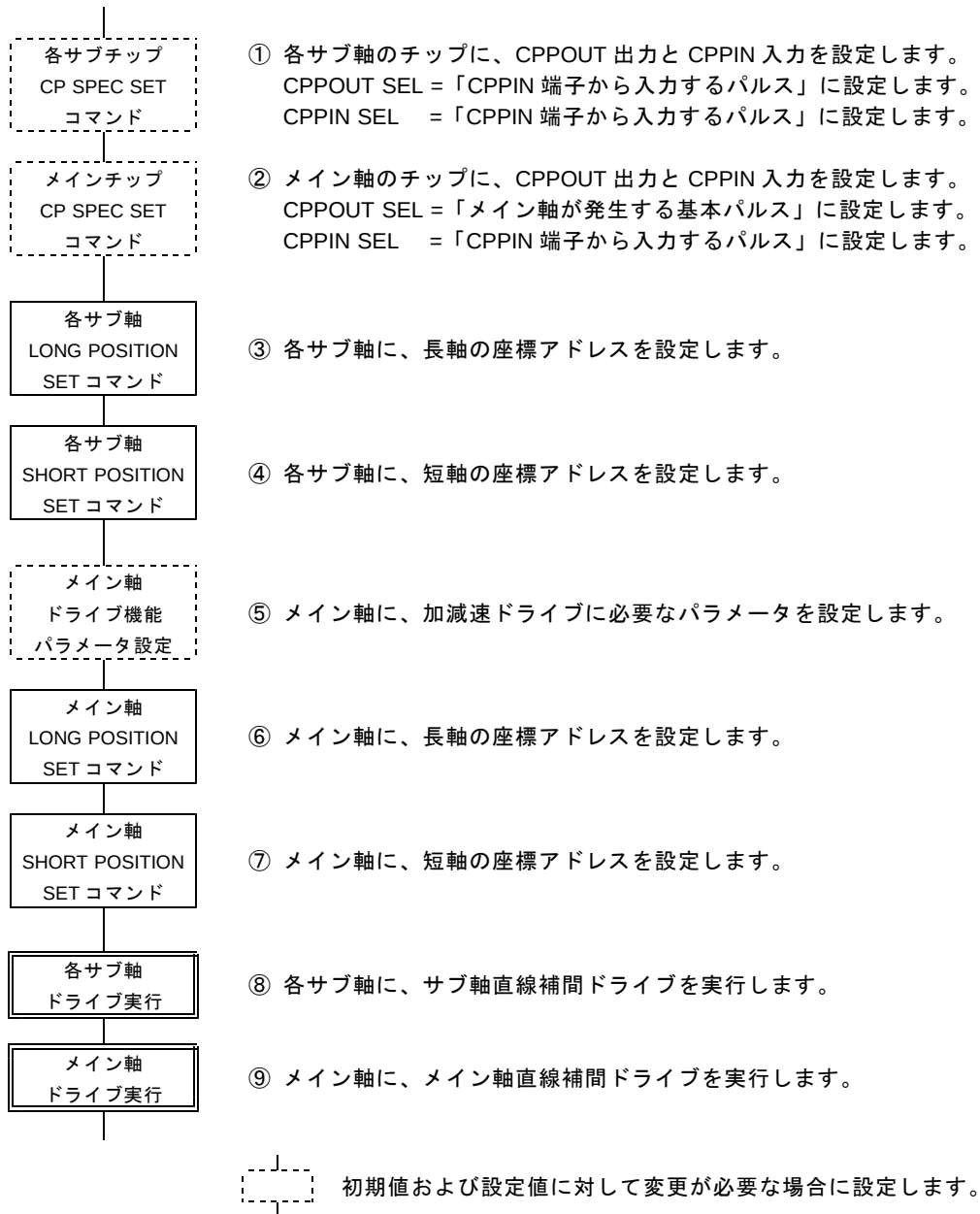


■ 1 チップ直線補間ドライブの実行シーケンス



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、基本パルスを出力して、補間ドライブを開始します。
サブ軸は、CPPIN に入力するパルスを基本パルスにして、補間ドライブを開始します。

■ マルチチップ直線補間ドライブの実行シーケンス

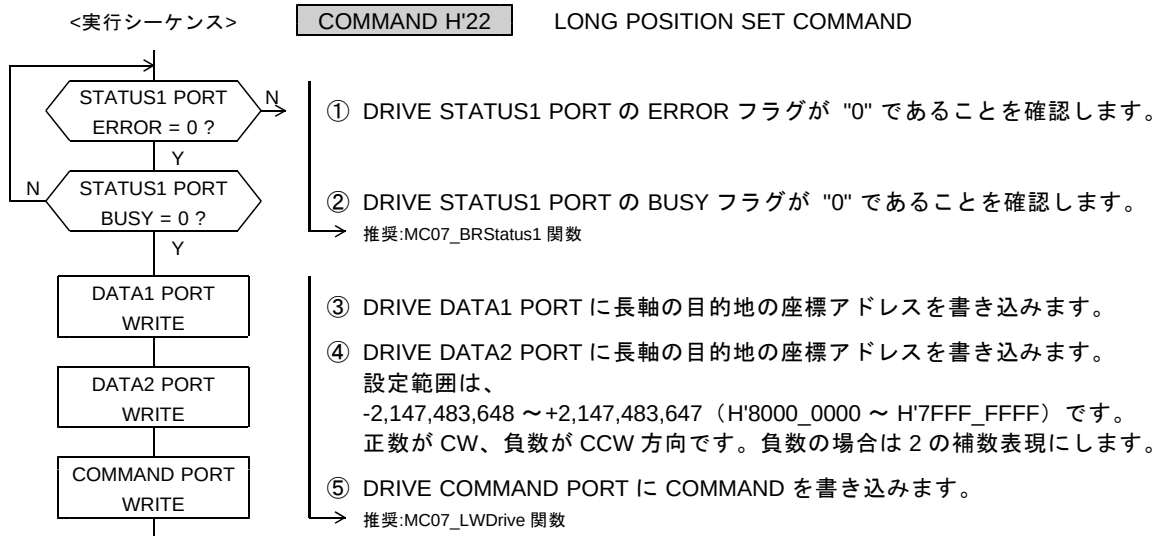


MCC09 の CPPIN 端子と CPPOUT 端子をデージーチェーン接続します。

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、基本パルスを出力して、補間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基本パルスとして、補間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															

●電源投入後の初期値は H'0000 0000 です。

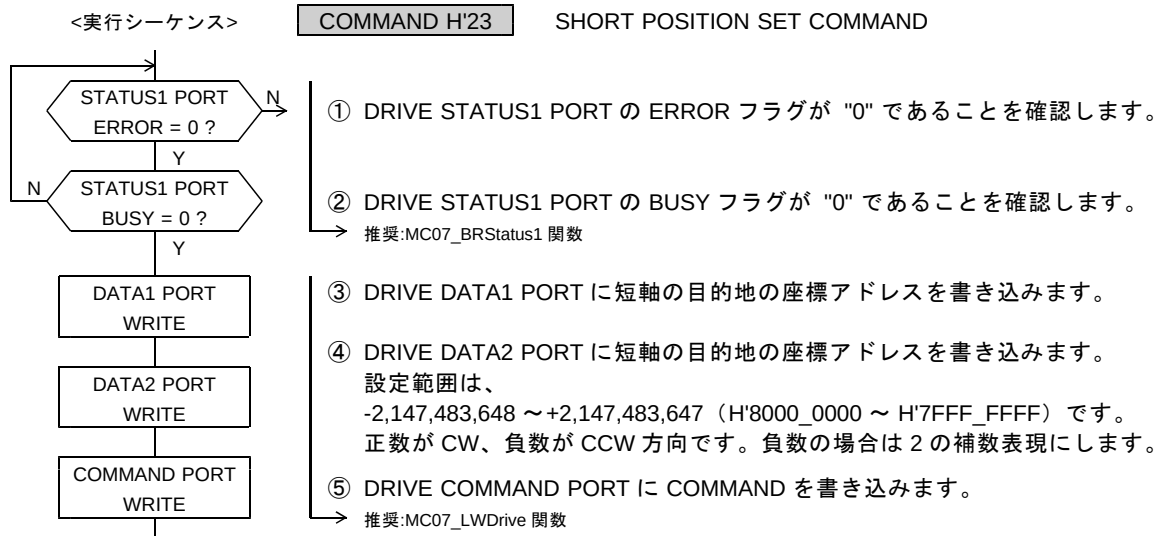
指定する座標アドレスは、現在位置を座標中心 (0, 0) とした相対アドレスです。

「長軸の目的地の座標アドレス」には、
各補間軸の中で補間パルス数が最も大きい補間軸（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（長軸／短軸）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															

●電源投入後の初期値は 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。
- ・「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」に設定します。

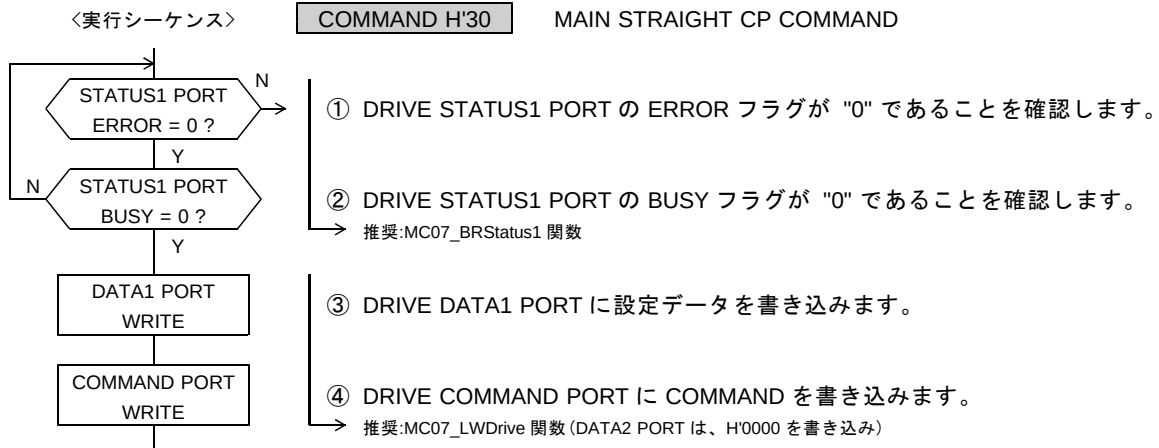
ドライブ実行コマンドの PULSE SEL で指定した軸（長軸／短軸）の座標アドレスの符号が、出力する補間パルスの動作方向になります。

(3) MAIN 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

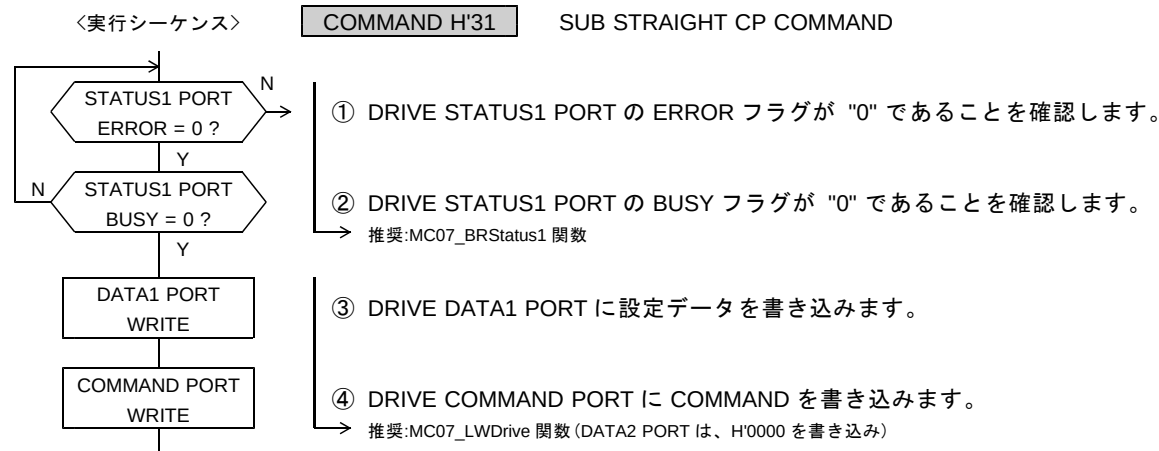
D4 : PULSE SEL

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

(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

3-1-6. 円弧補間ドライブの設定と実行

メイン軸円弧補間ドライブの実行軸には、加減速ドライブのパラメータを設定します。

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、
長軸側は基本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。

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、円弧補間ドライブを実行します。

■ 円弧補間ドライブのドライブ仕様

ドライブ仕様は、円弧補間ドライブのコマンド実行時に指定します。

- ・メイン軸は、指定の円弧半径と回転方向で基本パルスと補間パルスを発生し、
指定した座標アドレスの補間パルスを出力します。
- ・サブ軸は、指定の円弧半径と回転方向で補間パルスを発生し、
指定した座標アドレスの補間パルスを出力します。

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。

● 円弧補間 SCAN ドライブ

停止指令を検出するまで、連続して補間ドライブを行います。

停止指令を検出すると、メイン軸は、基本パルスと補間パルス出力を停止します。

サブ軸は、補間パルス出力を即時停止します。

● 円弧補間 INDEX ドライブ

基本パルスをカウントして、カウント値が短軸パルス数の設定値と一致すると、
補間ドライブを終了します。

D1 : CONST CP ENABLE

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。

メイン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。

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、
次の基本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。

D2 : CPP STOP ENABLE

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。

● メイン軸の CPP STOP 機能

メイン軸補間ドライブ実行中に機能します。

- ・メイン軸の CPP STOP 機能を有効にすると、
補間ドライブの基本パルスと CPPIN に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、
補間ドライブの基本パルス出力を停止して、メイン軸のドライブを終了します。
- ・CPP STOP 機能でドライブを終了した場合は、
メイン軸の ERROR STATUS の CPP STOP ERROR = 1 にします。
- ・メインは 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。

D2 : 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 CPPIN マスク機能

サブ軸補間ドライブ実行中に機能します。

- ・サブ軸の CPPIN マスク機能を有効にすると、

ERROR = 1 になると、MCC09 内部の CPPIN に入力するパルスをマスクします。

- ・出力中の補間パルスが OFF レベルのときにマスクします。

出力中の補間パルスのアクティブレベルが続く場合は、

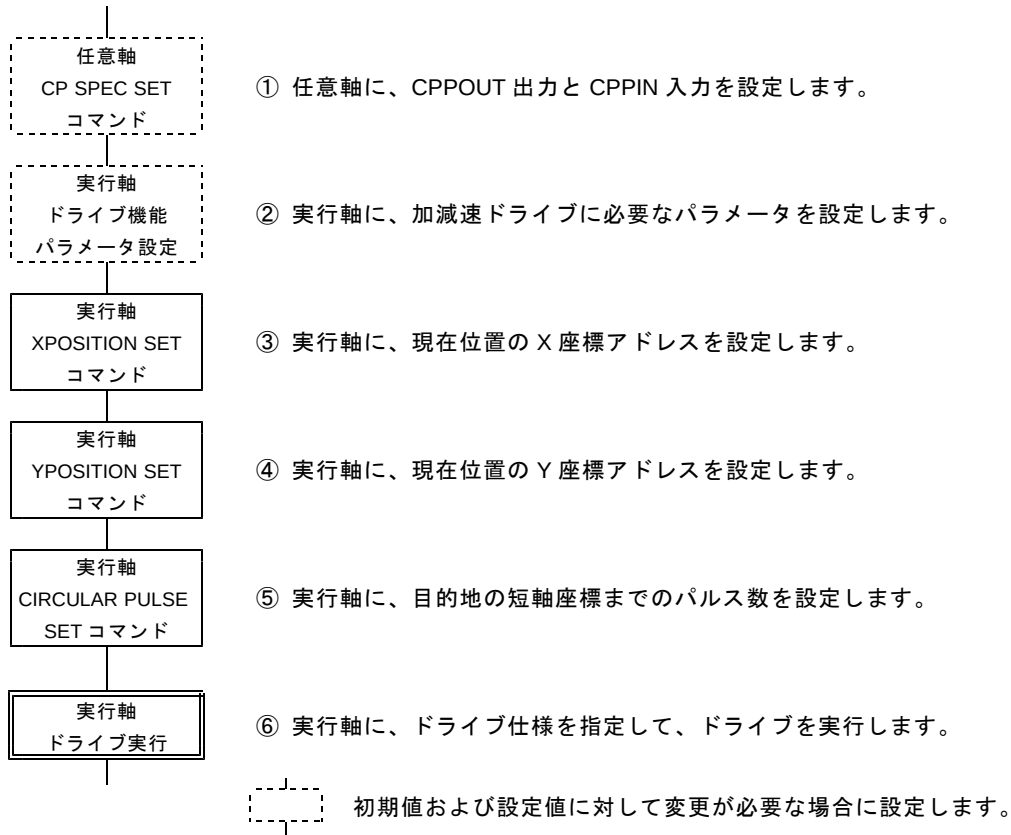
ERROR = 1 から 100 μ s 後に、補間パルス出力を OFF レベルにしてマスクします。

- ・CPPOUT SEL で CPPOUT 出力を「CPPIN 端子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。

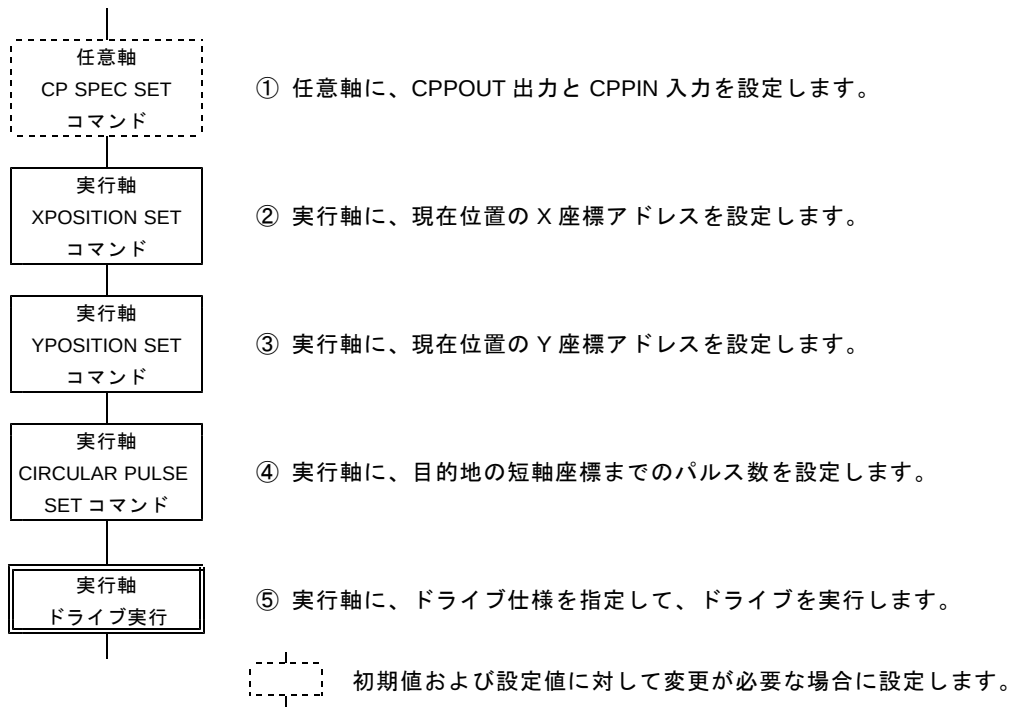
- ・CPPIN マスク機能で CPPIN をマスク中は、STATUS5 PORT の CPP MASK = 1 にします。

ERROR = 0 にクリアすると、CPP MASK = 0 にします。

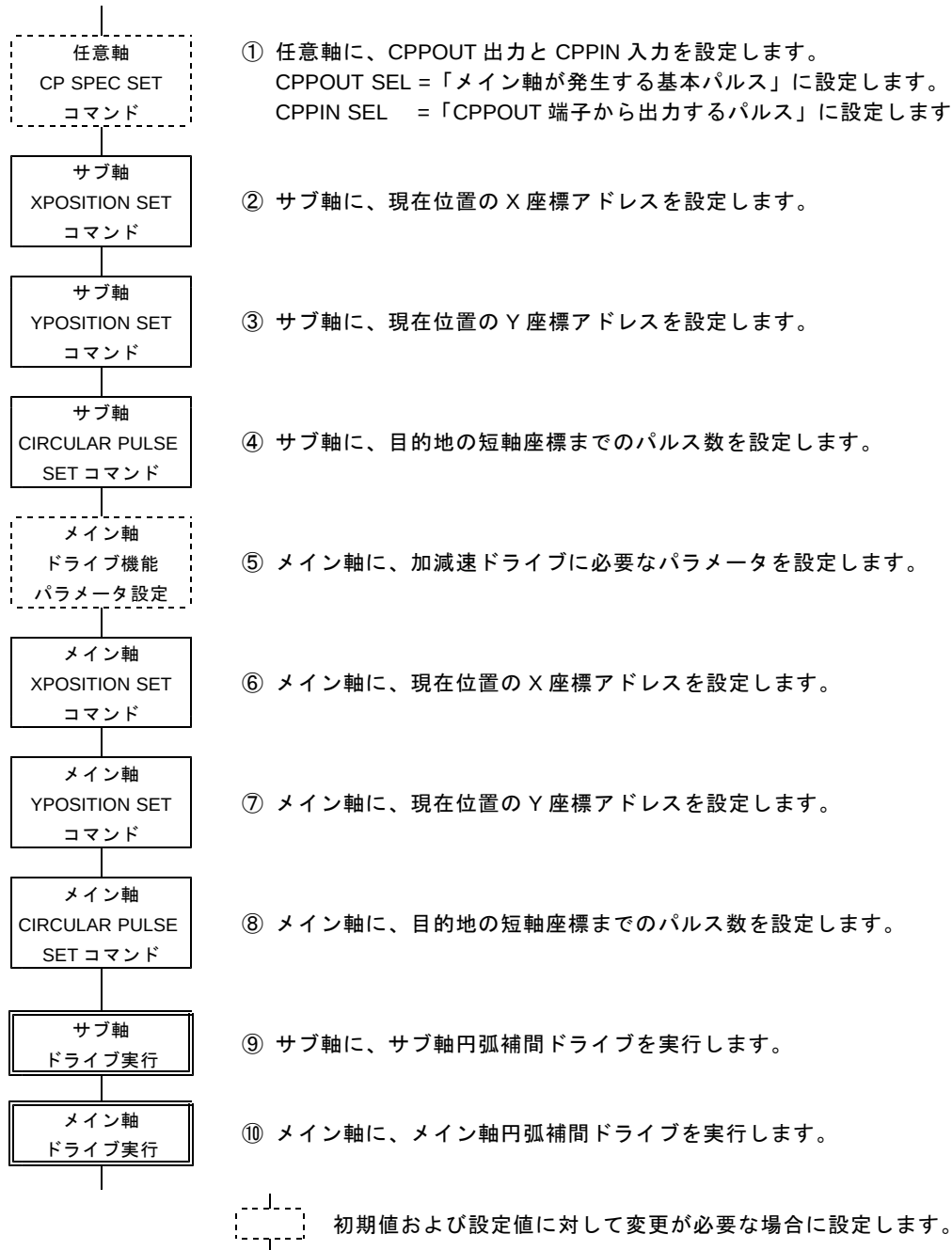
■ メイン軸円弧補間ドライブの実行シーケンス



■ サブ軸円弧補間ドライブの実行シーケンス

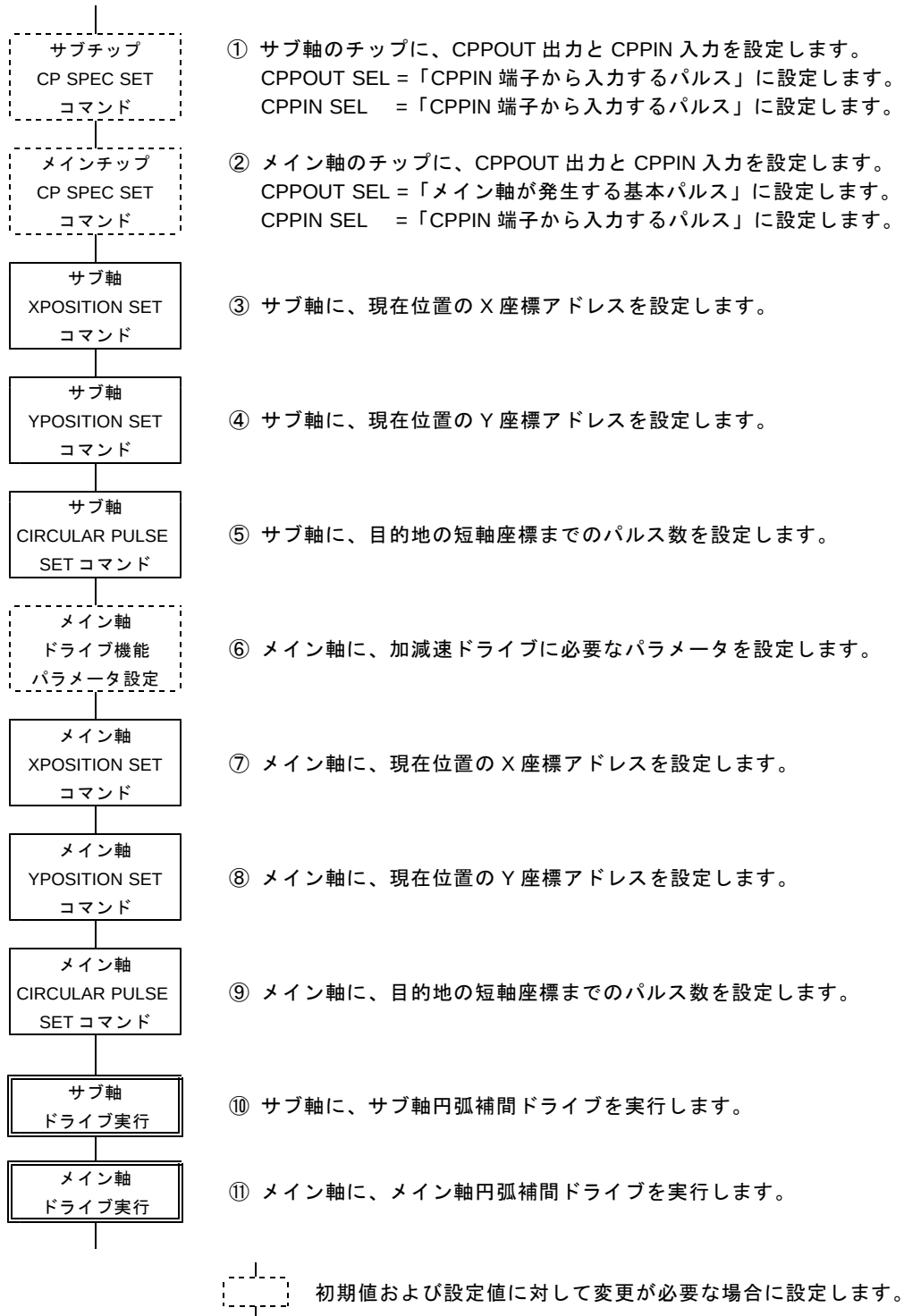


■ 1 チップ 2 軸円弧補間ドライブの実行シーケンス



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、基本パルスを出力して、補間ドライブを開始します。
サブ軸は、CPPIN に入力するパルスを基本パルスにして、補間ドライブを開始します。

