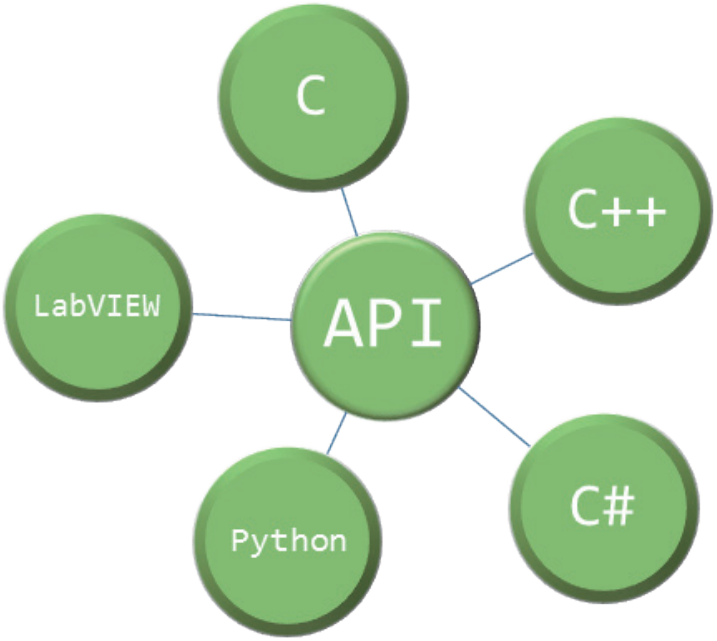
HIWIN® MIKROSYSTEM



HIMCシリーズ

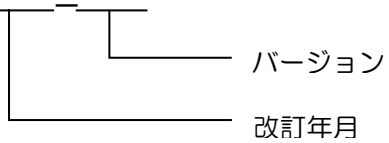
API ユーザーマニュアル

www.hiwinmikro.tw
MH05UJ01-2512_V1.2

改訂履歴

バージョンは表紙の下部にも記載されています。

MH05UJ01-2512_V1.2



日付	バージョン	適用機種	改訂内容
2025 年 12 月	1.2	iA Studio 3.3.0	(1) 章・セクションの追加： ■ 19 章 スレーブ関数 (2) 機能の追加： ■ 2 章： HIMC_IsConnected HIMC_GetModelNum HIMC_GetSerialNum ■ 3 章： HIMC_MovePVT HIMC_GetVel HIMC_GetAcc HIMC_GetDec HIMC_SetPosWindow HIMC_GetPosWindow HIMC_SetPosWindowTime HIMC_GetPosWindowTime ■ 6 章： HIMC_GetGrpVel HIMC_GetGrpAcc HIMC_GetGrpDec ■ 7 章： HIMC_GetSlvAllGPIByStartIdx HIMC_GetSlvAllGPOByStartIdx ■ 11 章： HIMC_EnableTouchProbe1 HIMC_EnableTouchProbe2 HIMC_DisableTouchProbe1 HIMC_DisableTouchProbe2 HIMC_IsTouchProbe1Enabled HIMC_IsTouchProbe2Enabled HIMC_IsTouchProbe1Triggered

日付	バージョン	適用機種	改訂内容
			HIMC_IsTouchProbe2Triggered HIMC_GetTouchProbe1Pos HIMC_GetTouchProbe2Pos HIMC_SetTouchProbe1Func HIMC_SetTouchProbe2Func ■ 18 章： HIMC_IsHomeSwitch (3) 機能変更 ■ 2 章： HIMC_GetFirmwareVer HIMC_GetFirmwareDate ■ 3 章： HIMC_MoveVel HIMC_SetVel HIMC_SetAcc HIMC_SetDec HIMC_SetOpMode ■ 6 章： HIMC_SetGrpMotionProfile HIMC_SetGrpVel HIMC_SetGrpAcc HIMC_SetGrpDec ■ 8 章： HIMC_SetSlvAORaw HIMC_GetSlvAIRaw HIMC_GetSlvAORaw ■ 16 章： HIMC_ReadSDO HIMC_ReadPDO ■ 17 章： HIMC_GetDriveErr (4) 機能内容の変更・追加 ■ 1 章： API の動作に関する説明を修正。 API 環境の設定を修正。 バージョンの説明を修正。 データ型の説明を追加。 ■ 10 章, 11 章, 18 章 位置トリガー関連機能、タッチプローブ関連機能、および

日付	バージョン	適用機種	改訂内容
			「MoveHome」機能は、シミュレータおよび仮想軸ではサポートされていません。 ■ 16 章： 変数の説明を修正 (HCV_ID_fb_ratio_pos, HCV_ID_fb_ratio_cnt, HCV_ID_fb_curr_ratio_curr, HCV_ID_fb_curr_ratio_cnt). (5) エラーメッセージを修正： セクション 17.1.1, セクション 17.1.2, セクション 17.1.3, セクション 22.1
2025 年 2 月 28 日	1.1	iA Studio 3.1.0	(1) 機能の追加 ■ 2 章： HIMC_IsSystemPreOp HIMC_GetFirmwareDate HIMC_GetFirmwareHash HIMC_GetModelNumber HIMC_GetApiVer ■ 3 章： HIMC_GetPosFbComp HIMC_GetBufferMode HIMC_GetCmdNum ■ 6 章： HIMC_SetGrpLookAheadPrm HIMC_SetGrpQueueSize ■ 10 章： HIMC_SetPT_PosArray HIMC_SetPT_StateArray HIMC_SetPT_StartIndex HIMC_SetPT_EndIndex ■ 18 章： HIMC_SetHomedStatus (2) 関数とデータ構造を変更 ■ 2 章： HIMC_IsSystemOper ■ 3 章： HIMC_GetCurrFb - the current feedback HIMC_SetOpMode

日付	バージョン	適用機種	改訂内容
			■ 8 章： HIMC_SetSlvAORaw HIMC_GetSlvAORaw HIMC_GetSlvAIRaw ■ 16 章： HIMC_ReadSDO HIMC_WriteSDO HIMC_ReadPDO HIMC_WritePDO HIMC_FroceWritePDO - parameter type ■ 20 章： MotionBufferMode MotionTransitionMode - add parameter (3) 機能説明の変更・追加 ■ 3 章： Axis 変数に CoE 変数を追加 ■ 6 章： バッファモードと遷移モードの説明を修正。先読み機能を追加 ■ 10 章： ランダム PT 変数の説明と手順を追加 ■ 18 章： 原点復帰方法の説明を変更 ■ 19 章： データ構造 Data_t を追加 (4) エラーメッセージを追加 セクション 17.1.1, セクション 17.1.2, セクション 17.1.3, セクション 21.1
2023 年 3 月 15 日	1.0	iA Studio 3.0.0	(1) HIMC は CoE 通信をサポートします。 (2) SDO および PDO の読み取り/書き込み機能を追加。関連するスレーブ変数の取得/設定機能と PDL の実行機能を削除。 (3) 章・セクションを追加 ■ 8 章 原点復帰機能 (4) 機能の追加 ■ 2 章： HIMC_IsSystemInit

日付	バージョン	適用機種	改訂内容
			HIMC_GetECATSt HIMC_GetSlvECATSt HIMC_ScanNetwork ■ 3 章： HIMC_SetOpMode HIMC_SetBufferMode ■ 8 章： HIMC_SetSlvAOHex HIMC_GetSlvAOHex HIMC_GetSlvAIHex ■ 11 章： HIMC_SetTouchProbeFunc ■ 16 章： HIMC_GetDriveErr (5) 10 章のランダム PT の関連機能を削除
2022 年 6 月 30 日	0.5	iA Studio 2.0	(1) 章・セクションの追加： 8 章 – AIO 関数 (2) API の追加： ■ 2 章 HIMC_SetEconMode HIMC_IsEconMode ■ 3 章 HIMC_MoveTrq HIMC_MovePVT HIMC_IsAcc ■ 6 章 HIMC_ HIMC_SetGrpAngMotionProfile HIMC_GetGrpCoordTrans HIMC_SetGrpCoordTrans HIMC_GetGrpPoseCmd HIMC_GetGrpPoseFb HIMC_CircleRel ■ 7 章 HIMC_SetGPIInvert HIMC_SetGPOInvert HIMC_BindEMO HIMC_GetAllGPIInvertSt HIMC_GetAllGPOInvertSt ■ 10 章 HIMC_SetPT_PosArray HIMC_SetPT_StateArray

日付	バージョン	適用機種	改訂内容
			HIMC_SetPT_StartIndex HIMC_SetPT_EndIndex ■ 12 章 HIMC_SetCompAlgType ■ 14 章 HIMC_LoadHMPLTask ■ 15 章 HIMC_SetHmplMsgEvtCallback (3) 変数を追加： セクション 3.1.1、セクション 6.1.1 (4) エラーメッセージの追加： セクション 17.1.1、セクション 17.1.2、セクション 17.1.3
2021 年 9 月 15 日	0.4	iA Studio 1.4	(1) 章・セクションの追加： セクション 7.1.1 – GPIO 変数 (2) API の追加： ■ 2 章 HIMC_GetFirmwareVer ■ 3 章 HIMC_GetVelFb HIMC_GetVelErr HIMC_GetCurrFb HIMC_SetVelScale HIMC_GetVelScale HIMC_SetRollover HIMC_GetRolloverTurns HIMC_IsDriveErr HIMC_IsPosErr ■ 6 章 HIMC_JogGroupHIMC_JogGroupAxis HIMC_SetGrpVelScale HIMC_GetGrpVelScale ■ 11 章 HIMC_SetupComp3D (3) 変数を追加： セクション 3.1.1、セクション 6.1.1、セクション 15.1.2 (4) エラーメッセージの追加： (1) セクション 16.1.1、セクション 16.1.2、セクション 16.1.3、セクション 19.1

関連文書

関連ドキュメントを通じて、ユーザーはこのマニュアルの位置付けと、マニュアルと製品との関連性をすぐに理解できます。詳細は、HIWIN MIKROSYSTEM の公式ウェブサイト → ダウンロード → マニュアル概要 (https://www.hiwinmikro.tw/Downloads/ManualOverview_EN.htm) をご覧ください。

目次

1. はじめに

1.1 API の仕組み

HIMC シリーズ API は、C/C++言語で開発されたダイナミックリンクライブラリ（DLL）であり、HIMC*1 の機能を利用するユーザーを支援するためのアプリケーションプログラミングインタフェース（API）セットを提供します。ユーザーは、DLL をインポートし、提供される API 関数を呼び出すことで、アプリケーションに必要なモーション制御ロジックとプログラムを作成し、モーション制御、ステータス読み出し、パラメーター設定などの操作を行うことができます。このリファレンスガイドで紹介する関数は、主に C/C++を対象としています。その他の API 環境については、その言語の DLL 呼び出しメカニズムを参照してください。対応する関数名については、各セクションの説明を参照してください。

HIMC シリーズ API は 32 ビット版と 64 ビット版の両方で利用可能です。アプリケーションのコンパイルアーキテクチャに基づいて適切なバージョンを選択してください。実行環境と DLL アーキテクチャに互換性がない場合、読み込みエラーや実行エラーが発生します。

- 32 bit: <インストール ディレクトリ>¥HIMC_API.dll
- 64 bit: <インストール ディレクトリ>¥x64_api¥HIMC_API.dll

注：

*1：HIMC シリーズとしたモデルも含まれます。

1.2 API 環境のセットアップ

API を実行する前に、環境に Visual Studio 2012 (VC++ 11.0) 再頒布可能パッケージがインストールされている必要があります。インストールパスは以下のとおりです：

- 32 bit: <インストールディレクトリ>¥vcredist_x86.exe
- 64bit: <インストールディレクトリ>¥x64_api¥vcredist_x64.exe

1.2.1 C / C++

1. <インストール ディレクトリ>\examples\API\cpp\vs_project に移動します。
2. copy_required_file.bat を実行します。
3. プロジェクト ファイル api_example.sln を開きます。

環境関連ファイル

Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll, msvcp110.dll, msvcr110.dll
HMPL Function File	HMPL_compiler.exe

1.2.2 C#

1. <インストールディレクトリ>¥examples¥API¥c_sharp に移動します。
2. copy_required_file.bat を実行します。
3. プロジェクト ファイル api_example.sln を開きます。

環境関連ファイル

C# Interface Definition	HIMC_API.cs
DLL	HIMC_API.dll, msvcp110.dll, msvcr110.dll
EXE	HMPL_compiler.exe

1.2.3 LabView

1. <インストールディレクトリ>¥examples¥API¥labview に移動します。
2. copy_required_file.bat を実行します。
3. プロジェクトファイル example.lvproj を開きます。

環境関連ファイル

LabVIEW Virtual Instrument File	<installation directory>\examples\API\labview\lib\Vis\#.vi
DLL	HIMC_API.dll, msvcp110.dll, msvcr110.dll
EXE	HMPL_compiler.exe

1.2.4 Python

1. <インストールディレクトリ>¥examples¥API¥python に移動します。
2. copy_required_file.bat を実行します。
3. サンプルの python example.py を直接実行します。

環境関連ファイル

Python Interface Definition	himc_api.py
DLL	HIMC_API.dll, msvcp110.dll, msvcr110.dll
EXE	HMPL_compiler.exe

1.3 バージョンの説明

CoE 通信対応 HIMC コントローラー（型番：MC-XX-XX-01-XX）は、ソフトウェアバージョン iA Studio 3.0 以降を適用する必要があります。MoE 通信対応 HIMC コントローラー（型番：MC-XX-01-00-XX）は、ソフトウェアバージョン iA Studio 2.X 以下を適用する必要があります。以下の表を参照して、対応するソフトウェアバージョンをご使用ください。

表 1.3.1 コントローラー製品に対応するソフトウェアバージョン

ソフトウェア バージョン	コントローラー		通信フォーマット	モデル番号
iA Studio 3.2 以上		HIMC3	EtherCAT	MC-XX-**03-01**-XX
iA Studio 3.1 iA Studio 3.0		HIMC	EtherCAT	MC-XX-**01-01**-XX
iA Studio 2.X 以下			mega-ulink	MC-XX-**01-00**-XX

注：

iA Studio 1.3 以降で採用されているモーション変数の単位：直線運動（mm）、回転運動（度）、時間（ms）

iA Studio 1.2 以前で採用されているモーション変数の単位：直線運動（m）、回転運動（rad）、時間（s）

1.4 法的免責事項

ユーザーは、本ガイドに記載されているサンプルコードを特定の用途に採用または改変することができます。ただし、様々なアプリケーションシナリオにおける正確性、有効性、安全性は保証されません。ソフトウェア実装の安全性と有効性については、ユーザーが全責任を負うものとします。

1.5 データ型

データ型は、変数を宣言したり、関数の戻り値を取得したりするために使用されます。変数の型によって、ストレージにおける占有領域のサイズと有効な値が決まります。データ型変換を行う際は、値が変換先のデータ型の範囲内に収まっている必要があります。そうでない場合、オーバーフローや切り捨てが発生し、値が変更される可能性があります。

表 1.5.1

タイプ	説明	サイズ (Byte)	有効な値
char int8_t	8bit 符号付き整数	1	-128 ~ 127
unsigned char uint8_t	8bit 符号なし整数	1	0 ~ 255
short int16_t	16bit 符号付き整数	2	-32768 ~ 32767
unsigned short uint16_t	16bit 符号なし整数	2	0 ~ 65535
int int32_t	32bit 符号付き整数	4	-2147483648 ~ 2147483647
unsigned int uint32_t	32bit 符号なし整数	4	0 ~ 4294967295
long long int64_t	64bit 符号付き整数	8	-9223372036854775808 ~ 9223372036854775807
unsigned long long uint64_t	64bit 符号なし整数	8	0 ~ 18446744073709551615
float	32bit 浮動小数点型 (6 桁の精度)	4	1.17549e-38 ~ 3.40282e+38
double	64bit 浮動小数点型 (15 桁の精度)	8	2.225074e-308 ~ 1.797693e+308
int* char* double* …	特定の型の変数の格納場所のアドレスを含むポインタ型	8	N/A
void	void 戻り値の型を持つ関数は値を返しません。	N/A	N/A
void*	あらゆる型の変数の格納場所のアドレスを含む汎用ポインタ型	8	N/A

例

```
void main()
{
	//	Value truncation
	int64_t i64 = 70000;
	int16_t i16 = (int16_t)i64;
	printf("%d", i16);		// 4464

	//	Value overflow
	int64_t i64 = -1;
	uint16_t u16 = (uint16_t)i64;
	printf("%d", u16);		// 65535

}
```

(このページはブランクになっています)

2. HIMC システムの関数

2.1 HIMC_ConnectCtrl

目的

コントローラーに接続します。

構文

```
int HIMC_ConnectCtrl(
    const ComInfo com_info,
    int *p_ctrl_id
);
```

パラメーター

com_info [in]　　接続の情報を格納する構造体

p_ctrl_id [out]　　正常に接続されたコントローラーID を受け取るバッファへのポインタ。
ユーザーはこの ID を他の API 関数で使用してこのコントローラーを操作できます。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

HIMC に接続するには、HIMC_ConnectToSimulator または HIMC_ConnectToEthernet を使用することをお勧めします。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ConnectCtrl
LabVIEW	-
Python	-

2.2 HIMC_ConnectToSimulator

目的

シミュレータに接続します。

構文

```
int HIMC_ConnectToSimulator(
    int *p_ctrl_id
);
```

パラメーター

p_ctrl_id [out] 正常に接続されたコントローラーID を受け取るバッファへのポインタ。
ユーザーはこの ID を他の API 関数で使用してこのコントローラーを操作できます。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ConnectToSimulator
LabVIEW	HIMC Connect To Simulator.vi
Python	Declare HimcAPI("Simulator")

2.3 HIMC_ConnectToEthernet

目的

イーサネットに接続します。

構文

```
int HIMC_ConnectToEthernet(
    const char *ip_address,
    const char *port,
    int *p_ctrl_id
);
```

パラメーター

ip_address [in]　IP アドレスの文字列

port [in]　ポートの文字列

p_ctrl_id [out]　正常に接続されたコントローラーID を受け取るバッファへのポインタ。
ユーザーはこの ID を他の API 関数で使用してこのコントローラーを操作できます。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ConnectToEthernet
LabVIEW	HIMC Connect To Ethernet.vi
Python	Declare HimcAPI("Ethernet")

2.4 HIMC_DisconnectCtrl

目的

コントローラーから切断します。

構文

```
int HIMC_DisconnectCtrl(
    int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisconnectCtrl
LabVIEW	HIMC Disconnect Ctrl.vi
Python	Disconnect when class is deleted

2.5 HIMC_RebootController

目的

コントローラーを再起動します。

構文

```
int HIMC_RebootController(
    int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_RebootController
LabVIEW	HIMC Reboot Controller.vi
Python	RebootController

2.6 HIMC_IsConnected

目的

コントローラーが「接続」状態にあるかどうかを照会します。

構文

```
bool HIMC_IsConnected(
    int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

コントローラーが「接続」状態の場合は bool 値 true を返し、コントローラーが他の状態の場合は bool 値 false を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsConnected
LabVIEW	-
Python	IsConnected

2.7 HIMC_IsSystemInit

目的

HIMC システムが「初期化」状態にあるかどうかを照会します。

構文

```
int HIMC_IsSystemInit(
        int ctrl_id,
        int *p_is_init
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_is_init [out] HIMC システムの「初期化」状態を受け取るバッファへのポインタ。
HIMC システムが「IsSystemInit」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsSystemInit
LabVIEW	HIMC Is System Init.vi
Python	IsSystemInit

2.8 HIMC_IsSystemOper

目的

HIMC システムが「動作」状態にあるかどうかを問い合わせます。動作状態にある場合、モーションおよび PDO 関連の関数を使用できます。

構文

```
int HIMC_IsSystemOper(
    int ctrl_id,
    int *p_is_op
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_is_op [out] HIMC システムの状態を受け取るバッファへのポインタ。
HIMC システムが「システム正常動作」状態にあり、EtherCAT が「動作」状態にある場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsSystemOper
LabVIEW	HIMC Is System Oper.vi
Python	IsSystemOper

2.9 HIMC_IsSystemPreOp

目的

HIMC システムが「動作前」状態にあるかどうかを問い合わせます。動作前状態の場合、HIMC とスレーブ間の通信は確立されていますが、モーションおよび PDO 関連機能は使用できません。

構文

```
int HIMC_IsSystemPreOp(
    int ctrl_id,
    int *p_is_preop
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_is_preop [out] HIMC システムの状態を受け取るバッファへのポインタ。
HIMC システムが「システム正常動作」状態にあり、EtherCAT が「動作前」状態にある場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsSystemPreOp
LabVIEW	HIMC Is System Pre Op.vi
Python	IsSystemPreOp

2.10 HIMC_IsSystemError

目的

HIMC システムが「エラー」状態にあるかどうかを照会します。

構文

```
int HIMC_IsSystemError(
    int ctrl_id,
    int *p_is_error
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_is_error [out] HIMC システムの「エラー」状態を受け取るバッファへのポインタ。
HIMC システムが「SystemError」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsSystemError
LabVIEW	HIMC Is System Error.vi
Python	IsSystemError

2.11 HIMC_GetECATSt

目的

コントローラーの EtherCAT ステート マシンを取得します。

構文

```
int HIMC_GetECATSt(
    int ctrl_id,
    int *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_state [out] コントローラーの EtherCAT ステート マシンを受信するためのバッファへのポインタ。
1: 初期化、2: 事前操作、4: 安全操作、8: 操作

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸および軸グループのモーション キューがクリアされます。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetECATSt
LabVIEW	HIMC Get ECAT St.vi
Python	GetECATSt

2.12 HIMC_GetSlvECATSt

目的

スレーブの EtherCAT ステート マシンを取得します。

構文

```
int HIMC_GetSlvECATSt(
    int ctrl_id,
    int slv_id
    int *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_id [in] スレーブインデックス。

p_state [out] スレーブの EtherCAT ステート マシンを受信するためのバッファへのポインタ。
1: 初期化、2: 事前操作、4: 安全操作、8: 操作

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸および軸グループのモーション キューがクリアされます。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvECATSt
LabVIEW	HIMC Get Slv ECAT St.vi
Python	GetSlvECATSt

2.13 HIMC_DisableAll

目的

すべての軸とすべての軸グループを無効にします。

構文

```
int HIMC_DisableAll(
    int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸および軸グループのモーション キューがクリアされます。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableAll
LabVIEW	HIMC Disable All.vi
Python	DisableAll

2.14 HIMC_StopAll

目的

すべての軸とすべての軸グループをキル減速で停止し、「有効」状態のままにします。

構文

```
int HIMC_StopAll(
    int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸および軸グループのモーション キューがクリアされます。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_StopAll
LabVIEW	HIMC Stop All.vi
Python	StopAll

2.15 HIMC_EStop

目的

コントローラー内のすべての実行プログラム（すべての HMPL タスクを含む）を停止し、すべての軸とすべての軸グループを無効にします。

構文

```
int HIMC_EStop(
        int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EStop
LabVIEW	HIMC E Stop.vi
Python	EStop