■ 2チップ2軸円弧補間ドライブの実行シーケンス

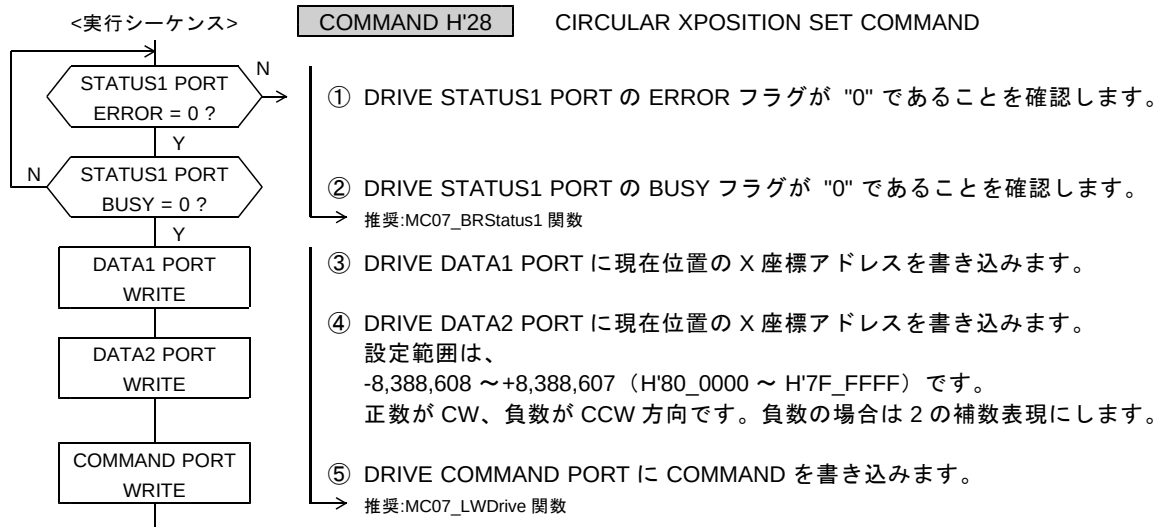


サブ軸円弧補間ドライブを実行すると、ドライブがスタンバイ状態になります。
メイン軸円弧補間ドライブを実行すると、基本パルスを出力して、補間ドライブを開始します。
サブ軸は、CPPIN 端子から入力するパルスを基本パルスとして、補間ドライブを開始します。

(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。

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

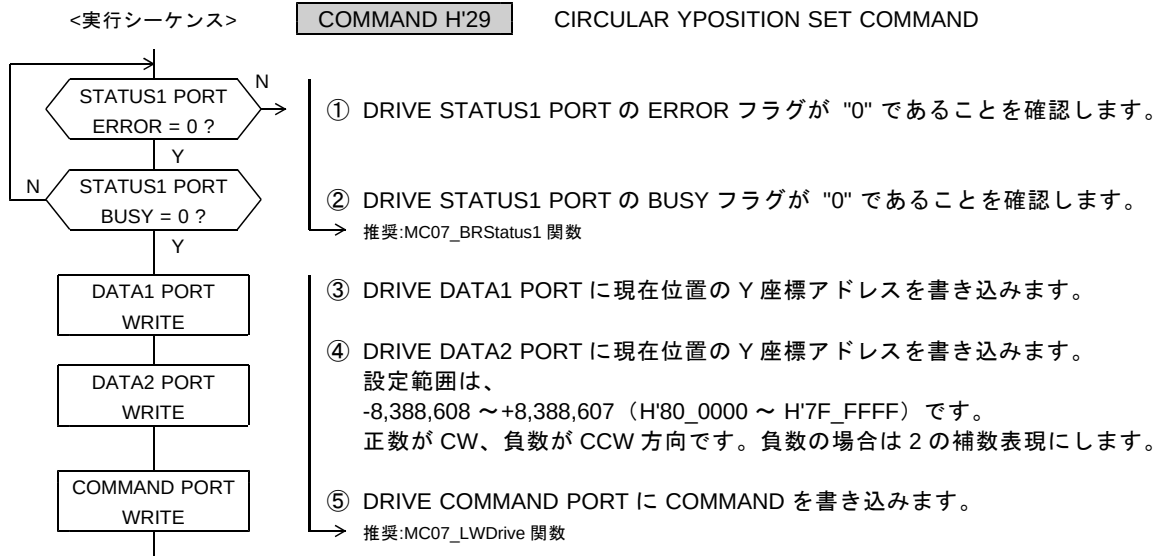
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
—	—	—	—	—	—	—	—	A23	← 現在位置の X 座標アドレス →						A16

●電源投入後の初期値は 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。
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															A0

← 現在位置の Y 座標アドレス →

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							A16

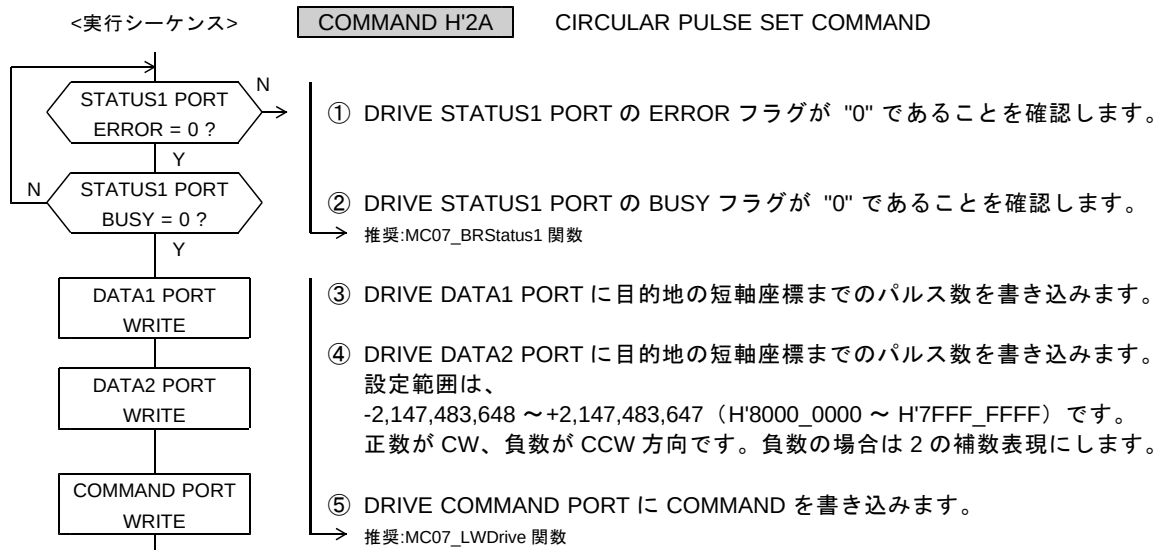
← 現在位置の Y 座標アドレス →

●電源投入後の初期値は 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。
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
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															

●電源投入後の初期値は 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。

・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。

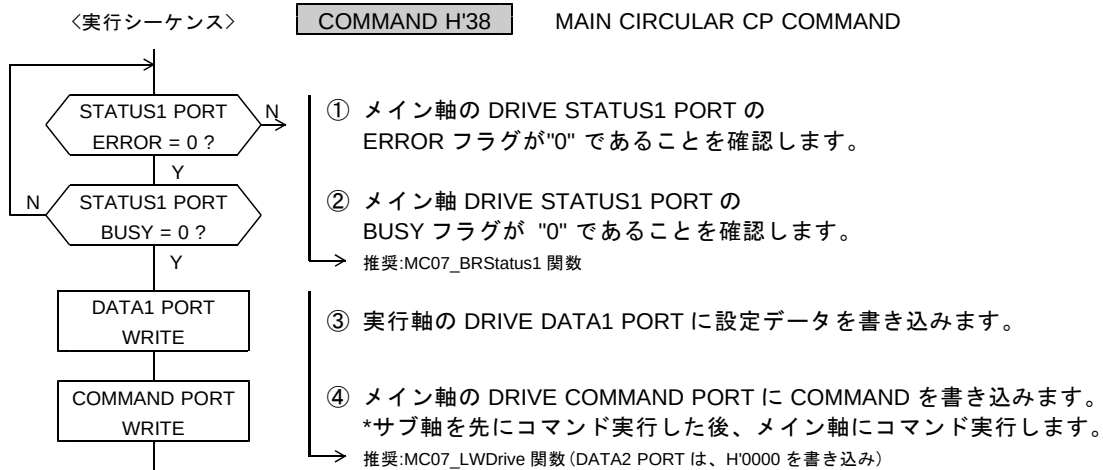
円弧を描く回転方向は、パルス数の符号で指定します。

・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。

(4) MAIN CIRCULAR CP

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。
メイン軸の加減速パラメータで発生するパルスの基本パルスにして動作します。



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

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

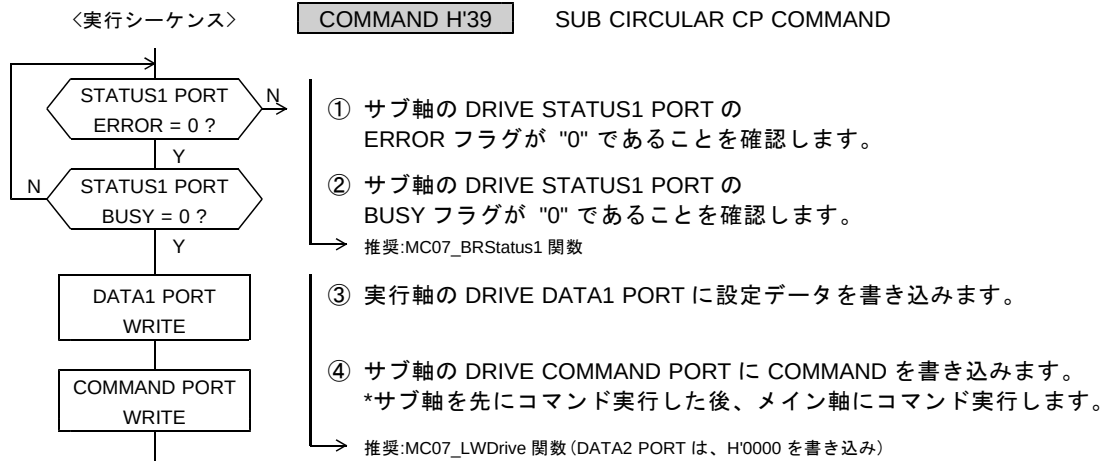
D4 : PULSE SEL

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

(5) SUB CIRCULAR CP

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。
サブ軸は、CPPIN 入力パルスの基本パルスにして動作します。



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

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。

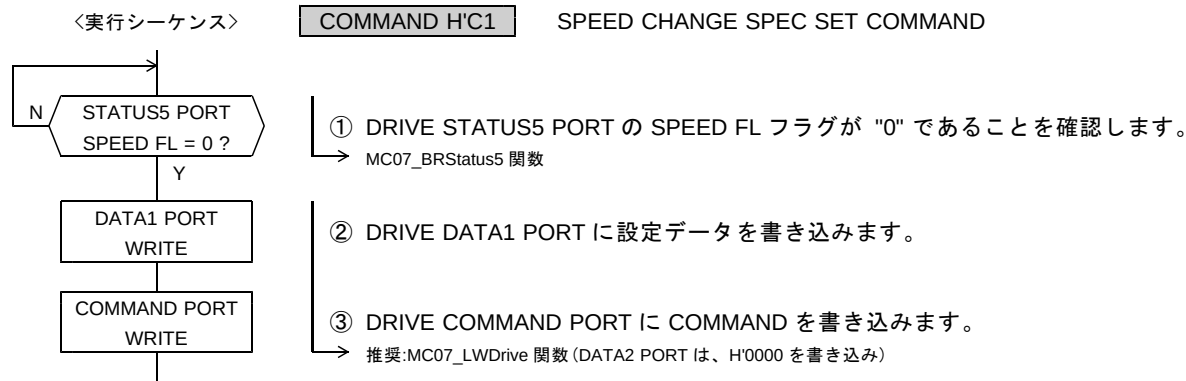
- 0 : 円弧補間演算の X 座標アドレスの補間パルス (XCP) を出力する
- 1 : 円弧補間演算の Y 座標アドレスの補間パルス (YCP) を出力する

3-1-7. 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

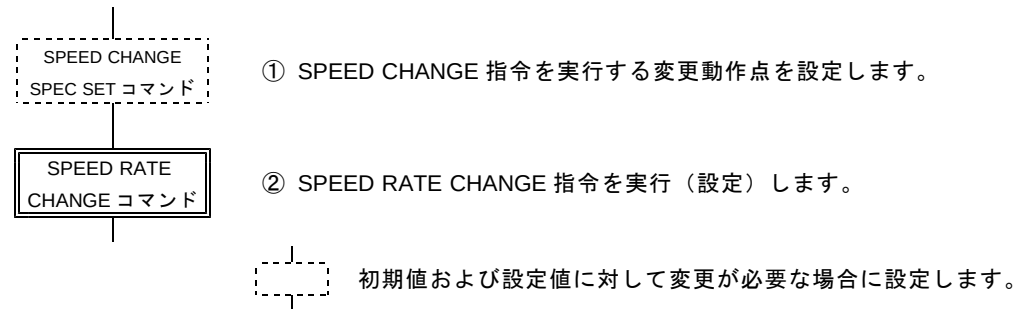
●電源投入後の初期値は H'0 (アンダーライン側) です。

D2--D0 : SPEED CHANGE TYPE2--0

SPEED CHANGE 指令を実行する変更動作点を選択します。

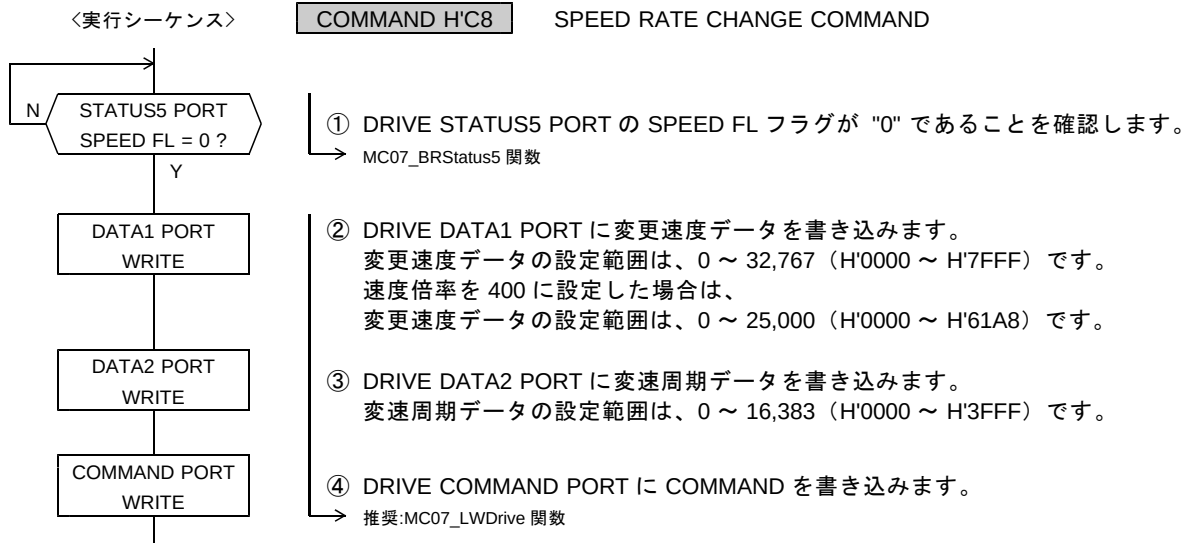
D2	D1	D0	SPEED CHANGE を実行する変更動作点	検出仕様
<u>0</u>	<u>0</u>	<u>0</u>	<u>STATUS1 PORT の DRIVE = 1 で実行する</u>	レベル検出
0	0	1	DRIVE = 1 のときに、他軸の STATUS3 PORT の OUT3 = 0 → 1 で実行する	エッジ検出
0	1	0	DRIVE = 1 のときに、STATUS3 PORT の GPIO2 = 0 → 1 で実行する	エッジ検出
0	1	1	DRIVE = 1 のときに、ORIGIN 停止機能の ORG エッジ信号の検出で実行する	エッジ検出
1	0	0	DRIVE = 1 のときに、STATUS3 PORT の OUT2 = 0 → 1 で実行する	エッジ検出
1	0	1	DRIVE = 1 のときに、STATUS3 PORT の OUT3 = 0 → 1 で実行する	エッジ検出
1	1	0	設定禁止	—
1	1	1	設定禁止	—

■ SPEED CHANGE の実行シーケンス



(2) SPEED RATE CHANGE

変更動作点の検出で、SPEED RATE CHANGE 指令を実行します。



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

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	← RATE CHANGE データ →												D0

SPEED CHANGE データの設定値を "0" にすると、CONST DRIVE 指令になります。

・ SPEED CHANGE の速度 (Hz) = SPEED CHANGE データ x RESOL : 1 ~ 10,000,000 Hz

RATE CHANGE データ (変速周期データ) の設定値が "0" の場合は、

「RATE CHANGE データ = RATE SET コマンドの UCYCLE または DCYCLE」にします。

・ 加速カーブの変速周期 (μs) = RATE CHANGE データ (または UCYCLE) x 0.5 μs

・ 減速カーブの変速周期 (μs) = RATE CHANGE データ (または DCYCLE) x 0.5 μs

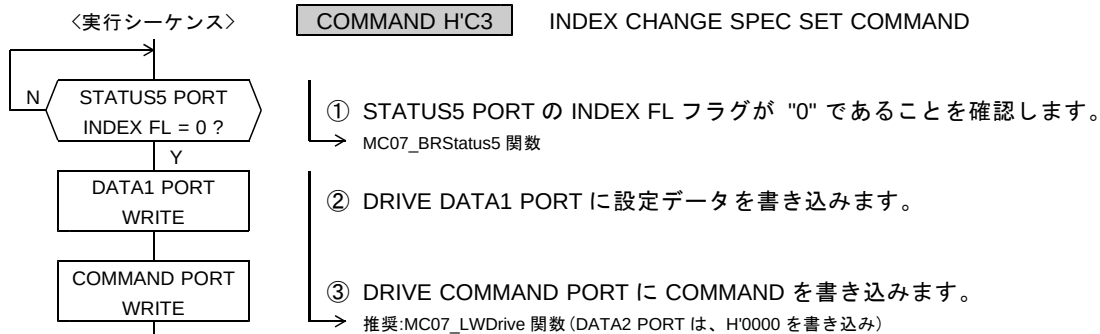
SPEED RATE CHANGE コマンドを実行しても、予め設定されている速度パラメータの値は変わりません。

3-1-8. 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と RFSPD を設定します。



DRIVE DATA1 PORT の設定データ

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

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

● 電源投入後の初期値は H'1400 (アンダーライン側) です。

D2--D0 : INDEX CHANGE TYPE2--0

INDEX CHANGE 指令を実行する変更動作点を選択します。

D2	D1	D0	INDEX CHANGE を実行する変更動作点	検出仕様
0	0	0	STATUS1 PORT の DRIVE = 1 で実行する	レベル検出
0	0	1	DRIVE = 1 のときに、他軸の STATUS3 PORT の OUT3 = 0 → 1 で実行する	エッジ検出
0	1	0	DRIVE = 1 のときに、STATUS3 PORT の GPIO2 = 0 → 1 で実行する	エッジ検出
0	1	1	DRIVE = 1 のときに、ORIGIN 停止機能の ORG エッジ信号の検出で実行する	エッジ検出
1	0	0	DRIVE = 1 のときに、STATUS3 PORT の OUT2 = 0 → 1 で実行する	エッジ検出
1	0	1	DRIVE = 1 のときに、STATUS3 PORT の OUT3 = 0 → 1 で実行する	エッジ検出
1	1	0	設定禁止	—
1	1	1	設定禁止	—

D15--D8 : RFSPD D7--D0

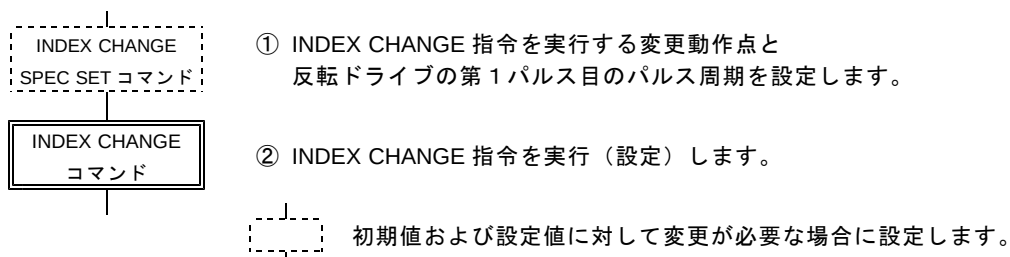
INDEX CHANGE 指令による反転方向の動作も可能です。

この反転ドライブの第 1 パルス目のパルス周期 (パルス速度) を設定します。

- ・設定範囲は、0 ~ 255 (H'00 ~ H'FF) です。設定値は、1 Hz 単位です。
- ・RFSPD の初期値は、20 Hz (H'14) です。

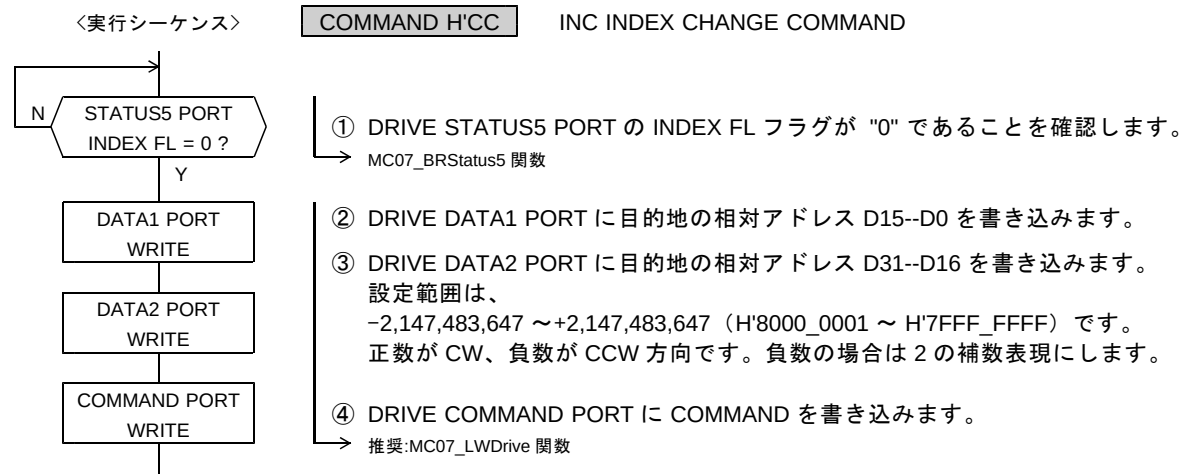
RFSPD の設定値が "0" の場合は、RFSPD を RFSPD = RSPD x RESOL に補正します。

INDEX CHANGE の実行シーケンス



(2) INC INDEX CHANGE

変更動作点の検出で、INC INDEX CHANGE 指令を実行します。
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															

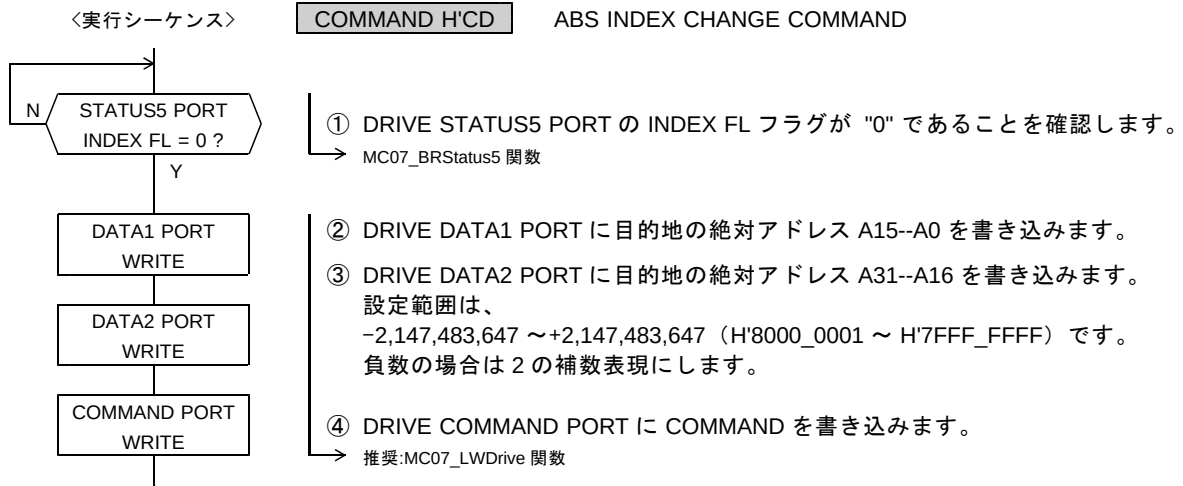
DRIVE DATA2 PORT の設定データ

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。

(3) ABS INDEX CHANGE

変更動作点の検出で、ABS INDEX CHANGE 指令を実行します。
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。



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

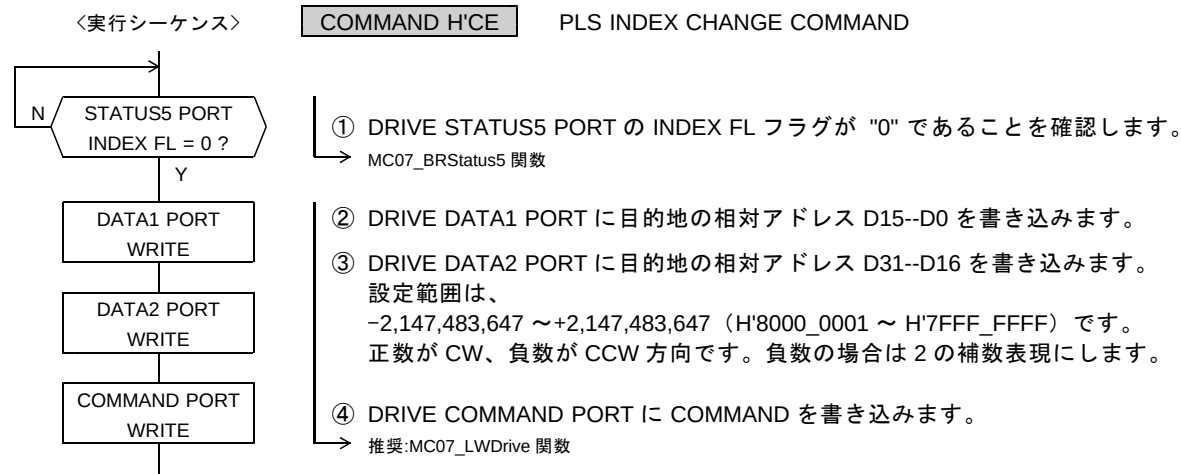
DRIVE DATA2 PORT の設定データ

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

変更動作点の検出で、PLS INDEX CHANGE 指令を実行します。
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。



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															

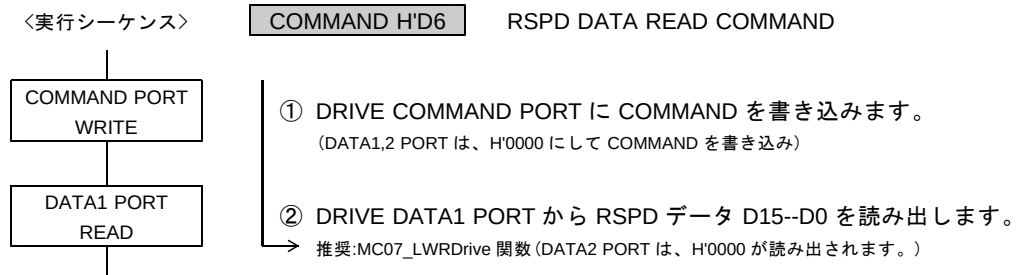
指定する相対アドレスは、変更動作点の検出位置から停止位置までのパルス数を、
変更動作点の検出位置を原点として符号付きで表現した値です。

3-1-9. RSPD データの読み出し

(1) RSPD DATA READ

RSPD データを読み出します。

このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。



読み出すデータは、「15 ビットのパルス速度データ」です。

RSPD DATA READ コマンドを実行すると、
RSPD データを DRIVE DATA1 PORT (READ) にセットします。

RSPD データ

- ・ RSPD は、HSPD, LSPD, ELSPD と同様の 15 ビットのパルス速度データです。
- ・ STATUS1 PORT の DRIVE = 1 → 0 になると、最終出力のパルス速度データを RSPD に記憶します。
ただし、最終出力のパルス速度が FSPD, RFSPD と JSPD の場合は、RSPD を書き換えません。
- ・ RSPD のリセット後の初期値は、H'012C (300) です。

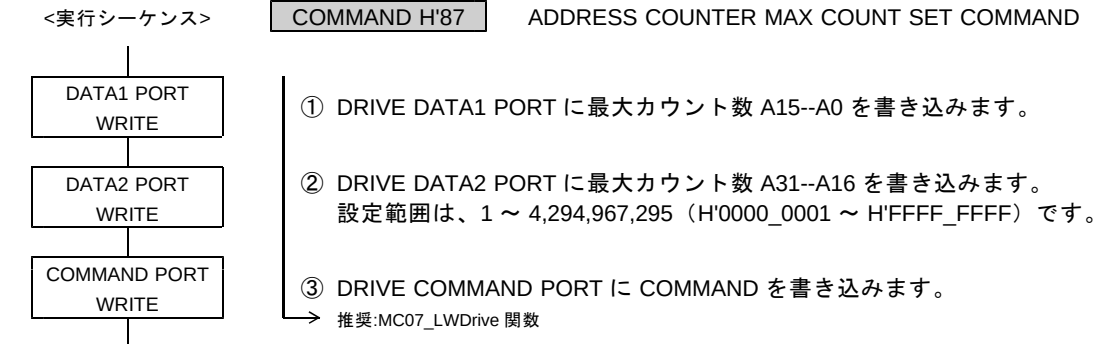
3-2. カウンタコマンド

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。

このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。



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
A15															A0
← 最大カウント数 →															

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															A16
← 最大カウント数 →															

●電源投入後の初期値は 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。

■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

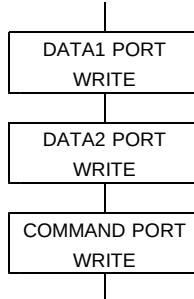
パルスカウンタの最大カウント数を設定します。

このコマンドの実行は DRIVE STATUS1 PORT の BUSY=1 のときも可能です。

<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															

- 電源投入後の初期値は 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。

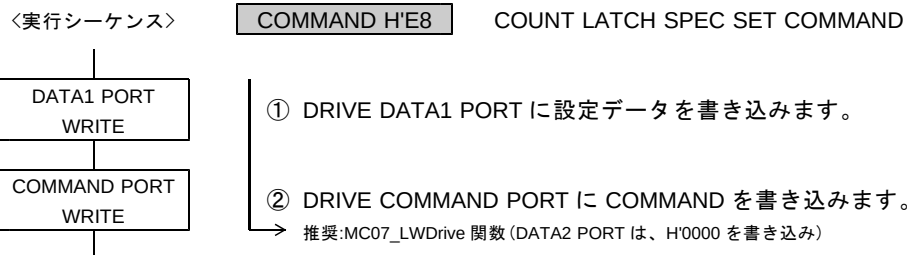
各カウンタには、ラッチタイミングによるカウンタのクリア機能があります。

■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 DRIVE STATUS1 PORT の BUSY=1 のときも可能です。



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

- 電源投入後の初期値は H'000 (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	STATUS3 PORT の GPIO2=0 → 1 でラッチする
0	1	1	ORIGIN 停止機能の ORG エッジ信号の検出でラッチする
1	0	0	STATUS3 PORT の OUT2=0 → 1 でラッチする
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タ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>
<u>0</u>	<u>0</u>	<u>0</u>	<u>PULSE LATCH DATA READ コマンドの実行でラッチする</u>
0	0	1	設定禁止
0	1	0	STATUS3 PORT の GPIO2=0 → 1 でラッチする
0	1	1	ORIGIN 停止機能の ORG エッジ信号の検出でラッチする
1	0	0	STATUS3 PORT の OUT2=0 → 1 でラッチする
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タ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>
<u>0</u>	<u>0</u>	<u>0</u>	<u>DFL LATCH DATA READ コマンドの実行でラッチする</u>
0	0	1	設定禁止
0	1	0	STATUS3 PORT の GPIO2=0 → 1 でラッチする
0	1	1	ORIGIN 停止機能の ORG エッジ信号の検出でラッチする
1	0	0	STATUS3 PORT の OUT2=0 → 1 でラッチする
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															D0

← ラッチデータ →

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															D16

← ラッチデータ →

- ・ 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 DRIVE STATUS1 PORT の BUSY=1 のときも可能です。

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 DRIVE STATUS1 PORT の BUSY=1 のときも可能です。

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 DRIVE STATUS1 PORT の BUSY=1 のときも可能です。

COMMAND H'DE

DFL LATCH DATA READ COMMAND

4. 機能説明

4-1. ドライブ仕様

4-1-1. コマンド予約機能

MCC09 の各軸には、20 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。

* DRIVE COMMAND の特殊コマンドは予約できません。

予約レジスタの状態は、DRIVE STATUS1 PORT の COMREG EP と COMREG FL フラグで確認します。

BUSY = 1、COMREG FL = 0 のときに、DRIVE COMMAND PORT に汎用コマンドを書き込むと、DRIVE DATA1, 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。

● コマンド予約機能が自動的に無効となる状態

以下の状態を検出すると、予約レジスタに格納している汎用コマンドをすべてクリアします。

同時に COMREG FL = 1、COMREG EP = 1 にして、汎用コマンドの書き込みを無効にします。

・ STATUS1 PORT の ERROR = 1 の検出

ERROR = 0 にクリアすると、

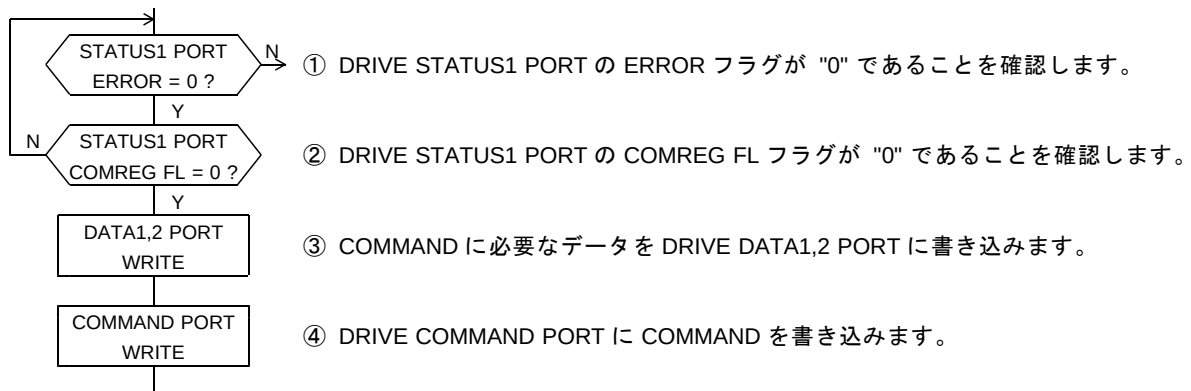
COMREG FL = 0、COMREG EP = 1 にして、汎用コマンドの書き込みを有効にします。

■ コマンド予約機能の実行例

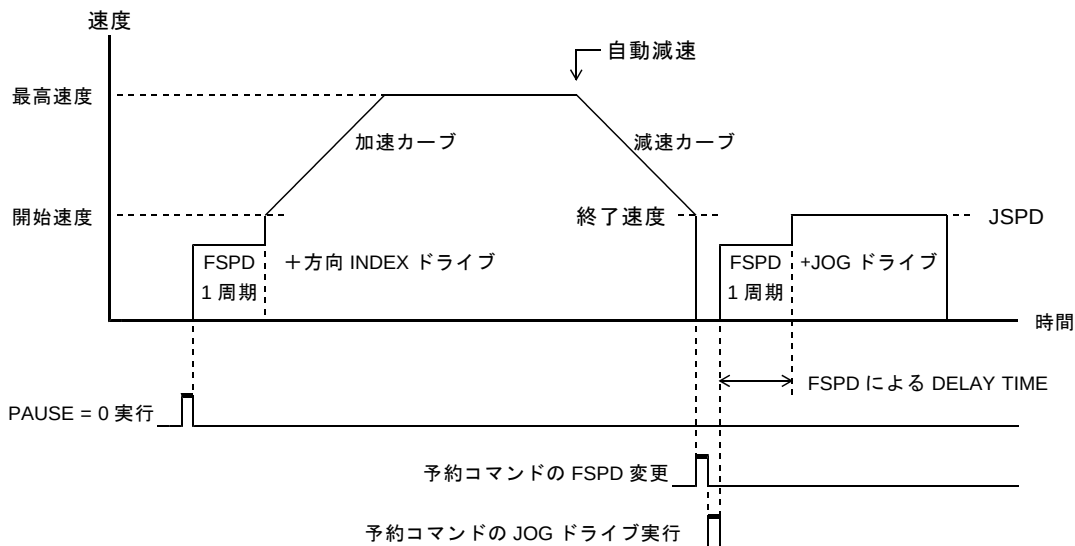
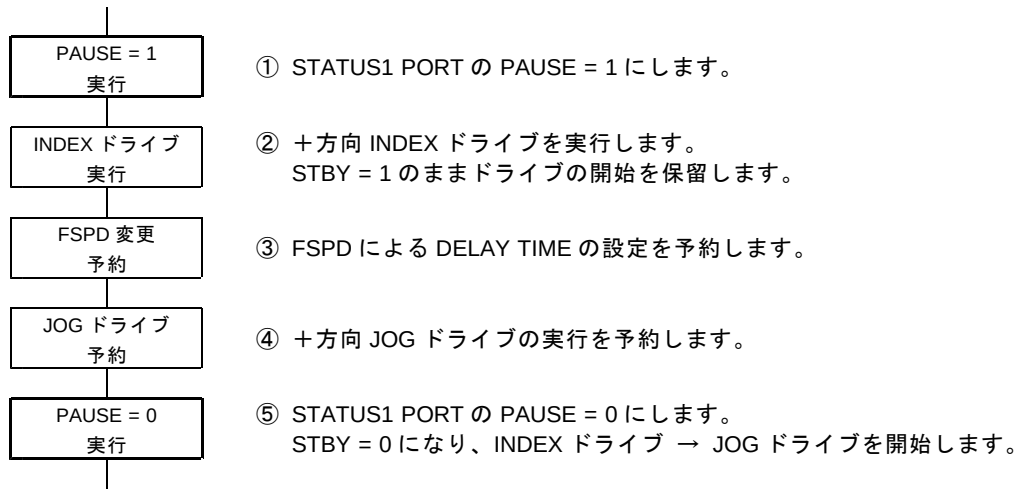
汎用コマンドを予約する場合は、BUSY = 0 の代わりに COMREG FL = 0 を確認します。

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になります。

● コマンド予約シーケンス



● INDEX ドライブ → JOG ドライブの連続実行シーケンス



4-1-2. 入出力仕様

(1) 入出力信号の論理切り替え機能

下記の入出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
FSSTOP 信号	正論理入力	HARD INITIALIZE7 コマンド
CWLM 信号	正論理入力	
CCWLM 信号	正論理入力	
DALM 信号	正論理入力	
ORG 信号	負論理入力	
NORG 信号	負論理入力	
PAUSE	1 でアクティブ	
CWP 信号	負論理出力	HARD INITIALIZE8 コマンド
CCWP 信号	負論理出力	

- ・ PAUSE 信号は外部とインターフェースする信号ではありません。
コマンド予約機能で操作する MCC09 の入力信号です。
- ・ アクティブ論理を変更すると、変更した信号の内部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

加減速ドライブは、加速カーブと減速カーブで加減速を行うドライブです。

- ・加速カーブは、S字加速の変速領域と直線加速の変速領域で構成します。
- ・減速カーブは、S字減速の変速領域と直線減速の変速領域で構成します。
- ・加速カーブと減速カーブを異なる設定にすると、非対称の加減速ドライブになります。

加減速ドライブには、以下のドライブパラメータの設定が必要です。

● 速度のパラメータ

- ・FSPD : 第1パルスの速度 (Hz)
- ・RFSPD : INDEX CHANGE による反転ドライブの第1パルスの速度 (Hz)
- ・RESOL : 速度データの速度倍率
- ・HSPD : 最高速時のパルス速度データ
- ・LSPD : 加速開始時のパルス速度データ
- ・ELSPD : 減速終了時のパルス速度データ
- ・JSPD : JOG ドライブと JSPD SCAN ドライブの速度 (Hz)
- ・RSPD : RSPD は、HSPD, LSPD, ELSPD と同様の 15 ビットのパルス速度データです。
DRIVE = 1 → 0 になると、最終出力のパルス速度データを RSPD に記憶します。
ただし、最終出力のパルス速度が FSPD, RFSPD と JSPD の場合は、RSPD を書き換えません。
RSPD のリセット後の初期値は、H'012C (300) です。

● 加減速カーブのパラメータ

- ・UCYCLE : 加速カーブの変速周期データ
- ・DCYCLE : 減速カーブの変速周期データ
- ・SUAREA : 加速カーブのS字変速領域データ
- ・SDAREA : 減速カーブのS字変速領域データ
- ・SUH : 加速カーブのS字加速終了部の変速領域データ (SCAREA MODE = 1 のときに有効)
- ・SDH : 減速カーブのS字減速開始部の変速領域データ (SCAREA MODE = 1 のときに有効)

● 減速パルス数の調整／設定パラメータ

- ・DOWN PULSE ADJUST : 自動減速パルス数のオフセットパルス数／マニュアル減速パルス数

■ 加減速ドライブの速度とS字領域

加減速ドライブのパルス速度は、速度データと速度倍率で設定します。

- ・最高速時の速度 (Hz) = HSPD × RESOL
- ・加速開始時の速度 (Hz) = LSPD × RESOL
- ・減速終了時の速度 (Hz) = ELSPD × RESOL
- ・SPEED RATE CHANGE の指定速度 (Hz) = SPEED CHANGE データ × RESOL

S字領域は、S字変速領域データと速度倍率で設定します。

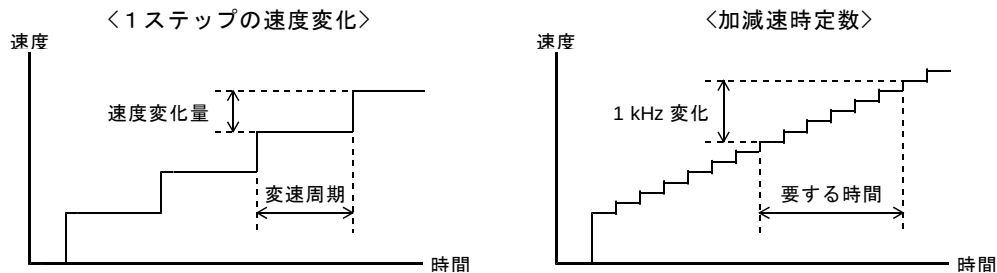
- ・S字加速開始部の変速領域 (Hz) = SUAREA × RESOL
- ・S字加速終了部の変速領域 (Hz) = SUAREA (または SUH) × RESOL
- ・S字減速開始部の変速領域 (Hz) = SDAREA (または SDH) × RESOL
- ・S字減速終了部の変速領域 (Hz) = SDAREA × RESOL

(2) 加減速時定数

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っています。

加減速時定数は、速度を 1 kHz 変化させるのに要する時間 (ms/kHz) で表しています。

本資料では、この時定数を RATE と呼称しています。



● 加減速時定数の設定方法

- ① 最高速度と速度倍率を設定します。設定した速度倍率データで、速度変化量が決定します。
速度倍率を小さくすると、加減速が滑らかになります。
- ② 開始速度と終了速度を設定します。
- ③ 加速の変速周期と減速の変速周期を設定します。設定した変速周期で、加減速時間が決定します。
速度変化量と変速周期で、目的に合った加減速時定数（加減速時間）を設定します。

■ 加減速 RATE の計算式

速度倍率データ (RESOL) と変速周期データ (UCYCLE) で、任意の加速 RATE を設定します。

速度倍率データ (RESOL) と変速周期データ (DCYCLE) で、任意の減速 RATE を設定します。

● 加速カーブの直線加速 RATE の計算式

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

$$\text{加速カーブの変速周期 (ms)} = \text{UCYCLE} * 0.5 * 10^{-3}$$

$$\text{直線加速カーブの速度変化量 (kHz)} = \text{RESOL} * 10^{-3}$$

● 減速カーブの直線減速 RATE の計算式

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

$$\text{減速カーブの変速周期 (ms)} = \text{DCYCLE} * 0.5 * 10^{-3}$$

$$\text{直線減速カーブの速度変化量 (kHz)} = \text{RESOL} * 10^{-3}$$

■ RATE DATA TABLE (参考)

RATE (ms/kHz)	RESOL = 1	RESOL = 5	RESOL = 10	RESOL = 20	RESOL = 50	RESOL = 200	RESOL = 400
	U/D CYCLE	U/D CYCLE	U/D CYCLE	U/D CYCLE	U/D CYCLE	U/D CYCLE	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	16,000
10	20	100	200	400	1,000	4,000	8,000
5	10	50	100	200	500	2,000	4,000
3	6	30	60	120	300	1,200	2,400
2	4	20	40	80	200	800	1,600
1	2	10	20	40	100	400	800
0.5	1	5	10	20	50	200	400
0.3		3	6	12	30	120	240
0.2		2	4	8	20	80	160
0.1		1	2	4	10	40	80
0.05			1	2	5	20	40
0.03					3	12	24
0.02					2	8	16
0.01					1	4	8
0.005						2	4
0.0025						1	2
0.00125							1

(3) 直線加減速ドライブ

直線加減速ドライブは、S字加減速の変速領域を "0" に設定して、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ます。
- ・ 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。

● 直線加速カーブ

SCAREA SET コマンドの SUAREA を "0" に設定します。

SHAREA SET コマンドの SUH を "0" に設定します。

- ・ SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合は、SUH の設定は不要です。

開始速度から最高速度まで、UCYCLE の直線加速カーブで加速します。

● 直線減速カーブ

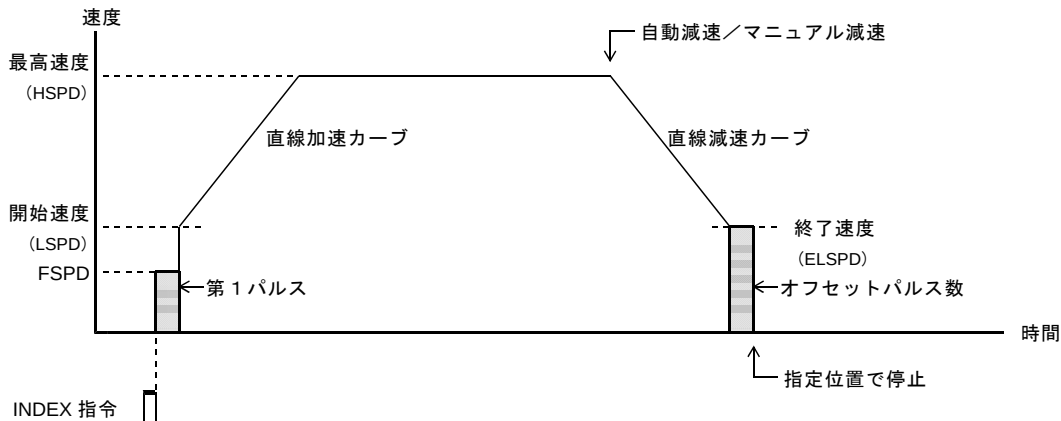
SCAREA SET コマンドの SDAREA を "0" に設定します。

SHAREA SET コマンドの SDH を "0" に設定します。

- ・ SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合は、SDH の設定は不要です。

最高速度から終了速度まで、DCYCLE の直線減速カーブで減速します。

■ 直線加減速ドライブの動作



オフセットパルス数の設定は、

SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 に設定している場合に有効です。

● 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) : $0 \leq TU < \text{最高速度の1周期}$

$$= (\text{UCYCLE} * 0.5 * 10^{-3}) * (\text{HSPD} - \text{LSPD} + 1) + TU + \text{第1パルスの周期 (ms)}$$

直線減速カーブの減速時間 (ms) : $0 \leq TD < \text{終了速度の1周期}$

$$= (\text{DCYCLE} * 0.5 * 10^{-3}) * (\text{HSPD} - \text{ELSPD} + 1) + TD$$

- ・ オフセットパルス数 = 0 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 (初期値 : +1) の増減があります。

(4) S 字加減速ドライブ

S 字加減速ドライブは、S 字加減速の変速領域を設定して、加減速を行うドライブです。

- ・ 加速開始部の S 字変速領域と加速終了部の S 字変速領域を、S 字加速カーブで加速します。
- ・ 減速開始部の S 字変速領域と減速終了部の S 字変速領域を、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 字加速開始部の変速領域を設定します。

SHAREA SET コマンドの SUH で S 字加速終了部の変速領域を設定します。

- ・ SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合は、SUH の設定は不要です。
SUH = SUAREA にして、S 字加速カーブを形成します。

SUAREA と SUH で設定した変速領域が、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 字減速終了部の変速領域を設定します。

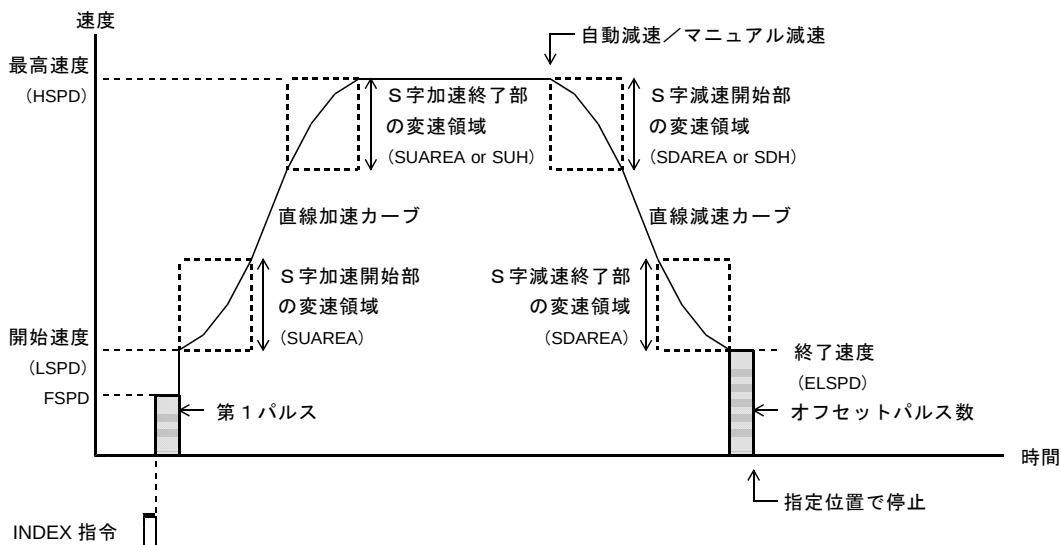
SHAREA SET コマンドの SDH で S 字減速開始部の変速領域を設定します。