2.16 HIMC_GetSlaveNum

目的

コントローラーに接続されているスレーブの数を取得します。

構文

```
int HIMC_GetSlaveNum(
    int ctrl_id,
    int *p_slv_num
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
p_slv_num [out]	コントローラーに接続されているスレーブの数を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlaveNum
LabVIEW	HIMC Get Slave Num.vi
Python	GetSlaveNum

2.17 HIMC_GetFirmwareVer

目的

コントローラーのファームウェア バージョンを取得します。

構文

```
int HIMC_GetFirmwareVer(
    int ctrl_id,
    char *p_ver_buf
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_ver_buf [out] 返されたファームウェア バージョンの文字列を受け取るバッファへのポインタ。
N/A の場合は、読み取りに失敗したか、データを取得できないことを示します。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetFirmwareVer
LabVIEW	HIMC Get Firmware Ver.vi
Python	GetFirmwareVer

2.18 HIMC_GetFirmwareDate

目的

コントローラーのファームウェア ファイルの日付を取得します。

構文

```
int HIMC_GetFirmwareDate(
    int ctrl_id,
    char *p_date_buf
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_date_buf [out] 返されたファームウェア ファイルの日付の文字列を受け取るバッファへのポインタ。
N/A の場合は、読み取りに失敗したか、データを取得できないことを示します。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetFirmwareDate
LabVIEW	-
Python	GetFirmwareDate

2.19 HIMC_GetFirmwareHash

目的

コントローラーのファームウェア ハッシュ ID を取得します。

構文

```
int HIMC_GetFirmwareHash(
    int ctrl_id,
    uint64_t *p_hash
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
p_hash [out]	返されたファームウェア ハッシュ ID を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetFirmwareHash
LabVIEW	-
Python	GetFirmwareHash

2.20 HIMC_GetModelNum

目的

コントローラーのモデル番号を取得します。

構文

```
int HIMC_GetModelNum(
    int ctrl_id,
    char *p_mod_buf
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_mod_buf [out] 返されたモデル番号の文字列を受け取るバッファへのポインタ。
N/A の場合は、読み取りに失敗したか、データを取得できないことを示します。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetModelNum
LabVIEW	-
Python	GetModelNum

2.21 HIMC_GetSerialNum

目的

コントローラーのシリアル番号を取得します。

構文

```
int HIMC_GetSerialNum(
    int ctrl_id,
    char *p_sn_buf
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_sn_buf [out] 返されたシリアル番号の文字列を受け取るバッファへのポインタ。
N/A の場合は、読み取りに失敗したか、データを取得できないことを示します。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSerialNum
LabVIEW	-
Python	GetSerialNum

2.22 HIMC_GetApiVer

目的

API のソフトウェア バージョンを取得します。

構文

```
int HIMC_GetApiVer(
    char *p_ver_buf
);
```

パラメーター

p_ver_buf [out] API のソフトウェア バージョンの返された文字列を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetApiVer
LabVIEW	-
Python	GetApiVer

2.23 HIMC_ScanNetwork

目的

コントローラーとスレーブ間の接続を再スキャンします。

構文

```
int HIMC_ScanNetwork(
    int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ScanNetwork
LabVIEW	HIMC Scan Network.vi
Python	ScanNetwork

2.24 HIMC_SetEconMode

目的

HIMC API パフォーマンス モードの設定。

構文

```
int HIMC_SetEconMode(
    int ctrl_id,
    bool mode
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

mode [in] HIMC API パフォーマンス モード。
0: 高パフォーマンスモード（初期値）
1: エコノミーモード

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) ハイパフォーマンスモードでは HIMC API の平均応答速度は速くなりますが、CPU 使用率はエコノミーモードよりも高くなります。
(2) エコノミーモードでは CPU 使用率を削減できますが、HIMC API の平均応答時間は長くなります。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetEconMode
LabVIEW	HIMC Set Econ Mode.vi
Python	SetEconMode

2.25 HIMC_IsEconMode

目的

HIMC API パフォーマンス モードが エコノミーモードであるかどうかを照会します。

構文

```
int HIMC_IsEconMode(
    int ctrl_id
    bool *p_is_econ
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_is_econ [in] HIMC API パフォーマンス モードを受信するバッファへのポインタ。
エコノミーモードの場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsEconMode
LabVIEW	HIMC Is Econ Mode.vi
Python	IsEconMode

3. 軸関数

3.1 概要

HIMC は、ポイントツーポイント（P2P）、JOG、同期動作などの単軸動作コマンドを提供します。ユーザーは、アプリケーションと要件に基づいて関連する動作機能を使用できます。図 3.1.1 は、HIMC の各軸動作制御のフロー図です。動作コマンドが内蔵プロファイルジェネレータ（PG）を通過した後、基準位置が生成され、出力された基準位置に誤差補正値が加算された後にドライバーへの位置出力が生成されます。

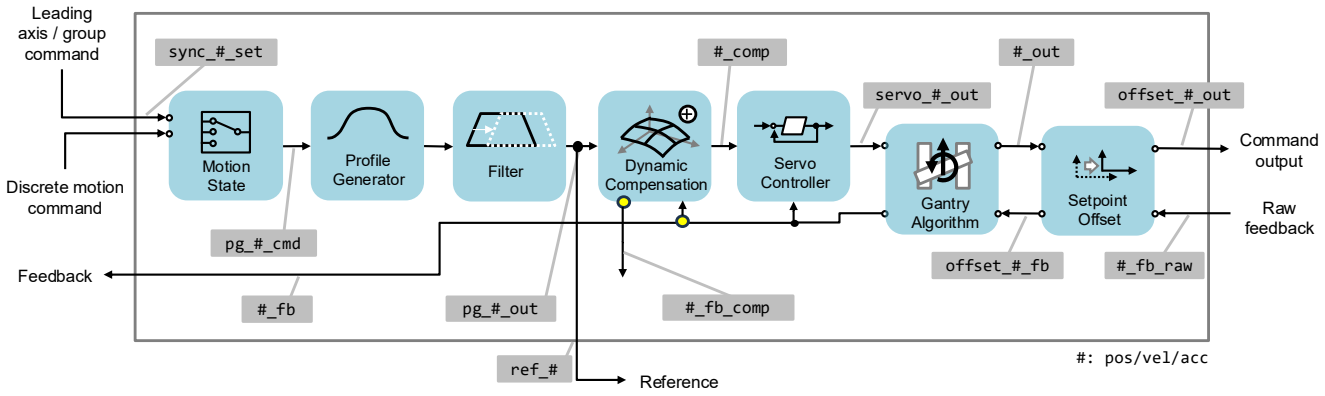


図 3.1.1

HIMC のプロファイルジェネレータには、図 3.1.2 に示すように、S カーブ速度計画機能が組み込まれています。ユーザーは、プロファイルジェネレータの最大速度、最大加速度、最大減速度、および平滑時間を設定できます。

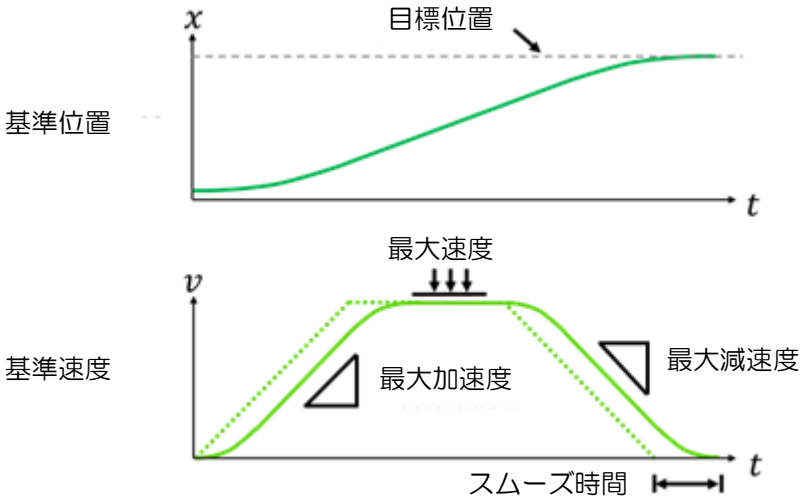


図 3.1.2

図 3.1.3 に示すように、平滑時間を追加すると基準速度は遅れますが、高加速度によって発生するジャークを効果的に低減し、システムの安定性を高めることができます。ジャーク、最大加速度、平滑時間の関係は以下のとおりです。

ジャーク = 最大加速度 / スムーズ時間 (Ts)

合計加速時間は以下の式で求められます。

合計加速時間 (T) = (Ta) + スムーズ時間 (Ts)

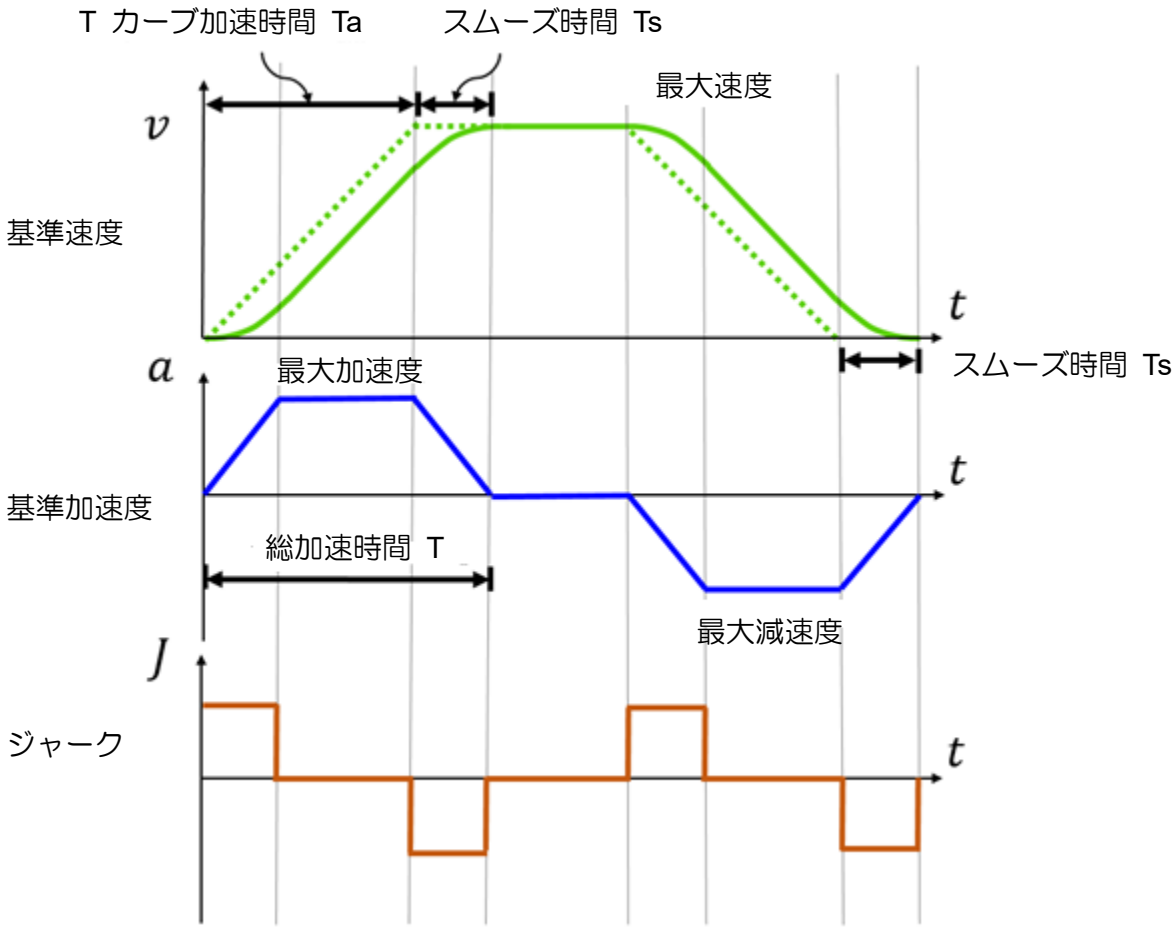


図 3.1.3

さらに、HIMC の軸動作コマンドは、動的な目標位置と速度計画の変更（オンザフライ変更）をサポートしています。軸動作中に、ユーザーは軸の目標位置、最大速度、最大加減速度を変更できます。HIMC のプロファイルジェネレータは、新しいコマンドと動作パラメーターに基づいて、新しい目標位置に移動します。

軸の動作状態は、図 3.1.4 に示すように、「移動中」と「インポジション」に分けられます。セクション I では軸の位置計画コマンドの送信を継続し、セクション II に入る前に終了します。コントローラーは、設定された位置ウィンドウと位置ウィンドウ時間に基づいて、軸がインポジションかどうかを判断します。

軸フィードバック位置が位置ウィンドウ時間経過後もリファレンス位置ウィンドウ内に留まっている場合、軸位置はインポジションとみなされます。この時点で、コントローラー内部では「軸インポジション」の状態が確立されます。ただし、位置ウィンドウ時間中に軸フィードバック位置がリファレンス位置ウィンドウを超えた場合、整定時間の計算はリセットされます。軸フィードバック位置が次に位置ウィンドウに入った時点で初めて、位置ウィンドウ時間のインポジション状態が確認されます。

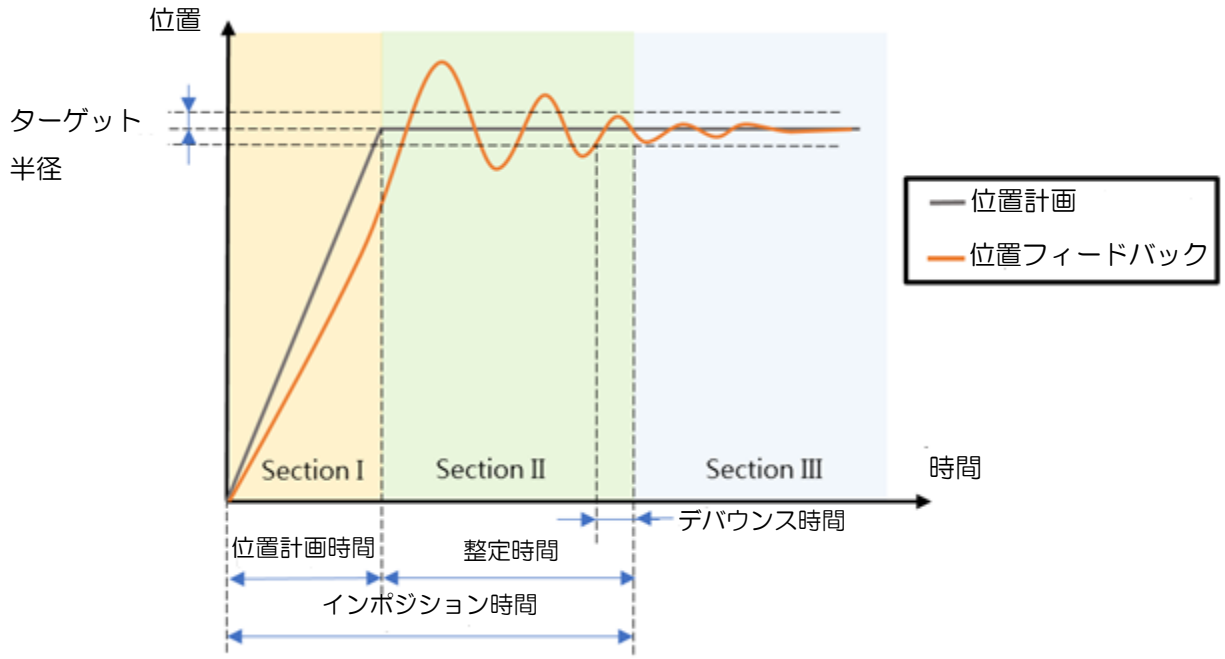


図 3.1.4

図 3.1.4 を参照すると、軸の動作ステータスは次のように説明されています。

1. セクション I: 軸が移動しており、所定の位置にありません。
2. セクション II: 軸は動いていませんが、所定の位置にありません。
3. セクション III: 軸は移動しておらず、所定の位置にありません。

軸が「同期」状態にある場合、軸グループまたはマスター軸が軸動作のモーションプロファイルを生成します。軸自体はモーションプロファイルを生成せず、軸グループまたはマスター軸によって計画された参照位置に従うだけです。

3.1.1 軸変数

軸変数は、動作コマンド変数、プロファイルジェネレータ変数、ステータス変数、および CoE オブジェクト変数に分類されます。ユーザーは iA Studio のスコープマネージャから必要な変数を選択できます（「iA Studio ユーザーガイド」のセクション 4.8 を参照）。詳細な説明は表 3.1.1.1 から表 3.1.1.6 に示されています。

表 3.1.1.1 軸のモーションコマンド変数

名称	変数	単位	説明
Sync Position Setpoint	sync_pos_set	mm または deg	同期位置設定値。軸が同期動作（軸グループ、カム、ギア操作など）を行っているときに追従する目標位置です。
Position Command	pg_pos_cmd	mm または deg	プロファイルジェネレータの位置コマンド。軸が離散動作（ポイントツーポイント）しているときに追従する目標位置です。
Reference Position	ref_pos	mm または deg	基準位置。これは、事前に定義されたモーションプロファイルに従ってプロファイルジェネレータから生成された位置設定点です。
Reference Velocity	ref_vel	mm/s または deg/s	基準速度。これは、事前に定義された動作プロファイルに従ってプロファイルジェネレータから生成される速度設定値です。
Reference Acceleration	ref_acc	mm/s^2 または deg/s^2	基準加速度。これは、事前に定義された動作プロファイルに従ってプロファイルジェネレータから生成される加速度設定値です。
Position Compensation	pos_comp_set	mm または deg	位置補正値。動的誤差補正機能の誤差マップの出力です。機能が無効になっている場合はゼロになります。
Compensated Position	pos_comp	mm または deg	補正された位置指令値。基準位置と位置補正値の合計です。
Position Offset	pos_offset	mm または deg	位置オフセット。デフォルト値はゼロです。モーターを動かさずに新しい軸位置を設定した場合、オフセットはゼロ以外の値になります。
Position Output	pos_out	mm または deg	位置出力。位置オフセットを除いた軸位置コマンドです。
Velocity Output	vel_out	mm/s または deg/s	速度出力。速度オフセットを除いた軸速度コマンドです。
Acceleration Output	acc_out	mm/s^2 または deg/s^2	加速度出力。加速度オフセットを除いた軸加速度コマンドです。
Offsetted Position Output	offset_pos_out	mm または deg	オフセット付き位置出力。位置オフセットを加算した最終的な軸位置コマンドです。値は「カウント」単位に変換され、対応するスレーブに送信されます。
Raw Position	pos_fb_raw	mm また	生の位置フィードバック。スレーブから読み取ったエンコ

名称	変数	単位	説明
Feedback		は deg	ーダフィードバックです。
Offsetted Position Feedback	offset_pos_fb	mm または deg	オフセット位置フィードバック。位置オフセットを付加した位置フィードバックです。
Position Feedback	pos_fb	mm または deg	位置フィードバック。軸座標系で行われます。
Velocity Feedback	vel_fb	mm/s または deg/s	速度フィードバック。軸座標系です。
Position Error	pos_err	mm または deg	位置誤差。位置出力と生の位置フィードバックの差です。
Velocity Error	vel_err	mm/s または deg/s	速度誤差。速度出力と生の速度フィードバックの差です。
Move Time	movetime	ms	移動時間
Settling Time	settlingtime	ms	整定時間

表 3.1.1.2 軸のプロファイルジェネレータ変数

名称	変数	単位	説明
Max. Profile Velocity	max_vel	mm/s または deg/s	最大プロファイル速度。必ずしも到達するとは限りません。
Max. Profile Acceleration	max_acc	mm/s^2 または deg/s^2	最大プロファイル加速度。必ずしも到達するとは限りません。
Profile Deceleration	max_dec	mm/s^2 または deg/s^2	最大プロファイル減速。必ずしも到達するとは限りません。
Smooth Time	sm_factor	ms	スムーズタイム。入力範囲は 0～500 です。値を大きくすると動作中の機械振動を軽減できますが、動作時間全体に影響します。

表 3.1.1.3 軸の状態変数

名称	変数	単位	説明
Fault Status	fault_status	N/A	軸のエラー状態。ビットの定義については表 3.1.1.4 を参照してください。
Motion Status	motion_status	N/A	軸の動作ステータス。ビットの定義については表 3.1.1.5 を参照してください。

表 3.1.1.4 軸エラーステータスのビット定義

Bit	名称	説明	デフォルトの応答
0	Error Stop	軸が「エラー停止」状態	N/A
1	Drive fault	スレーブドライバー障害	コントローラーは軸を無効にします
2	Position error	位置誤差が保護限度を超えています	コントローラーは軸を無効にします
3	Hardware right limit	軸ハードウェア右制限がトリガーされました	コントローラーは動作を停止します
4	Hardware left limit	軸ハードウェアの左制限がトリガーされました	コントローラーは動作を停止します
5	Software right limit	軸ソフトウェアの右制限がトリガーされました	コントローラーは動作を停止します
6	Software left limit	軸ソフトウェアの左制限がトリガーされました	コントローラーは動作を停止します

表 3.1.1.5 軸動作ステータスのビット定義

Bit	名称	説明	備考
0	Enabled	軸が有効になります	N/A
1	Moving	軸が動いています	セクション 3.1 を参照
2	In Position	軸は所定の位置にあります	セクション 3.1 を参照
3	Synchronous	軸は「同期」状態にあります	軸は軸グループ内にあるか、同期動作中のスレーブ軸です
4	Group	軸は軸グループ内にあります	セクション 6.1 を参照
5	Gantry	軸はガントリー軸です	セクション 5.1 を参照
6	Input Shape	入力シェイプ フィルターを有効にします	セクション 13.1 を参照
7	VSF	VSF フィルターを有効にします	セクション 13.1 を参照
8	Gear	軸は電子ギアスレーブ軸です	セクション 4.1 を参照
9	Cam	軸は電子カムスレーブ軸です	まだサポートされていません
10	Accelerating	軸が加速しています	セクション 3.1 を参照
11	Homed	軸が原点復帰を完了します	N/A
12	Homing	軸が原点復帰中です	N/A
13	Home Switch	軸の原点スイッチがオンになっています	N/A

表 3.1.1.6 軸の CoE オブジェクト変数

名称	変数	単位	説明
Power Drive State	power_drive_st	N/A	パワー ドライブ ステート (PDS)
Controlword	coe_controlword	N/A	軸制御オブジェクト
Statusword	coe_statusword	N/A	軸ステータス オブジェクト
Modes of Operation	servo_type	N/A	軸制御モードの書き込まれた値
Modes of Operational Display	coe_op_modes_display	N/A	軸制御モードの表示値
Target Position	coe_pos_cmd	count	目標位置コマンド
Target Velocity	coe_vel_cmd	count/s	目標速度コマンド
Target Torque	coe_trq_cmd	N/A	目標トルクコマンド
Position Actual Value	coe_pos_fb	count	位置フィードバック値
Velocity Actual Value	coe_vel_fb	count/s	速度フィードバック値
Torque Actual Value	coe_trq_fb	N/A	トルクフィードバック値
Touch Probe Function	tp_function	N/A	タッチプローブ機能の設定値
Touch Probe Status	tp_status	N/A	タッチプローブ機能のステータス
Touch Probe 1 Positive Value	tp1_positive_edge	count	タッチプローブ 1 の正エッジの位置の値
Touch Probe 1 Negative Value	tp1_negative_edge	count	タッチプローブ 1 の負エッジの位置の値
Touch Probe 2 Positive Value	tp2_positive_edge	count	タッチプローブ 2 の正エッジの位置の値
Touch Probe 2 Negative Value	tp2_negative_edge	count	タッチプローブ 2 の負エッジの位置の値
Negative Limit Switch	di_bit_HWLL	N/A	負リミットスイッチの信号
Positive Limit Switch	di_bit_HWRL	N/A	正リミットスイッチの信号
Home Switch	di_bit_HOME	N/A	原点スイッチの信号

3.2 軸モーション制御

3.2.1 HIMC_Enable

目的
軸を有効にします

構文

```
int HIMC_Enable(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値
関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考
この機能を使用する場合、ユーザーはオブジェクト 0x6040 (制御ワード) とオブジェクト 0x6041 (ステータス ワード) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Enable
LabVIEW	HIMC Enable.vi
Python	Enable

3.2.2 HIMC_Disable

目的

軸を無効にします

構文

```
int HIMC_Disable(
        int ctrl_id,
        int axis_id
);
```

パラメーター

ctrl_id [in]　　HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in]　　軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) 軸のモーションキューがクリアされます。
(2) この機能を使用する場合、ユーザーはオブジェクト 0x6040 (制御ワード) とオブジェクト 0x6041 (ステータス ワード) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Disable
LabVIEW	HIMC Disable.vi
Python	Disable

3.2.3 HIMC_Reset

目的

軸が「エラー停止」状態のときに軸をリセットします。

構文

```
int HIMC_Reset(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) 軸がエラー停止モードのときにこの機能を操作します。
(2) この機能を使用する場合、ユーザーはオブジェクト 0x6040 (制御ワード) とオブジェクト 0x6041 (ステータス ワード) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Reset
LabVIEW	HIMC Reset.vi
Python	Reset

3.2.4 HIMC_MoveAbs

目的

軸を絶対的な目標位置に移動します。

構文

```
int HIMC_MoveAbs(
    int    ctrl_id,
    int    axis_id,
    double pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
pos [in]	絶対ターゲット位置の値 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーは対応するコマンドを PDO として設定する必要があります。
例えば、CSP モードを 0x607A（ターゲット位置）として設定します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_MoveAbs
LabVIEW	HIMC Move Abs.vi
Python	MoveAbs

3.2.5 HIMC_MoveRel

目的

軸を相対距離だけ移動します

構文

```
int HIMC_MoveRel(
    int    ctrl_id,
    int    axis_id,
    double rel_dist
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

rel_dist [in] 相対距離の値
パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーは対応するコマンドを PDO として設定する必要があります。
例えば、CSP モードを 0x607A（ターゲット位置）として設定します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_MoveRel
LabVIEW	HIMC Move Rel.vi
Python	MoveRel

3.2.6 HIMC_MoveVel

目的

特定の速度を設定し、その速度で終わりのない動作を開始します。

構文

```
int HIMC_MoveVel(
    int    ctrl_id,
    int    axis_id,
    double vel
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス.
vel [in]	特定の速度の値 正負の値によって動きの方向が決まります。 パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーは対応するコマンドを PDO として設定する必要があります。
たとえば、CSP モードを 0x607A (ターゲット位置) として構成します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_MoveVel
LabVIEW	HIMC Move Vel.vi
Python	MoveVel

3.2.7 HIMC_MoveTrq

目的

特定のトルクで終わりのない動きを開始します。

構文

```
int HIMC_MoveTrq(
        int      ctrl_id,
        int      axis_id,
        double trq_cmd
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス.

trq_cmd [in] トルクコマンド
パラメーター 単位: N-m

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) この機能は「プロファイルトルク」モードにのみ適用されます。
(2) トルク指令がモーターの連続トルクより大きい場合、モーターは連続トルクの値で動きます。
(3) ユーザーはオブジェクト 0x6071（目標トルク）を PDO として設定し、モーターの力定数を設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_MoveTrq
LabVIEW	HIMC Move Trq.vi
Python	MoveTrq

3.2.8 HIMC_MovePVT

目的

指定された位置 (P)、速度 (V)、時間 (T) に基づいて軸を指定された位置に移動します。

構文

```
int HIMC_MovePVT(
        int      ctrl_id,
        int      axis_id,
        double *time,
        double *pos,
        double *vel,
        int      num_pt
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス.
time [in]	ユーザーが指定した時間配列へのポインタ。 パラメーター 単位: ms
pos [in]	ユーザーが指定した位置配列へのポインタ。 パラメーター単位: mm または deg
vel [in]	ユーザーが指定した速度配列へのポインタ。 パラメーター単位: mm/s または deg/s
num_pt [in]	PVT モーションポイントの数。最大値は 32 です。 時間の長さ、位置、速度の配列はこのパラメーターと同じである必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーは対応するコマンドを PDO として設定する必要があります。
たとえば、CSP モードを 0x607A (ターゲット位置) として構成します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_MovePVT
LabVIEW	-
Python	MovePVT

3.2.9 HIMC_Stop

目的

軸の動きを停止します。

構文

```
int HIMC_Stop(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸のモーションキューがクリアされます。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Stop
LabVIEW	HIMC Stop.vi
Python	Stop

3.2.10 HIMC_Halt

目的

軸の動きを停止するには、その速度を 0 に設定します。

構文

```
int HIMC_Halt(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはオブジェクト 0x6040 (制御ワード) とオブジェクト 0x6041 (ステータス ワード) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Halt
LabVIEW	HIMC Halt.vi
Python	Halt

3.2.11 HIMC_Resume

目的

軸の動作を「停止」状態から再開します。

構文

```
int HIMC_Resume(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはオブジェクト 0x6040 (制御ワード) とオブジェクト 0x6041 (ステータス ワード) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Resume
LabVIEW	HIMC Resume.vi
Python	Resume

3.3 軸設定

3.3.1 HIMC_GetVel

目的

軸の移動速度の目標値を取得します。

構文

```
int HIMC_GetVel(
    int     ctrl_id,
    int     axis_id,
    double *p_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_vel [out] 軸の移動速度の目標値を受け取るバッファへのポインタ
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetVel
LabVIEW	-
Python	GetVel

3.3.2 HIMC_SetVel

目的

軸の動作速度の目標値を設定します。

構文

```
int HIMC_SetVel(
    int    ctrl_id,
    int    axis_id,
    double vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

vel [in] 軸の新しい動作速度の目標値
パラメーター単位: mm/s または deg/s
入力範囲: ゼロ以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetVel
LabVIEW	HIMC Set Vel.vi
Python	SetVel

3.3.3 HIMC_GetAcc

目的

軸のモーション加速度の目標値を取得します。

構文

```
int HIMC_GetAcc(
    int    ctrl_id,
    int    axis_id,
    double *p_acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_acc [out] 軸のモーション加速度の目標値を受け取るバッファへのポインタ。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetAcc
LabVIEW	-
Python	GetAcc

3.3.4 HIMC_SetAcc

目的

軸のモーション加速度の目標値を設定します。

構文

```
int HIMC_SetAcc(
    int    ctrl_id,
    int    axis_id,
    double acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

acc [in] 軸の新しいモーション加速の目標値。
パラメーター単位: mm/s^2 または deg/s^2
入力範囲: ゼロ以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetAcc
LabVIEW	HIMC Set Acc.vi
Python	SetAcc

3.3.5 HIMC_SetAccTime

目的

軸の加速時間を設定します。

構文

```
int HIMC_SetAccTime(
    int    ctrl_id,
    int    axis_id,
    double acc_time
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

acc_time [in] 軸の加速時間
パラメーター単位: ms
入力範囲: ゼロ以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetAccTime
LabVIEW	HIMC Set Acc Time.vi
Python	SetAccTime

3.3.6 HIMC_GetDec

目的

軸のモーション減速の目標値を取得します。

構文

```
int HIMC_GetDec(
    int      ctrl_id,
    int      axis_id,
    double *p_dec
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_dec [out] 軸のモーション減速の目標値を受け取るバッファへのポインタ。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetDec
LabVIEW	-
Python	GetDec

3.3.7 HIMC_SetDec

目的

軸のモーション減速の目標値を設定します。

構文

```
int HIMC_SetDec(
    int    ctrl_id,
    int    axis_id,
    double dec
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

dec [in] 軸の新しいモーション減速の目標値。
パラメーター単位: mm/s^2 または deg/s^2
入力範囲: ゼロ以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetDec
LabVIEW	HIMC Set Dec.vi
Python	SetDec

3.3.8 HIMC_SetDecTime

目的

軸の減速時間を設定します。

構文

```
int HIMC_SetDecTime(
    int    ctrl_id,
    int    axis_id,
    double dec_time
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

dec_time [in] 軸の減速時間
パラメーター単位: ms
入力範囲: ゼロ以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetDecTime
LabVIEW	HIMC Set Dec Time.vi
Python	SetDecTime

3.3.9 HIMC_GetKillDec

目的

軸のキル減速を取得します。

構文

```
int HIMC_GetKillDec(
    int      ctrl_id,
    int      axis_id,
    double *p_kill_dec
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_kill_dec [out] 軸のキル減速を受け取るバッファへのポインタ。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetKillDec
LabVIEW	HIMC Get Kill Dec.vi
Python	GetKillDec

3.3.10 HIMC_SetKillDec

目的

軸のキル減速を設定します。

構文

```
int HIMC_SetKillDec(
    int    ctrl_id,
    int    axis_id,
    double kill_dec
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

kill_dec [in] 軸の新しいキル減速。
パラメーター単位: mm/s^2 または deg/s^2
入力範囲: ゼロ以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetKillDec
LabVIEW	HIMC Set Kill Dec.vi
Python	SetKillDec

3.3.11 HIMC_GetSWRL

目的

軸のソフトウェアの正しい制限位置を取得します。

構文

```
int HIMC_GetSWRL(
    int    ctrl_id,
    int    axis_id,
    double *p_right_limit_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_right_limit_pos [out]	軸のソフトウェア右限界位置を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSWRL
LabVIEW	HIMC Get SWRL.vi
Python	GetSWRL

3.3.12 HIMC_SetSWRL

目的

軸のソフトウェア右限界位置を設定します。

構文

```
int HIMC_SetSWRL(
    int    ctrl_id,
    int    axis_id,
    double right_limit_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
right_limit_pos [in]	軸の新しいソフトウェア右限界位置。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSWRL
LabVIEW	HIMC Set SWRL.vi
Python	SetSWRL

3.3.13 HIMC_GetSWLL

目的

軸のソフトウェア左限界位置を取得します。

構文

```
int HIMC_GetSWLL(
    int    ctrl_id,
    int    axis_id,
    double *p_left_limit_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_left_limit_pos [out]	軸のソフトウェア左限界位置を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSWLL
LabVIEW	HIMC Get SWLL.vi
Python	GetSWLL

3.3.14 HIMC_SetSWLL

目的

軸のソフトウェア左限界位置を設定します。

構文

```
int HIMC_SetSWLL(
    int    ctrl_id,
    int    axis_id,
    double left_limit_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
left_limit_pos [in]	新しいソフトウェアは軸の限界位置を左にしました。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSWLL
LabVIEW	HIMC Set SWLL.vi
Python	SetSWLL

3.3.15 HIMC_GetSMTime

目的

軸のプロファイルのスムーズ時間を取得します。

構文

```
int HIMC_GetSMTime(
        int      ctrl_id,
        int      axis_id,
        double *p_smooth_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_smooth_time [out]	軸のプロファイルのスムーズ時間を受け取るバッファへのポインタ。 パラメーター単位: ms

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSMTime
LabVIEW	HIMC Get SM Time.vi
Python	GetSMTime

3.3.16 HIMC_SetSMTime

目的

軸のプロファイルのスムーズ時間を設定します。

構文

```
int HIMC_SetSMTime(
        int      ctrl_id,
        int      axis_id,
        double smooth_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
smooth_time [in]	軸の新しいプロファイルのスムーズ時間。 パラメーター単位: ms 入力範囲: 0 ~ 500

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能は軸が移動している場合には適用できません。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSMTime
LabVIEW	HIMC Set SM Time.vi
Python	SetSMTime

3.3.17 HIMC_GetMoveTime

目的

軸の移動時間を取得します。

構文

```
int HIMC_GetMoveTime(
    int     ctrl_id,
    int     axis_id,
    double *p_move_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_move_time [out]	軸の移動時間を受け取るバッファへのポインタ。 パラメーター単位: ms

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetMoveTime
LabVIEW	HIMC Get Move Time.vi
Python	GetMoveTime

3.3.18 HIMC_GetSettlingTime

目的

軸の安定時間を取得します。

構文

```
int HIMC_GetSettlingTime(
    int      ctrl_id,
    int      axis_id,
    double *p_settling_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_settling_time [out]	軸の安定時間を受け取るバッファへのポインタ。 パラメーター単位: ms

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSettlingTime
LabVIEW	HIMC Get Settling Time.vi
Python	GetSettlingTime

3.3.19 HIMC_SetPos

目的

軸の位置を設定し、原点オフセットを変更します。

構文

```
int HIMC_SetPos(
    int    ctrl_id,
    int    axis_id,
    double pos
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

pos [in] 軸の現在の位置の値。
パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能は、軸が「同期済み」状態、軸グループに追加されている状態、またはエラー状態の場合に適用されません。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPos
LabVIEW	HIMC Set Pos.vi
Python	SetPos

3.3.20 HIMC_GetPosFb

目的

軸のフィードバック位置を取得します。

構文

```
int HIMC_GetPosFb(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_pos [out] 軸のフィードバック位置を受け取るバッファへのポインタ。
パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この関数を使用する場合、ユーザーはオブジェクト 0x6064 (位置の実際の値) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetPosFb
LabVIEW	HIMC Get Pos Fb.vi
Python	GetPosFb

3.3.21 HIMC_GetPosFbComp

目的

軸の位置補正値を使用してフィードバック位置を取得します。

構文

```
int HIMC_GetPosFbComp(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_pos [out]	軸の位置補正値を含むフィードバック位置を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この関数を使用する場合、ユーザーはオブジェクト 0x6064 (位置の実際の値) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetPosFbComp
LabVIEW	-
Python	GetPosFbComp

3.3.22 HIMC_GetPosOffset

目的

軸の位置オフセットを取得します。

構文

```
int HIMC_GetPosOffset(
    int      ctrl_id,
    int      axis_id,
    double *p_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_pos_err [out]	軸の位置誤差を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetPosOffset
LabVIEW	HIMC Get Pos Offset.vi
Python	GetPosOffset

3.3.23 HIMC_GetPosErr

目的

軸の位置誤差を取得します。

構文

```
int HIMC_GetPosErr(
    int    ctrl_id,
    int    axis_id,
    double *p_pos_err
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_pos_err [out]	軸の位置誤差を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetPosErr
LabVIEW	HIMC Get Pos Err.vi
Python	GetPosErr

3.3.24 HIMC_GetVelFb

目的

軸の速度フィードバックを取得します。

構文

```
int HIMC_GetVelFb(
    int     ctrl_id,
    int     axis_id,
    double *p_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_vel [out] 軸の速度フィードバックを受信するバッファへのポインタ。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetVelFb
LabVIEW	HIMC Get Vel Fb.vi
Python	GetVelFb

3.3.25 HIMC_GetVelErr

目的

軸の速度誤差を取得します。

構文

```
int HIMC_GetVelErr(
    int     ctrl_id,
    int     axis_id,
    double *p_vel_err
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_vel_err [out] 軸の速度誤差を受け取るバッファへのポインタ。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetVelErr
LabVIEW	HIMC Get Vel Err.vi
Python	GetVelErr

3.3.26 HIMC_GetCurrFb

目的

軸の現在のフィードバックを取得します。

構文

```
int HIMC_GetCurrFb(
        int      ctrl_id,
        int      axis_id,
        double *p_curr
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_curr [out] 軸の現在のフィードバックを受け取るバッファへのポインタ。
パラメーター単位：アンペア（A）

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはオブジェクト 0x6077 (トルクの実際の値) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetCurrFb
LabVIEW	-
Python	GetCurrFb

3.3.27 HIMC_GetRefPos

目的

軸の基準位置を取得します。

構文

```
int HIMC_GetRefPos(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_pos [out] 軸の参照位置を受け取るバッファへのポインタ。
パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetRefPos
LabVIEW	HIMC Get Ref Pos.vi
Python	GetRefPos

3.3.28 HIMC_GetRefVel

目的

軸の基準速度を取得します。

構文

```
int HIMC_GetRefVel(
    int    ctrl_id,
    int    axis_id,
    double *p_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_vel [out] 軸の参照速度を受け取るバッファへのポインタ。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetRefVel
LabVIEW	HIMC Get Ref Vel.vi
Python	GetRefVel

3.3.29 HIMC_GetRefAcc

目的

軸の基準加速度を取得します。

構文

```
int HIMC_GetRefAcc(
    int    ctrl_id,
    int    axis_id,
    double *p_acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_acc [out] 軸の参照加速度を受け取るバッファへのポインタ。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetRefAcc
LabVIEW	HIMC Get Ref Acc.vi
Python	GetRefAcc

3.3.30 HIMC_GetPosOut

目的

コントローラーからスレーブ ドライバーに送信された軸の位置コマンド出力を取得します。

構文

```
int HIMC_GetPosOut(
    int    ctrl_id,
    int    axis_id,
    double *p_pos
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_pos [out] 軸の位置コマンド出力を受け取るバッファへのポインタ。
パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetPosOut
LabVIEW	HIMC Get Pos Out.vi
Python	GetPosOut

3.3.31 HIMC_GetVelOut

目的

コントローラーからスレーブ ドライバーに送信された軸の速度コマンド出力を取得します。

構文

```
int HIMC_GetVelOut(
    int    ctrl_id,
    int    axis_id,
    double *p_vel
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_vel [out]	軸の速度コマンド出力を受け取るバッファへのポインタ。 パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetVelOut
LabVIEW	HIMC Get Vel Out.vi
Python	GetVelOut

3.3.32 HIMC_GetAccOut

目的

コントローラーからスレーブ ドライバーに送信された軸の加速コマンド出力を取得します。

構文

```
int HIMC_GetAccOut(
    int    ctrl_id,
    int    axis_id,
    double *p_acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_acc [out] 軸の加速コマンド出力を受け取るバッファへのポインタ。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetAccOut
LabVIEW	HIMC Get Acc Out.vi
Python	GetAccOut

3.3.33 HIMC_IgnoreHWL

目的

ハードウェア制限保護の警告メッセージを無視します。

構文

```
int HIMC_IgnoreHWL(
    int ctrl_id,
    int axis_id,
    int cmd
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

cmd [in] メッセージを無視するには「1」に設定します。
メッセージを復元するには「0」に設定します（初期設定）。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IgnoreHWL
LabVIEW	HIMC Ignore HWL.vi
Python	IgnoreHWL

3.3.34 HIMC_IgnoreSWL

目的

ソフトウェア制限保護の警告メッセージを無視します。

構文

```
int HIMC_IgnoreSWL(
    int ctrl_id,
    int axis_id,
    int cmd
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

cmd [in] メッセージを無視するには「1」に設定します。
メッセージを復元するには「0」に設定します（初期値）。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IgnoreSWL
LabVIEW	HIMC Ignore SWL.vi
Python	IgnoreSWL

3.3.35 HIMC_IgnorePE

目的

位置誤差制限の警告メッセージを無視します。

構文

```
int HIMC_IgnorePE(
    int ctrl_id,
    int axis_id,
    int cmd
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
cmd [in]	メッセージを無視するには「1」に設定します。 メッセージを復元するには「0」に設定します（初期値）。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IgnorePE
LabVIEW	HIMC Ignore PE.vi
Python	IgnorePE

3.3.36 HIMC_GetAxisNum

目的

コントローラーに接続されている軸の数を取得します。

構文

```
int HIMC_GetAxisNum(
    int ctrl_id,
    int *num
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

num [out] コントローラーに接続された軸の数を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetAxisNum
LabVIEW	HIMC Get Axis Num.vi
Python	GetAxisNum

3.3.37 HIMC_SetVelScale

目的

軸動作の速度スケールを設定します。

構文

```
int HIMC_SetVelScale(
    int    ctrl_id,
    int    axis_id,
    double vel_scale
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

vel_scale [in] 軸動作の新しい速度スケール。
入力範囲: 0 ~ 100

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetVelScale
LabVIEW	HIMC Set Vel Scale.vi
Python	SetVelScale

3.3.38 HIMC_GetVelScale

目的

軸動作の速度スケールを取得します。

構文

```
int HIMC_GetVelScale(
    int    ctrl_id,
    int    axis_id,
    double *p_vel_scale
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_vel_scale [out]	軸動作の速度スケールを受け取るバッファへのポインタ。 範囲は 0 から 100 です。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetVelScale
LabVIEW	HIMC Get Vel Scale.vi
Python	GetVelScale

3.3.39 HIMC_SetRollover

目的

軸の位置ロールオーバー値を設定します。

構文

```
int HIMC_SetRollover(
        int     ctrl_id,
        int     axis_id,
        double rollover_val
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
rollover_val [in]	軸の位置ロールオーバー値。 パラメーター単位: mm または deg 入力範囲: ゼロまたは正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) パラメーター「rollover_val」が 0 に設定されている場合、関数は閉じられます。
(2) この機能は軸が無効になっている場合にのみ適用されます。
(3) 軸が軸グループに追加されている場合、この機能は適用されません。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetRollover
LabVIEW	HIMC Set Rollover.vi
Python	SetRollover

3.3.40 HIMC_GetRolloverTurns

目的

軸がロールオーバー モードのときに回転数を取得します。

構文

```
int HIMC_GetRolloverTurns(
    int ctrl_id,
    int axis_id,
    int *p_turns
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_turns [out] 軸がロールオーバー モードのときに回転数を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetRolloverTurns
LabVIEW	HIMC Get Rollover Turns.vi
Python	GetRolloverTurns

3.3.41 HIMC_SetOpMode

目的

軸の動作モードを設定します。

構文

```
int HIMC_SetOpMode(
    int ctrl_id,
    int axis_id,
    int op_mode
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

op_mode [in] 軸の新しい操作モード。
入力範囲: 8(CSP), 9(CSV), 10(CST)

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetOpMode
LabVIEW	HIMC Set Op Mode.vi
Python	SetOpMode

3.3.42 HIMC_SetBufferMode

目的
軸のバッファ モードを設定します。

構文

```
int HIMC_SetBufferMode(
    int ctrl_id,
    int axis_id,
    int buf_mode
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
buf_mode [in]	軸の新しいバッファ モード。 入力範囲: 0 (即時停止モード)、1 (バッファモード)

戻りの値
関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetBufferMode
LabVIEW	HIMC Set Buffer Mode.vi
Python	SetBufferMode

3.3.43 HIMC_GetBufferMode

目的

軸のバッファ モードを取得します。

構文