- ・ SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合は、SDH の設定は不要です。
SDH = SDAREA にして、S 字減速カーブを形成します。

SDAREA と SDH で設定した変速領域が、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。

■ S 字加減速ドライブの動作



オフセットパルス数の設定は、SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 に設定している場合に有効です。

● 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) : $SUL = SUAREA$, $0 \leq TU < \text{最高速度の 1 周期}$

$= T1 + T2 + T3 + TU + \text{第 1 パルスの周期 (ms)}$

$= (UCYCLE * 0.5 * 10^{-3}) * (HSPD - LSPD + 1 + SUH + SUL) + TU + \text{第 1 パルスの周期 (ms)}$

- ・ S 字加速開始部の変速領域 (ms) : $T1 = (UCYCLE * 0.5 * 10^{-3}) * SUL * 2$
- ・ 直線加速カーブの変速領域 (ms) : $T2 = (UCYCLE * 0.5 * 10^{-3}) * (HSPD - LSPD + 1 - SUH - SUL)$
- ・ S 字加速終了部の変速領域 (ms) : $T3 = (UCYCLE * 0.5 * 10^{-3}) * SUH * 2$
- ・ $(HSPD - LSPD) \geq (SUH + SUL)$ 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) : $SDL = SDAREA$, $0 \leq TD < \text{終了速度の 1 周期}$

$= T4 + T5 + T6 + TD$

$= (DCYCLE * 0.5 * 10^{-3}) * (HSPD - ELSPD + 1 + SDH + SDL) + TD$

- ・ S 字減速開始部の変速領域 (ms) : $T4 = (DCYCLE * 0.5 * 10^{-3}) * SDH * 2$
- ・ 直線減速カーブの変速領域 (ms) : $T5 = (DCYCLE * 0.5 * 10^{-3}) * (HSPD - ELSPD + 1 - SDH - SDL)$
- ・ S 字減速終了部の変速領域 (ms) : $T6 = (DCYCLE * 0.5 * 10^{-3}) * SDL * 2$
- ・ $(HSPD - ELSPD) \geq (SDH + SDL)$ 、オフセットパルス数 = 0 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 (初期値 : +1) の増減があります。

● S 字加減速ドライブの加速パルス数と減速パルス数

S 字加速カーブの加速パルス数 : $SUL = SUAREA$

$= P1 + P2 + P3 + \text{第 1 パルス (+1)}$

$= (UCYCLE * 0.5 * 10^{-6}) * RESOL * (\frac{1}{2} * (HSPD - LSPD + 1) * (HSPD + LSPD) + SUH * HSPD + SUL * LSPD - \frac{1}{6} * (SUH - SUL) * (SUH + SUL + 3)) + 1$

- ・ S 字加速開始部のパルス数 : $P1 = (LSPD + \frac{1}{3} * SUL) * RESOL * T1 * 10^{-3}$
- ・ 直線加速カーブのパルス数 : $P2 = (HSPD - SUH + LSPD + SUL) * RESOL * \frac{1}{2} * T2 * 10^{-3}$
- ・ S 字加速終了部のパルス数 : $P3 = (HSPD - \frac{1}{3} * SUH) * RESOL * T3 * 10^{-3}$
- ・ $(HSPD - LSPD) \geq (SUH + SUL)$ で加速する場合の加速パルス数 (小数点以下は切り上げ) です。

S 字減速カーブの減速パルス数 : $SDL = SDAREA$

$= P4 + P5 + P6 + \text{ADJUST PULSE (+2)}$

$= (DCYCLE * 0.5 * 10^{-6}) * RESOL * (\frac{1}{2} * (HSPD - ELSPD + 1) * (HSPD + ELSPD) + SDH * HSPD + SDL * ELSPD - \frac{1}{6} * (SDH - SDL) * (SDH + SDL + 3)) + 2$
 $= (3 * (HSPD - ELSPD + 1) * (HSPD + ELSPD) + 6 * (SDH * HSPD + SDL * ELSPD) - (SDH - SDL) * (SDH + SDL + 3)) * RESOL * DCYCLE * \frac{1}{12} * 10^{-6} + 2$

- ・ S 字減速開始部のパルス数 : $P4 = (HSPD - \frac{1}{3} * SDH) * RESOL * T4 * 10^{-3}$
- ・ 直線減速カーブのパルス数 : $P5 = (HSPD - SDH + ELSPD + SDL) * RESOL * \frac{1}{2} * T5 * 10^{-3}$
- ・ S 字減速終了部のパルス数 : $P6 = (ELSPD + \frac{1}{3} * SDL) * RESOL * T6 * 10^{-3}$
- ・ $(HSPD - ELSPD) \geq (SDH + SDL)$ で減速停止する場合の減速パルス数 (小数点以下は切り上げ) です。

<参考> S 字減速カーブの減速パルス数 : $SD = SDL = SDH$ の場合

$= P4 + P5 + P6 + 2$

$= (DCYCLE * 0.5 * 10^{-6}) * RESOL * (\frac{1}{2} * (HSPD - ELSPD + 1) * (HSPD + ELSPD) + SD * HSPD + SD * ELSPD - \frac{1}{6} * (SD - SD) * (SD + SD + 3)) + 2$

$= (DCYCLE * 0.5 * 10^{-6}) * RESOL * \frac{1}{2} * (HSPD - ELSPD + 1 + 2 * SD) * (HSPD + ELSPD) + 2$

$= (HSPD - ELSPD + 1 + 2 * SD) * (HSPD + ELSPD) * RESOL * DCYCLE * \frac{1}{4} * 10^{-6} + 2$

■ S 字加減速 INDEX ドライブの三角駆動回避動作

● SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合

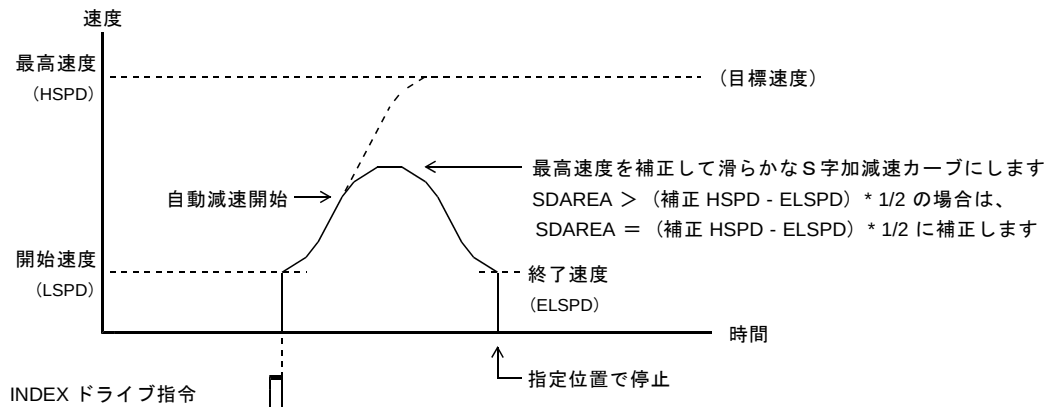
S 字加減速の INDEX ドライブで、

停止位置までのパルス数が少なく、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 字加速カーブで加速を行い、

S 字加速終了後から S 字減速カーブ（または補正した S 字減速カーブ）で減速を開始し、

指定位置で INDEX ドライブを停止します。



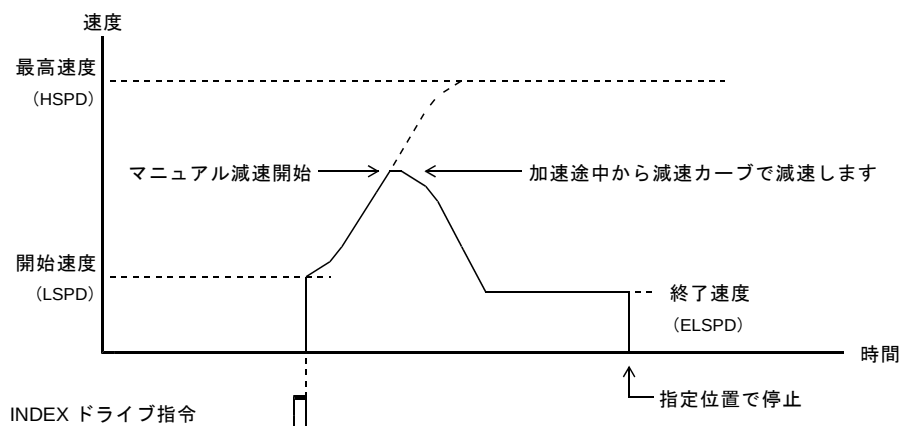
● SPEC INITIALIZE3 コマンドの SCAREA MODE = 1 の場合

このモードの INDEX ドライブでは、マニュアル設定した減速パルス数で減速を開始します。

停止位置までのパルス数が少なく、最高速度に達しないまま減速を開始した場合は、

加速途中から補正しない減速カーブで終了速度まで減速し、

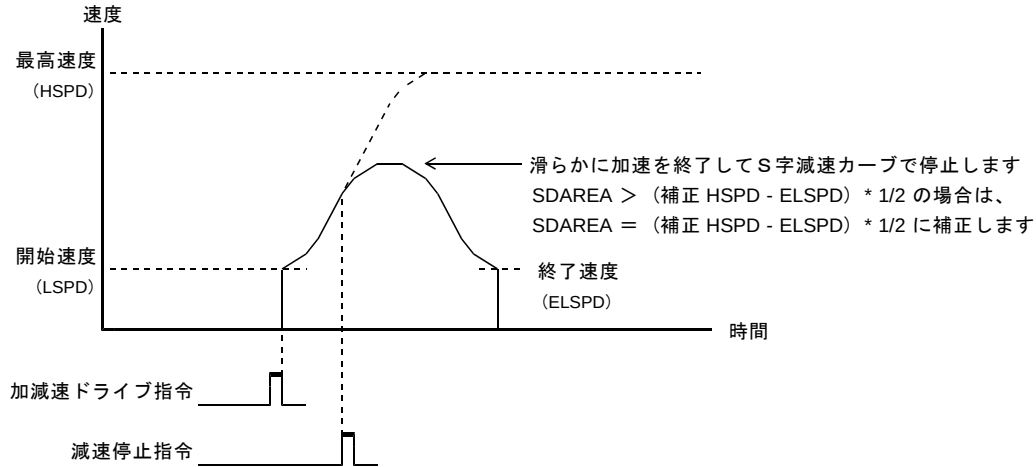
指定位置で INDEX ドライブを停止します。



■ 減速停止指令検出時の三角駆動回避動作

● SPEC INITIALIZE3 コマンドの SCAREA MODE = 0 の場合

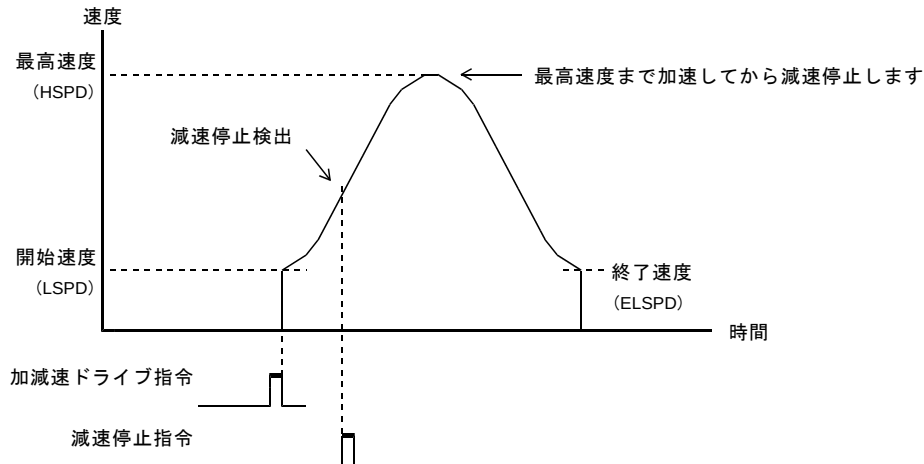
S字加速中に減速停止指令を検出した場合は、
SUAREA の S字加速終了カーブで滑らかに加速を終了し、
S字減速カーブ（または補正した S字減速カーブ）で減速停止します。



S字加減速の INDEX ドライブでは、減速停止指令を検出すると、自動減速機能をマスクして、
S字減速カーブ（または補正した S字減速カーブ）で減速停止します。
減速中に INDEX ドライブの指定位置を検出した場合は、指定位置で即時停止します。

● SPEC INITIALIZE3 コマンドの SCAREA MODE = 1 の場合

加速中に減速停止指令を検出した場合は、最高速度まで加速してから減速停止します。



減速停止指令検出後の加速動作中に、
INDEX ドライブのマニュアル設定した減速パルス数で減速を開始した場合は、
加速途中から補正しない減速カーブで減速を開始し、
終了速度または指定位置で INDEX ドライブを停止します。

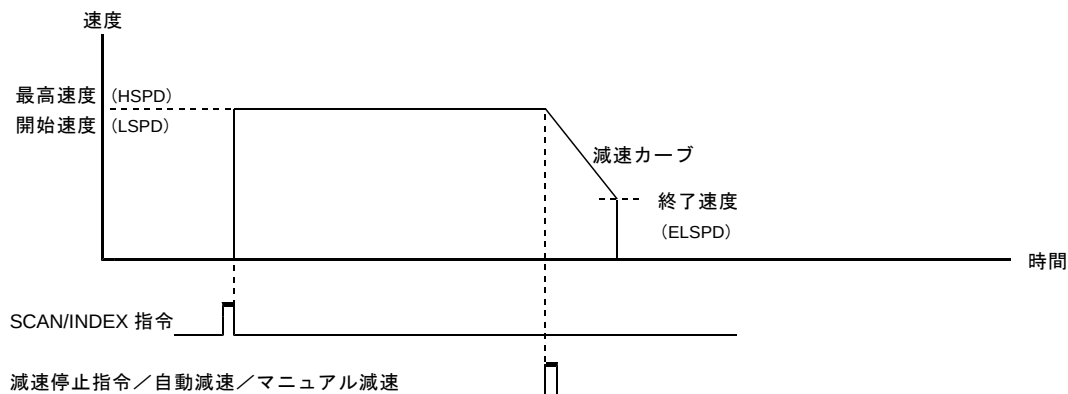
(5) 加速ドライブ

「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。



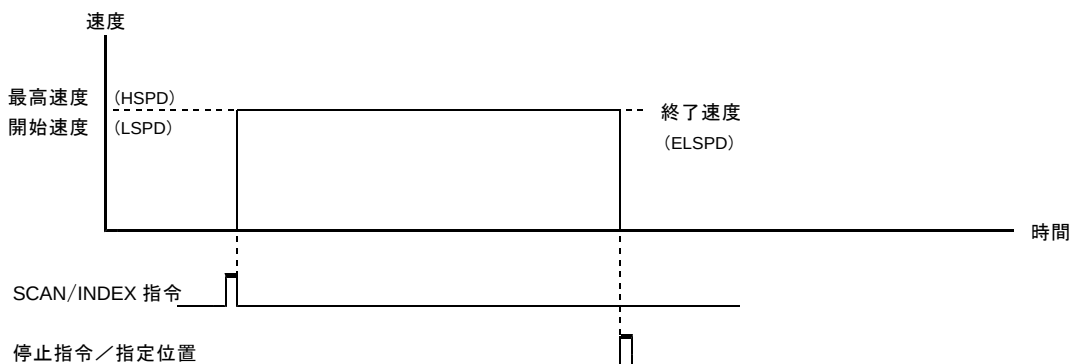
(6) 減速ドライブ

「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。



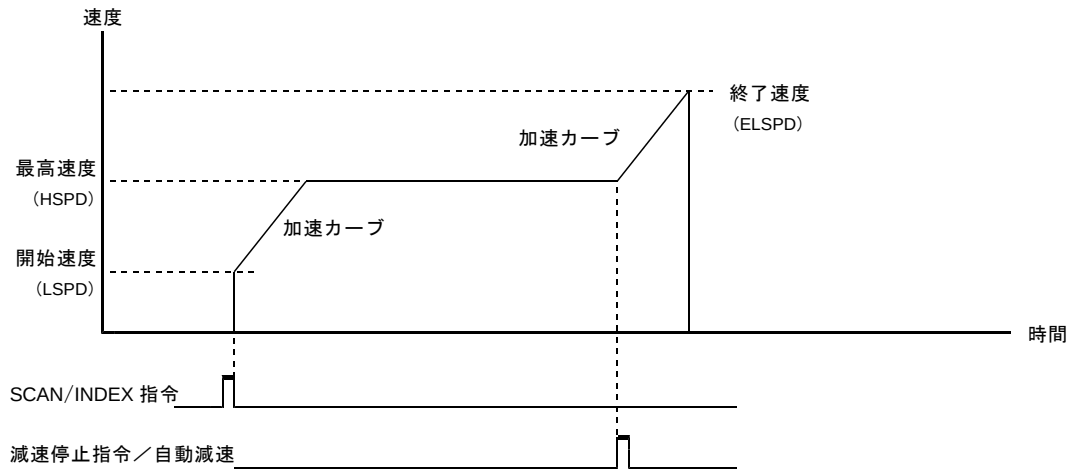
(7) 一定速ドライブ

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

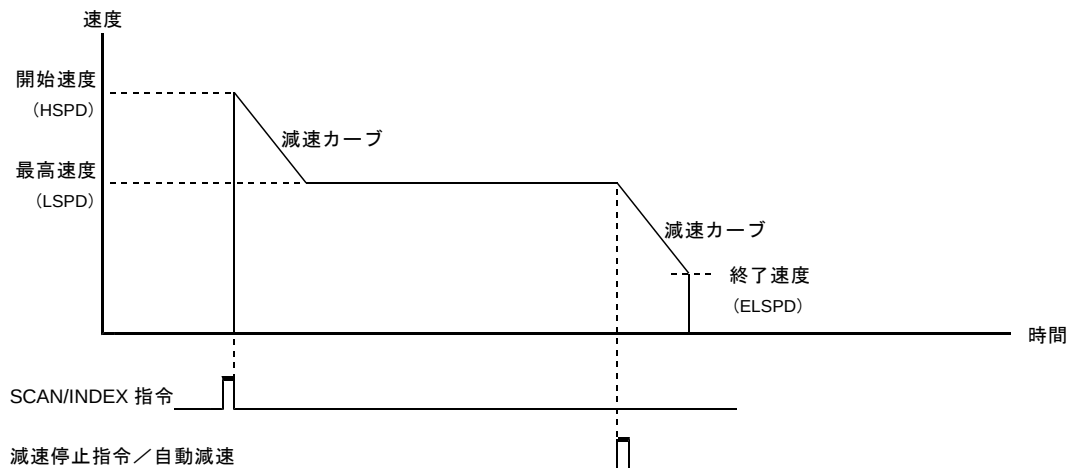


(8) その他のドライブ (SCAREA MODE = 0 の場合に可能)

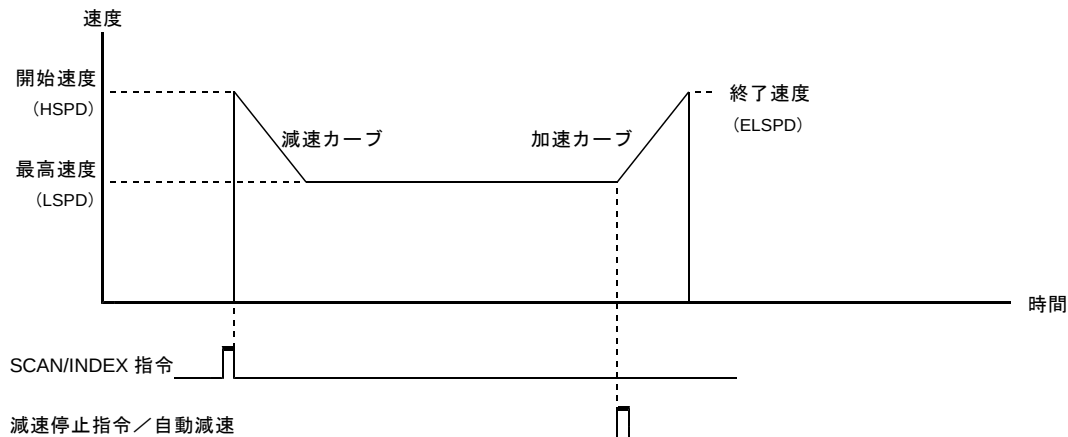
- 「開始速度<最高速度<終了速度」に設定すると、以下の加速ドライブを行います。



- 「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 (MCC09 チップの機械原点検出機能)

関数による ORIGIN ドライブ機能は、取扱説明書をご覧ください。

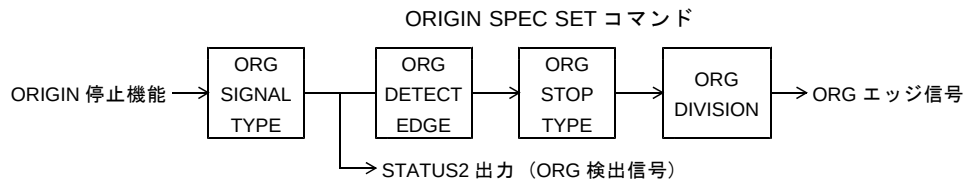
ORG エッジ信号を検出して、ドライブパルス出力を減速停止、即時停止、1 パルス停止します。

また、検出のみ行うことで、MCC09 の各種機能を実行するトリガ信号として使用できます。

検出する ORG 検出信号は、 $\overline{\text{ORG}}$ 信号、 $\pm \text{ZORG}$ 信号、 $\overline{\text{DEND}}/\text{PO}$ 信号、 NORG 信号の合成信号から選択します。

ORG エッジ信号の検出条件と停止機能は、ORIGIN SPEC SET コマンドで設定します。

- ・ ORG エッジ信号による停止機能は、STATUS1 PORT の DRIVE = 1 のときに有効です。
 - ・ ORG エッジ信号の停止機能が動作すると、STATUS2 PORT の ORGEND = 1 にします。
- STATUS1 PORT の SSEND, FSEND フラグは変化しません。



■ ORIGIN 停止機能による原点検出工程

任意のドライブを実行し、ORG エッジ信号の停止機能で原点検出を行います。

＜原点検出工程の実行例＞

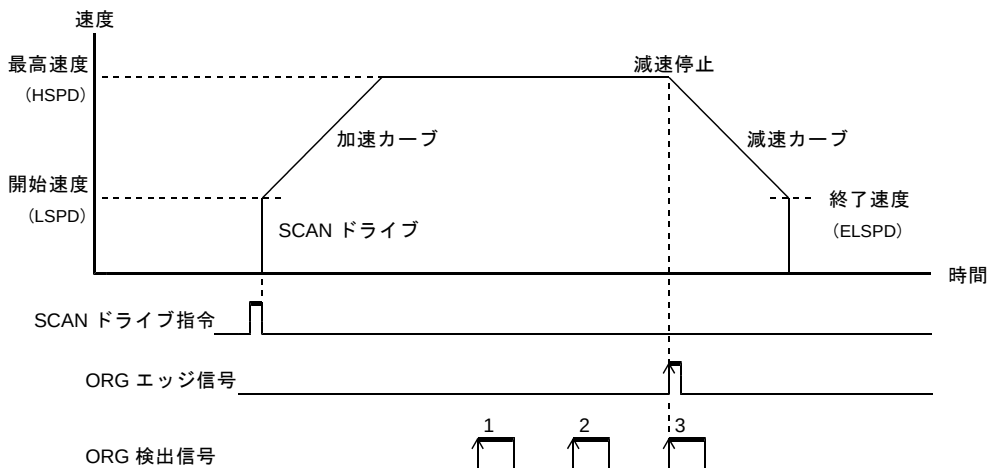
ORG 検出信号の 3 カウント目のアクティブエッジ検出で、減速停止動作させます。

- ・ 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
- ・ ORG STOP TYPE = "01" : ORG エッジ信号の停止機能を、減速停止にする
- ・ ORG DIVISION = H'02 : 3 カウント毎のエッジ信号 (ORG エッジ信号) を出力する

● ORIGIN SCAN ドライブ

加減速ドライブのパラメータで、SCAN (または INDEX) ドライブを実行します。

ORIGIN 停止機能の ORG エッジ信号を検出すると、減速停止します。



● ORIGIN CONSTANT SCAN ドライブ

JSPD SCAN（または JOG）ドライブを実行します。

または、一定速ドライブのパラメータで、SCAN（または INDEX）ドライブを実行します。

ORIGIN 停止機能の ORG エッジ信号を検出すると、減速なしで停止します。

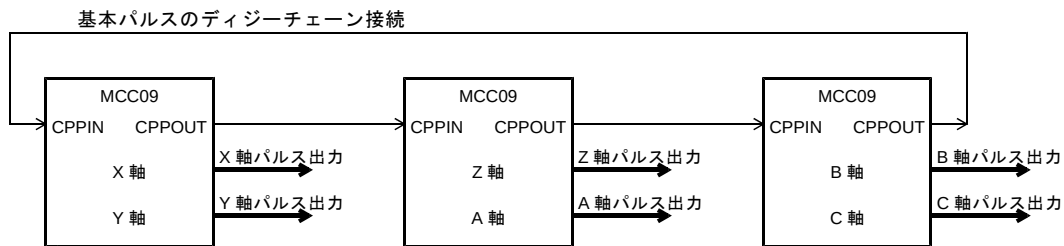
4-1-5. 補間ドライブ

補間ドライブには、1 軸単位で直線補間ドライブを実行するコマンドと、
1 軸単位で円弧補間ドライブを実行するコマンドがあります。
マルチチップの多軸直線補間ドライブと任意 2 軸の円弧補間ドライブができます。