```
int HIMC_GetBufferMode(
    int ctrl_id,
    int axis_id,
    int* buf_mode
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

buf_mode [out] 軸のバッファ モードを受け取るバッファへのポインタ。
0: 即時停止モード、1: バッファモード

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetBufferMode
LabVIEW	HIMC Get Buffer Mode.vi
Python	GetBufferMode

3.3.44 HIMC_GetCmdNum

目的

軸のバッファリングコマンドの数を取得します。

構文

```
int HIMC_GetCmdNum(
    int ctrl_id,
    int axis_id,
    int* cmd_num
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

cmd_num [out] 軸のバッファリング コマンドの番号を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetCmdNum
LabVIEW	HIMC Get Cmd Num.vi
Python	GetCmdNum

3.3.45 HIMC_GetMultiAxesFeedbackPos

目的

軸リストのフィードバック位置を取得します。

構文

```
int HIMC_GetMultiAxesFeedbackPos(
        int ctrl_id,
        int *p_axes_id_array,
        int num_of_axes,
        int *p_pos_array
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
p_axes_id_array [in]	軸 ID のリストを格納するバッファへのポインタ。
num_of_axes [in]	軸の数
p_pos_array [out]	軸リストのフィードバック位置を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetMultiAxesFeedbackPos
LabVIEW	HIMC Get Multi Axes Feedback Pos.vi
Python	GetMultiAxesFeedbackPos

3.3.46 HIMC_SetPosWindow

目的

軸の位置ウィンドウを設定します。

構文

```
int HIMC_SetPosWindow(
    int ctrl_id,
    int axis_id,
    double pos_window
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
pos_window [in]	軸の新しい位置ウィンドウ。 パラメーター単位: mm または deg 入力範囲: 0 以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPosWindow
LabVIEW	-
Python	SetPosWindow

3.3.47 HIMC_GetPosWindow

目的

軸の位置ウィンドウを取得します。

構文

```
int HIMC_GetPosWindow(
    int ctrl_id,
    int axis_id,
    double *pos_window
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
pos_window [out]	軸の位置ウィンドウを受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetPosWindow
LabVIEW	-
Python	GetPosWindow

3.3.48 HIMC_SetPosWindowTime

目的

軸の位置ウィンドウ時間を設定します。

構文

```
int HIMC_SetPosWindowTime(
    int ctrl_id,
    int axis_id,
    double pos_window_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
pos_window_time [in]	軸の新しい位置ウィンドウ時間。 パラメーター単位: ms 入力範囲: 0 以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPosWindowTime
LabVIEW	-
Python	SetPosWindowTime

3.3.49 HIMC_GetPosWindowTime

目的

軸の位置ウィンドウの時間を取得します。

構文

```
int HIMC_GetPosWindowTime(
    int ctrl_id,
    int axis_id,
    double *pos_window_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
pos_window_time [out]	軸の位置ウィンドウ時間を受け取るバッファへのポインタ。 パラメーター単位: ms

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetPosWindowTime
LabVIEW	-
Python	GetPosWindowTime

3.4 軸ステータス

3.4.1 HIMC_IsEnabled

目的
軸の「有効」ステータスを照会します。

構文

```
int HIMC_IsEnabled(
        int ctrl_id,
        int axis_id,
        int *p_enabled
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_enabled [out] 軸の有効ステータスを受け取るバッファへのポインタ。
軸が「有効」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値
関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考
この機能を使用する場合、ユーザーはオブジェクト 0x6041 (ステータス ワード) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsEnabled
LabVIEW	HIMC Is Enabled.vi
Python	IsEnabled

3.4.2 HIMC_IsMoving

目的

軸の「移動」状態を問い合わせます。軸が移動中の場合、PG（プロファイルジェネレータ）は新しい位置を出力し続けます。

構文

```
int HIMC_IsMoving(
        int ctrl_id,
        int axis_id,
        int *p_is_moving
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_moving [out]	軸の移動状態を受け取るバッファへのポインタ。 軸が「移動中」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsMoving
LabVIEW	HIMC Is Moving.vi
Python	IsMoving

3.4.3 HIMC_IsInPos

目的

軸の「インポジション」状態を問い合わせます。軸がインポジション状態の場合、位置誤差は特定の期間（位置ウィンドウ時間）にわたって位置ウィンドウ内に維持されます。

構文

```
int HIMC_IsInPos(
        int ctrl_id,
        int axis_id,
        int *p_in_position
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_in_position [out]	軸のインポジションステータスを受け取るバッファへのポインタ。 軸が「InPos」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsInPos
LabVIEW	HIMC Is In Pos.vi
Python	IsInPos

3.4.4 HIMC_IsErrorStop

目的

軸が「エラー停止」状態にあるかどうかを照会します。

構文

```
int HIMC_IsErrorStop(
        int ctrl_id,
        int axis_id,
        int *p_is_errorstop
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_errorstop [out]	軸のエラー停止ステータスを受け取るバッファへのポインタ。 軸が「ErrorStop」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsErrorStop
LabVIEW	HIMC Is Error Stop.vi
Python	IsErrorStop

3.4.5 HIMC_IsGantry

目的

軸が「ガントリー」状態にあるかどうかを照会します。

構文

```
int HIMC_IsGantry(
        int ctrl_id,
        int axis_id,
        int *p_is_gantry
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_gantry [out]	軸のガントリー ステータスを受け取るバッファへのポインタ。 軸が「ガントリー」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGantry
LabVIEW	HIMC Is Gantry.vi
Python	IsGantry

3.4.6 HIMC_IsGrouped

目的

軸が軸グループにグループ化されているかどうかを照会します。

構文

```
int HIMC_IsGrouped(
        int ctrl_id,
        int axis_id,
        int *p_is_grouped
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_grouped [out]	軸のグループ化されたステータスを受け取るバッファへのポインタ。 軸が「グループ化」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGrouped
LabVIEW	HIMC Is Grouped.vi
Python	IsGrouped

3.4.7 HIMC_IsSync

目的

軸が「同期」状態にあるかどうかを照会します。

構文

```
int HIMC_IsSync(
        int ctrl_id,
        int axis_id,
        int *p_is_sync
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_sync [out] 軸の同期ステータスを受け取るバッファへのポインタ。
軸が「同期」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsSync
LabVIEW	HIMC Is Sync.vi
Python	IsSync

3.4.8 HIMC_IsHWLL

目的

軸がハードウェアの左限界に達したかどうかを照会します。

構文

```
int HIMC_IsHWLL(
        int ctrl_id,
        int axis_id,
        int *p_is_hwll
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_hwll [out] 軸の HWLL ステータスを受信するバッファへのポインタ。
軸が「HWLL」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはオブジェクト 0x60FD (デジタル入力) を PDO として設定し、ビット 0 を左制限入力として指定する必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsHWLL
LabVIEW	HIMC Is HWLL.vi
Python	IsHWLL

3.4.9 HIMC_IsHWRL

目的

軸がハードウェアの右制限に達したかどうかを照会します。

構文

```
int HIMC_IsHWRL(
        int ctrl_id,
        int axis_id,
        int *p_is_hwrl
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_hwrl [out] 軸の HWRL ステータスを受け取るバッファへのポインタ。
軸が「HWRL」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはオブジェクト 0x60FD (デジタル入力) を PDO として設定し、ビット 1 を右制限入力として指定する必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsHWRL
LabVIEW	HIMC Is HWRL.vi
Python	IsHWRL

3.4.10 HIMC_IsSWLL

目的

軸がソフトウェアの左限界に達したかどうかを照会します。

構文

```
int HIMC_IsSWLL(
        int ctrl_id,
        int axis_id,
        int *p_is_swll
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_swll [out] 軸の SWLL ステータスを受信するバッファへのポインタ。
軸が「SWLL」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsSWLL
LabVIEW	HIMC Is SWLL.vi
Python	IsSWLL

3.4.11 HIMC_IsSWRL

目的

軸がソフトウェア権利制限に達したかどうかを照会します。

構文

```
int HIMC_IsSWRL(
        int ctrl_id,
        int axis_id,
        int *p_is_swrl
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_swrl [out] 軸の SWRL ステータスを受信するバッファへのポインタ。
軸が「SWRL」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsSWRL
LabVIEW	HIMC Is SWRL.vi
Python	IsSWRL

3.4.12 HIMC_IsDriveErr

目的

軸がドライバーアラームをトリガーするかどうかを照会します。

構文

```
int HIMC_IsDriveErr(
        int ctrl_id,
        int axis_id,
        int *p_is_driveerr
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_driveerr [out]	軸の DriveErr ステータスを受け取るバッファへのポインタ。 軸が「DriveErr」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはオブジェクト 0x6041 (ステータス ワード) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsDriveErr
LabVIEW	HIMC Is Drive Err.vi
Python	IsDriveErr

3.4.13 HIMC_IsPosErr

目的

軸の位置誤差が保護限度を超えているかどうかを照会します。

構文

```
int HIMC_IsPosErr(
        int ctrl_id,
        int axis_id,
        int *p_is_poserr
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_poserr [out]	軸の PosErr ステータスを受け取るバッファへのポインタ。 軸が「PosErr」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

エラー保護制限は、コントローラー内の軸の位置エラー許容範囲を示します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsPosErr
LabVIEW	HIMC Is Pos Err.vi
Python	IsPosErr

3.4.14 HIMC_IsCompActive

目的

補正機能がアクティブかどうかを照会します。

構文

```
int HIMC_IsCompActive(
    int ctrl_id,
    int axis_id,
    int *p_is_compactive
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_compactive [out]	軸の補正アクティブ ステータスを受け取るバッファへのポインタ。 軸が「CompActive」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsCompActive
LabVIEW	HIMC Is Comp Active.vi
Python	IsCompActive

3.4.15 HIMC_IsAcc

目的

軸が加速しているかどうかを照会します。

構文

```
int HIMC_IsAcc(
        int ctrl_id,
        int axis_id,
        int *p_is_acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_acc [out] 軸の加速ステータスを受け取るバッファへのポインタ。
軸が「Acc」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsAcc
LabVIEW	HIMC Is Acc.vi
Python	IsAcc

4. 同期モーション機能

4.1 概要

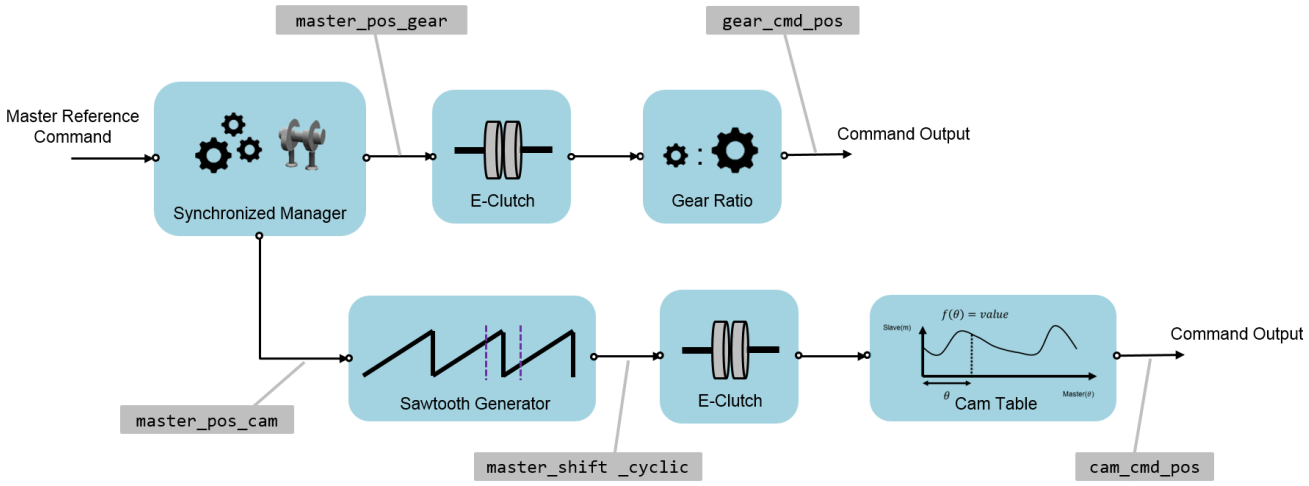


図 4.1.1

ユーザーは、ある軸と別の軸間の同期動作を定義できます。マスター軸（リーディング軸）が最初に位置コマンドを生成し、スレーブ軸は動作設定に基づいてマスター軸を参照します。マスターとスレーブの関係が一定である場合、動作は電子ギアリングです。一方、スレーブ軸がパターンに従う必要がある場合、動作は電子カムです。図 4.1.2 では、軸 0 がマスター軸として機能し、リーディング軸は軸 1、2、3、4 です。軸 1、2、3 は電子ギアリングを採用し、軸 4 は電子カムを採用しています。

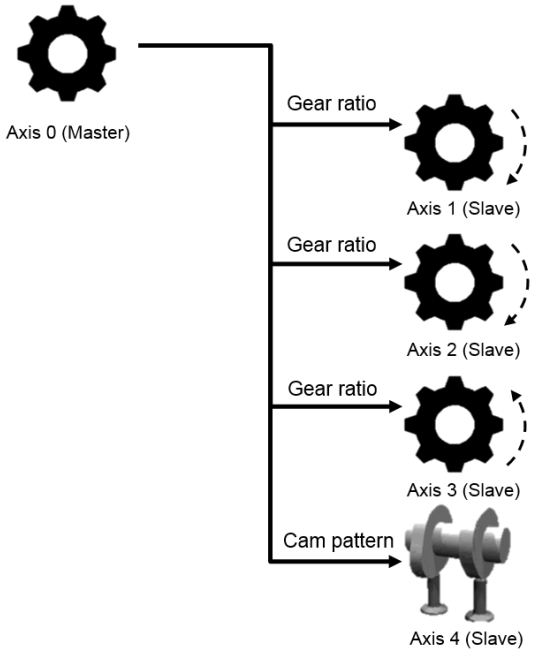


図 4.1.2

4.1.1 同期動作変数

一般的な同期動作変数は表 4.1.1.1 に示されています。ユーザーは iA Studio のスコープマネージャから必要な変数を選択できます（「iA Studio ユーザーガイド」のセクション 4.8 を参照）。

表 4.1.1.1

名称	変数	単位	説明
Raw Master Position	master_pos_gear	mm または deg	マスター軸の位置コマンド
Gear Command Position	gear_cmd_pos	mm または deg	スレーブ軸の位置コマンド出力
Gear Ratio	gear_ratio	mm または deg	ギア比

4.2 HIMC_EnableGear

目的

2 つの軸をマスターとスレーブの関係で結合します。

構文

```
int HIMC_EnableGear(
    int ctrl_id,
    int axis_master_id,
    int axis_slv_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_master_id [in] マスター軸インデックス

axis_slv_id [in] スレーブ軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能は、両方の軸が有効になっている場合にのみ適用されます。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableGear
LabVIEW	HIMC Enable Gear.vi
Python	EnableGear

4.3 HIMC_DisableGear

目的

2 つの軸をマスター/スレーブ関係から切り離し、 2 つの独立した軸にします。

構文

```
int HIMC_DisableGear(
    int ctrl_id,
    int axis_slv_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_slv_id [in] スレーブ軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableGear
LabVIEW	HIMC Disable Gear.vi
Python	DisableGear

4.4 HIMC_GearIn

目的

スレーブ軸の状態を解除から接続に変更します。

構文

```
int HIMC_GearIn(
    int ctrl_id,
    int axis_master_id,
    int axis_slv_id,
    double gear_ratio
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_master_id [in] マスター軸インデックス

axis_slv_id [in] スレーブ軸インデックス

gear_ratio [in] ギア比の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能は、両方の軸が有効になっている場合にのみ適用されます。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GearIn
LabVIEW	HIMC Gear In.vi
Python	GearIn

4.5 HIMC_GearOut

目的

スレーブ軸の状態を接続から解除に変更します。

構文

```
int HIMC_GearOut(
    int ctrl_id,
    int axis_slv_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_slv_id [in] スレーブ軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GearOut
LabVIEW	HIMC Gear Out.vi
Python	GearOut

4.6 HIMC_GetGearRatio

目的

スレーブ軸のギア比を取得します。

構文

```
int HIMC_GetGearRatio(
    int ctrl_id,
    int axis_slv_id,
    double *p_ratio
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_slv_id [in]	スレーブ軸インデックス
p_ratio [out]	スレーブ軸のギア比を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGearRatio
LabVIEW	HIMC Get Gear Ratio.vi
Python	GetGearRatio

4.7 HIMC_IsInGear

目的

スレーブ軸が「接続」状態にあるかどうかを照会します。

構文

```
int HIMC_IsInGear(
        int ctrl_id,
        int axis_id,
        int *p_is_in_gear
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_in_gear [out]	スレーブ軸の接続状態を受信するためのバッファへのポインタ。 スレーブ軸が「InGear」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsInGear
LabVIEW	HIMC Is In Gear.vi
Python	IsInGear

4.8 HIMC_IsGearMaster

目的

軸がマスター軸であるかどうかを照会します。

構文

```
int HIMC_IsGearMaster(
        int ctrl_id,
        int axis_id,
        int *p_is_gear_master
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_gear_master [out]	軸がマスター軸であるかどうかのステータスを受け取るバッファへのポインタ。 軸が「GearMaster」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGearMaster
LabVIEW	HIMC Is Gear Master.vi
Python	IsGearMaster

4.9 HIMC_IsGearSlave

目的

軸がスレーブ軸であるかどうかを照会します。

構文

```
int HIMC_IsGearSlave(
        int ctrl_id,
        int axis_id,
        int *p_is_gear_slv
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_gear_slv [out]	軸がスレーブ軸であるかどうかのステータスを受け取るバッファへのポインタ。 軸が「GearSlave」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGearSlave
LabVIEW	HIMC Is Gear Slave.vi
Python	IsGearSlave

(このページはブランクになっています)

5. ガントリー機能

5.1 概要

ガントリー構成は、図 5.1.1 に示すように、右側（RHS）軸と左側（LHS）軸のペアを、仮想的な直線軸とヨー軸のペアに変換します。ガントリー構成を確立した後、ユーザーは RHS 軸に直線軸方向のコマンドを与えて RHS 軸と LHS 軸を同じ方向に駆動し、LHS 軸にヨー軸方向の回転動作コマンドを与えることができます。

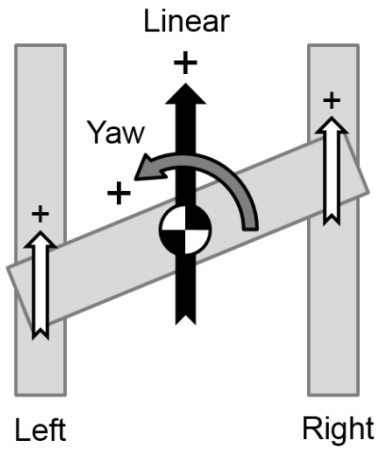


図 5.1.1

ガントリー構成では、直線軸とヨー軸の位置フィードバックは次のように定義されます。

$$Pos_{linear} = \frac{Pos_{RHS} + Pos_{LHS}}{2}; \qquad Pos_{yaw} = \frac{Pos_{RHS} - Pos_{LHS}}{2}$$

Pos_{linear}	：直線軸の位置フィードバック	Pos_{yaw}	：ヨー軸の位置フィードバック
Pos_{RHS}	：RHS 軸の位置フィードバック	Pos_{LHS}	：LHS 軸の位置フィードバック

図 5.1.2 は直線軸、ヨー軸、RHS 軸、LHS 軸の位置フィードバックの概略図です。

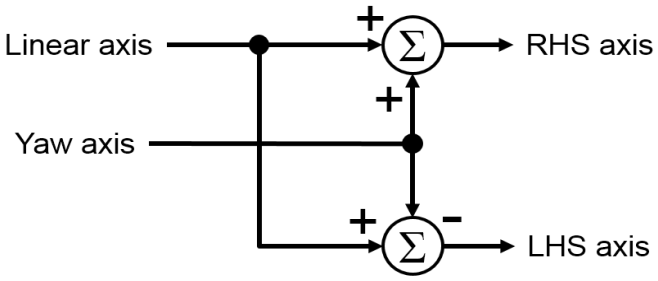


図 5.1.2

5.2 HIMC_EnableGantryPair

目的

ガントリーペアをセットアップします。

構文

```
int HIMC_EnableGantryPair(
    int ctrl_id,
    int lhs_axis_id,
    int rhs_axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

lhs_axis_id [in] 左側の軸インデックス

rhs_axis_id [in] 右側の軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能は、両方の軸が無効になっている場合にのみ適用されます。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableGantryPair
LabVIEW	HIMC Enable Gantry Pair.vi
Python	EnableGantryPair

5.3 HIMC_DisableGantryPair

目的
ガントリーペアを分割します。

構文

```
int HIMC_DisableGantryPair(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] ガントリーペアのいずれかの軸インデックス

戻りの値
関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考
この機能は、両方の軸が無効になっている場合にのみ適用されます。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableGantryPair
LabVIEW	HIMC Disable Gantry Pair.vi
Python	DisableGantryPair

5.4 HIMC_GetGantryPairID

目的

任意のガントリー軸のガントリー ペア ID を取得します。

構文

```
int HIMC_GetGantryPairID(
    int ctrl_id,
    int axis_id,
    int *p_id
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	ガントリーペアのいずれかの軸インデックス。
p_id [out]	ガントリー ペア ID を受け取るバッファへのポインタ。 入力軸がガントリー軸でない場合は -1 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGantryPairID
LabVIEW	HIMC Get Gantry Pair ID.vi
Python	GetGantryPairID

5.5 HIMC_IsGantryPair

目的

2 つの軸がガントリー ペアであるかどうかを照会します。

構文

```
int HIMC_IsGantryPair(
        int ctrl_id,
        int axis_id_1,
        int axis_id_2,
        int *p_is_gantry_pair
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id_1 [in]	軸インデックス 1
axis_id_2 [in]	軸インデックス 2
p_is_gantry_pair [out]	2 軸がガントリーペアであるかどうかの状態を受け取るバッファへのポインタ。2 軸が「GantryPair」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGantryPair
LabVIEW	HIMC Is Gantry Pair.vi
Python	IsGantryPair

6. グループ関数

6.1 概要

HIMC は、LineAbs2D / 3D、LineRel2D / 3D、Arc2D、Circle2D など、多軸直線・円弧同時補間機能の軸グループ動作コマンドを提供します。軸動作コマンドと比較して、軸グループ動作コマンドはグループ内の各軸の同期を保証します。各軸動作の開始時間と停止時間は同じで、コントローラーはユーザーが指定した参照速度に基づいて各軸の動作速度を調整します。HIMC コントローラーの基本機能は、軸グループ動作コマンドで最大 4 軸をサポートします（型番：MC-XX-XX-XX-00）。軸グループ動作機能で 5 軸（またはそれ以上）の同時加工が必要な場合は、HIWIN Mikrosystem または最寄りの販売代理店にお問い合わせください。

図 6.1.1 は、HIMC 軸グループ動作コマンドのパラメーターフロー図です。各軸の位置フィードバックが順運動学計算を経ることで、機械座標系における軸グループの位置フィードバック（直交位置フィードバック）が得られます。コントローラーは、ユーザーが指定した目標コマンドに基づいて、軸グループの動作プロファイル（図 6.1.2 参照）に基づいて空間補間コマンド（直交位置コマンド）を計画し、逆運動学を用いて各軸の対応する位置コマンドを計算します。

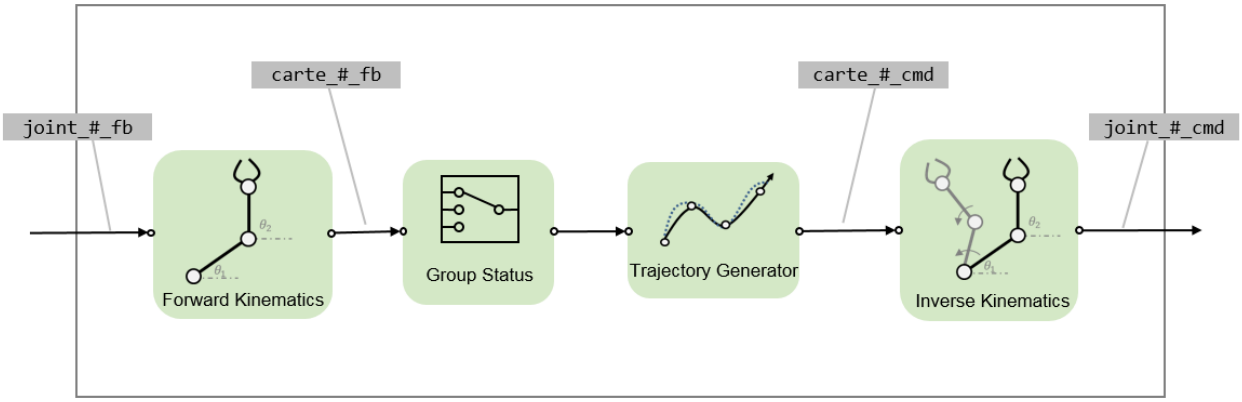


図 6.1.1

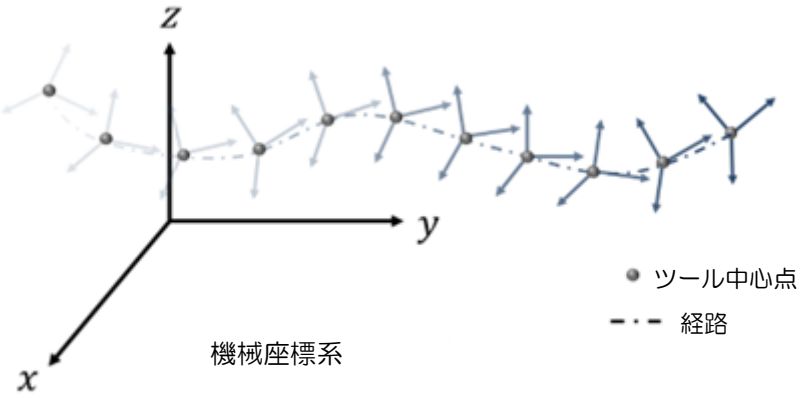


図 6.1.2

軸グループ移動コマンドでは、HIMC は各セグメントの空間内での移動距離を計算します。軸移動コマンドとは異なり、速度計画は軸グループの空間内での移動方向に沿って計画され、移動方向は移動コマンドの方向に基づいて変化します。

軸グループ動作コマンドは、軸動作コマンド（第 3 章参照）に類似しています。図 6.1.3 に示すように、S カーブ速度計画も採用しています。空間における軸グループ動作は、並進と回転の 2 つの部分で構成されます。並進コマンドは XYZ の位置コマンドで構成され、回転コマンドは ABC の回転コマンドで構成されます。軸グループでは、プロファイルジェネレータの最大速度、最大加速度、最大減速度、スムージング時間など、並進と回転の速度計画パラメーターを設定できます。

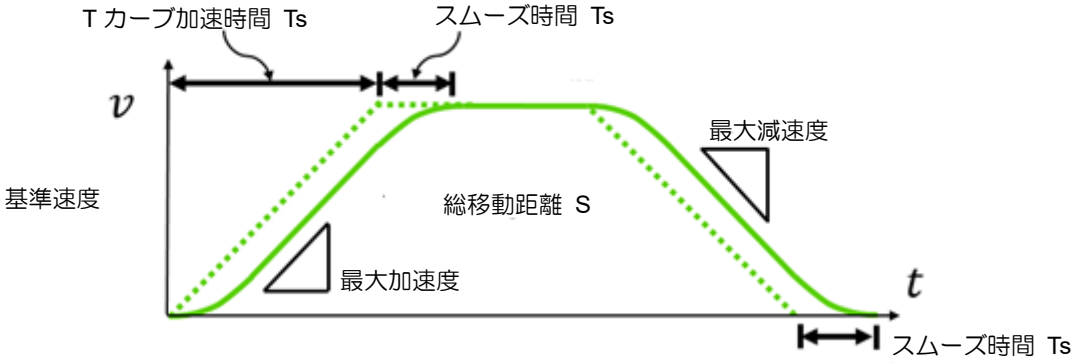


図 6.1.3

図 6.1.4 に示すように、各軸グループの移動コマンドは 1 つのセグメントとして表示されます。移動中、HIMC は各セグメントの移動コマンドと回転コマンド、およびユーザーが設定した速度計画パラメーターに基づいて、移動コマンドと回転コマンドの移動時間を計算します。移動時間の長い方の速度計画パラメーターは、軸グループの送り速度として表示されます。移動時間の短い方については、送り速度コマンドに割り当てられた移動コマンドに従って移動します。

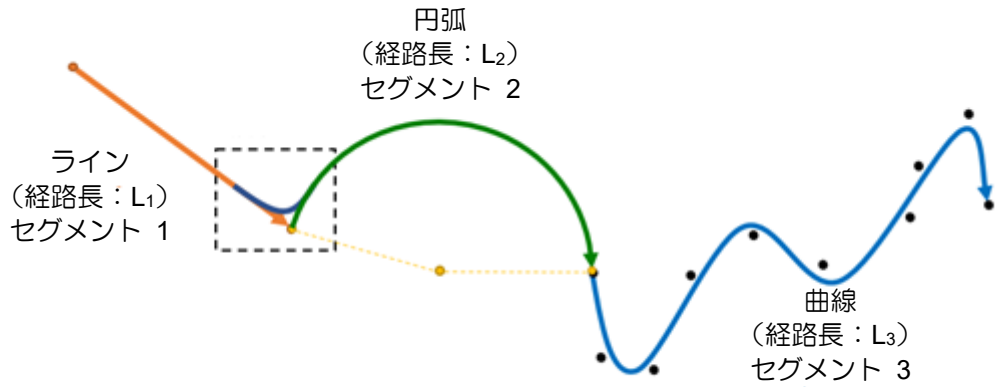


図 6.1.4

HIMC には、軸グループコマンド用のバッファが組み込まれています。各動作コマンドのセグメントはこのバッファに配置され、最大 512 個の動作コマンドを同時に受け入れることができます。この容量制限を超える動作コマンドはコントローラーによって破棄され、エラーメッセージが表示されます。セグメントの動作コマンド間では、ユーザーが設定したバッファモードと遷移モードに基づいて速度と経路が計画されます。計画された速度と経路は、選択したモードによって変更される場合があります。図 6.1.4 を例にとると、バッファモードを使用して各セグメントの速度の引き継ぎを設定する場合、この軸動作の合計長さは「S = L_1（直線）+ L_2（円弧）+ L_3（曲線）」になります。詳細については、セクション 6.1.4 および 6.1.5 を参照してください。

軸グループの動作ステータスは軸の動作ステータスに似ており、「移動中」と「位置決め中」に分けられます。動作中には、図 3.1.4 に示すように、以下の 3 つのフェーズがあります。

1. 軸グループが移動しており、位置がずれています。
2. 軸グループは移動していませんが、位置がずれています。
3. 軸グループが移動しておらず、所定の位置にありません。

軸移動コマンドが位置ウィンドウと位置ウィンドウ時間を用いて軸がインポジション状態にあるかどうかをチェックするのに対し、軸グループ移動コマンドはグループ内のすべての軸がインポジション状態にあるかどうかをチェックします。つまり、軸グループがインポジション状態にある場合、グループ内のすべての軸は「インポジション」状態にあります。

6.1.1 グループ変数

軸グループ変数は、モーションコマンド変数、プロファイルジェネレータ変数、ステータス変数の 3 つのカテゴリに分類されます。ユーザーは iA Studio のスコープマネージャから必要な変数を選択できます（「iA Studio ユーザーガイド」のセクション 4.8 を参照）。詳細な説明は表 6.1.1.1 から表 6.1.1.5 に示されています。

表 6.1.1.1 軸グループの動作コマンド変数

名称	変数	単位	説明
Cartesian Position Command	carte_pose_cmd	mm または deg	機械座標系（MCS）における軸グループの空間位置コマンド。[X Y Z A B C]の値を含む配列です。
Cartesian Velocity Command	carte_vel_cmd	mm/s または deg/s	機械座標系（MCS）における軸グループの空間速度コマンド。[X Y Z A B C]の値を含む配列です。
Cartesian Position Feedback	carte_pose_fb	mm または deg	機械座標系（MCS）における軸グループの空間位置フィードバック。[X Y Z A B C]の値を含む配列です。
Axis Position Command	joint_pos_cmd	mm または deg	軸座標系（ACS）における軸グループの軸位置コマンド。配列です。
Axis Velocity Command	joint_vel_cmd	mm/s または deg/s	軸座標系（ACS）における軸グループの軸速度コマンド。配列です。
Axis Acceleration Command	joint_acc_cmd	mm/s^2 または deg/s^2	軸座標系（ACS）における軸グループの軸加速コマンド。配列です。
Axis Position Feedback	joint_pos_fb	mm または deg	軸座標系（ACS）における軸グループの軸位置フィードバック。配列です。
Cartesian Position Error	carte_pose_err	mm または deg	機械座標系（MCS）における軸グループの空間位置誤差。[X Y Z A B C]の値を含む配列です。
Reference Group Position	grp_pg_pos	mm または deg	軸グループの参照位置。軸グループコマンドの動作プロファイルに基づいてプロファイルジェネレータから生成される位置設定値です。
Reference Group Velocity	grp_pg_vel	mm/s または deg/s	軸グループの基準速度。軸グループコマンドの動作プロファイルに基づいてプロファイルジェネレータから生成される速度設定値です。
Reference Group Acceleration	grp_pg_acc	mm/s^2 または deg/s^2	軸グループの基準加速度。軸グループコマンドの動作プロファイルに基づいてプロファイルジェネレータから生成される加速度設定値です。

表 6.1.1.2 軸グループのプロファイルジェネレータ変数

名称	変数	単位	説明
Group Max. Linear Profile Velocity	grp_lin_vel	mm/s	軸グループの最大線形プロファイル速度。 必ずしも到達するとは限りません。
Group Max. Linear Profile Acceleration	grp_lin_acc	mm/s^2	軸グループの最大線形プロファイル加速度。 必ずしも到達するとは限りません。
Group Max. Linear Profile Deceleration	grp_lin_dec	mm/s^2	軸グループの最大線形プロファイル減速。 必ずしも到達するとは限りません。
Group Linear Smooth Time	grp_lin_sf	ms	軸グループの線形プロファイルのスムーズ時間。入力範囲は 0～500 です。値を大きくすると動作中の機械振動を軽減できますが、動作時間全体に影響します。
Group Max. Angular Profile Velocity	grp_ang_vel	deg/s	軸グループの最大角プロファイル速度。 必ずしも到達するとは限りません。
Group Max. Angular Profile Acceleration	grp_ang_acc	deg/s^2	軸グループの最大角プロファイル加速度。 必ずしも到達するとは限りません。
Group Max. Angular Profile Deceleration	grp_ang_dec	deg/s^2	軸グループの最大角度プロファイル減速。 必ずしも到達するとは限りません。
Group Angular Smooth Time	grp_ang_sf	ms	軸グループの角度プロファイルの平滑化時間。入力範囲は 0～500 です。値を大きくすると動作中の機械振動を軽減できますが、動作時間全体に影響します。

表 6.1.1.3 軸グループの状態変数

名称	変数	単位	説明
Group Fault Status	grp_fault_status	N/A	軸グループのエラー状態。ビットの定義については表 6.1.1.4 を参照してください。
Group Motion Status	grp_motion_status	N/A	軸グループの動作ステータス。ビットの定義については表 6.1.1.5 を参照してください。

表 6.1.1.4 軸グループエラーステータスのビット定義

Bit	名称	説明	デフォルトの応答
0	Error Stop	軸グループが「エラー停止」状態	N/A
1	Axis Fault	スレーブドライバー障害	コントローラーが軸を無効にしているため、軸グループが同期されていません。
2	Software Limit	軸ソフトウェア制限がトリガーされました	コントローラーは動作を停止します。軸グループは同期していません。

表 6.1.1.5 軸グループ動作ステータスのビット定義

Bit	名称	説明	備考
0	Enabled	軸グループが有効になっています。	N/A
1	Moving	軸グループが移動しています。	N/A
2	In Position	軸グループは所定の位置にあります。	軸グループ内のすべての軸が位置にあります。
3	Input Shape	軸グループの入力形状フィルターを有効にします。	セクション 13.1 を参照してください。

6.1.2 座標系

表 6.1.2.1 は、軸座標系（ACS）、マシン座標系（MCS）、製品座標系（PCS）、ワークピース座標系（WCS）、グローバル座標系、座標オフセットを含む HIMC の座標系の定義と説明を示しています。

表 6.1.2.1

HMPL 定義	説明
CS_ACS	軸座標系。 個々のモーターの動きに関係します。
CS_MCS	機械座標系 （「ワールド座標系」または「ベース座標系」と呼ばれることもあります） 機械上に固定された原点を持つ座標系であり、運動学変換を介して ACS にリンクされています（セクション 6.1.3 を参照）。空間内の位置と方向を示す合計 6 次元（並進 3 次元、回転 3 次元）を持ちます。
CS_PCS	製品座標系（CNC プログラムでは「プログラム座標系」）。 製品またはワークピースに付属し、座標変換パラメーターを設定できます。
CS_WCS# (#=1~15)	ワークピース座標系。 ワークピースの原点を設定するために使用され、最大 15 個の独立したワークピース座標系を提供します。デフォルトではオフセットなしです。製品座標系に依存するため、座標変換パラメーターを設定できます。
CS_GLOBAL	グローバル座標系。 グローバルゼロ点を設定するために使用され、各軸グループのグローバルな空間関係を確立できます。 まだサポートされていません。
CS_OFFSET	座標オフセット。 一時的な原点を設定するために使用されます。デフォルトではオフセットなし、つまりオフセットの座標原点は機械の座標原点となります。これは製品座標系に依存するため、座標変換パラメーターを設定できます。

図 6.1.2.1 は、2 つの回転軸を持つ SCARA ロボットの ACS、MCS、PCS の関係の例を示しています。ACS と MCS は、順方向運動学と逆方向運動学によって変換されます（6.1.3 節参照）。一方、MCS と PCS の間には座標変換関係があります。座標系上の位置は、座標の移動と回転によって得られます。

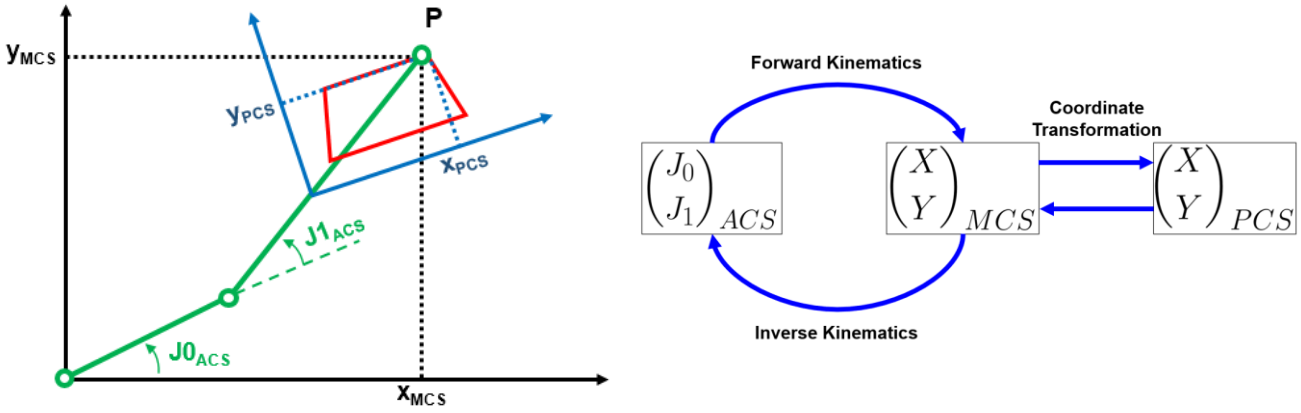


図 6.1.2.1

MCS を PCS に変換する際、HIMC は要件に応じて機械のワーク座標（WCS1～15）と座標オフセット（OFFSET）を設定できます。座標系の設定には、3 つの並進自由度（X、Y、Z）と 3 つの回転自由度（A、B、C）を使用して空間内の姿勢を示します。

HIMC は、固定角度における「ロール-ピッチ-ヨー」回転規則を採用しています。図 6.1.2.2 に示すように、X 軸に沿った回転の自由度はロール（角度 A）、Y 軸に沿った回転の自由度はピッチ（角度 B）、Z 軸に沿った回転の自由度はヨー（角度 C）です。この回転規則は、図 6.1.2.3 に示すように、空間における物体の向きを示すために Tait-Bryan 角の ZYX の順序を使用するのと同じです。

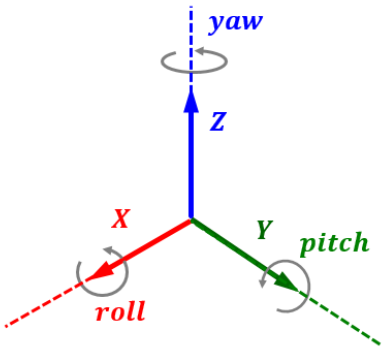


図 6.1.2.2

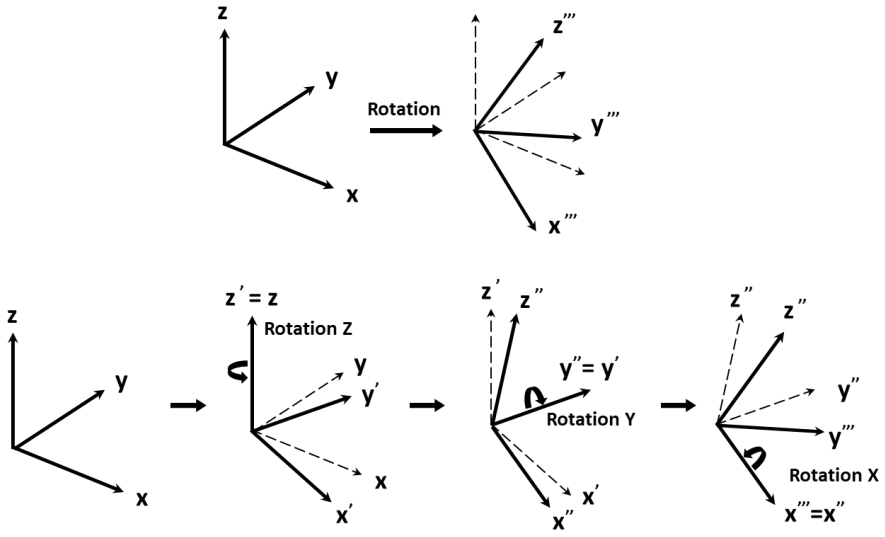


図 6.1.2.3

座標オフセットがない場合、各 WCS と MCS の関係は図 6.1.2.4 に示されます。

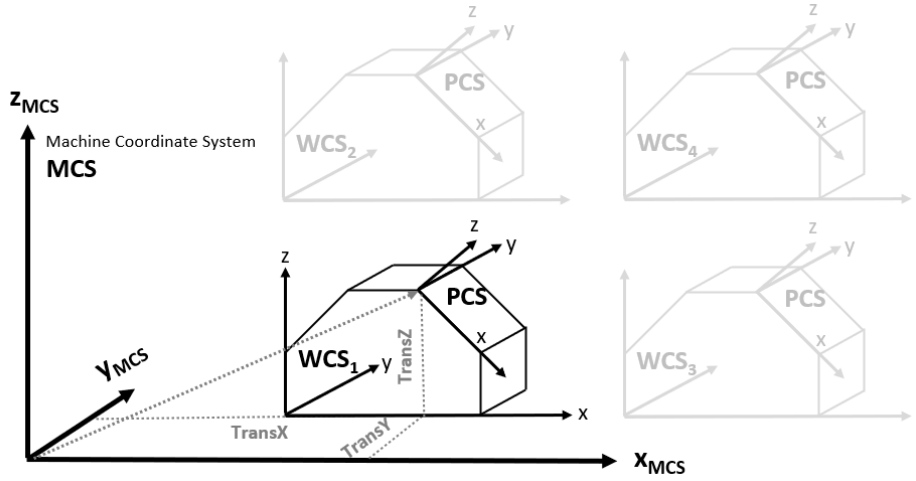


図 6.1.2.4

座標オフセットが追加された場合、WCS と MCS の関係は図 6.1.2.5 のようになります。座標オフセットの変換が追加されます。

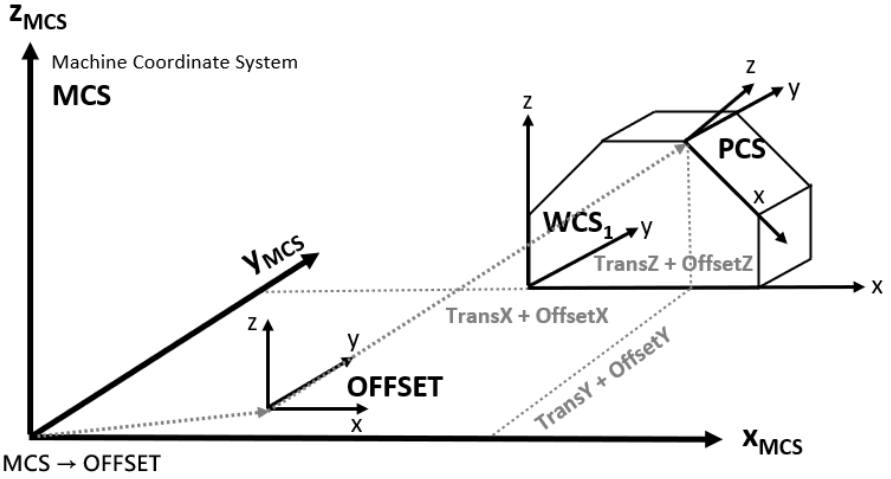


図 6.1.2.5

上記の機能に基づいて、ユーザーは HIMC で座標変換のパラメーターを定義し、座標系間の変換関係を確立することができます。図 6.1.2.6 は座標系間の関係を示しています。分かりやすくするために、この図では XY 平面の座標のみを示しています。実際のアプリケーションでは、座標変換に 6 つの自由度（X、Y、Z、A、B、C）を設定できます。

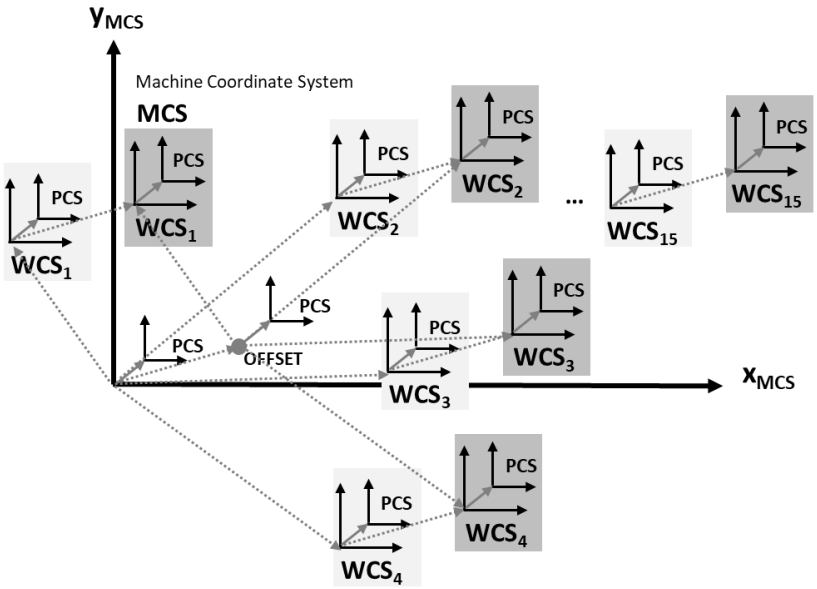


図 6.1.2.6

6.1.3 運動学

運動学は主に ACS（軸座標系）と MCS（機械座標系）間の変換を扱います。順運動学は、ACS における各軸の位置フィードバックから MCS の座標位置への計算です。一方、逆運動学は、MCS の座標位置から ACS における各軸の位置への計算です。表 6.1.3.1 は、HIMC が提供する運動学構成の定義を示しています。

表 6.1.3.1

ID	名称	説明
1	Cartesian	協調動作グループ内の各軸を、それぞれ直交座標系の X、Y、Z、A、B、C 軸にマッピングします。関節空間における軸の最大許容数は 6 です。（軸グループのデフォルト）
2	SCARA	（サポートされていません）
3	WAFER	（サポートされていません）
4	6-Axis Articulated Robot	（サポートされていません）

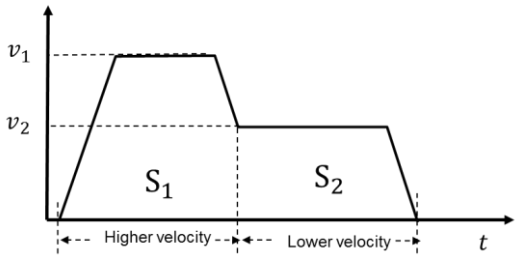
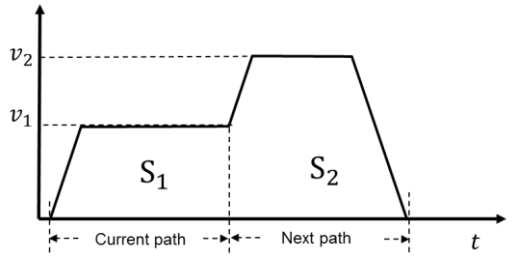
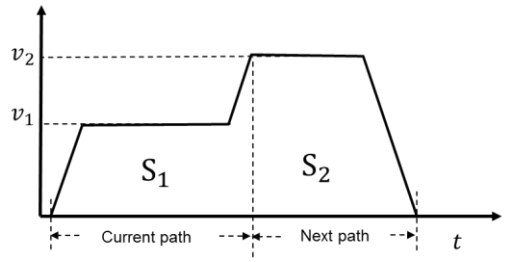
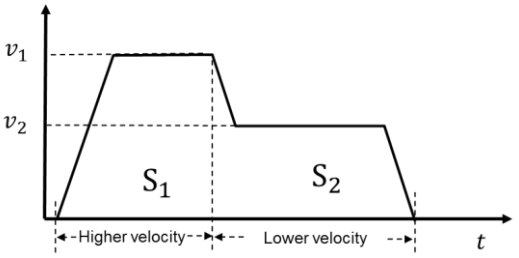
6.1.4 バッファモード

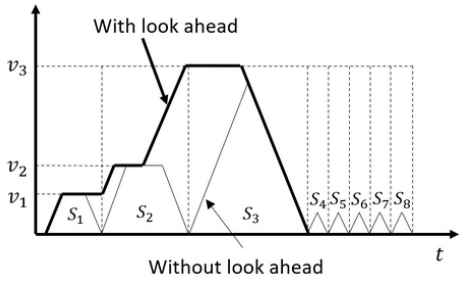
バッファモードは、隣接するパスの終点における速度プロファイルを決定します。ユーザーはこの設定を使用して、隣接する 2 つのパスのプロファイル速度を計画できます。表 6.1.4.1 に、HIMC が提供するバッファモードの定義を示します。

注: 現在のパスが位置合わせされる前に軸グループが次のパスのコマンドを受信しない場合、バッファ モード機能は無視されます。

表 6.1.4.1

API 定義	説明
kBM_Aborting	進行中の動作を中止し、すぐに次の動作を開始します。
kBM_Buffered	現在のパスが完了したら次のパスを開始します。

kBM_BlendingLow	速度は、2 つのパスのうちの低い方の速度とブレンドされます。(ブレンディング) 
kBM_BlendingPrevious	速度は現在のパスの速度とブレンドされます。（ブレンディング） 
kBM_BlendingNext	速度は次のパスの速度とブレンドされます。(ブレンディング) 
kBM_BlendingHigh	速度は 2 つのパスのうち、より高い速度のパスとブレンドされます。(ブレンディング) 

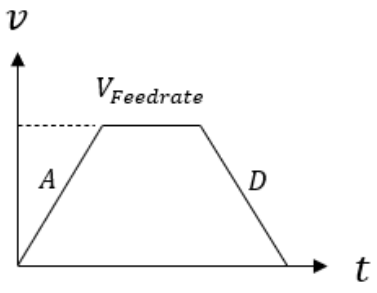
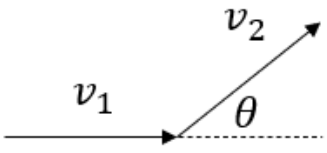
kBM_LookAhead	コントローラーは、先読み機能によって計算された移動距離と指定された制約に基づいて速度を決定します。 

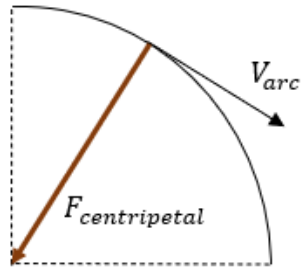
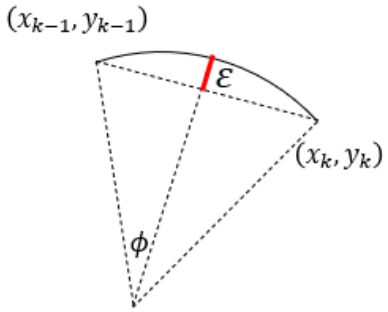
先読みモーション

先読み機能は、軸グループコマンドバッファ内のコントローラーの動作コマンドを先読みすることで、動作コマンドの移動距離を計算します。さらに、機構の加速限界に応じて、動作パス上の折り返し点における加減速動作を事前に実行します。表 6.1.4.2 は、動作中に先読み動作機能によって計算される速度パラメーターを示しています。

(a) 送り速度 ($V_{Feedrate}$): 軸グループ コマンドの速度条件。
(b) コーナー速度(V_{corner})：ユーザーが設定したコーナー加速度(A_{corner})の制限値と、動作方向間の角度に基づいて計算されます。コーナー加速度のデフォルトの制限値は 5000 mm/s²です。
(c) アーク速度(V_{arc})：ユーザーが設定したアーク加速度(A_{arc})の制限に基づいて計算されます。アーク加速度のデフォルト値は 100 mm/s²です。
(d) 最大コードエラー速度(V_{chord})：ユーザーが設定したコードエラー（ε）の条件に基づいて計算されます。コードエラーのデフォルト値（0）は使用されません。

表 6.1.4.2

	
(a) 送り速度 $V_{Feedrate}$	(b) コーナー速度 $V_{corner} = \frac{A_{corner} * \Delta T}{1 - cos\theta}$

	
(c) アーク速度 $V_{arc} = \sqrt{A_{arc} * r}$	(d) 最大弦誤差速度 $V_{chord} = \frac{2 * r * \phi}{\Delta T}$

先読みモーションを使用する場合、コマンドバッファ内の命令数が増えるほど、コントローラーのモーションコアの計算時間が長くなり、MCK Overload エラーメッセージが表示されることがあります。この場合、通信周期を長くするか、バッファサイズを小さくすることでエラーを解消できます。

6.1.5 遷移モード

遷移モードは、表 6.1.5.1 に示すように、隣接するパス間の遷移曲線の種類を決定します。この設定により、コントローラーはユーザーが設定したモードとパラメーターに基づいて、前者と後者の動作コマンド間のコーナースムージングを計算します。これにより、スムーズパスの開始点と終了点が決定され、バッファモードと動作プロファイルに基づいて結果が生成されます（表 6.1.5.2 を参照）。この計画方法は、元の動作プロファイルに影響を与えることに注意してください。

表 6.1.5.1

API 定義	説明
KTM_None	なし: 遷移曲線を挿入しません。(デフォルトモード)
kTM_StartVelocity (Not supported)	遷移パスの開始速度として速度パラメーターを使用します。この機能はまだサポートされていません。
kTM_Velocity	速度パラメーターを遷移パスの一定速度として使用します。
kTM_CornerDistance	距離パラメーターを使用して、スムーズな遷移パスの開始点からコーナーまでの距離を設定します。
kTM_MaxCornerDeviation	偏差パラメーターを使用して、スムーズな遷移パスと元のパスの間の最大偏差を決定します。
kTM_MaxCornerCurvature	曲率パラメーターを使用して、遷移パスの最大曲率値を決定します。

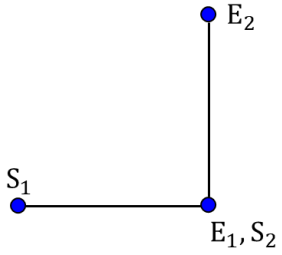
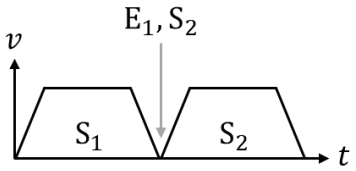
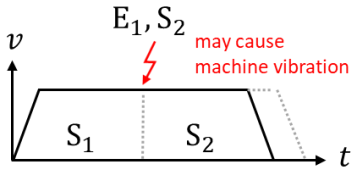
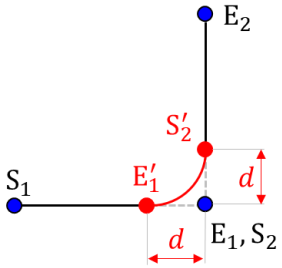
注：

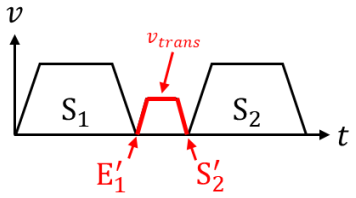
(1) 遷移モードによって計算されたトリム範囲がいずれかのセグメントの長さを超える場合、セグメント間の遷移モードの機能は無視されます。

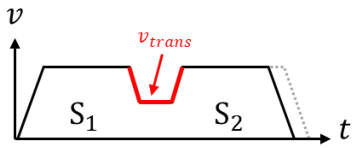
(2) 遷移モードのパラメーターの設定については、6.3.26 項を参照のこと。

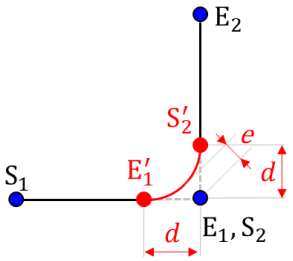
(3) kTM_Velocity と kTM_CornerDistance の動作は同じです。

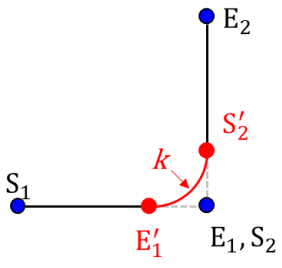
表 6.1.5.2 バッファモードと遷移モード

<table>
<tr><th rowspan="2">遷移
モード</th><th rowspan="2">TCP の軌跡</th><th colspan="2">TCP の速度</th></tr>
<tr><th>kBM_バッファリング</th><th>その他</th></tr>
<tr><td>kTM_None</td><td> 
</td><td> 
</td><td> 
</td></tr>
<tr><td>kTM_StartVelocity
kTM_CornerDistance</td><td rowspan="3"> 

d = trans_dis</td><td rowspan="3"> 

v_{trans} = trans_vel</td><td rowspan="3"> 

v_{trans} = trans_vel</td></tr>
<tr><td>kTM_MaxCornerDeviation</td><td> 

d = trans_dis
e = trans_dev</td></tr>
<tr><td>kTM_MaxCornerCurvature</td><td> 

k = trans_curv</td></tr>
</table>

6.2 グループモーション制御

6.2.1 HIMC_EnableGroup

目的
軸グループを有効にします。

構文