(1) 任意軸補間ドライブ

MCC09 の CPPIN 端子と CPPOUT 端子をデジチェーン接続しており、
マルチチップ補間ドライブができます。

各軸 MCC09 の CPPOUT 端子と CPPIN 端子はデジ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 端子に出します。



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)

補間ドライブの基本となる加減速パルスは、メイン軸に設定した加減速パラメータで発生します。

- ・メイン軸のチップは、基本パルスを CPPOUT 端子から出力する設定にします。
- ・サブ軸のチップは、基本パルスを CPPIN 端子から入力して CPPOUT 端子に出力する設定にします。

マルチチップ直線補間ドライブは、1 つのメイン軸とその他のサブ軸で構成します。

- ・メイン軸には、メイン軸直線補間ドライブを実行します。
- ・サブ軸には、サブ軸直線補間ドライブを実行します。

任意 2 軸円弧補間ドライブは、1 つのメイン軸と 1 つのサブ軸で構成します。

- ・メイン軸には、メイン軸円弧補間ドライブを実行します。
- ・サブ軸には、サブ軸円弧補間ドライブを実行します。

● CPPOUT 出力

CP SPEC SET コマンドの CPPOUT SEL で選択したパルスを出します。

● CPPIN 入力

補間ドライブの基本パルスを入力します。

CPPIN 端子に入力できるパルス速度の最高値は、5 MHz です。

■ 直線補間ドライブの実行と停止機能

直線補間ドライブでは、各補間軸が独立して直線補間パルス（長軸・短軸パルス）を出力します。

- ・ 停止指令またはエラーの発生した補間軸は、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。
- ・ 補間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は、他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。

● メイン軸直線補間ドライブを実行した場合

マルチチップの直線補間ドライブが実行できます。

- ・ メイン軸直線補間ドライブのコマンドは、各サブ軸にサブ軸直線補間ドライブを実行した後（または同時に）、メイン軸に実行します。メイン軸はコマンドの実行でドライブを開始します。
- ・ メイン軸直線補間ドライブを実行すると、コマンド実行軸の加減速パラメータで、直線補間ドライブの基本パルスと直線補間パルスを発生します。メイン軸直線補間ドライブでは、線速一定制御した基本パルスを発生することができます。

メイン軸直線補間ドライブ実行中に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、出力中の基本パルスが OFF レベルのときに、基本パルス出力を停止して、ドライブを終了します。

● サブ軸直線補間ドライブを実行した場合

CPPIN 入力によるマルチチップの直線補間ドライブが実行できます。

- ・ サブ軸直線補間ドライブを実行すると、CPPIN に入力するパルスを直線補間ドライブの基本パルスにして、直線補間パルスを発生します。サブ軸直線補間ドライブは、コマンドの実行で STBY = 1 になります。CPPIN のハイレベルを検出すると、STBY = 0、DRIVE = 1 にして、ドライブを開始します。

サブ軸直線補間ドライブ実行中に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 OFF 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 OFF レベルのときに、ドライブを終了します。
- ・ 出力中の補間パルスが OFF レベルの場合は、そのまますぐにドライブを終了します。
- ・ 出力中の補間パルスがアクティブレベルの場合は、補間パルス出力が OFF レベルになると、ドライブを終了します。

● CPP STOP 機能と CPPIN マスク機能によるパルス出力の停止

この機能は、補間ドライブ実行コマンドのドライブ仕様で設定します。

CPPIN 端子と CPPOUT 端子をディジーチェーン接続したマルチチップ補間ドライブで、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。
- ・ エラーを検出すると、出力中の補間パルスが OFF レベルのときにドライブを終了します。出力中の補間パルスのアクティブレベルが続く場合は、エラー検出から 100 μ s 後に、補間パルス出力を OFF レベルにしてドライブを終了します。
- ・ 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。

■ 円弧補間ドライブの実行と停止機能

円弧補間ドライブでは、各補間軸が独立して円弧補間パルス（X, Y 座標パルス）を出力します。

- ・停止指令またはエラーの発生した補間軸は、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。
- ・補間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は、他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。

● メイン軸円弧補間ドライブを実行した場合

マルチチップの円弧補間ドライブが実行できます。

- ・メイン軸円弧補間ドライブのコマンドは、サブ軸にサブ軸円弧補間ドライブを実行した後（または同時）に、メイン軸に実行します。メイン軸はコマンドの実行でドライブを開始します。
- ・メイン軸円弧補間ドライブを実行すると、コマンド実行軸の加減速パラメータで、円弧補間ドライブの基本パルスと円弧補間パルスを発生します。メイン軸円弧補間ドライブでは、線速一定制御した基本パルスを発生することができます。

メイン軸円弧補間ドライブ実行中に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、出力中の基本パルスが OFF レベルのときに、基本パルス出力を停止して、ドライブを終了します。

● サブ軸円弧補間ドライブを実行した場合

CPPIN 入力によるマルチチップの円弧補間ドライブが実行できます。

- ・サブ軸円弧補間ドライブを実行すると、CPPIN に入力するパルスを円弧補間ドライブの基本パルスにして、円弧補間パルスを発生します。サブ軸円弧補間ドライブは、コマンドの実行で STBY = 1 になります。CPPIN のハイレベルを検出すると、STBY = 0、DRIVE = 1 にして、ドライブを開始します。

サブ軸円弧補間ドライブ実行中に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 OFF 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 OFF レベルのときに、ドライブを終了します。
- ・出力中の補間パルスが OFF レベルの場合は、そのまますぐにドライブを終了します。
- ・出力中の補間パルスがアクティブレベルの場合は、補間パルス出力が OFF レベルになると、ドライブを終了します。

● CPP STOP 機能と CPPIN マスク機能によるパルス出力の停止

この機能は、補間ドライブ実行コマンドのドライブ仕様で設定します。

CPPIN 端子と CPPOUT 端子をディジーチェーン接続したマルチチップ補間ドライブで、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、メイン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。
- ・エラーを検出すると、出力中の補間パルスが OFF レベルのときにドライブを終了します。出力中の補間パルスのアクティブレベルが続く場合は、エラー検出から 100 μ s 後に、補間パルス出力を OFF レベルにしてドライブを終了します。
- ・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。

(2) 直線補間ドライブ

マルチチップの多軸直線補間ドライブができます。

各補間軸は任意の長軸と短軸で座標を構成し、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、5 MHz 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、-2,147,483,647 ～ +2,147,483,647（32 ビット）です。

● マルチチップ用の 1 軸単位の補間ドライブ

H'30	MAIN STRAIGHT CP	メイン軸直線補間ドライブの実行
H'31	SUB STRAIGHT CP	サブ軸直線補間ドライブの実行

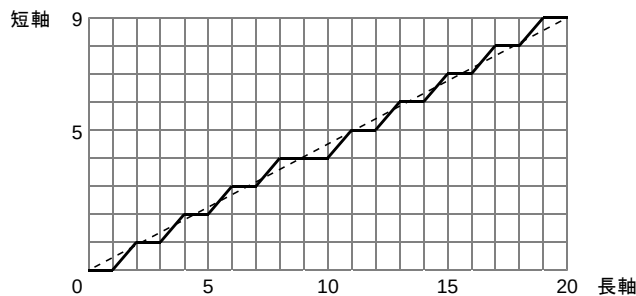
メイン軸直線補間ドライブには、以下のドライブパラメータの設定が必要です。

- ・実行軸の加減速ドライブのパラメータ
- ・CP SPEC : CPPOUT 出力、CPPIN 入力
- ・LONG POSITION : 補間軸の長軸の座標アドレス
- ・SHORT POSITION : 補間軸の短軸の座標アドレス

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、CPPIN 入力
- ・LONG POSITION : 補間軸の長軸の座標アドレス
- ・SHORT POSITION : 補間軸の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。

(3) 円弧補間ドライブ

マルチチップの任意 2 軸円弧補間ドライブができます。
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、5 MHz 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、-2,147,483,647 ~ +2,147,483,647 (32 ビット) です。

● 任意 2 軸用の 1 軸単位の円弧補間ドライブ

H'38	MAIN CIRCULAR CP	メイン軸円弧補間ドライブの実行
H'39	SUB CIRCULAR CP	サブ軸円弧補間ドライブの実行

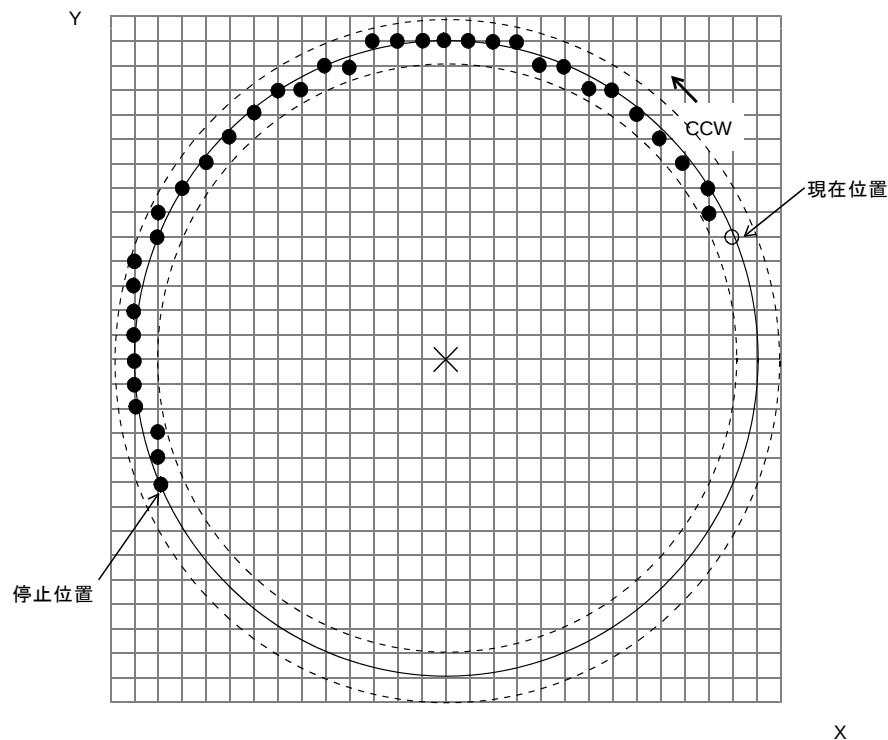
メイン軸円弧補間ドライブには、以下のドライブパラメータの設定が必要です。

- ・ 実行軸の加減速ドライブのパラメータ
- ・ CP SPEC : CPPOUT 出力、CPPIN 入力
- ・ 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

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、CPPIN 入力
- ・ 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。

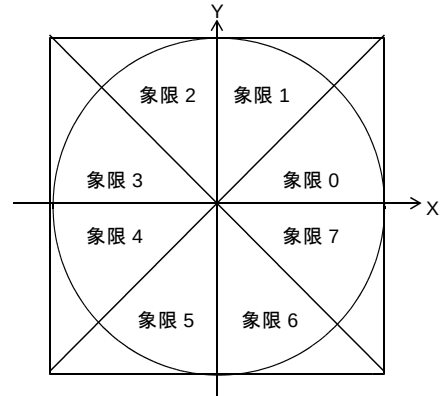
●円弧補間の短軸

円弧補間の中心点 (0, 0) としたときに

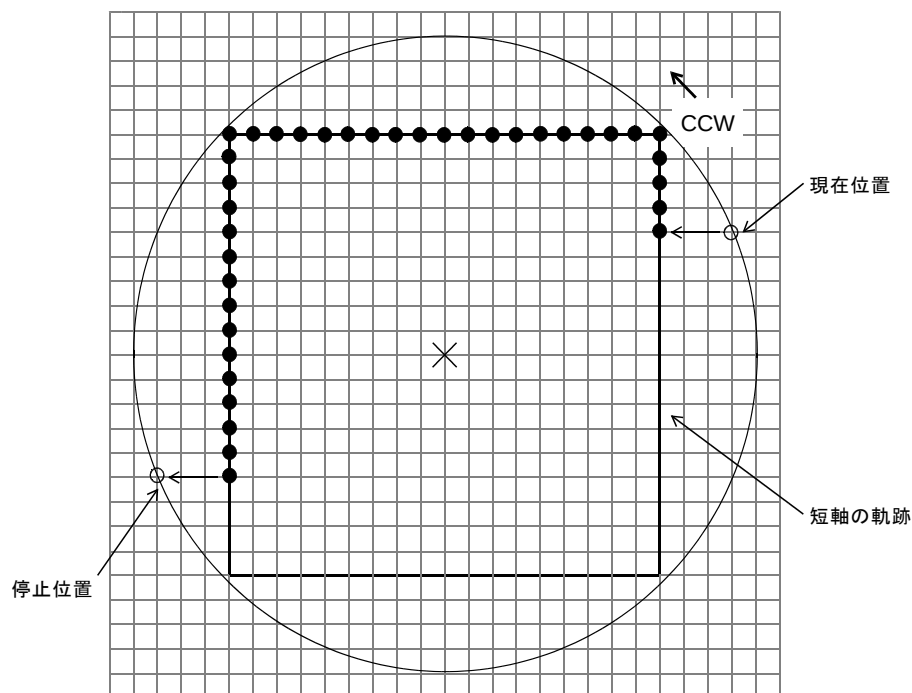
補間座標 (X, Y) の絶対値が小さい方の軸が短軸になります。

右図の 1, 2, 5, 6 象限は X 軸が短軸です。

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。

$$K = \text{int}(R / \sqrt{2}) \quad : \text{int}() \text{ は小数点以下を切り捨てた整数}$$

$$(1) R^2 \leq K^2 + (K + 1)^2 \text{ のとき}$$

$$\text{条件式: } |K^2 + (K + 1)^2 - R^2| > |K^2 + K^2 - R^2| \text{ のときは, } Ps = K$$

$$|K^2 + (K + 1)^2 - R^2| \leq |K^2 + K^2 - R^2| \text{ のときは, } Ps = K + 1/2$$

$$(2) R^2 > K^2 + (K + 1)^2 \text{ のとき}$$

$$\text{条件式: } |K^2 + (K + 1)^2 - R^2| > |(K + 1)^2 + (K + 1)^2 - R^2| \text{ のときは, } Ps = K + 1$$

$$|K^2 + (K + 1)^2 - R^2| \leq |(K + 1)^2 + (K + 1)^2 - R^2| \text{ のときは, } Ps = K + 1/2$$

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。

$$R = \sqrt{10^2 + 5^2} = \sqrt{125}, \quad K = \text{int}(R / \sqrt{2}) = \text{int}(7.9) = 7$$

$$R^2 = 125, \quad K^2 + (K + 1)^2 = 113, \quad \text{なので } R^2 > K^2 + (K + 1)^2$$

$$|K^2 + (K + 1)^2 - R^2| = 12, \quad |(K + 1)^2 + (K + 1)^2 - R^2| = 3, \quad \text{なので}$$

$$|K^2 + (K + 1)^2 - R^2| > |(K + 1)^2 + (K + 1)^2 - R^2| \text{ のときは, } Ps = K + 1 = 8$$

$$\begin{aligned} \text{短軸パルス数 } P &= \text{int}(\text{象限 0 のパルス数} + \text{象限 1, 2, 3 のパルス数} + \text{象限 4 のパルス数}) \\ &= (8 - 5) + (8 + 8 + 8) + 5 = 32 \end{aligned}$$

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。

$$R = \sqrt{20^2 + 5^2} = \sqrt{425}, \quad K = \text{int}(R / \sqrt{2}) = \text{int}(14.6) = 14$$

$$R^2 = 425, \quad K^2 + (K + 1)^2 = 421, \quad \text{なので } R^2 > K^2 + (K + 1)^2$$

$$|K^2 + (K + 1)^2 - R^2| = 4, \quad |(K + 1)^2 + (K + 1)^2 - R^2| = 25, \quad \text{なので}$$

$$|K^2 + (K + 1)^2 - R^2| \leq |(K + 1)^2 + (K + 1)^2 - R^2| \text{ のときは, } Ps = K + 1/2 = 14.5$$

$$\begin{aligned} \text{短軸パルス数 } P &= \text{int}(\text{象限 0 のパルス数} + \text{象限 7, 6, 5 のパルス数} + \text{象限 4 のパルス数}) \\ &= 5 + (14.5 + 14.5 + 14.5) + (14.5 - 5) = 58 \end{aligned}$$

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

(4) 線速一定制御

メイン軸直線補間ドライブで有効です。

メイン軸円弧補間ドライブで有効です。

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。

メイン軸直線／円弧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。

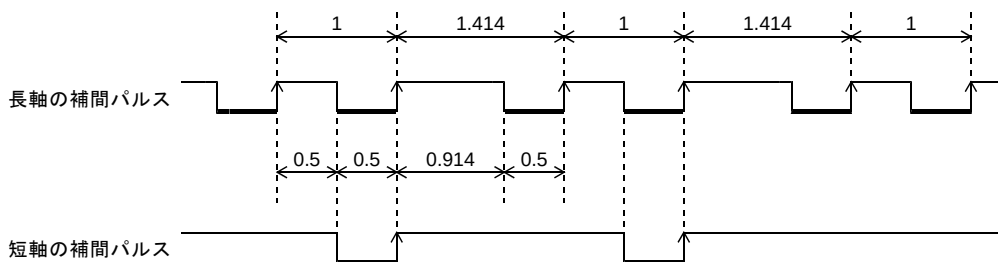
- ・直線補間ドライブでは、メイン軸の長軸と短軸の2軸間で、線速一定制御します。
- ・円弧補間ドライブでは、X座標軸とY座標軸の2軸間で、線速一定制御します。

線速一定で加減速ドライブを行うと、減速パルス数が増加します。

INDEXドライブの自動減速では、減速後の終了速度でのドライブが長くなります。

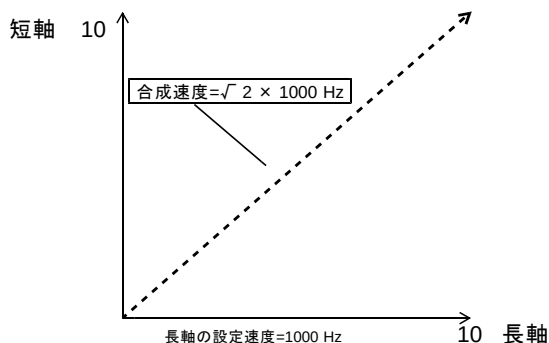
■ 線速一定の補間パルス出力（2軸直線補間ドライブの例）

アクティブレベルの幅はそのまま、OFFレベルの幅を長くします。



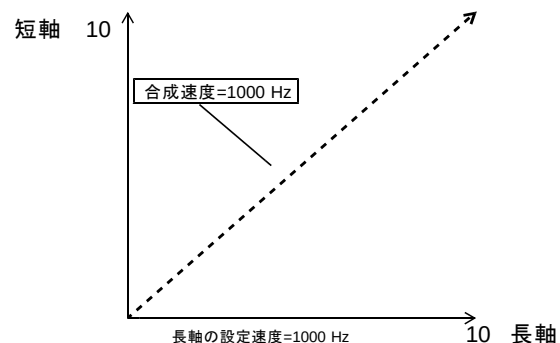
直線補間ドライブの軌跡(長軸 10:短軸 10 の例)

＜線速一定制御なしのとき＞



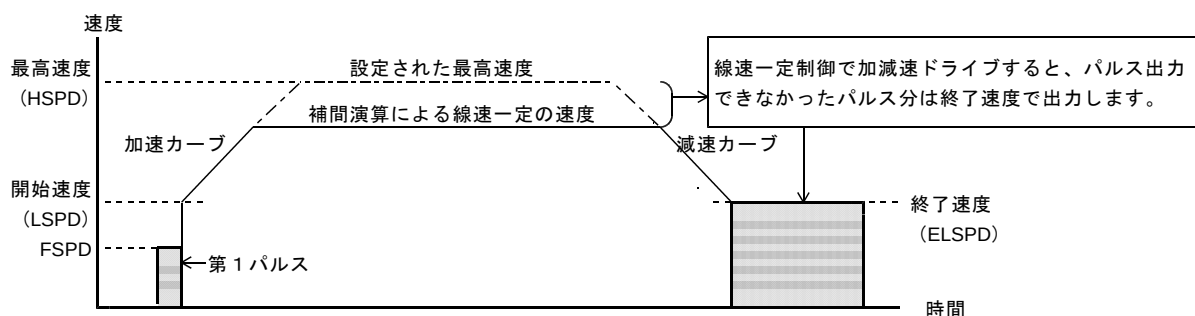
長軸の速度を一定速の 1000Hz とすると、
2 軸直線補間で描かれる軌跡の合成速度は
 $\sqrt{2} \times 1000\text{Hz}$ でドライブします。

＜線速一定制御ありのとき＞



コマンド実行軸の速度を一定速の 1000Hz として
線速一定制御を設定すると、2 軸直線補間で
描かれる軌跡の合成速度が 1000Hz となるように
ドライブします。

※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。



4-1-6. ドライブ CHANGE 機能

ドライブ CHANGE 機能には、SPEED CHANGE 機能と INDEX CHANGE 機能があります。
ドライブ CHANGE 指令は、予約して実行することができます。

(1) SPEED CHANGE 機能

SPEED CHANGE 指令には、以下のコマンドがあります。

- ・ 設定コマンド
SPEED CHANGE SPEC SET コマンド : SPEED CHANGE 指令を実行する変更動作点
- ・ 実行コマンド
SPEED RATE CHANGE コマンド : SPEED & RATE CHANGE 指令

SPEED CHANGE の状態は、STATUS5 PORT の SPEED EP と SPEED FL フラグで確認します。

- ・ SPEED FL = 0 のときに、変更動作点を設定して、SPEED RATE CHANGE コマンドを書き込みます。
SPEED CHANGE SPEC SET コマンドを実行しても、SPEED FL = 0 のままです。
SPEED RATE CHANGE コマンドを書き込むと、SPEED CHANGE 専用のレジスタに格納します。
同時に SPEED FL = 1、SPEED EP = 0 にします。
- ・ DRIVE = 1 のときに、変更動作点を検出して、SPEED RATE CHANGE コマンドを実行します。
同時に SPEED FL = 0、SPEED EP = 1 にします。

変更動作点を検出して、SPEED RATE CHANGE コマンドを実行すると、

指定したドライブパルス速度まで (SPEED CHANGE)、

指定した変速周期データで (RATE CHANGE)、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。
- ・ 直線加減速ドライブ (SUAREA = 0, SDAREA = 0) の加減速中に CHANGE 指令を実行した場合は、
現在の速度から、指定した速度まで指定した変速周期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 CHANGE 指令を実行した場合は、以下のようになります。
S 字加減速ドライブでは、現在の加減速カーブを終了するまでは SPEED FL = 1 のままです。

現在の加速または減速状態で、指定した速度に到達できる場合は、

現在の変速周期のまま加速または減速します。この場合は、指定した変速周期は無効になります。

現在の加速または減速状態で、指定した速度に到達できない場合は、

現在の加速または減速状態を、現在の変速周期の S 字カーブで滑らかに終了させてから、
指定した速度まで指定した変速周期で加速または減速します。

- ・ 減速停止動作時および自動減速動作中の INDEX CHANGE による再加速時の変速周期は、
RATE SET コマンドで設定した UCYCLE または DCYCLE の変速周期で動作します。

指定するドライブパルス速度を "0" に設定すると、CONST DRIVE 指令になります。

CONST DRIVE 指令を実行すると、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、CONST DRIVE 指令の実行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、CONST DRIVE 指令を実行すると、
現在の変速周期の S 字カーブで滑らかに加速または減速を終了して、一定速にします。

SPEED RATE CHANGE コマンドを実行しても、ドライブパラメータの設定は変わりません。

■ SPEED CHANGE 指令が有効となるドライブ（コマンド）

補間ドライブでは、基本パルス発生軸の加減速パラメータに対して CHANGE 指令を実行します。

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'30	MAIN STRAIGHT CP	*P	メイン軸直線補間ドライブの実行
H'38	MAIN CIRCULAR CP	*P	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 DRIVE = 1 から、変更動作点の検出が有効になります。
変更動作点を検出すると、格納している SPEED RATE CHANGE コマンドを実行します。

● SPEED CHANGE 指令が自動的に無効となる状態

以下の状態を検出すると、格納している SPEED RATE CHANGE コマンドをクリアします。

SPEED FL = 1、SPEED EP = 1 にして、SPEED CHANGE コマンドの書き込みを無効にします。

- ・ SPEC INITIALIZE3 コマンドの SCAREA MODE = 1 の検出
- ・ STATUS1 PORT の ERROR = 1 の検出
- ・ DRIVE = 1 のときに、LSEND = 1、SSEND = 1、FSEND = 1、ORGEND = 1 の検出
- ・ INDEX ドライブの自動減速地点の検出
- ・ 反転動作が必要な INDEX CHANGE 指令の検出
- ・ MANUAL SCAN ドライブの減速地点の検出
- ・ DRIVE = 1 → 0 の検出（格納している実行コマンドのクリアのみ行います）
- ・ SERVO RESET コマンドの実行

SCAREA MODE = 0、ERROR = 0、DRIVE = 0 になると、

SPEED FL = 0、SPEED EP = 1 にして、SPEED CHANGE コマンドの書き込みを有効にします。

(2) INDEX CHANGE 機能

INDEX CHANGE 指令には、以下のコマンドがあります。

- ・ 設定コマンド
INDEX CHANGE SPEC SET コマンド : INDEX CHANGE 指令を実行する変更動作点
反転ドライブの第 1 パルス周期 (RFSPD)
- ・ 実行コマンド
INC INDEX CHANGE コマンド : INC INDEX CHANGE 指令
ABS INDEX CHANGE コマンド : ABS INDEX CHANGE 指令
PLS INDEX CHANGE コマンド : PLS INDEX CHANGE 指令

INDEX CHANGE の状態は、STATUS5 PORT の INDEX EP と INDEX FL フラグで確認します。

- ・ INDEX FL = 0 のときに、変更動作点を設定して、INDEX CHANGE 実行コマンドを書き込みます。
INDEX CHANGE SPEC SET コマンドを実行しても、INDEX FL = 0 のままです。
INDEX CHANGE 実行コマンドを書き込むと、INDEX CHANGE 専用のレジスタに格納します。
同時に INDEX FL = 1、INDEX EP = 0 にします。
- ・ DRIVE = 1 のときに、変更動作点を検出して、INDEX CHANGE 実行コマンドを実行します。
同時に INDEX FL = 0、INDEX EP = 1 にします。