```
int HIMC_EnableGroup(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値
関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考
この関数を実行する前に、グループ内のすべての軸を有効にする必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableGroup
LabVIEW	HIMC Enable Group.vi
Python	EnableGroup

6.2.2 HIMC_DisableGroup

目的

軸グループを無効にします。

構文

```
int HIMC_DisableGroup(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableGroup
LabVIEW	HIMC Disable Group.vi
Python	DisableGroup

6.2.3 HIMC_ResetGroup

目的

軸グループの状態を「グループエラー停止」から「グループスタンバイ」に変更します。

構文

```
int HIMC_ResetGroup(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in]　HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in]　軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能は、すべてのエラーがクリアされた後にのみ使用できます。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ResetGroup
LabVIEW	HIMC Reset Group.vi
Python	ResetGroup

6.2.4 HIMC_StopGroup

目的

軸グループの動きを停止します。

構文

```
int HIMC_StopGroup(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸グループのモーションキューがクリアされます。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_StopGroup
LabVIEW	HIMC Stop Group.vi
Python	StopGroup

6.2.5 HIMC_HaltGroup

目的

軸グループの動きを停止するには、その速度を 0 に設定します。

構文

```
int HIMC_HaltGroup(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸グループが所定の位置にない場合は、移動し続けます。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_HaltGroup
LabVIEW	HIMC Halt Group.vi
Python	HaltGroup

6.2.6 HIMC_ResumeGroup

目的

軸グループの動作を「停止」状態から再開します。

構文

```
int HIMC_ResumeGroup(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ResumeGroup
LabVIEW	HIMC Resume Group.vi
Python	ResumeGroup

6.2.7 HIMC_JogGroup

目的

軸グループを機械座標系の指定された方向に特定の速度で無限に動かします。

構文

```
int HIMC_JogGroup(
    int    ctrl_id,
    int    group_id,
    int    carte_dir,
    double jog_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

carte_dir [in] 機械座標系における動作の方向。
0～5 の数字は順に機械座標系の 6 自由度{X、Y、Z、A、B、C}を表します。

jog_vel [in] 特定の速度の値。
その正/負の値は、移動方向の同じ/逆方向の動きを表します。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_JogGroup
LabVIEW	HIMC Jog Group.vi
Python	JogGroup

6.2.8 HIMC_JogGroupAxis

目的

軸グループ内の特定の軸が、軸座標系で特定の速度で無限の運動を開始するようにします。

構文

```
int HIMC_JogGroupAxis(
        int     ctrl_id,
        int     group_id,
        int     grp_axis,
        double jog_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

grp_axis [in] 軸グループ内の軸インデックス
0～8 の数字は、各軸が軸グループに追加される順序を表します。

jog_vel [in] 特定の速度の値
その正/負の値は、移動方向の同じ/逆方向の動きを表します。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_JogGroupAxis
LabVIEW	HIMC Jog Group Axis.vi
Python	JogGroupAxis

6.2.9 HIMC_LineAbs2D

目的

機械座標系内の絶対目標位置に向かって、軸グループ上で補間された 2 次元直線移動を指令します。

構文

```
int HIMC_LineAbs2D(
        int     ctrl_id,
        int     group_id,
        double end_x,
        double end_y
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

end_x [in] X 座標における絶対ターゲット位置の値。

end_y [in] Y 座標における絶対ターゲット位置の値。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LineAbs2D
LabVIEW	HIMC Line Abs2D.vi
Python	LineAbs2D

6.2.10 HIMC_LineAbs3D

目的

機械座標系における絶対目標位置に向かって、軸グループ上で補間された 3 次元直線移動を指令します。

構文

```
int HIMC_LineAbs3D(
        int     ctrl_id,
        int     group_id,
        double end_x,
        double end_y,
        double end_z
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

end_x [in] X 座標における絶対ターゲット位置の値

end_y [in] Y 座標における絶対ターゲット位置の値

end_z [in] Z 座標における絶対ターゲット位置の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LineAbs3D
LabVIEW	HIMC Line Abs3D.vi
Python	LineAbs3D

6.2.11 HIMC_LineRel2D

目的

機械座標系内の相対位置に向かって、軸グループ上で補間された 2 次元直線移動を指令します。

構文

```
int HIMC_LineRel2D(
        int     ctrl_id,
        int     group_id,
     double distance_x,
     double distance_y
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

distance_x [in] X 座標における相対距離の値

distance_y [in] Y 座標における相対距離の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LineRel2D
LabVIEW	HIMC Line Rel2D.vi
Python	LineRel2D

6.2.12 HIMC_LineRel3D

目的

機械座標系内の相対位置に向かって、軸グループ上で補間された 3 次元直線移動を指令します。

構文

```
int HIMC_LineRel3D(
        int      ctrl_id,
        int      group_id,
        double distance_x,
        double distance_y,
        double distance_z
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
distance_x [in]	X 座標における相対距離の値
distance_y [in]	Y 座標における相対距離の値
distance_z [in]	Z 座標における相対距離の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LineRel3D
LabVIEW	HIMC Line Rel3D.vi
Python	LineRel3D

6.2.13 HIMC_Arc2D

目的

機械座標系内の絶対目標位置に向かって、軸グループ上で補間された 2 次元の円運動を指令します。

構文

```
int HIMC_Arc2D(
        int     ctrl_id,
        int     group_id,
        double border_x,
        double border_y,
        double end_x,
        double end_y
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
border_x [in]	X 座標における絶対境界位置の値
border_y [in]	Y 座標における絶対境界位置の値
end_x [in]	X 座標における絶対終了位置の値
end_y [in]	Y 座標における絶対終了位置の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

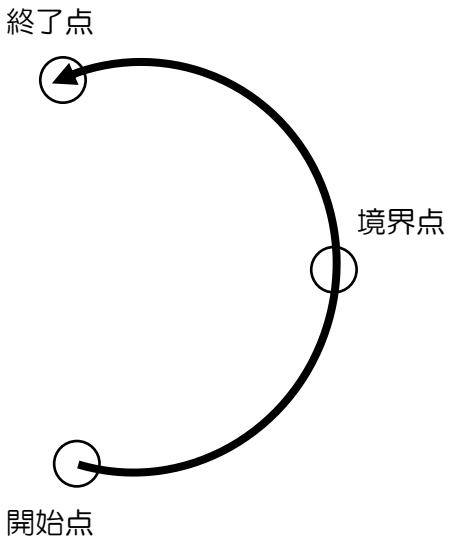
C#	HIMC_Arc2D
LabVIEW	HIMC Arc2D.vi
Python	Arc2D

長所

ユーザーは境界点（動作の最遠点）を指定し、機械がその境界点に到達できることを確認できます。

短所

単一コマンドでは角度 < 2π に制限されます。



6.2.14 HIMC_ArcCW2D

目的

機械座標系内の絶対目標位置に向かって、軸グループ上で時計回りに補間された 2 次元円運動を指令します。

構文

```
int HIMC_ArcCW2D(
        int     ctrl_id,
        int     group_id,
        double center_x,
        double center_y,
        double end_x,
        double end_y
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

center_x [in] X 座標における絶対中心位置の値

center_y [in] Y 座標における絶対中心位置の値

end_x [in] X 座標における絶対終了位置の値

end_y [in] Y 座標における絶対終了位置の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ArcCW2D
LabVIEW	HIMC ArcCW2D.vi
Python	ArcCW2D

6.2.15 HIMC_ArcCCW2D

目的

機械座標系内の絶対目標位置に向かって、軸グループ上で反時計回りに補間された 2 次元円運動を指令します。

構文

```
int HIMC_ArcCCW2D(
        int     ctrl_id,
        int     group_id,
        double center_x,
        double center_y,
        double end_x,
        double end_y
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
center_x [in]	X 座標における絶対中心位置の値
center_y [in]	Y 座標における絶対中心位置の値
end_x [in]	X 座標における絶対終了位置の値
end_y [in]	Y 座標における絶対終了位置の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ArcCCW2D
LabVIEW	HIMC ArcCCW2D.vi
Python	ArcCCW2D

6.2.16 HIMC_ArcAngle2D

目的

指定された角度に基づいて、機械座標系の絶対目標位置に向かって軸グループ上で補間された 2 次元の円運動を指令します。

構文

```
int HIMC_ArcAngle2D(
    int    ctrl_id,
    int    group_id,
    double center_x,
    double center_y,
    double angle
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
center_x [in]	X 座標における絶対中心位置の値
center_y [in]	Y 座標における絶対中心位置の値
angle [in]	絶対中心位置に対する開始点と終了点の相対的な角度。 円運動の方向と回転角度を決定します。 パラメーター単位: 度

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

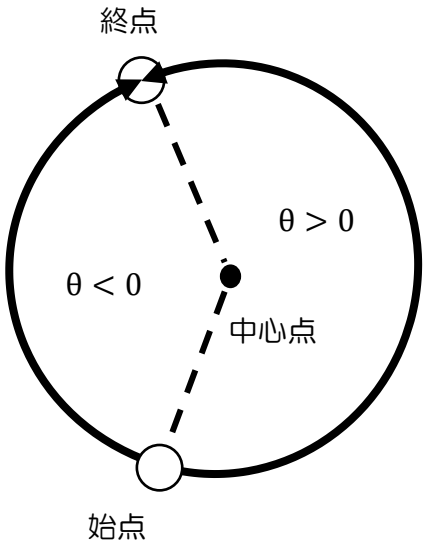
パラメーター「angle」は円軌道の回転方向を表します。
「angle」> 0 の場合は反時計回りに移動し、「angle」< 0 の場合は時計回りに移動します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ArcAngle2D
LabVIEW	HIMC Arc Angle2D.vi
Python	ArcAngle2D



θ の値によって円運動の方向が決まります。

$\theta > 0$: 反時計回りに動く

$\theta < 0$: 時計回りに動く

6.2.17 HIMC_Circle2D

目的

機械座標系内の絶対目標位置に向かって、軸グループ上で補間された 2 次元の円運動を指令します。

構文