任意の変更動作点の検出で、INDEX CHANGE コマンドを実行します。

- ・ 変更動作点を検出して、INC INDEX CHANGE コマンドを実行すると、
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。
- ・ 変更動作点を検出して、ABS INDEX CHANGE コマンドを実行すると、
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。
- ・ 変更動作点を検出して、PLS INDEX CHANGE コマンドを実行すると、
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。

PLS INDEX CHANGE コマンドを実行すると、INC INDEX ドライブの原点を変更します。

- ・ PLS INDEX CHANGE コマンド実行後に、INC INDEX CHANGE コマンドを実行すると、
PLS INDEX CHANGE コマンドの変更動作点の検出位置を原点として INC INDEX ドライブを行います。

反転動作が必要な INDEX CHANGE 指令を検出した場合は、
終了速度まで減速停止後に反転ドライブを行い、指定の停止位置まで移動します。

■ INDEX CHANGE 指令が有効となるドライブ (コマンド)

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 DRIVE = 1 から、変更動作点の検出が有効になります。
変更動作点を検出すると、格納している INDEX CHANGE 実行コマンドを実行します。

● INDEX CHANGE 指令が自動的に無効となる状態

以下の状態を検出すると、格納している INDEX CHANGE 実行コマンドをクリアします。

INDEX FL = 1、INDEX EP = 1 にして、INDEX CHANGE コマンドの書き込みを無効にします。

- ・ SPEC INITIALIZE3 コマンドの DOWN PULSE MASK = 1 の検出
- ・ SPEC INITIALIZE3 コマンドの SCAREA MODE = 1 の検出
- ・ STATUS1 PORT の ERROR = 1 の検出
- ・ DRIVE = 1 のときに、LSEND = 1、SSEND = 1、FSEND = 1、ORGEND = 1 の検出
- ・ DRIVE = 1 → 0 の検出 (格納している実行コマンドのクリアのみ行います)
- ・ SERVO RESET コマンドの実行

DOWN PULSE MASK = 0、SCAREA MODE = 0、ERROR = 0、DRIVE = 0 になると、
INDEX FL = 0、INDEX EP = 1 にして、INDEX CHANGE コマンドの書き込みを有効にします。

● INDEX CHANGE 指令によるエラー

内部カウンタの相対アドレス範囲は、-2,147,483,647 ～ +2,147,483,647 (32 ビット) です。
出力パルス数範囲は、0 ～ 2,147,483,647 (31 ビット) です。

反転動作が必要な INDEX CHANGE 指令を検出して、反転ドライブを行う場合に、
反転ドライブの出力パルス数が 2,147,483,647 を超えて、オーバーフローしてしまう場合があります。
この場合は、ERROR STATUS の INDEX ERROR = 1 にします。

STATUS1 PORT の ERROR = 1 となり、実行中のドライブを終了速度まで減速して停止します。

- ・ INDEX CHANGE 指令検出後の反転ドライブで、出力パルス数がオーバーフローした

SCAN ドライブの実行または反転動作が必要な INC INDEX CHANGE 指令の実行により、
内部カウンタの相対アドレスがオーバーフローしてしまう場合があります。

この場合は、ERROR STATUS の INDEX ERROR = 1 にします。

STATUS1 PORT の ERROR = 1 となり、実行中のドライブを終了速度まで減速して停止します。

- ・ INC INDEX CHANGE 指令検出後のドライブで、内部の相対アドレスがオーバーフローした

SCAN ドライブの実行または反転動作が必要な ABS INDEX CHANGE 指令の実行により、
アドレスカウンタがオーバーフローしてしまう場合があります。

この場合は、ERROR STATUS の INDEX ERROR = 1 にします。

STATUS1 PORT の ERROR = 1 となり、実行中のドライブを終了速度まで減速して停止します。

- ・ ABS INDEX CHANGE 指令検出後のドライブで、アドレスカウンタがオーバーフローした

■ INDEX CHANGE コマンド実行前と実行後のドライブの停止

INDEX CHANGE コマンド実行前と実行後では、INDEX ドライブの減速停止動作が異なります。

● INDEX CHANGE コマンド実行前の INDEX ドライブの停止

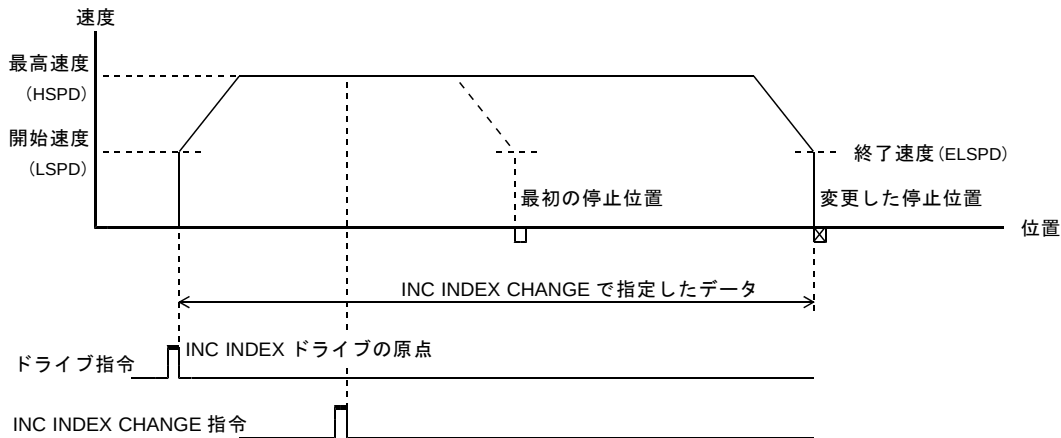
- ・ 自動減速停止動作開始後に指定アドレスを検出すると即時停止します。
- ・ ドライブ中に減速停止指令を検出した場合は、終了速度まで減速して指定アドレス内で停止します。
減速中に指定アドレスを検出した場合は、指定アドレスで即時停止します。
- ・ ドライブ中に ERROR = 1 を検出した場合は、終了速度まで減速して指定アドレス内で停止します。
減速中に指定アドレスを検出した場合は、指定アドレスで即時停止します。

● INDEX CHANGE コマンド実行後の INDEX ドライブの停止

- ・ INDEX CHANGE コマンドの実行で "指定アドレスの検出で停止" をマスクします。
終了速度になるとマスクを一時解除します。DRIVE = 0 の検出でマスクを完全に解除します。
- ・ 自動減速停止動作開始後は、終了速度まで減速すると指定アドレスを検出して停止します。
ドライブ中に指定アドレスを超えた場合は、終了速度まで減速停止後に反転ドライブを行います。
反転ドライブでは、自動減速停止動作開始後に指定アドレスを検出すると即時停止します。
- ・ ドライブ中に減速停止指令を検出した場合は、終了速度まで減速して停止します。
減速中に指定アドレスを検出しても停止しません。終了速度で停止します。
- ・ ドライブ中に ERROR = 1 を検出した場合は、終了速度まで減速して停止します。
減速中に指定アドレスを検出しても停止しません。終了速度で停止します。

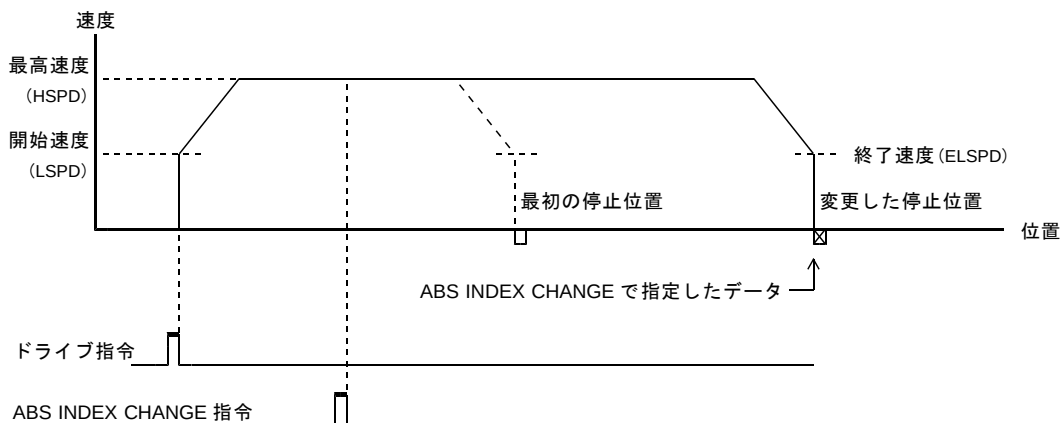
■ INC INDEX CHANGE の動作

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。



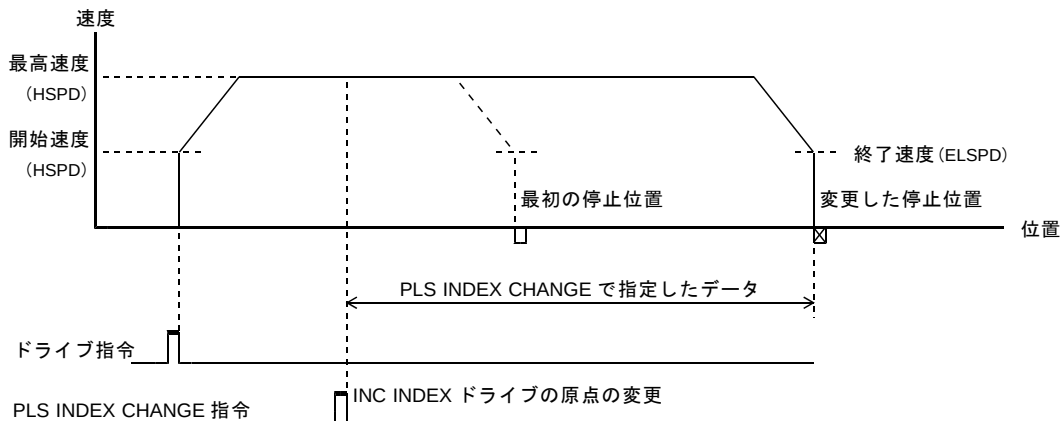
■ ABS INDEX CHANGE の動作

指定したデータを、アドレスカウンタが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

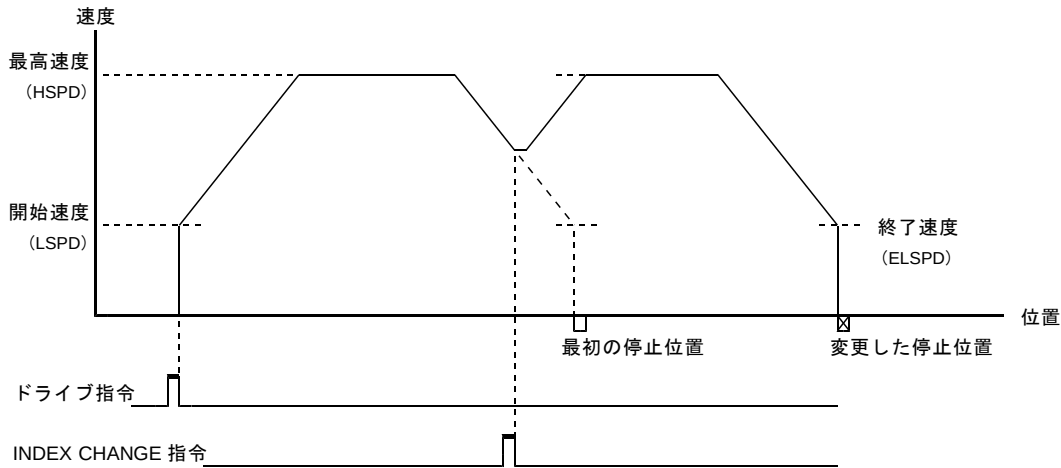
指定したデータを、変更動作点の検出位置を原点とする相対アドレスの停止位置にします。



■ 減速中の INDEX CHANGE 動作

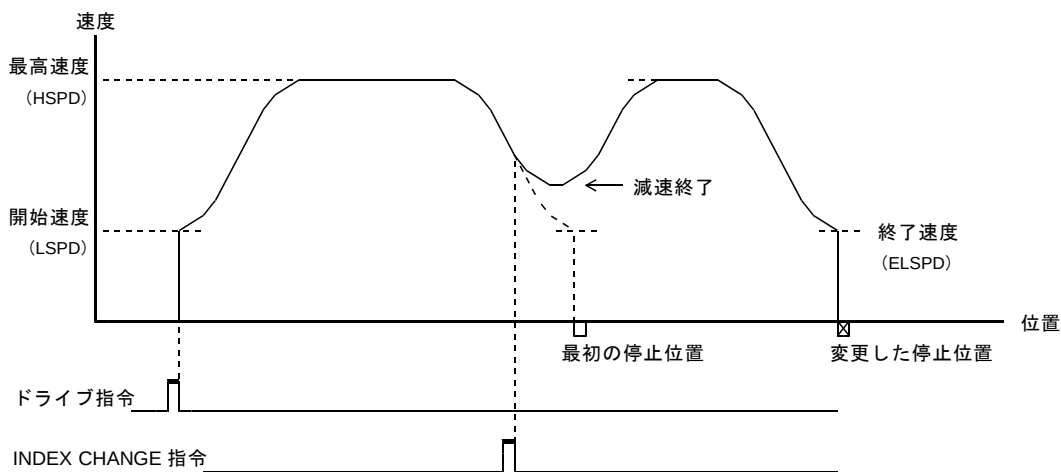
● 直線加減速ドライブの減速中の INDEX CHANGE

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。



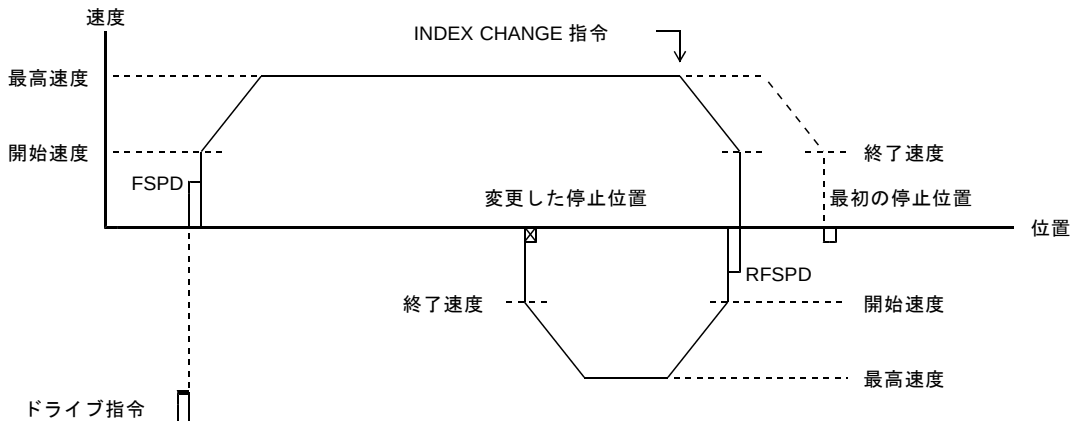
● S字加減速ドライブの減速中の INDEX CHANGE

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。



■ INDEX CHANGE の反転ドライブ

終了速度まで減速停止後に反転ドライブを行い、指定の停止位置まで移動します。



● INDEX CHANGE 指令による反転ドライブ

反転動作が必要な INDEX CHANGE 指令を検出した場合は、以下ようになります。

- ・ INDEX CHANGE コマンドの実行で "指定アドレスの検出で停止" をマスクします。
終了速度になるとマスクを一時解除します。DRIVE = 0 の検出でマスクを完全に解除します。
- ・ 実行中のドライブを終了速度まで減速して停止します。
 - ・ STATUS1 PORT の DRIVE = 1 → 0 になります。
 - ・ DRIVE = 0 の検出で "指定アドレスの検出で停止" のマスクを完全に解除します。
 - ・ DEND 信号の<サーボ対応>も実行します。
 - ・ この間は、BUSY = 1 のままです。
- ・ LSEND = 0、SSEND = 0、FSEND = 0、ORGEND = 0、ERROR = 0 のときに、
指定アドレスを超えて停止した場合は、反転ドライブを実行し、指定の停止位置まで移動します。
 - ・ 反転ドライブの第1パルス周期は、FSPD ではなく RFSPD になります。
RFSPD は、INDEX CHANGE SPEC SET コマンドで設定します。
 - ・ その他のドライブパラメータは、INC/ABS INDEX ドライブの設定値で動作します。
 - ・ 反転ドライブの自動減速停止動作開始後に指定アドレスを検出すると即時停止します。
 - ・ 停止後には、DEND 信号の<サーボ対応>も実行します。

反転ドライブ中に、反転動作が必要な INDEX CHANGE 指令を検出した場合は、上記の動作を繰り返します。

以下の状態を検出した場合は、指定アドレスを超えて停止しても、反転ドライブを実行しません。

- ・ STATUS1 PORT の ERROR = 1、LSEND = 1、SSEND = 1、FSEND = 1 の検出
- ・ STATUS2 PORT の ORGEND = 1 の検出

4-1-7. 読み出し機能

下記データの読み出しは、取扱説明書(標準)の 3-7.章の制限事項を除き、常時可能です。

下記の関数にて、コマンドの書き込みと、データの読み出しを一括で処理することができます。
これらの関数は、コマンドの書き込みからデータの読み出しまで、関数内で排他処理されています。
マルチスレッドプログラミングのように、複数のスレッドで処理される場合は、この関数によって読み出しすることを推奨します。

- ・ デバイス単位 … DRIVE COMMAND 32 ビット書き込み／読み出し関数
DRIVE COMMAND PORT 書き込み／読み出し関数

(1) カウントデータのラッチデータ読み出し

MCCC09 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。

4-1-8. RSPD 機能

RSPD とは、ドライブ終了時に MCC09 が記憶している終了速度です。

RSPD 設定方法

- ・ JOG パルス速度 (JSPD) を "0" に設定すると、 $JSPD = RSPD \times RESOL$ に補正して JSPD が設定されます。
 - ・ 第 1 パルス周期 (FSPD) を "0" に設定すると、 $FSPD = RSPD \times RESOL$ に補正して FSPD が設定されます。
 - ・ INDEX CHANGE の反転開始速度 (RFSPD) を "0" に設定すると、 $RFSPD = RSPD \times RESOL$ に補正して RFSPD が設定されます。
 - ・ 最高速度 (HSPD) を "0" に設定すると、 $HSPD = RSPD \times RESOL$ に補正して HSPD が設定されます。
 - ・ 最低速度 (LSPD) を "0" に設定すると、 $LSPD = RSPD \times RESOL$ に補正して LSPD が設定されます。
- 終了速度 (ELSPD) を "0" に設定した場合は、ELSPD は、 $ELSPD = LSPD$ に補正されます。

但し、以下の場合は、RSPD は書き換えません。

- ・ 最終出力のパルス速度が FSPD のとき
- ・ 最終出力のパルス速度が JSPD のとき

RSPD の電源投入後の初期値は、H'012C (300) です。

RSPD の読み出し

RSPD DATA READ コマンドにより、FSPD、HSPD、LSPD と同様のパルス速度データが読み出しできます。

● 応用例

連続したドライブなど、2 回目の開始速度を MCC09 が記憶している速度でドライブを再開して、変速のないドライブパターンのつなぎ方が可能です。

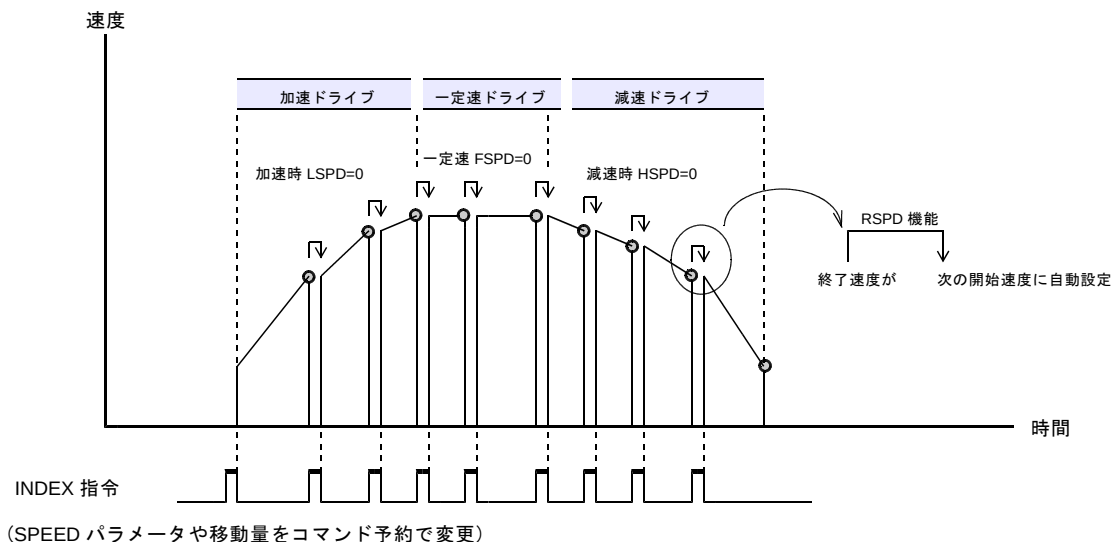
一回の位置決め量が少ないときは、加速ドライブパターンで設定した最高速度、減速ドライブパターンで設定した終了速度まで到達せずに、ドライブを終了する可能性があります。

このときの終了した速度は、通常アプリケーション側から再設定することはできません。

これを RSPD 機能により、前の区間のドライブの終了速度を、次のドライブ開始速度として、自動的に再設定することを可能にしています。

RSPD 機能と加減速時定数 (RATE) の変更や任意なドライブパターンを組み合わせると、コマンド予約機能を使用すると、任意なドライブ形状を作ることが可能です。

下記の例で示す○印が MCC09 が記憶する終了速度 (RSPD) です。



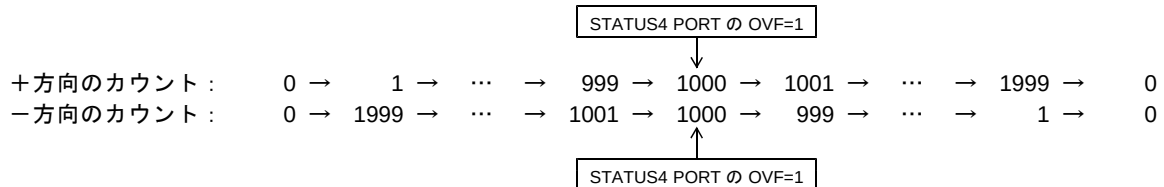
4-2. カウンタ仕様

4-2-1. リングカウンタ機能

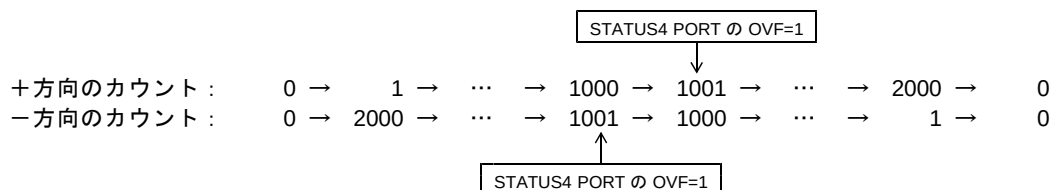
アドレスカウンタ、およびパルスカウンタは、カウント最大値を設定すると、設定値をカウンタの最大値としてリングカウントします。

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
・	他軸の STATUS3 PORT の OUT3 = 0 → 1 でラッチする
・	STATUS3 PORT の GPIO2 = 0 → 1 でラッチする
・	ORIGIN 停止機能の ORG エッジ信号の検出でラッチする
・	STATUS3 PORT の OUT2 = 0 → 1 でラッチする
・	STATUS3 PORT の OUT3 = 0 → 1 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。

(注) GPIO2 (SS0) 信号は SPEC INITIALIZE2 コマンドで SS0 機能が「汎用入力」に設定している場合に有効です。

・SS0 信号は下記の軸に対応しています。(初期値)

/IN0 信号	X 軸の SS0 信号
/IN1 信号	Y 軸の SS0 信号

■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。

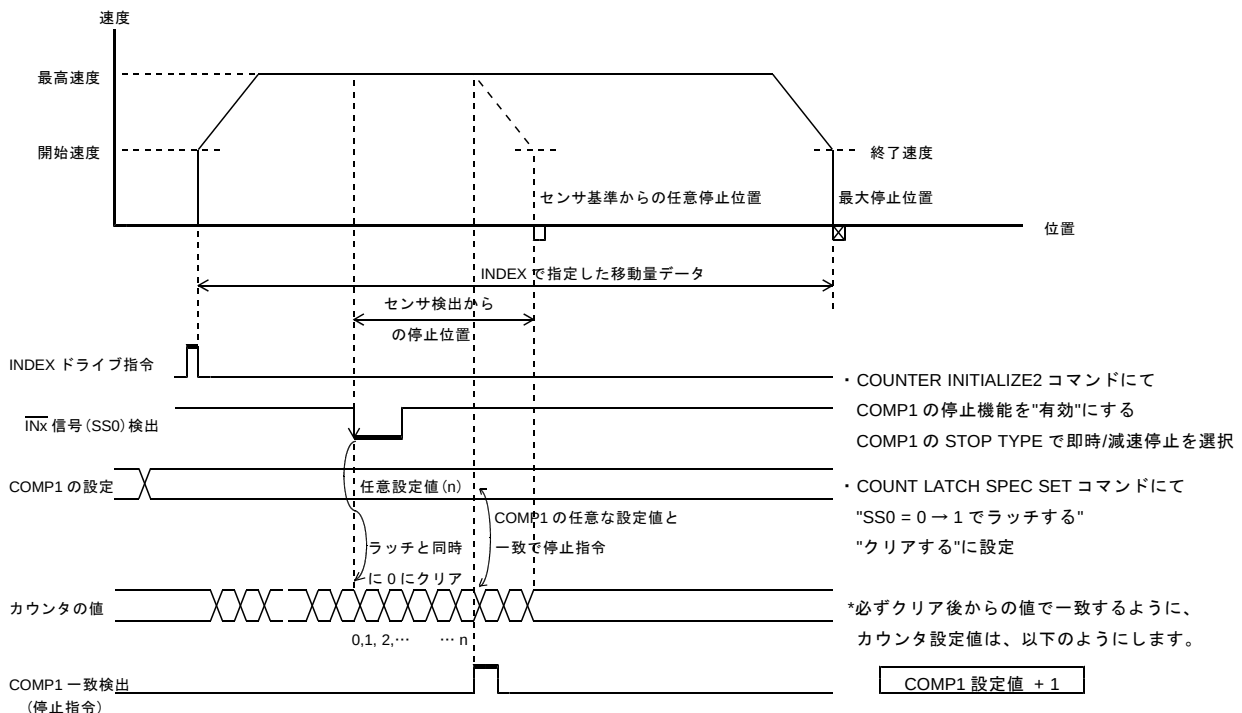
カウンタのカウントタイミングとクリアタイミングが同時に発生した場合はクリアを優先します。

・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。

●カウンタラッチとクリア機能を応用したドライブ例

センサ (/IN0 信号または /IN1 信号) の検出を基準とした位置から、カウンタのコンパレータ一致による停止機能でドライブを停止させることが可能です。

- ・ドライブ起動前に必要な設定を済ませることができ、ドライブ中の Windows OS や AL-II 通信などの影響を受けない繰り返し精度の高いセンサ検出によるドライブが可能です。
- ・予め INDEX ドライブを起動することで、センサ信号検出しない場合でもメカの最大移動量は保証できます。
- ・センサ検出から減速開始または即時停止するまでの移動量をコンパレータの設定値により調整できます。
減速停止時の減速開始から出力されるパルス数は保証されます。
- ・ドライブ中の他の停止信号や停止コマンドも有効です。



4-3. I/O 信号割付機能

4-3-1. MCM の設定

MCM SPEC を設定すると、MCM で次のような制御が可能になります。

● MCM SPEC0 SET 関数

OUTx信号の出力仕様を、汎用出力かSIGNAL OUT機能に設定することができます。

● MCM SPEC2 SET

外部信号から制御を行うときの設定です。

外部信号から MCC09 の SS0 に入力する信号を設定することができます。

MCC09 の SS0 は、停止機能または外部トリガとして使用することができます。

SIGNAL1 と SIGNAL2 には、それぞれ次の信号を割り付けることができます。

- ・ $\overline{\text{INO}}$ 信号 : +24V I/F フォトカプラ入力
- ・ $\overline{\text{IN1}}$ 信号 : +24V I/F フォトカプラ入力
- ・ $\overline{\text{SIN0}}$ 信号 : TTL レベル入力
- ・ $\overline{\text{SIN1}}$ 信号 : TTL レベル入力
- ・ $\overline{\text{SIN2}}$ 信号 : TTL レベル入力
- ・ $\overline{\text{SIN3}}$ 信号 : TTL レベル入力

SS0 SEL により、上記 SIGNAL1 または SIGNAL2 に割り付けされた外部入力信号を SS0 信号に入力する条件を加えて接続することができます。

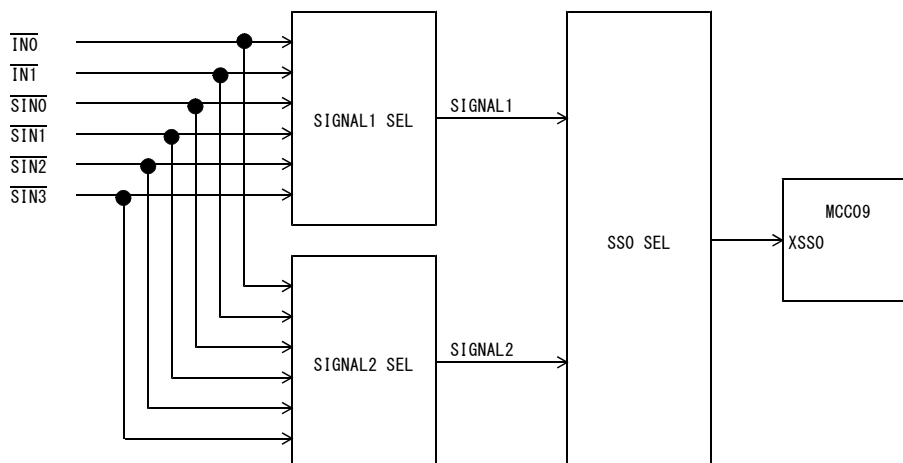
- ・ SIGNAL1 側の信号をスルー出力で SS0 に接続する
- ・ SIGNAL1 側の信号と SIGNAL2 側の信号を論理積 (AND) を取って SS0 に接続する
- ・ SIGNAL1 側の信号と SIGNAL2 側の信号を論理和 (OR) を取って SS0 に接続する

外部信号と SS0 信号の初期値は次のようになっています。

- ・ X 軸 SS0 : $\overline{\text{INO}}$ 信号
- ・ Y 軸 SS0 : $\overline{\text{IN1}}$ 信号
- ・ Z 軸 SS0 : 外部信号割り当てなし
- ・ A 軸 SS0 : 外部信号割り当てなし

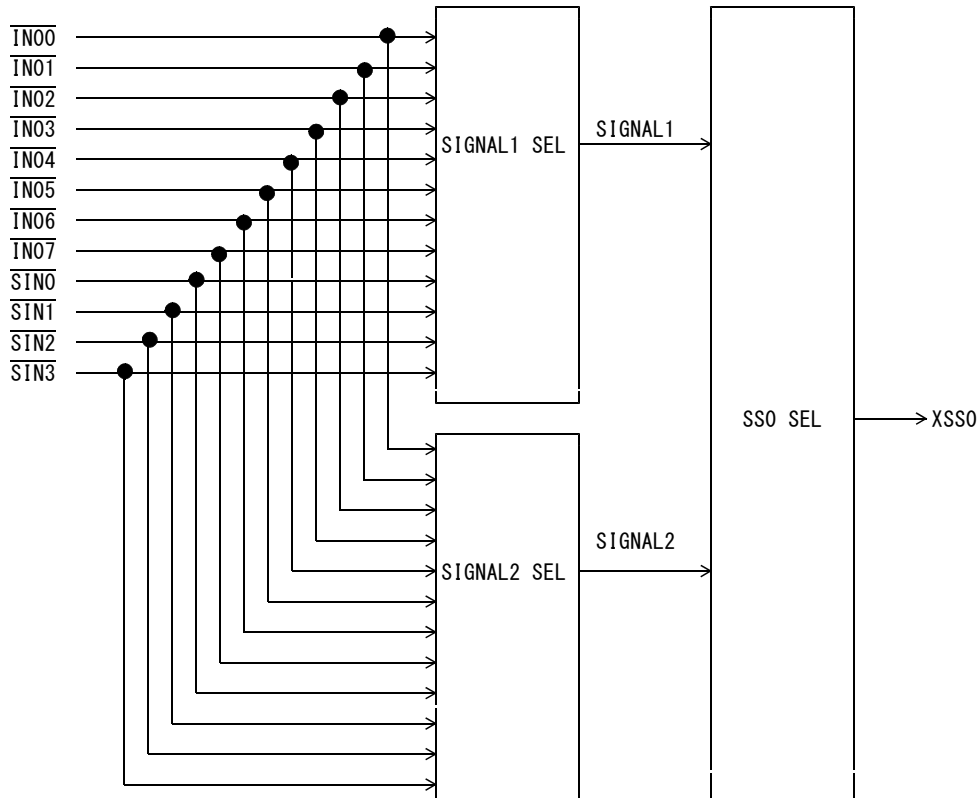
● 2C-7760、2CD-7720、2CD-7721、2CD-7730

【 停止センサ機能 (X 軸の場合) 】



● 2CD-7730G02

【 停止センサ機能（X 軸の場合）】



UNIT MCM SPEC2 SET 関数により、MCM SPEC2 を設定します

- ・ SS0 を停止機能として使用する場合、SPEC INITIALIZE2 コマンドで SS0 信号を停止機能に設定します。
- ・ SS0 をトリガ信号として使用する場合、SPEC INITIALIZE2 コマンドで SS0 信号を汎用入力に設定します。
MCM SPEC2 で MCC09 の SS0 に信号を割り付けると、外部信号を GPIO2 の SS0 フラグにより、次のタイミングを取ることができます。

PAUSE 機能の STBY 解除条件

SPEED RATE CHANGE の変更動作点

INDEX CHANGE の変更動作点

各カウンタのラッチタイミング

パルス偏差カウンタのカウント信号

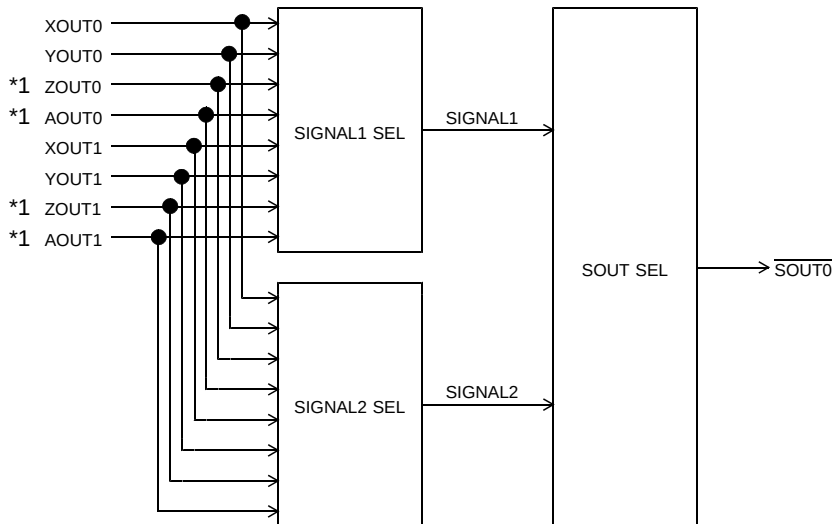
RAM 機能の JUMP 条件

RAM 機能のメモリーデータの NO OPERATION コマンド実行条件

● MCM SPEC3 SET

MCC09 の内部ステータス信号 OUT0 信号, OUT1 信号を $\overline{\text{SOUT}}$ 信号に出力する信号として設定できます。
UNIT MCM SPEC3 SET 関数により、MCM SPEC3 を設定します

【 ステータス外部出力機能 ($\overline{\text{SOUT0}}$ 信号の場合) 】



*1 ZOUT0,AOUT0,ZOUT1,AOUT1 は、4 軸ユニットのみ選択が出来ます。

OUT0, OUT1 信号の選択は、HARD INITIALIZE1 コマンドで設定します。

OUT0,1 信号の出力機能

TYPE3	TYPE2	TYPE1	TYPE0	OUT0,1 の出力機能	
0	0	0	0	ADRINT	カウンタ割り込み要求の ADRINT 出力
0	0	0	1	CNTINT	カウンタ割り込み要求の CNTINT 出力
0	0	1	0	DFLINT	カウンタ割り込み要求の DFLINT 出力
0	0	1	1	RDYINT	コマンド終了割り込み要求の RDYINT 出力
0	1	0	0	STBY	STATUS1 の STBY フラグ出力
0	1	0	1	DRIVE	STATUS1 の DRIVE フラグ出力
0	1	1	0	nSPEED FL	STATUS5 の SPEED FL フラグの反転出力
0	1	1	1	nINDEX FL	STATUS5 の INDEX FL フラグの反転出力
1	0	0	0	UP	STATUS1 の UP フラグ出力
1	0	0	1	DOWN	STATUS1 の DOWN フラグ出力
1	0	1	0	CONST	STATUS1 の CONST フラグ出力
1	0	1	1	EXT PULSE	STATUS1 の EXT PULSE フラグ出力
1	1	0	0	nPULSE MASK	STATUS2 の PULSE MASK フラグの反転出力
1	1	0	1	ORG SIGNAL	STATUS2 の ORG SIGNAL フラグ出力
1	1	1	0	汎用出力	汎用出力として使用する
1	1	1	1	設定禁止	—

OUT0 初期値 = CNTINT

OUT1 初期値 = RDYINT

SOUT 信号には、割り当てる軸、ステータス信号の種、および任意な 2 つの信号の論理和・論理積を加えて出力させることができます。

SOUT 信号の初期値は次のようになっています。

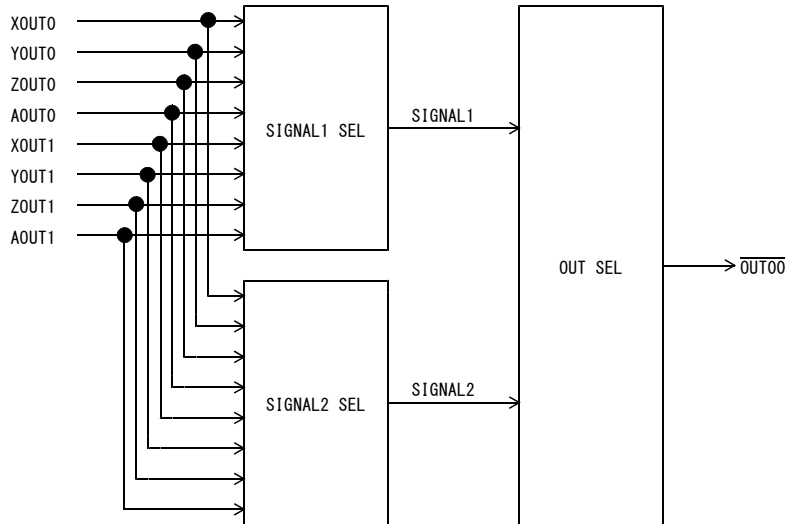
外部出力信号	軸と出力 (割付信号名)		SIGNAL1 SEL
	4 軸ユニット	2 軸ユニット	
$\overline{\text{SOUT0}}$	XOUT0 (XCNTINT)	XOUT0 (XCNTINT)	信号をスルー出力
$\overline{\text{SOUT1}}$	YOUT0 (YCNTINT)	YOUT0 (YCNTINT)	信号をスルー出力
$\overline{\text{SOUT2}}$	ZOUT0 (ZCNTINT)	XOUT0 (XCNTINT)	信号をスルー出力
$\overline{\text{SOUT3}}$	AOUT0 (ACNTINT)	YOUT0 (YCNTINT)	信号をスルー出力

MCM SPEC3 で MCC09 内部ステータス信号 SOUT 信号を割り付けると、OUT0, OUT1 の内部ステータスのタイミングで外部機器に信号出力することができます。

● MCM SPEC4 SET

MCC09 の内部ステータス信号 OUT0 信号, OUT1 信号を $\overline{\text{OUT00}}$ ~ $\overline{\text{OUT03}}$ 信号に出力する信号として設定できます。
UNIT MCM SPEC4 SET 関数により、MCM SPEC4 を設定します

【 ステータス外部出力機能 ($\overline{\text{OUT00}}$ 信号の場合) 】



OUT0, OUT1 信号の選択は、HARD INITIALIZE1 コマンドで設定します。

OUT0,1 信号の出力機能

TYPE3	TYPE2	TYPE1	TYPE0	OUT0,1 の出力機能	
0	0	0	0	ADRINT	カウンタ割り込み要求の ADRINT 出力
0	0	0	1	CNTINT	カウンタ割り込み要求の CNTINT 出力
0	0	1	0	DFLINT	カウンタ割り込み要求の DFLINT 出力
0	0	1	1	RDYINT	コマンド終了割り込み要求の RDYINT 出力
0	1	0	0	STBY	STATUS1 の STBY フラグ出力
0	1	0	1	DRIVE	STATUS1 の DRIVE フラグ出力
0	1	1	0	nSPEED FL	STATUS5 の SPEED FL フラグの反転出力
0	1	1	1	nINDEX FL	STATUS5 の INDEX FL フラグの反転出力
1	0	0	0	UP	STATUS1 の UP フラグ出力
1	0	0	1	DOWN	STATUS1 の DOWN フラグ出力
1	0	1	0	CONST	STATUS1 の CONST フラグ出力
1	0	1	1	EXT PULSE	STATUS1 の EXT PULSE フラグ出力
1	1	0	0	nPULSE MASK	STATUS2 の PULSE MASK フラグの反転出力
1	1	0	1	ORG SIGNAL	STATUS2 の ORG SIGNAL フラグ出力
1	1	1	0	汎用出力	汎用出力として使用する
1	1	1	1	設定禁止	—

OUT0 初期値 = CNTINT

OUT1 初期値 = RDYINT

OUT 信号には、割り当てる軸、ステータス信号の種、および任意な 2 つの信号の論理和・論理積を加えて出力させることができます。

OUT 信号の初期値は次のようになっています。

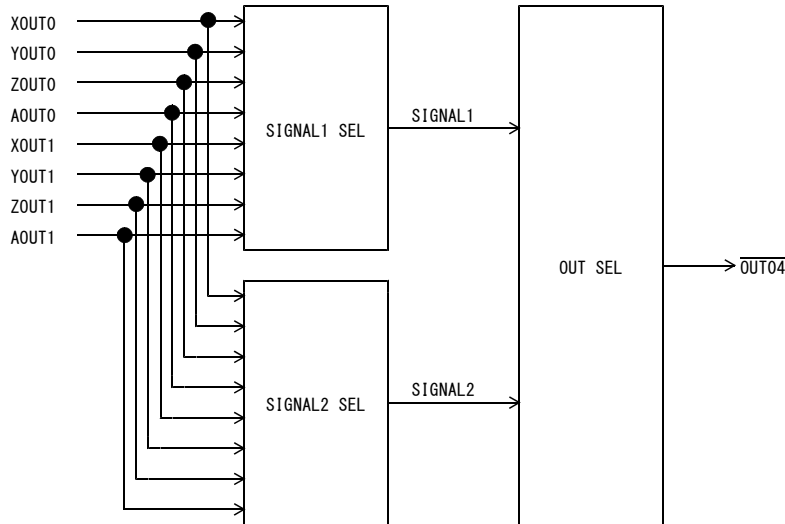
外部出力信号	出力 (割付信号名)	SIGNAL1 SEL
$\overline{\text{OUT00}}$	XOUT0 (XCNTINT)	信号をスルー出力
$\overline{\text{OUT01}}$	YOUT0 (YCNTINT)	信号をスルー出力
$\overline{\text{OUT02}}$	ZOUT0 (ZCNTINT)	信号をスルー出力
$\overline{\text{OUT03}}$	AOUT0 (ACNTINT)	信号をスルー出力

MCM SPEC4 で MCC09 内部ステータス信号 OUT 信号を割り付けると、OUT0, OUT1 の内部ステータスのタイミングで外部機器に信号出力することができます。

● MCM SPEC5 SET

MCC09 の内部ステータス信号 OUT0 信号, OUT1 信号を $\overline{\text{OUT04}}$ ~ $\overline{\text{OUT07}}$ 信号に出力する信号として設定できます。
UNIT MCM SPEC5 SET 関数により、MCM SPEC5 を設定します

【 ステータス外部出力機能 ($\overline{\text{OUT04}}$ 信号の場合) 】



OUT0, OUT1 信号の選択は、HARD INITIALIZE1 コマンドで設定します。

OUT0,1 信号の出力機能

TYPE3	TYPE2	TYPE1	TYPE0	OUT0,1 の出力機能	
0	0	0	0	ADRINT	カウンタ割り込み要求の ADRINT 出力
0	0	0	1	CNTINT	カウンタ割り込み要求の CNTINT 出力
0	0	1	0	DFLINT	カウンタ割り込み要求の DFLINT 出力
0	0	1	1	RDYINT	コマンド終了割り込み要求の RDYINT 出力
0	1	0	0	STBY	STATUS1 の STBY フラグ出力
0	1	0	1	DRIVE	STATUS1 の DRIVE フラグ出力
0	1	1	0	nSPEED FL	STATUS5 の SPEED FL フラグの反転出力
0	1	1	1	nINDEX FL	STATUS5 の INDEX FL フラグの反転出力
1	0	0	0	UP	STATUS1 の UP フラグ出力
1	0	0	1	DOWN	STATUS1 の DOWN フラグ出力
1	0	1	0	CONST	STATUS1 の CONST フラグ出力
1	0	1	1	EXT PULSE	STATUS1 の EXT PULSE フラグ出力
1	1	0	0	nPULSE MASK	STATUS2 の PULSE MASK フラグの反転出力
1	1	0	1	ORG SIGNAL	STATUS2 の ORG SIGNAL フラグ出力
1	1	1	0	汎用出力	汎用出力として使用する
1	1	1	1	設定禁止	—

OUT0 初期値 = CNTINT

OUT1 初期値 = RDYINT

OUT 信号には、割り当てる軸、ステータス信号の種、および任意な 2 つの信号の論理和・論理積を加えて出力させることができます。

OUT 信号の初期値は次のようになっています。

外部出力信号	出力 (割付信号名)	SIGNAL1 SEL
$\overline{\text{OUT04}}$	XOUT0 (XCNTINT)	信号をスルー出力
$\overline{\text{OUT05}}$	YOUT0 (YCNTINT)	信号をスルー出力
$\overline{\text{OUT06}}$	ZOUT0 (ZCNTINT)	信号をスルー出力
$\overline{\text{OUT07}}$	AOUT0 (ACNTINT)	信号をスルー出力

MCM SPEC5 で MCC09 内部ステータス信号 OUT 信号を割り付けると、OUT0, OUT1 の内部ステータスのタイミングで外部機器に信号出力することができます。

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		
■入出力信号のアクティブ論理		
FSSTOP 信号	正論理入力	HARD INITIALIZE7 コマンド
CWLM 信号	正論理入力	
CCWLM 信号	正論理入力	
DALM 信号	正論理入力	
ORG 信号	負論理入力	
NORG 信号	負論理入力	
PAUSE 信号	1 でアクティブ	
CWP 信号	負論理出力	HARD INITIALIZE8 コマンド
CCWP 信号	負論理出力	
■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	
■RAM アクセス機能		
直接 RAM アクセス機能	無効	MCM SPEC0 SET 関数
■多用途センサ機能		
各軸 SS0 入力機能	汎用入力	SPEC INITIALIZE2 コマンド
X 軸 SS1 接続信号	割り当てなし (LOW 固定)	MCM SPEC1 SET 関数
Y 軸 SS1 接続信号	割り当てなし (LOW 固定)	
Z 軸 SS1 接続信号	割り当てなし (LOW 固定)	
A 軸 SS1 接続信号	割り当てなし (LOW 固定)	
X 軸 SS0 接続信号	/IN0 信号のスルー	MCM SPEC2 SET 関数
Y 軸 SS0 接続信号	/IN1 信号のスルー	
Z 軸 SS0 接続信号	割り当てなし (LOW 固定)	
A 軸 SS0 接続信号	割り当てなし (LOW 固定)	
■ステータス外部出力機能		
各軸 OUT0 出力ステータス	CNTINT	HARD INITIALIZE1 コマンド
各軸 OUT1 出力ステータス	RDYINT	
各軸 OUT2 ステータス	汎用出力	
各軸 OUT3 ステータス	汎用出力	
各軸 GPIO2 ステータス	汎用入力	HARD INITIALIZE2 コマンド
各軸 GPIO3 ステータス	汎用入力	HARD INITIALIZE3 コマンド
SOUT0 出力信号	X 軸 OUT0 のスルー	MCM SPEC3 SET 関数
SOUT1 出力信号	Y 軸 OUT0 のスルー	
SOUT2 出力信号	Z 軸 OUT0 のスルー	
SOUT3 出力信号	A 軸 OUT0 のスルー	
■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 SS1 (同期) 機能	機能なし	MCM SPEC1 SET 関数

(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	
減速終了時の速度データ (ELSPD)	300	LOW SPEED SET コマンド
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)	
加速カーブS字変速領域データ (SUH)	0 (変速領域なし)	SHAREA SET コマンド
減速カーブS字変速領域データ (SDH)	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	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 STOP 型式	常時検出/エッジ検出で停止しない	
停止時 DRST 出力	出力しない	
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	
■補間ドライブパラメータ		
CPPOUT 端子から出力するパルス	パルス出力なし	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		
SPEED RATE CHANGE 変更動作点	STATUS1 PORT の DRIVE = 1 で実行	SPEED CHANGE SPEC SET コマンド
INDEX CHANGE 変更動作点	STATUS1 PORT の DRIVE = 1 で実行	INDEX CHANGE SPEC SET コマンド