```
int HIMC_Circle2D(
        int     ctrl_id,
        int     group_id,
        double  center_x,
        double  center_y,
        double  end_x,
        double  end_y,
        int     turns
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
center_x [in]	X 座標における絶対中心位置の値
center_y [in]	Y 座標における絶対中心位置の値
end_x [in]	X 座標における絶対終了位置の値
end_y [in]	Y 座標における絶対終了位置の値
turns [in]	開始点を基準とした円形パスの回転数 円軌道の方向と合計角度を決定します。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) パラメーター「turns」は円軌道の回転方向を表す。
「turns」>= 0 の場合は反時計回りに移動し、「turns」< 0 の場合は時計回りに移動します。

(2) ||turns|| <= 1 のとき、円軌道の全移動角度は<360°となる。
円軌道の合計移動角度が >= 360° (つまり、1 回転または 1 回転以上) の場合、||turns|| は >= 2 である必要があります。

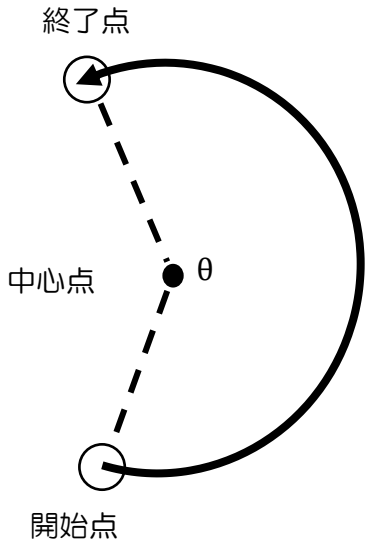
(3) この関数を使用する場合、「turns = 0」と「turns = 1」の動作は同じです。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Circle2D
LabVIEW	HIMC Circle2D.vi
Python	Circle2D



長所
角度に制限はありません。

短所
ユーザーは境界点(動きの最も遠い点)を指定できません。そのため、境界点に到達しない場合があります。

回転角の値は回転方向を決定します。※角度=θ+回転角×360
回転角≥0 は反時計回り、回転角<0 は時計回りを示します。回転角=0 の動きは回転角=1 の動きと同じであることに注意してください。以下の表は θ =210°を例にしています。

Turns	計算	角度 (度)
-2	$210 - 2 \times 360°$	$-510°$
-1	$210 - 1 \times 360°$	$-150°$
0	$210 + 0 \times 360°$	$210°$
1	$210 + 0 \times 360°$	$210°$
2	$210 + 1 \times 360°$	$570°$

6.3 グループ設定

6.3.1 HIMC_AddAxesToGrp

目的

特定の順序で軸グループに軸を追加します。

構文

```
int HIMC_AddAxesToGrp(
    int ctrl_id,
    int group_id,
    int num_of_axes,
    int *p_axis_list,
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
num_of_axes [in]	軸グループに追加する軸の数。(最大: 9)
p_axis_list [in]	軸 ID のシーケンス リストを格納するバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_AddAxesToGrp
LabVIEW	HIMC Add Axes To Grp.vi
Python	AddAxesToGrp

6.3.2 HIMC_RemoveAxisFromGrp

目的

軸グループから最後の軸を削除します。

構文

```
int HIMC_RemoveAxisFromGrp(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_RemoveAxisFromGrp
LabVIEW	HIMC Remove Axis From Grp.vi
Python	RemoveAxisFromGrp

6.3.3 HIMC_SetupGroup

目的

特定のシーケンスを持つ軸グループを設定します。

構文

```
int HIMC_SetupGroup(
        int ctrl_id,
        int group_id,
        int num_of_axes,
        int *p_axis_list,
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
num_of_axes [in]	軸の数。（最大：9）
p_axis_list [in]	軸 ID のシーケンス リストを格納するバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetupGroup
LabVIEW	HIMC Setup Group.vi
Python	SetupGroup

6.3.4 HIMC_UngrpAllAxes

目的

軸グループをグループ解除して無効にします。

構文

```
int HIMC_UngrpAllAxes(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_UngrpAllAxes
LabVIEW	HIMC Ungrp All Axes.vi
Python	UngrpAllAxes

6.3.5 HIMC_GetGroupID

目的

軸が属する軸グループ ID を取得します。

構文

```
int HIMC_GetGroupID(
    int ctrl_id,
    int axis_id,
    int *p_group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_group_id [out] 軸が属する軸グループ ID を受け取るバッファへのポインタ。
値が -1 の場合、軸がどの軸グループにも属していないことを示します。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGroupID
LabVIEW	HIMC Get Group ID.vi
Python	GetGroupID

6.3.6 HIMC_SetGrpMotionProfile

目的

軸グループの TCP 直線モーション パラメーターを設定します。

構文

```
int HIMC_SetGrpMotionProfile(
        int     ctrl_id,
        int     group_id,
     double max_velocity,
     double max_acceleration,
     double max_deceleration,
     double smooth_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
max_velocity [in]	軸グループの最大線形プロファイル速度。 パラメーター単位: mm/s 入力範囲: 0 ~ 5000
max_acceleration [in]	軸グループの最大線形プロファイル加速度。 パラメーター単位: mm/s^2 入力範囲: >0 ~ 50000 (加速度は 0 にできません)
max_deceleration [in]	軸グループの最大線形プロファイル減速。 パラメーター単位: mm/s^2 入力範囲: >0 ~ 50000 (減速は 0 にできません)
smooth_time [in]	軸グループの線形プロファイルのスムーズ時間。 パラメーター単位: ms 入力範囲: 0 ~ 500

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

線形グループモーションプロファイルのデフォルト値は、速度、加速度、減速、スムーズ時間に対してそれぞれ [100, 5000, 5000, 50] です。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpMotionProfile
LabVIEW	HIMC Set Grp Motion Profile.vi
Python	SetGrpMotionProfile

6.3.7 HIMC_SetGrpAngMotionProfile

目的

軸グループの TCP 角度モーション パラメーターを設定します。

構文

```
int HIMC_SetGrpAngMotionProfile(
    int    ctrl_id,
    int    group_id,
    double max_velocity,
    double max_acceleration,
    double max_deceleration,
    double smooth_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
max_velocity [in]	軸グループの最大角プロファイル速度。 パラメーター単位: deg/s 入力範囲: 0 ~ 7200
max_acceleration [in]	軸グループの最大角プロファイル加速度。 パラメーター単位: deg/s^2 入力範囲: >0 ~ 72000 (加速度は 0 にできません)
max_deceleration [in]	軸グループの最大角度プロファイル減速。 パラメーター単位: deg/s^2 入力範囲: >0 ~ 72000 (減速は 0 にできません)
smooth_time [in]	軸グループの角度プロファイルのスムーズ時間。 パラメーター単位: m 入力範囲: 0 ~ 500

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

角度グループモーションプロファイルのデフォルト値は、速度、加速度、減速、スムーズタイムがそれぞれ [360, 1800, 1800, 50] です。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpAngMotionProfile
LabVIEW	HIMC Set Grp Ang Motion Profile.vi
Python	SetGrpAngMotionProfile

6.3.8 HIMC_GetGrpKin

目的

軸グループの運動学タイプを取得します。

構文

```
int HIMC_GetGrpKin(
        int ctrl_id,
        int group_id,
        int *p_grp_kin
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

p_grp_kin [out] 軸グループの運動学タイプを受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpKin
LabVIEW	HIMC Get Grp Kin.vi
Python	GetGrpKin

6.3.9 HIMC_SetGrpKin

目的

軸グループの運動学タイプを設定します。

構文

```
int HIMC_SetGrpKin(
        int ctrl_id,
        int group_id,
        int kin_type
);
```

パラメーター

ctrl_id [in]　HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in]　軸グループインデックス

kin_type [in]　軸グループの新しい運動学タイプ。詳細はセクション 6.1.3 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpKin
LabVIEW	HIMC Set Grp Kin.vi
Python	SetGrpKin

6.3.10 HIMC_GetGrpVel

目的

軸グループの移動速度の目標値を取得します。

構文

```
int HIMC_GetGrpVel(
        int      ctrl_id,
        int      group_id,
        double *p_grp_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

p_grp_vel [out] 軸グループの移動速度の目標値を受け取るバッファへのポインタ。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpVel
LabVIEW	-
Python	GetGrpVel

6.3.11 HIMC_SetGrpVel

目的

軸グループの移動速度の目標値を設定します。

構文

```
int HIMC_SetGrpVel(
    int    ctrl_id,
    int    group_id,
    double vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

vel [in] 軸グループの新しいモーション速度の目標値。
パラメーター単位: mm/s または deg/s
入力範囲: 0 ~ 5000

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpVel
LabVIEW	HIMC Set Grp Vel.vi
Python	SetGrpVel

6.3.12 HIMC_GetGrpAcc

目的

軸グループのモーション加速度の目標値を取得します。

構文

```
int HIMC_GetGrpAcc(
        int      ctrl_id,
        int      group_id,
        double *p_grp_acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

p_grp_acc [out] 軸グループのモーション加速度の目標値を受け取るバッファへのポインタ。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpAcc
LabVIEW	-
Python	GetGrpAcc

6.3.13 HIMC_SetGrpAcc

目的

軸グループのモーション加速度の目標値を設定します。

構文

```
int HIMC_SetGrpAcc(
    int    ctrl_id,
    int    group_id,
    double acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

acc [in] 軸グループの新しいモーション加速のターゲット値。
パラメーター単位: mm/s^2 または deg/s^2
入力範囲: >0 ~ 50000 (加速度は 0 にできません)

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpAcc
LabVIEW	HIMC Set Grp Acc.vi
Python	SetGrpAcc

6.3.14 HIMC_SetGrpAccTime

目的

軸グループの加速時間を設定します。

構文

```
int HIMC_SetGrpAccTime(
    int    ctrl_id,
    int    group_id,
    double acc_time
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

acc_time [in] 軸グループの加速時間
パラメーター単位: ms
入力範囲: ゼロ以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpAccTime
LabVIEW	HIMC Set Grp Acc Time.vi
Python	SetGrpAccTime

6.3.15 HIMC_GetGrpDec

目的

軸グループのモーション減速の目標値を取得します。

構文

```
int HIMC_GetGrpDec(
        int      ctrl_id,
        int      group_id,
        double *p_grp_dec
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

p_grp_dec [out] 軸グループのモーション減速の目標値を受け取るバッファへのポインタ。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpDec
LabVIEW	-
Python	GetGrpDec

6.3.16 HIMC_SetGrpDec

目的

軸グループのモーション減速の目標値を設定します。

構文

```
int HIMC_SetGrpDec(
    int    ctrl_id,
    int    group_id,
    double dec
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

dec [in] 軸グループの新しいモーション減速の目標値。
パラメーター単位: mm/s^2 または deg/s^2
入力範囲: >0 ~ 50000 (減速は 0 にできません)

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpDec
LabVIEW	HIMC Set Grp Dec.vi
Python	SetGrpDec

6.3.17 HIMC_SetGrpDecTime

目的

軸グループの減速時間を設定します。

構文

```
int HIMC_SetGrpDecTime(
    int    ctrl_id,
    int    group_id,
    double dec_time
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

dec_time [in] 軸グループの減速時間
パラメーター単位: ms
入力範囲: 0 以外の正の値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpDecTime
LabVIEW	HIMC Set Grp Dec Time.vi
Python	SetGrpDecTime

6.3.18 HIMC_GetGrpSMTime

目的

軸グループのプロファイルのスムーズ時間を取得します。

構文

```
int HIMC_GetGrpSMTime(
        int      ctrl_id,
        int      group_id,
        double *p_grp_smooth_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_grp_smooth_time [out]	軸グループのプロファイルのスムーズ時間を受け取るバッファへのポインタ。 パラメーター単位: ms

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpSMTime
LabVIEW	HIMC Get Grp SM Time.vi
Python	GetGrpSMTime

6.3.19 HIMC_SetGrpSMTime

目的

軸グループのプロファイルのスムーズ時間を設定します。

構文

```
int HIMC_SetGrpSMTime(
    int    ctrl_id,
    int    group_id,
    double smooth_time
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
smooth_time [in]	軸グループの新しいプロファイルのスムーズ時間。 パラメーター単位: ms 入力範囲: 0 ~ 500

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpSMTime
LabVIEW	HIMC Set Grp SM Time.vi
Python	SetGrpSMTime

6.3.20 HIMC_GetGrpCoordSys

目的

軸グループの座標系を取得します。

構文

```
int HIMC_GetGrpCoordSys(
        int ctrl_id,
        int group_id,
        int *p_grp_coord_sys
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_grp_coord_sys [out]	軸グループの座標系を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpCoordSys
LabVIEW	HIMC Get Coord Sys.vi
Python	GetGrpCoordSys

6.3.21 HIMC_SetGrpCoordSys

目的

軸グループの座標系を設定します。

構文

```
int HIMC_SetGrpCoordSys(
    int ctrl_id,
    int group_id,
    int coord_sys
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

coord_sys [in] 軸グループの新しい座標系。詳細はセクション 6.1.2 を参照してください。
例: 1. kCoord_MCS
2. kCoord_WCS1 | kCoord_PCS
3. kCoord_OFFSET | kCoord_WCS2 | kCoord_PCS

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpCoordSys
LabVIEW	HIMC Set Grp Coord Sys.vi
Python	SetGrpCoordSys

6.3.22 HIMC_GetGrpBufferMode

目的

軸グループのバッファ モードを取得します。

構文

```
int HIMC_GetGrpBufferMode(
    int ctrl_id,
    int group_id,
    int *p_grp_buffer_mode
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_grp_buffer_mode [out]	軸グループのバッファ モードを受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpBufferMode
LabVIEW	HIMC Get Buffer Mode.vi
Python	GetGrpBufferMode

6.3.23 HIMC_SetGrpBufferMode

目的

軸グループのバッファ モードを設定します。

構文

```
int HIMC_SetGrpBufferMode(
    int ctrl_id,
    int group_id,
    int buffer_mode
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

buffer_mode [in] 軸グループの新しいバッファモード。詳細はセクション 6.1.4 を参照してください。
入力範囲: 0 ~ 5

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpBufferMode
LabVIEW	HIMC Set Grp Buffer Mode.vi
Python	SetGrpBufferMode

6.3.24 HIMC_GetGrpTransMode

目的

軸グループの遷移モードを取得します。

構文

```
int HIMC_GetGrpTransMode(
        int ctrl_id,
        int group_id,
        int *p_grp_trans_mode
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_grp_trans_mode [out]	軸グループの遷移モードを受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpTransMode
LabVIEW	HIMC Get Trans Mode.vi
Python	GetGrpTransMode

6.3.25 HIMC_SetGrpTransMode

目的

軸グループの遷移モードを設定します。

構文

```
int HIMC_SetGrpTransMode(
    int ctrl_id,
    int group_id,
    int trans_mode
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

trans_mode [in] 軸グループの新しい遷移モード。詳細はセクション 6.1.5 を参照してください。
入力範囲: 0 ~ 4

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpTransMode
LabVIEW	HIMC Set Grp Trans Mode.vi
Python	SetGrpTransMode

6.3.26 HIMC_SetGrpTransPrm

目的

軸グループの遷移モードのパラメーターを設定します。

構文

```
int HIMC_SetGrpTransPrm(
        int     ctrl_id,
        int     group_id,
    double trans_vel,
    double trans_dis
    double trans_dev,
    double trans_curv
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

trans_vel [in] 軸グループの新しい遷移モードの速度パラメーター。
詳細についてはセクション 6.1.5 を参照してください。

trans_dis [in] 軸グループの新しい遷移モードの距離パラメーター。
詳細についてはセクション 6.1.5 を参照してください。

trans_dev [in] 軸グループの新しい遷移モードの最大偏差パラメーター。
詳細についてはセクション 6.1.5 を参照してください。

trans_curv [in] 軸グループの新しい遷移モードの最大曲率パラメーター。
詳細についてはセクション 6.1.5 を参照してください。

表 6.3.26.1 遷移パラメーターの設定範囲

遷移パラメーター	単位	最大	最小
trans_vel	mm/s	< 5000	> 0
trans_dis	mm	< [len_{min}] = MIN([len_{S1}],[len_{S2}])	> 0
trans_dev	mm	< [trans_dis] $\times \frac{1}{2} cos[\frac{\theta}{2}]$ **注: θ はコーナー角度です**	> 0
trans_curv	mm^{-1}	< 10^7	> $\frac{6 \times \sin([\theta])}{[len_{min}]\sqrt{2-2\cos([\theta])}^3}$

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpTransPrm
LabVIEW	HIMC Set Grp Trans Prm.vi
Python	SetGrpTransPrm

6.3.27 HIMC_GetGrpCmdNum

目的

コマンド バッファ内の軸グループのコマンドの数を取得します。

構文

```
int HIMC_GetGrpCmdNum(
        int ctrl_id,
        int group_id,
        int *p_grp_cmd_num
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_grp_cmd_num [out]	コマンド バッファ内の軸グループのコマンドの数を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpCmdNum
LabVIEW	HIMC Get Grp Cmd Num.vi
Python	GetGrpCmdNum

6.3.28 HIMC_SetGrpVelScale

目的

軸グループモーションの速度スケールを設定します。

構文

```
int HIMC_SetGrpVelScale(
    int    ctrl_id,
    int    group_id,
    double vel_scale
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

vel_scale [in] 軸グループモーションの新しい速度スケール。
入力範囲: 0 ~ 100

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpVelScale
LabVIEW	HIMC Set Grp Vel Scale.vi
Python	SetGrpVelScale

6.3.29 HIMC_GetGrpVelScale

目的

軸グループモーションの速度スケールを取得します。

構文

```
int HIMC_GetGrpVelScale(
    int    ctrl_id,
    int    group_id,
    double *p_grp_vel_scale
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_grp_vel_scale [out]	軸グループモーションの速度スケールを受け取るバッファへのポインタ。 範囲は 0 から 100 です。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpVelScale
LabVIEW	HIMC Get Grp Vel Scale.vi
Python	GetGrpVelScale

6.3.30 HIMC_GetGrpCoordTrans

目的

軸グループの座標系の変換パラメーターを取得します。

構文

```
int HIMC_GetGrpCoordTrans(
    int     ctrl_id,
    int     group_id,
    int     coord_sys,
    double *trans_param
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
coord_sys [in]	座標系 詳細については、セクション 21.2 CoordSystem を参照してください。
trans_param [out]	6-DOF {X、Y、Z、A、B、C} の変換パラメーターを含む 6 要素配列へのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpCoordTrans
LabVIEW	HIMC Get Grp Coord Trans.vi
Python	GetGrpCoordTrans

6.3.31 HIMC_SetGrpCoordTrans

目的

軸グループの座標系の変換パラメーターを設定します。

構文

```
int HIMC_SetGrpCoordTrans(
    int     ctrl_id,
    int     group_id,
    int     coord_sys,
    double *trans_param
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
coord_sys [in]	座標系 詳細については、セクション 21.2 CoordSystem を参照してください。
trans_param [in]	6-DOF {X、Y、Z、A、B、C} の変換パラメーターを含む 6 要素配列へのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpCoordTrans
LabVIEW	HIMC Set Grp Coord Trans.vi
Python	SetGrpCoordTrans

6.3.32 HIMC_GetGrpPoseCmd

目的

軸グループの座標系のポーズコマンドを取得します。

構文

```
int HIMC_GetGrpPoseCmd(
    int     ctrl_id,
    int     group_id,
    int     coord_sys,
    double *pose_cmd
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
coord_sys [in]	座標系 詳細については、セクション 21.2 CoordSystem を参照してください。
pose_cmd [out]	6-DOF {X、Y、Z、A、B、C} のポーズコマンドを含む 6 要素配列へのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpPoseCmd
LabVIEW	HIMC Get Grp Pose Cmd.vi
Python	GetGrpPoseCmd

6.3.33 HIMC_GetGrpPoseFb

目的

軸グループの座標系のポーズフィードバックを取得します。

構文

```
int HIMC_GetGrpPoseFb(
        int       ctrl_id,
        int       group_id,
        int       coord_sys,
        double *pose_fb
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
coord_sys [in]	座標系 詳細については、セクション 21.2 CoordSystem を参照してください。
pose_fb [out]	6-DOF {X、Y、Z、A、B、C} のポーズフィードバックを含む 6 要素配列へのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGrpPoseFb
LabVIEW	HIMC Get Grp Pose Fb.vi
Python	GetGrpPoseFb

6.3.34 HIMC_SetGrpLookAheadPrm

目的

軸グループの先読み機能の動作パラメーターを設定します。

構文

```
int HIMC_SetGrpLookAheadPrm(
    int     ctrl_id,
    int     group_id,
    int     prm_id,
    double value
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
prm_id [in]	先読み関数のパラメーター。 0: 最大コーナー加速度、1: 最大円弧加速度、2: 最大弦誤差
value [in]	先読み機能のパラメーター設定値。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpLookAheadPrm
LabVIEW	HIMC Set Grp Look Ahead Prm.vi
Python	SetGrpLookAheadPrm

6.3.35 HIMC_SetGrpQueueSize

目的

軸グループコマンドのバッファサイズを設定します。

構文

```
int HIMC_SetGrpQueueSize(
    int ctrl_id,
    int group_id,
    int queue_size
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
queue_size [in]	軸グループコマンドのバッファサイズ。 入力値が n の場合、バッファのサイズは 2^n になります。 入力範囲: 0 ~ 10

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpQueueSize
LabVIEW	HIMC Set Grp Queue Size.vi
Python	SetGrpQueueSize

6.4 グループステータス

6.4.1 HIMC_IsGrpEnabled

目的

軸グループの「有効」ステータスを照会します。

構文

```
int HIMC_IsGrpEnabled(
        int ctrl_id,
        int group_id,
        int *p_is_grp_enabled
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_is_grp_enabled [out]	軸グループの有効ステータスを受け取るバッファへのポインタ。 軸グループが「GrpEnabled」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGrpEnabled
LabVIEW	HIMC Is Grp Enabled.vi
Python	IsGrpEnabled

6.4.2 HIMC_IsGrpMoving

目的

軸グループの「移動」ステータスを照会します。軸グループが移動中の場合、PG（プロファイルジェネレータ）は新しい位置を出力し続けます。

構文

```
int HIMC_IsGrpMoving(
        int ctrl_id,
        int group_id,
        int *p_is_grp_moving
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_is_grp_moving [out]	軸グループの移動ステータスを受け取るバッファへのポインタ。 軸グループが「GrpMoving」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGrpMoving
LabVIEW	HIMC Is Grp Moving.vi
Python	IsGrpMoving

6.4.3 HIMC_IsGrpInPos

目的

軸グループの「インポジション」ステータスを照会します。軸グループがインポジションの場合、グループ内のすべての軸がインポジションになります。

構文

```
int HIMC_IsGrpInPos(
        int ctrl_id,
        int group_id,
        int *p_is_grp_inpos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_is_grp_inpos [out]	軸グループのインポジションステータスを受け取るバッファへのポインタ。 軸グループが「GrpInPos」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGrpInPos
LabVIEW	HIMC Is Grp In Pos.vi
Python	IsGrpInPos

6.4.4 HIMC_IsGrpErrorStop

目的

軸グループが「エラー停止」状態にあるかどうかを照会します。

構文