5-2. 関数一覧

R1

種別	名称	記号	ページ	
			標準編	応用編
基本機能	RESULT構造体	MC07_S_RESULT	32	
	関数を実行した結果を格納する			
	コマンドデータ構造体	MC07_S_COMMAND_DATA	35	
	DRIVE COMMAND PORT、DRIVE DATA1 PORT、DRIVE DATA2 PORTに書き込むデータを格納する			
基本機能	ステータスデータ構造体	MC07_S_STATUS_DATA	36	
	DRIVE STATUS1 PORT、DRIVE DATA1 PORT、DRIVE DATA2 PORTから読み出した内容を格納する			
AL-II通信	スレーブ情報構造体	MC07_S_SLAVE_INFO	37	
	AL-II通信に接続される全スレーブユニットのタイプを格納する			
環境設定	環境設定関数	MC07_Environment	39	
	AL-II通信に接続されている全スレーブユニットを対象に環境設定を行う			
AL-II通信	スレーブ情報読み出し関数	MC07_ReadUnitInfo	40	
	AL-II通信に接続されている全スレーブユニットのタイプを読み出す			
	AL-II通信エラー累計回数読み出し関数	MC07_ErrCount	41	
	AL-II通信のエラー累計回数を読み出す			
	AL-II通信エラー累計回数クリア関数	MC07_ClrErrCount	42	
	AL-II通信のエラー累計回数を0にクリアする			
ユニット操作	ユニットオープン関数	MC07_UOpen	43	
	指定ユニット番号でユニットオープンし、引数 $phUnit$ の変数にユニットハンドルを格納する			
ユニット操作	ユニットクローズ関数	MC07_UClose	44	
	指定されたユニットをクローズする			
ユニット操作	ユニット動作エラークリア関数	MC07_UClrError	45	
	指定ユニットに対し、指定された軸の動作エラークリア処理を一括で行う			
拡張ユニット通信	拡張ユニット通信設定関数	MC07_UWExUnitCommMode	46	
	指定されたユニットと拡張ユニット間の通信設定を行う			
拡張ユニット通信	拡張ユニット通信制御関数	MC07_UWExUnitCommControl	47	
	指定されたユニットと拡張ユニット間の通信を制御する			
拡張ユニット通信	拡張ユニット通信ステータス読み出し関数	MC07_URExUnitCommStatus	48	
	指定されたユニットと拡張ユニット間の通信の状態を読み出す			
拡張ユニット通信	拡張ユニット通信設定読み出し関数	MC07_URExUnitCommMode	49	
	指定されたユニットと拡張ユニット間の通信設定を読み出す			
拡張GI/0ユニット通信	拡張GI/0ユニット通信制御関数	MC07_UWGExUnitCommControl	50	
	指定されたスレーブGI/0ユニットと拡張GI/0ユニット間の通信を制御する			
拡張GI/0ユニット通信	拡張GI/0ユニット通信ステータス読み出し関数	MC07_URGExUnitCommStatus	51	
	指定されたユニットと拡張GI/0ユニット間の通信の状態を読み出す			
ユニットステータス	ユニットステータス構造体	MC07_S_UNIT_STATUS	52	
	ユニットのステータスの内容を格納する			
6軸ユニットステータス	ユニットステータス構造体 (6軸)	MC07_S_UNIT_STATUS_6	53	
	6軸ユニットのステータスの内容を格納する			
ユニットコマンド	ユニットコマンド構造体	MC07_S_UNIT_COMMAND	54	
	ユニットのコマンドを格納する			
入力PORT	入力PORT構造体	MC07_S_IN_PORT	55	
	ユニットの入力PORTから読み出された内容を格納する			
出力PORT	出力PORT構造体	MC07_S_OUT_PORT	58	
	ユニットの出力PORTに書き込むデータ、OR書き込みデータ、AND書き込みデータを格納する			
ユニットDRIVE COMMAND・I/O書き込み	ユニットDRIVE COMMAND・I/O書き込み関数	MC07_UWDriveIo	60	
	指定されたユニットに対し、各軸のDATA、COMMAND、各I/O PORTデータの書き込みを一括で行う			
ユニットDRIVE COMMAND書き込み/読み出し	ユニットDRIVE COMMAND書き込み/読み出し関数	MC07_UWRDrive	62	
	指定されたユニットに対し、各軸のDATA、COMMANDを書き込み、読み出しする順の処理を一括で行う			
ユニットSTATUS1・I/O読み出し	ユニットSTATUS1・I/O読み出し関数	MC07_URStatusIo	63	
	指定されたユニットに対し、各軸のSTATUS1 PORT、I/O PORTの読み出しを一括で行う			
ユニットSTATUS1・I/O読み出し (6軸)	ユニットSTATUS1・I/O読み出し関数 (6軸)	MC07_URStatusIo_6	64	
	指定された6軸ユニットに対し、各軸のSTATUS1 PORT、I/O PORTの読み出しを一括で行う			
ユニットSTATUS1・パルスカウンタ・I/O読み出し	ユニットSTATUS1・パルスカウンタ・I/O読み出し関数	MC07_URStatusIPentIo	65	
	指定されたユニットに対し、各軸のSTATUS1 PORT、パルスカウンタ値、I/O PORTの読み出しを一括で行う			
ユニットSTATUS1・パルスカウンタ・I/O読み出し (6軸)	ユニットSTATUS1・パルスカウンタ・I/O読み出し関数 (6軸)	MC07_URStatusIPentIo_6	66	
	指定された6軸ユニットに対し各軸のSTATUS1 PORT、パルスカウンタ値、I/O PORTの読み出しを一括で行う			

取扱説明書 応用機能編

(AL-II シリーズ MCC09 ユニット編)

R1

種別	名称	記号	ページ	
	説明		標準編	応用編
K ヤ ク ア ム シ ハ コ	ユニットI/O PORT書き込み関数 指定されたユニットに対し、各I/O PORT毎の個別データを書き込む	MC07_UPortOUT	67	
	ユニットI/O PORT OR書き込み関数 指定されたユニットに対し、各I/O PORT毎の個別データをOR書き込む	MC07_UPortOrOUT	68	
	ユニットI/O PORT AND書き込み関数 指定されたユニットに対し、各I/O PORT毎の個別データをAND書き込む	MC07_UPortAndOUT	69	
ハ コ	ユニットI/O PORT読み出し関数 指定されたユニットに対し、各I/O PORTの内容を一括で読み出す	MC07_UPortIn	70	
	デバイスオープン関数 指定ユニット番号、軸でデバイスオープンし、引数 $phDev$ の変数にデバイスハンドルを格納する	MC07_BOpen	71	
	デバイスクローズ関数 指定されたデバイスをクローズする	MC07_BClose	72	
ト	動作エラークリア関数 指定されたデバイスの動作エラーをクリアする	MC07_ClrError	73	
	DRIVE COMMAND 32ビット書き込み関数 指定デバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTにデータを書き込んだ後コマンドを書き込む	MC07_LWDrive	76	
	DRIVE COMMAND PORT書き込み関数 指定デバイスのDRIVE COMMAND PORTにコマンドコードを書き込む	MC07_BWDriveCommand	77	
ト	DRIVE STATUS1 PORT読み出し関数 指定デバイスのDRIVE STATUS1 PORTを読み出す	MC07_BRStatus1	78	
	DRIVE STATUS2 PORT読み出し関数 指定デバイスのDRIVE STATUS2 PORTを読み出す	MC07_BRStatus2	82	
	DRIVE STATUS3 PORT読み出し関数 指定デバイスのDRIVE STATUS3 PORTを読み出す。	MC07_BRStatus3	84	
ト	DRIVE STATUS4 PORT読み出し関数 指定デバイスのDRIVE STATUS4 PORTを読み出す。	MC07_BRStatus4	85	
	DRIVE STATUS5 PORT読み出し関数 指定デバイスのDRIVE STATUS5 PORTを読み出す	MC07_BRStatus5	87	
	STATUS PORTバッファ読み出し関数 指定デバイスのDRIVE STATUS1、STATUS2、STATUS3、STATUS4、STATUS5 PORT、ORIGIN STATUSを読み出す	MC07_BRStatusBuf	90	
ト	DRIVE COMMAND 32ビット書き込み/読み出し関数 指定デバイスのDRIVE DATA、COMMAND PORTにデータ、コマンドを書き込み、DRIVE DATA PORTを読み出す	MC07_LWRDrive	91	
	DRIVE COMMAND PORT書き込み/読み出し関数 指定デバイスのDRIVE COMMAND PORTにコマンドを書き込み、DRIVE DATA PORTを読み出す。	MC07_BWRDrive	92	
	DEVICE I/O書き込み関数 指定デバイスのDEVICE I/O へのデータを書き込む。	MC07_WriteDeviceIo	93	
ト	DEVICE I/O読み出し関数 指定デバイスのDEVICE I/O へのデータを読み出す。	MC07_ReadDeviceIo	94	
	NOP DATA PORT読み出し関数 指定デバイスのNOP DATA PORTを読み出す	MC07_BRNopData	95	
ト	DRIVE DATA 32ビット書き込み関数 指定デバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTにデータを書き込む。	MC07_LWDATA		9
	DRIVE DATA1 PORT書き込み関数 指定デバイスのDRIVE DATA1 PORTにデータを書き込む。	MC07_BWDriveData1		10
	DRIVE DATA2 PORT書き込み関数 指定デバイスのDRIVE DATA2 PORTにデータを書き込む。	MC07_BWDriveData2		11
ト	DRIVE DATA 32ビット読み出し関数 指定デバイスのDRIVE DATA1 PORT、DRIVE DATA2 PORTを一括で読み出す。	MC07_LRDrive		12
	DRIVE DATA1 PORT読み出し関数 指定デバイスのDRIVE DATA1 PORTを読み出す。	MC07_BRDriveData1		13
	DRIVE DATA2 PORT読み出し関数 指定デバイスのDRIVE DATA2 PORTを読み出す。	MC07_BRDriveData2		14
ト	データ構造体 DRIVE DATA1 PORT、DRIVE DATA2 PORTに書き込むデータを格納する	MC07_S_DATA		15
	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にデータ構造体の内容を書き込み後、コマンドコードを書き込む	MC07_IWDRIVE		16

取扱説明書 応用機能編

(AL-IIシリーズ MCC09 ユニット編)

種別	名称	記号	ページ	
	説明		標準編	応用編
イ ン テ リ フ ェ ー ス	データセット関数 32ビットデータを引数 $psData$ で示されるデータ構造体に格納する	MC07_SetData		19
	データゲット関数 データ構造体の内容を引数 $psData$ で示される32ビットデータに変換して返す	MC07_GetData		20
T I M E R	READY WAIT関数 指定デバイスがREADY (STATUS1 BUSY BIT=0) まで待機し、最大待ち時間を超えるとエラー終了する	MC07_BWaitDriveCommand	97	
	WAIT状態読み出し関数 指定デバイスのWAIT状態を返す	MC07_BIsWait	98	
	WAIT中止関数 指定デバイスのREADY WAIT関数またはCOMREG NOT FULL WAIT関数の実行を中止する	MC07_BBreakWait	99	
S P E E D	SPEED・RATE構造体 SPEED・RATEセット関数で使用する	MC07_S_SPEED_RATE	100	
	SPEED・RATEセット関数 指定のRESOL No. とSPEED・RATE構造体を元にSPEEDパラメータ設定、加減速時定数(RATE)設定を実行する	MC07_SetSpeedRate	102	
	SPEED・RATE読み出し関数 指定デバイスからSPEEDパラメータ、加減速時定数設定を読み出し、SPEED・RATE構造体に格納する	MC07_ReadSpeedRate	103	
P O S I T I O N	POSITION構造体 X・Y座標を指定するときに使用する	MC07_S_XY_POSITION	104	
	2軸相対アドレス直線補間ドライブ関数 相対アドレスで指定された目的地まで任意2軸直線補間ドライブを行う	MC07_IncStrCp	105	
	2軸相対アドレス円弧補間ドライブ関数 相対アドレスで指定された目的地まで任意2軸円弧補間ドライブを行う	MC07_IncCirCp	107	
	円の中心点ゲット関数 円弧の通過点相対アドレス、目的地相対アドレスを元に中心点相対アドレス、回転方向を求める	MC07_GetCirCenterPosition	109	
	相対アドレス変換関数 指定された絶対アドレスを相対アドレス(絶対アドレス - 現在位置)に変換する	MC07_IncFromAbs	110	
	円弧補間短軸PULSE数ゲット関数 指定円弧の中心点相対アドレス、目的地相対アドレス、回転方向を元に目的地短軸座標までのパルス数を計算	MC07_GetCirShortPulse		23
O R I G I N	ORIGINドライブパラメータ構造体 ORIGINドライブパラメータ読み出し関数で読み出した内容を格納する	MC07_TAG_S_ORG_PARAM	111	
	ORIGINドライブステータス読み出し関数 ORIGIN STATUSの内容を読み出す	MC07_ReadOrgStatus	112	
	ORIGIN SPEC SET関数 ORIGINドライブの動作仕様を設定する	MC07_SetOrgSpec	114	
	ORIGIN MARGIN PULSE SET関数 ORIGINドライブの機械原点信号検出後のMARGINパルス数を設定する	MC07_SetOrgMarginPulse	117	
	ORIGIN DELAY SET関数 ORIGINドライブの各工程後で挿入するDELAYを設定する	MC07_SetOrgDelay	118	
	ORIGIN ERROR PULSE SET関数 CONSTANT SCAN工程時および1PULSE送り工程時にエラー判定する各最大パルス数を設定する	MC07_SetOrgErrorPulse	119	
	ORIGIN OFFSET PULSE SET関数 機械原点近傍アドレスのOFFSETパルス数を設定する	MC07_SetOrgOffsetPulse	120	
	ORIGIN PRESET PULSE SET関数 機械原点検出終了後に実行するORIGINドライブのPRESETパルスを設定する	MC07_SetOrgPresetPulse	121	
	ORIGINドライブパラメータ読み出し関数 設定されたORIGINドライブパラメータを読み出し、ORIGINドライブパラメータ構造体に格納する	MC07_ReadOrgParam	122	
	ORIGIN FLAG RESET関数 ORIGIN FLAGをRESETする	MC07_ResetOrgFlag	123	
	ORIGINドライブ関数 指定された機械原点の型式に従いORIGINドライブを行う	MC07_Org	124	
約 々	COMREG NOT FULL WAIT関数 指定デバイスがコマンド予約可能 (STATUS1 COMREG FULL BIT=0) まで待機し最大待ち時間超えてエラー終了	MC07_BWaitComregNotFull		21
	DRIVE COMMANDバッファ書き込み関数 コマンドデータ構造体配列に格納したコマンド、データを指定デバイスのDATA1, 2, COMMANDに指定個数書込む。	MC07_LWDriveBuf		22

種別	名称	記号	ページ	
	説明		標準編	応用編
O ／ I	I/O PORTオープン関数 拡張・汎用・制御I/O PORTをオープンし、引数 $phPort$ の変数にPORTハンドルを格納する	MC07_BPortOpen	125	
	I/O PORTクローズ関数 指定された拡張・汎用・制御I/O PORTをクローズする	MC07_BPortClose	127	
	I/O PORT書き込み関数 指定された拡張・汎用・制御・GユニットのI/O PORTにデータを書き込む	MC07_BPortOut	128	
	I/O PORT OR書き込み関数 指定された拡張・汎用・制御・GユニットのI/O PORTにORデータを書き込む	MC07_BPortOrOut	129	
	I/O PORT AND書き込み関数 指定された拡張・汎用・制御・GユニットのI/O PORTにANDデータを書き込む	MC07_BPortAndOut	130	
	I/O PORT 読み出し関数 指定された拡張・汎用・制御・GユニットのI/O PORTのデータを読み出す	MC07_BPortIn	131	
	I/O PORTラッチエッジ選択書き込み関数 指定された汎用I/O PORTのラッチのエッジを設定する	MC07_BWLatchEdge	132	
	I/O PORTラッチエッジ選択読み出し関数 指定された汎用I/O PORTのラッチのエッジの設定を読み出す	MC07_BRLatchEdge	133	
	I/O PORTラッチクリア書き込み関数 指定された汎用I/O PORTのラッチデータをクリアする	MC07_BWLatchClr	134	
	I/O PORTラッチデータ読み出し関数 指定された汎用I/O PORTのラッチデータを読み出す	MC07_BRLatchData	135	
数 値 の 変 換	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する。	MC07_Unsigned16		24
	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ビット符号付きデータに変換する。	MC07_signed16		25
	32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する。	MC07_Unsigned32		26
	32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ビット符号付きデータに変換する。	MC07_signed32		27
M C M	AXIS MCM SPEC構造体 指定された軸のMCM基本設定の内容を格納する	MC07_S_AXIS_MCM_SPEC		28
	UNIT MCM SPEC構造体 指定されたユニットのMCM基本設定の内容を格納する	MC07_S_UNIT_MCM_SPEC		29
	UNIT MCM SPEC0 SET関数 指定されたユニットのMCM SPEC0を設定する	MC07_USetMcmSpec0		30
	UNIT MCM SPEC0 GET関数 指定されたユニットのMCM SPEC0を設定を読み出す	MC07_UGetMcmSpec0		31
	UNIT MCM SPEC2 SET関数 指定されたユニットのMCM SPEC2を設定する	MC07_USetMcmSpec2		32
	UNIT MCM SPEC2 GET関数 指定されたユニットのMCM SPEC2を設定を読み出す	MC07_UGetMcmSpec2		35
	UNIT MCM SPEC3 SET関数 指定されたユニットのMCM SPEC3を設定する	MC07_USetMcmSpec3		36
	UNIT MCM SPEC3 GET関数 指定されたユニットのMCM SPEC3を設定を読み出す	MC07_UGetMcmSpec3		38
	UNIT MCM SPEC4 SET関数 指定されたユニットのMCM SPEC4を設定する	MC07_USetMcmSpec4		39
	UNIT MCM SPEC4 GET関数 指定されたユニットのMCM SPEC4を設定を読み出す	MC07_UGetMcmSpec4		41
	UNIT MCM SPEC5 SET関数 指定されたユニットのMCM SPEC5を設定する	MC07_USetMcmSpec5		42
	UNIT MCM SPEC5 GET関数 指定されたユニットのMCM SPEC5を設定を読み出す	MC07_UGetMcmSpec5		44

5-3. コマンド一覧

(1) 汎用コマンド

種別	コマンド名	コマンドコード	ページ	
			標準編	応用編
汎用	説明			
	NO OPERATION	H' 00	159	
	機能なし			
	SPEC_INITIALIZE1	H' 01	136	
	ドライブパルスの出力仕様の設定			
	SPEC_INITIALIZE2	H' 02	137	
	CWLM, CCWLM, SSOの設定			
	SPEC_INITIALIZE3	H' 03	139	
	DRST, DEND/PO, DALMの設定			
	FSPD SET	H' 05		52
	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			
	HIGH SPEED SET	H' 06		53
	加減速ドライブの最高速時のパルス速度データ (HSPD)、速度データの速度倍率 (RESOL)を設定			
	LOW SPEED SET	H' 07		54
	加減速ドライブの加速開始時パルス速度データ (LSPD)、減速終了時パルス速度データ (ELSPD)を設定			
	RATE SET	H' 08		55
	加速カーブの変速周期データ (UCYCLE)、減速カーブの変速周期データ (DCYCLE)を設定			
	SCAREA SET	H' 09		56
	加速カーブのS字変速領域データ (SUAREA)、減速カーブのS字変速領域データ (SDAREA)を設定			
	DOWN PULSE ADJUST	H' 0A		58
	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			
	SHAREA SET	H' 0B		57
	加速カーブのS字変速領域データ (SUH)、減速カーブのS字変速領域データ (SDH)を設定			
	JSPD SET	H' 0C	144	
	JOGドライブのパルス速度の設定			
	JOG PULSE SET	H' 0D	145	
	JOGドライブのパルス数の設定			
	ORIGIN SPEC SET	H' 0F		59
	ORIGINドライブの動作仕様を設定			
	+JOG	H' 10	145	
	+方向JOGドライブの実行			
	-JOG	H' 11	146	
	-方向JOGドライブの実行			
	+SCAN	H' 12	147	
	+方向SCANドライブの実行			
	-SCAN	H' 13	147	
	-方向SCANドライブの実行			
	INC INDEX	H' 14	148	
	相対アドレスINDEXドライブの実行			
	ABS INDEX	H' 15	149	
	絶対アドレスINDEXドライブの実行			
	+JSPD SCAN	H' 16	146	
	+方向JSPD SCANドライブの実行			
	-JSPD SCAN	H' 17	146	
	-方向JSPD SCANドライブの実行			
	SERVO RESET	H' 1F	152	
	DRST信号出力に10ms間アクティブを出力			

種別	コマンド名	コマンドコード	ページ	
			標準編	応用編
ユ ハ ト ロ ロ	CP SPEC SET	H' 20		61
	相関2軸1チップのCPPOUT端子から出力するパルスを設定			
	LONG POSITION SET	H' 22		68
	直線補間ドライブの、長軸の座標アドレスを設定			
	SHORT POSITION SET	H' 23		69
	直線補間ドライブの、短軸の座標アドレスを設定			
	CIRCULAR XPOSITION SET	H' 28		77
	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			
	CIRCULAR YPOSITION SET	H' 29		78
	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			
	CIRCULAR PULSE SET	H' 2A		79
	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			
	MAIN STRAIGHT CP	H' 30		80
	任意軸、複数軸の直線補間ドライブのときのメイン軸に実行			
	SUB STRAIGHT CP	H' 31		81
	任意軸、複数軸の直線補間ドライブのときのサブ軸に実行			
ユ ハ ト ロ ロ	MAIN CIRCULART CP	H' 38		80
	任意軸の円弧補間ドライブのときのメイン軸に実行			
	SUB CIRCULART CP	H' 39		81
	任意軸の円弧補間ドライブのときのサブ軸に実行			

(2) 特殊コマンド

種別	コマンド名	コマンドコード	ページ	
			標準編	応用編
ミ ハ ト ロ カ サ サ	説明			
	ADDRESS COUNTER PRESET アドレスカウンタの現在位置設定	H' 80	165	
	ADDRESS COUNTER INITIALIZE1 アドレスカウンタの各機能の設定	H' 81	160	
	ADDRESS COUNTER INITIALIZE2 アドレスカウンタの各機能の設定	H' 82	163	
	ADDRESS COUNTER MAX COUNT SET アドレスカウンタの最大カウント数の設定	H' 87		89
	ADRINT COMPARE REGISTER1 SET ADRINTのコンペアレジスタ1の設定	H' 88	166	
	ADRINT COMPARE REGISTER2 SET ADRINTのコンペアレジスタ2の設定	H' 89	166	
	ADRINT COMPARE REGISTER3 SET ADRINTのコンペアレジスタ3の設定	H' 8A	166	
	ADRINT COMP1 ADD DATA SET ADRINTのCOMP1 ADDデータの設定	H' 8C	167	
	PULSE COUNTER PRESET パルスカウンタのカウント初期値の設定	H' 90	173	
	PULSE COUNTER INITIALIZE1 パルスカウンタの各機能の設定	H' 91	168	
	PULSE COUNTER INITIALIZE2 パルスカウンタの各機能の設定	H' 92	171	
	PULSE COUNTER MAX COUNT SET パルスカウンタの最大カウント数の設定	H' 97		90
	CNTINT COMPARE REGISTER1 SET CNTINTのコンペアレジスタ1の設定	H' 98	174	
	CNTINT COMPARE REGISTER2 SET CNTINTのコンペアレジスタ2の設定	H' 99	174	
	CNTINT COMPARE REGISTER3 SET CNTINTのコンペアレジスタ3の設定	H' 9A	174	
	CNTINT COMP1 ADD DATA SET CNTINTのCOMP1 ADDデータの設定	H' 9C	175	
	DFL COUNTER PRESET パルス偏差カウンタのカウント初期値の設定	H' A0	184	
	DFL COUNTER INITIALIZE1 パルス偏差カウンタの各機能の設定	H' A1	176	
	DFL COUNTER INITIALIZE2 パルス偏差カウンタの各機能の設定	H' A2	179	
	DFL COUNTER INITIALIZE3 パルス偏差カウンタの各機能の設定	H' A3	182	
	DFLINT COMPARE REGISTER1 SET DFLINTのコンペアレジスタ1の設定	H' A8	185	
	DFLINT COMPARE REGISTER2 SET DFLINTのコンペアレジスタ2の設定	H' A9	185	
	DFLINT COMPARE REGISTER3 SET DFLINTのコンペアレジスタ3の設定	H' AA	185	
	DFLINT COMP1 ADD DATA SET DFLINTのCOMP1 ADDデータの設定	H' AC	186	

種別	コマンド名	コマンドコード	ページ	
	説明		標準編	応用編
ユ ハ ト ロ タ	SPEED CHANGE SPEC SET	H' C1		82
	SPEED CHANGE指令を実行する変更動作点を設定			
	INDEX CHANGE SPEC SET	H' C3		84
	INDEX CHANGE指令を実行する変更動作点を設定する			
	SPEED RATE CHANGE	H' C8		83
	実行中のパルス出力を指定ドライブパルス速度とRATEで加速または減速させる			
	INC INDEX CHANGE	H' CC		85
	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を行う			
	ABS INDEX CHANGE	H' CD		86
	指定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を行う			
	PLS INDEX CHANGE	H' CE		87
	指定データを変更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を行う			
	ERROR STATUS READ	H' D1	154	
	ERROR STATUSの読み出し			
	MCC SPEED READ	H' D4	156	
	ドライブパルス速度の読み出し			
	MCC SET DATA READ	H' D5	157	
	設定データの読み出し			
	RSPD DATA READ	H' D6		88
	RSPDデータの読み出し			
	ADDRESS COUNTER READ	H' D8	187	
	アドレスカウンタの読み出し			
	PULSE COUNTER READ	H' D9	187	
	パルスカウンタの読み出し			
	DFL COUNTER READ	H' DA	187	
	パルス偏差カウンタの読み出し			
	ADDRESS LATCH DATA READ	H' DC		93
	アドレスカウンタのラッチデータを読み出す			
	PULSE LATCH DATA READ	H' DD		93
	パルスカウンタのラッチデータを読み出す			
	DFL LATCH DATA READ	H' DE		93
	パルス偏差カウンタのラッチデータを読み出す			
	ERROR STATUS MASK	H' E5	153	
	ERRORに出力するERROR STATUSのマスク			
	COUNT LATCH SPEC SET	H' E8		91
	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			

種別	コマンド名	コマンドコード	ページ	
	説明		標準編	応用編
	HARD_INITIALIZE1 ステータス信号(SOUT信号)に出力する機能を設定	H' F1		45
	HARD_INITIALIZE2 GPIO2信号の入出力機能を設定	H' F2		47
	HARD_INITIALIZE3 GPIO3信号の入出力機能を設定	H' F3		48
	HARD_INITIALIZE7 CWLM信号、CCWLM信号、FSSTOP信号、DALM信号の入力アクティブ論理を設定	H' F7		49
	HARD_INITIALIZE8 CWP信号、CCWP信号の出力アクティブ論理を設定	H' F8		51
	SIGNAL_OUT 汎用出力信号の操作	H' FC	151	
	DRST_OUT DRSTに10ms間のアクティブレベルを出力	H' FD	152	
	SLOW_STOP 減速停止の実行	H' FE	150	
	FAST_STOP 即時停止の実行	H' FF	150	

本版で改訂された主な箇所

箇所	内容
P7, 135, 136	【R1】 誤記訂正

■ 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いただきます。

- (1) お客様の不適當な取り扱い、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) お客様の改造、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、保証外とさせていただきます。

技術相談のお問い合わせ

TEL. (042) 664-5382 FAX. (042) 666-5664
E-mail s-support@melec-inc.com

販売に関するお問い合わせ

TEL. (042) 664-5384 FAX. (042) 666-2031

株式会社 **メレック** 制御機器営業部
〒193-0834 東京都八王子市東浅川町516-10

URL: <http://www.melec-inc.com>