```
int HIMC_IsGrpErrorStop(
        int ctrl_id,
        int group_id,
        int *p_is_grp_errorstop
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_is_grp_errorstop [out]	軸グループのエラー停止ステータスを受け取るバッファへのポインタ。 軸グループが「GrpErrorStop」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsGrpErrorStop
LabVIEW	HIMC Is Grp Error Stop.vi
Python	IsGrpErrorStop

6.5 高度なグループモーション制御

6.5.1 HIMC_LineAbs

目的

特定の座標系内の絶対位置に向かって軸グループ上で補間された直線移動を命令します。

構文

```
int HIMC_LineAbs(
        int ctrl_id,
        int group_id,
        CoordPosition *target_pos,
        MotionProfile *motion_profile,
        CoordSystem coord_sys,
        MotionBufferMode buff_mode,
        MotionTransitionMode trans_mode,
        TransPrm *trans_prm
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
target_pos [in]	ツールの中心点 (エンドエフェクタ) の 6-DOF ターゲット位置を格納するバッファへのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg 詳細については、セクション 20.2 CoordPosition を参照してください。
motion_profile [in]	ツールの中心点 (エンドエフェクタ) のモーション プロファイル設定を格納するバッファへのポインタ。 詳細については、セクション 20.3 MotionProfile を参照してください。
coord_sys [in]	適用可能な座標系を指定します。 詳細については、セクション 21.2 CoordSystem を参照してください。
buff_mode [in]	バッファ モードを指定します。 詳細については、セクション 21.3 MotionBufferMode を参照してください。
trans_mode [in]	遷移モードを指定します。

詳細については、セクション 21.4 MotionTransitionMode を参照してください。

trans_prm [in] 遷移モードのパラメーターへのポインタを指定します。
詳細については、セクション 20.6 TransPrm を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LineAbs
LabVIEW	HIMC Line Abs.vi
Python	LineAbs

6.5.2 HIMC_LineRel

目的

特定の座標系内の相対位置に向かって軸グループ上で補間された直線移動を命令します。

構文

```
int HIMC_LineRel(
        int ctrl_id,
        int group_id,
        CoordPosition *relative_dist,
        MotionProfile *motion_profile,
        CoordSystem coord_sys,
        MotionBufferMode buff_mode,
        MotionTransitionMode trans_mode,
        TransPrm *trans_prm
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
relative_dist [in]	ツール中心点 (エンドエフェクタ) の 6-DOF の相対距離を格納するバッファへのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg 詳細については、セクション 20.2 CoordPosition を参照してください。
motion_profile [in]	ツールの中心点 (エンドエフェクタ) のモーション プロファイル設定を格納するバッファへのポインタ。 詳細については、セクション 20.3 MotionProfile を参照してください。
coord_sys [in]	適用可能な座標系を指定します。 詳細については、セクション 21.2 CoordSystem を参照してください。
buff_mode [in]	バッファ モードを指定します。 詳細については、セクション 21.3 MotionBufferMode を参照してください。
trans_mode [in]	遷移モードを指定します。 詳細については、セクション 21.4 MotionTransitionMode を参照してください。

trans_prm [in] 遷移モードのパラメーターへのポインタを指定します。
詳細については、セクション 20.6 TransPrm を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LineRel
LabVIEW	HIMC Line Rel.vi
Python	LineRel

6.5.3 HIMC_CircleAbs

目的

特定の座標系内の絶対位置に向かって軸グループ上で補間された円運動を指令します。

構文

```
int HIMC_CircleAbs(
    int ctrl_id,
    int group_id,
    CenterPosition *center_pos,
    NormalVector *normal_vector,
    int turns,
    CoordPosition *target_pos,
    MotionProfile *motion_profile,
    CoordSystem coord_sys,
    MotionBufferMode buff_mode,
    MotionTransitionMode trans_mode,
    TransPrm *trans_prm
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
center_pos [in]	中心点の位置を格納するバッファへのポインタ。 パラメーター単位: mm 詳細については、セクション 20.4 CenterPosition を参照してください。
normal_vector [in]	円の法線ベクトルを格納するバッファへのポインタ。 詳細については、セクション 20.5 NormalVector を参照してください。
turns [in]	開始点を基準とした円形パスの回転数。 円軌道の方向と合計角度を決定します。
target_pos [in]	ツールの中心点 (エンドエフェクタ) の 6-DOF ターゲット位置を格納するバッファへのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg 詳細については、セクション 20.2 CoordPosition を参照してください。

motion_profile [in] ツールの中心点 (エンドエフェクタ) のモーション プロファイル設定を格納するバッファへのポインタ。
詳細については、セクション 20.3 MotionProfile を参照してください。

coord_sys [in] 適用可能な座標系を指定します。
詳細については、セクション 21.2 CoordSystem を参照してください。

buff_mode [in] バッファ モードを指定します。
詳細については、セクション 21.3 MotionBufferMode を参照してください。

trans_mode [in] 遷移モードを指定します。
詳細については、セクション 21.4 MotionTransitionMode を参照してください。

trans_prm [in] 遷移モードのパラメーターへのポインタを指定します。
詳細については、セクション 20.6 TransPrm を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_CircleAbs
LabVIEW	HIMC Circle Abs.vi
Python	CircleAbs

6.5.4 HIMC_CircleRel

目的

特定の座標系内の相対位置に向かって軸グループ上で補間された円運動を指令します。

構文

```
int HIMC_CircleRel(
    int ctrl_id,
    int group_id,
    CenterPosition *center_pos,
    NormalVector *normal_vector,
    int turns,
    CoordPosition *relative_dist,
    MotionProfile *motion_profile,
    CoordSystem coord_sys,
    MotionBufferMode buff_mode,
    MotionTransitionMode trans_mode,
    TransPrm *trans_prm
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
center_pos [in]	中心点の位置を格納するバッファへのポインタ。 パラメーター単位: mm 詳細については、セクション 20.4 CenterPosition を参照してください。
normal_vector [in]	円の法線ベクトルを格納するバッファへのポインタ。 詳細については、セクション 20.5 NormalVector を参照してください。
turns [in]	開始点を基準とした円形パスの回転数。 円軌道の方向と合計角度を決定します。
relative_dist [in]	ツール中心点 (エンドエフェクタ) の 6-DOF の相対距離を格納するバッファへのポインタ。 パラメーターの単位: X、Y、Z の場合は mm、A、B、C の場合は deg 詳細については、セクション 20.2 CoordPosition を参照してください。

motion_profile [in] ツールの中心点 (エンドエフェクタ) のモーション プロファイル設定を格納するバッファへのポインタ。
詳細については、セクション 20.3 MotionProfile を参照してください。

coord_sys [in] 適用可能な座標系を指定します。
詳細については、セクション 21.2 CoordSystem を参照してください。

buff_mode [in] バッファ モードを指定します。
詳細については、セクション 21.3 MotionBufferMode を参照してください。

trans_mode [in] 遷移モードを指定します。
詳細については、セクション 21.4 MotionTransitionMode を参照してください。

trans_prm [in] 遷移モードのパラメーターへのポインタを指定します。
詳細については、セクション 20.6 TransPrm を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_CircleRel
LabVIEW	HIMC Circle Rel.vi
Python	CircleRel

7. GPIO 関数

7.1 概要

HIMC は 8 セットの汎用入出力（GPIO）ピンを備えています。ハードウェア遅延時間は 1ms 以内、電源は 24V です。サブデバイスは CoE 通信を介してコントローラーに接続し、IO ステータスを更新できます。IO の数はサブデバイスによって異なります。この章で提供される「SetGPO」や「SetSlvGPO」などの関数を使用すると、ユーザーは HIMC とスレーブのそれぞれの出力ピンの信号を設定できます。さらに、ユーザーは入出力ピンの信号状態を照会できます。iA Studio の機能モジュール「デジタル IO」（「iA Studio ユーザーガイド」のセクション 4.4 を参照）を使用すると、ユーザーは HIMC とスレーブの入出力状態を監視および設定できます。

HIMC のデジタル入力「I8」は E-stop 信号（「HIMC インストールガイド」のセクション 3.3 を参照）であり、立ち上がりエッジ信号を受信するとトリガーされます。このとき、すべての軸が無効になり、すべての HMPL タスクが停止します。

注: 立ち上がりエッジ信号がトリガーされた後、ユーザーは軸を再度有効にするか、HMPL タスクを再実行できます。

7.1.1 GPIO 変数

ユーザーは、iA Studio のスコープマネージャを介して、必要なコントローラーの汎用入出力システム変数を選択できます（「iA Studio ユーザーガイド」のセクション 4.8 を参照）。詳細な説明は
表 7.1.1.1 に示されています。

表 7.1.1.1 コントローラーの汎用入出力変数

名称	変数	単位	説明
HIMC GPO	himc_gpo_bits	N/A	コントローラーの汎用出力ピンの状態。
HIMC GPI	himc_gpi_bits	N/A	コントローラーの汎用入力ピンの状態。

7.2 Controller IO setting

7.2.1 HIMC_SetGPO

目的

コントローラーの汎用出力の状態を設定します。

構文

```
int HIMC_SetGPO(
    int   ctrl_id,
    int   gpo_idx,
    char state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

gpo_idx [in] 汎用出力インデックス

state [in] 設定する状態

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGPO
LabVIEW	HIMC Set GPO.vi
Python	SetGPO

7.2.2 HIMC_ToggleGPO

目的

コントローラーの汎用出力の状態を切り替えます。

構文

```
int HIMC_ToggleGPO(
    int ctrl_id,
    int gpo_idx
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

gpo_idx [in] 汎用出力インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ToggleGPO
LabVIEW	HIMC Toggle GPO.vi
Python	ToggleGPO

7.2.3 HIMC_SetAllGPO

目的

コントローラーの複数の汎用出力の状態を設定します。

構文

```
int HIMC_SetAllGPO(
    int ctrl_id,
    int all_gpo_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

all_gpo_state [in] すべての汎用出力の状態値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetAllGPO
LabVIEW	HIMC Set All GPO.vi
Python	SetAllGPO

7.2.4 HIMC_SetGPIInvert

目的

コントローラーの汎用入力の反転状態を設定します。

構文

```
int HIMC_SetGPIInvert(
    int  ctrl_id,
    int  gpi_idx,
    char invert
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

gpi_idx [in] 汎用入力インデックス

invert [in] 設定する反転状態

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGPIInvert
LabVIEW	HIMC Set GPI Invert.vi
Python	SetGPIInvert

7.2.5 HIMC_SetGPOInvert

目的

コントローラーの汎用出力の反転状態を設定します。

構文

```
int HIMC_SetGPOInvert(
    int  ctrl_id,
    int  gpo_idx,
    char invert
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

gpo_idx [in] 汎用出力インデックス

invert [in] 設定する反転状態

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGPOInvert
LabVIEW	HIMC Set GPO Invert.vi
Python	SetGPOInvert

7.2.6 HIMC_BindEMO

目的

汎用入力ピンを E-Stop にバインドするように設定します。

構文

```
int HIMC_BindEMO(
    int ctrl_id,
    int gpi_idx
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

gpi_idx [in] 汎用入力インデックス。デフォルト値は 8 です。
0 に設定すると、すべての汎用入力ピンが E-Stop にバインドされません。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_BindEMO
LabVIEW	HIMC Bind EMO.vi
Python	BindEMO

7.3 スレーブ IO 設定

7.3.1 HIMC_SetSlvGPO

目的
スレーブの汎用出力の状態を設定します。

構文

```
int HIMC_SetSlvGPO(
    int ctrl_id,
    int slv_slot_id,
    int gpo_idx,
    int on_off
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
gpo_idx [in]	汎用出力インデックス
on_off [in]	設定する状態

戻りの値
関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考
この機能を使用する場合、ユーザーはデジタル出力オブジェクトを PDO として設定する必要があります。
たとえば、ドライバーの 0x60FE (デジタル出力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvGPO
LabVIEW	HIMC Set Slv GPO.vi
Python	SetSlvGPO

7.3.2 HIMC_ToggleSlvGPO

目的

スレーブの汎用出力の状態を切り替えます。

構文

```
int HIMC_ToggleSlvGPO(
        int ctrl_id,
        int slv_slot_id,
        int gpo_idx
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

gpo_idx [in] 汎用出力インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル出力オブジェクトを PDO として設定する必要があります。
たとえば、ドライバーの 0x60FE (デジタル出力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ToggleSlvGPO
LabVIEW	HIMC Toggle Slv GPO.vi
Python	ToggleSlvGPO

7.3.3 HIMC_SetSlvAllGPO

目的

スレーブの複数の汎用出力の状態を設定します。

構文

```
int HIMC_SetSlvAllGPO(
    int ctrl_id,
    int slv_slot_id,
    int all_gpo_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

all_gpo_state [in] すべての汎用出力の状態値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル出力オブジェクトを PDO として設定する必要があります。
たとえば、ドライバーの 0x60FE (デジタル出力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAllGPO
LabVIEW	HIMC Set Slv All GPO.vi
Python	SetSlvAllGPO

7.4 コントローラーIO ステータス

7.4.1 HIMC_GetGPI

目的

コントローラーの汎用入力の状態を取得します。

構文

```
int HIMC_GetGPI(
    int     ctrl_id,
    int     gpi_idx,
    char *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

gpi_idx [in] 汎用入力インデックス

p_state [out] 特定の入力の状態を受け取るバッファへのポインタ。
入力が「オン」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGPI
LabVIEW	HIMC Get GPI.vi
Python	GetGPI

7.4.2 HIMC_GetGPO

目的

コントローラーの汎用出力の状態を取得します。

構文

```
int HIMC_GetGPO(
        int     ctrl_id,
        int     gpo_idx,
        char *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

gpo_idx [in] 汎用出力インデックス

p_state [out] 特定の出力の状態を受け取るバッファへのポインタ。
出力が「オン」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGPO
LabVIEW	HIMC Get GPO.vi
Python	GetGPO

7.4.3 HIMC_GetAllGPI

目的

コントローラーの複数の汎用入力の状態を取得します。

構文

```
int HIMC_GetAllGPI(
    int    ctrl_id,
    int *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_state [out] コントローラーの汎用入力の状態を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetAllGPI
LabVIEW	HIMC Get All GPI.vi
Python	GetAllGPI

7.4.4 HIMC_GetAllGPO

目的

コントローラーの複数の汎用出力の状態を取得します。

構文

```
int HIMC_GetAllGPO(
    int    ctrl_id,
    int *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_state [out] コントローラーの汎用出力の状態を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetAllGPO
LabVIEW	HIMC Get All GPO.vi
Python	GetAllGPO

7.5 スレーブ IO ステータス

7.5.1 HIMC_GetSlvGPIz

目的

スレーブの汎用入力の状態を取得します。

構文

```
int HIMC_GetSlvGPI(
    int   ctrl_id,
    int   slv_slot_id,
    int   gpi_idx,
    int *p_state
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
gpi_idx [in]	汎用入力インデックス
p_state [out]	特定の入力の状態を受け取るバッファへのポインタ。 入力が「オン」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル入力オブジェクトを PDO として設定する必要があります。
たとえば、ドライバーの 0x60FE (デジタル入力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvGPI
LabVIEW	HIMC Get Slv GPI.vi
Python	GetSlvGPI

7.5.2 HIMC_GetSlvGPO

目的

スレーブの汎用出力の状態を取得します。

構文

```
int HIMC_GetSlvGPO(
    int   ctrl_id,
    int   slv_slot_id,
    int   gpo_idx,
    int *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

gpo_idx [in] 汎用出力インデックス

p_state [out] 特定の出力の状態を受け取るバッファへのポインタ。
出力が「オン」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル出力オブジェクトを PDO として設定する必要があります。
たとえば、ドライバーの 0x60FE (デジタル出力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvGPO
LabVIEW	HIMC Get Slv GPO.vi
Python	GetSlvGPO

7.5.3 HIMC_GetSlvAllGPI

目的

スレーブのすべての汎用入力状態を照会します。

構文

```
int HIMC_GetSlvAllGPI(
        int ctrl_id,
        int slv_slot_id,
        int *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

p_state [out] スレーブのすべての汎用入力状態を受信するバッファへのポインタ。
1 番目と 4 番目の GPI ピンが TRUE の場合、戻り値は 9 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル入力オブジェクトを PDO として設定する必要があります。たとえば、ドライバーの 0x60FD (デジタル入力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAllGPI
LabVIEW	HIMC Get Slv All GPI.vi
Python	GetSlvAllGPI

7.5.4 HIMC_GetSlvAllGPO

目的

スレーブのすべての汎用出力状態を照会します。

構文

```
int HIMC_GetSlvAllGPO(
    int ctrl_id,
    int slv_slot_id,
    int *p_state
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

p_state [out] 特定の出力の状態を受け取るバッファへのポインタ。
2 番目と 3 番目の GPO ピンが TRUE の場合、戻り値は 6 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル出力オブジェクトを PDO として設定する必要があります。たとえば、ドライバーの 0x60FE (デジタル出力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAllGPO
LabVIEW	HIMC Get Slv All GPO.vi
Python	GetSlvAllGPO

7.5.5 HIMC_GetSlvAllGPIByStartIdx

目的

開始インデックスに基づいてスレーブのすべての汎用入力状態を取得します。

構文

```
int HIMC_GetSlvAllGPIByStartIdx(
        int ctrl_id,
        int slv_slot_id,
        int start_idx
        int *p_state
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
start_idx [in]	汎用入力の開始インデックス
p_state [out]	スレーブの開始インデックスに続くすべての汎用入力状態を受信するバッファへのポインタ。開始インデックスが 8 で、8 番目と 11 番目の GPI ピンが TRUE の場合、戻り値は 9 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル入力オブジェクトを PDO として設定する必要があります。
たとえば、ドライバーの 0x60FD (デジタル入力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAllGPIByStartIdx
LabVIEW	-
Python	GetSlvAllGPIByStartIdx

7.5.6 HIMC_GetSlvAllGPOByStartIdx

目的

開始インデックスに基づいてスレーブのすべての汎用出力状態を取得します。

構文

```
int HIMC_GetSlvAllGPOByStartIdx(
        int ctrl_id,
        int slv_slot_id,
        int start_idx
        int *p_state
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
start_idx [in]	汎用出力の開始インデックス
p_state [out]	スレーブの開始インデックスに続くすべての汎用出力状態を受信するバッファへのポインタ。開始インデックスが 8 で、9 番目と 10 番目の GPO ピンが TRUE の場合、戻り値は 6 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはデジタル出力オブジェクトを PDO として設定する必要があります。
たとえば、ドライバーの 0x60FE (デジタル出力) を PDO として設定します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAllGPOByStartIdx
LabVIEW	-
Python	GetSlvAllGPOByStartIdx

（このページはブランクになっています）

8. AIO 関数

8.1 概要

AIO 機能により、アナログ入力（AI）またはアナログ出力（AO）機能を持つスレーブは、関連パラメーターの読み取りと設定を行うことができます。その中で、HMPL は、デジタルとアナログ間の変換タイプを指定する設定を提供します。詳細な仕様は表 8.1 に示されています。

表 8.1.1 アナログデータ変換タイプの定義

<table>
<tr><th>変換タイプ</th><th>変換の説明</th><th>仕様</th><th>備考</th></tr>
<tr><td>kDAC_NONE</td><td>N/A</td><td>この変換を実行するには、Raw 関数のみを使用してください。</td><td>-</td></tr>
<tr><td>kDAC_N10_P10</td><td>-10~10</td><td>元のアナログ値を -10 ～ 10 に変換します。</td><td rowspan="4">アナログ電圧モジュールに共通</td></tr>
<tr><td>kDAC_P0_P10</td><td>0~10</td><td>元のアナログ値を 0 ～ 10 に変換します。</td></tr>
<tr><td>kDAC_N5_P5</td><td>-5~5</td><td>元のアナログ値を -5～5 に変換します。</td></tr>
<tr><td>kDAC_P0_P5</td><td>0~5</td><td>元のアナログ値を 0～5 に変換します。</td></tr>
<tr><td>kDAC_P0_P20</td><td>0~20</td><td>元のアナログ値を 0～20 に変換します。</td><td rowspan="3">アナログ電流モジュールの共通</td></tr>
<tr><td>kDAC_P4_P20</td><td>4~20</td><td>元のアナログ値を 4～20 に変換します。</td></tr>
<tr><td>kDAC_N20_P20</td><td>-20~20</td><td>元のアナログ値を -20 ～ 20 に変換します。</td></tr>
<tr><td>kDAC_SIGNED</td><td>上位ビットは符号が正か負かを定義します。</td><td>符号付き変換。
6 ビット ±10V アナログ デバイスを例にとると、kDAC_N10_P10 を設定すると、変換テーブルは次のようになります：
<table>
<tr><th>元の値</th><th>返還後の値</th></tr>
<tr><td>65535</td><td>10</td></tr>
<tr><td>49152</td><td>5</td></tr>
<tr><td>32768</td><td>0</td></tr>
<tr><td>16384</td><td>-5</td></tr>
<tr><td>0</td><td>-10</td></tr>
</table>
kDAC_SIGNED | kDAC_N10_P10 を設定すると、変換テーブルは次のようになります:
<table>
<tr><th>元の値</th><th>返還後の値</th></tr>
<tr><td>32767</td><td>10</td></tr>
<tr><td>16384</td><td>5</td></tr>
<tr><td>0</td><td>0</td></tr>
<tr><td>-16384</td><td>-5</td></tr>
<tr><td>-32768</td><td>-10</td></tr>
</table></td><td>変換方法についてはサブデバイスのユーザーマニュアルを参照してください。</td></tr>
</table>

8.2 スレーブ AIO 設定

8.2.1 HIMC_SetSlvAIType

目的

スレーブのアナログ入力値の変換タイプを設定します。

構文

```
int HIMC_SetSlvAIType(
        int ctrl_id,
        int slv_slot_id,
        int ai_idx,
        int range_type
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

Ai idx [in] アナログ入力チャンネル

range_type [in] アナログデータ変換タイプ。詳細は表 8.1 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAOType
LabVIEW	HIMC Set Slv AO Type.vi
Python	SetSlvAOType

8.2.2 HIMC_SetSlvAOType

目的

スレーブのアナログ出力値の変換タイプを設定します。

構文

```
int HIMC_SetSlvAOType(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        int range_type
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ai idx [in] アナログ出力チャンネル

range_type [in] アナログデータ変換タイプ。詳細は表 8.1 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAOType
LabVIEW	HIMC Set Slv AO Type.vi
Python	SetSlvAOType

8.2.3 HIMC_SetSlvAORaw

目的

スレーブのアナログ出力生の値を設定します。

構文

```
int HIMC_SetSlvAORaw(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int ao_raw_val
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とそのスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ao_idx [in] アナログ出力チャンネル

ao_raw_val [in] アナログ出力の元の値。
値の表示範囲を調整するには、セクション 1.5 を参照してデータ型の変換を実行してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはアナログ出力オブジェクトを PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAORaw
LabVIEW	-
Python	SetSlvAORaw

8.2.4 HIMC_SetSlvAO

目的

スレーブのアナログ出力値を設定します。

構文

```
int HIMC_SetSlvAO(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        double ao_val
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とそのスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ao_idx [in] アナログ出力チャンネル

ao_val [in] アナログ出力値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはアナログ出力オブジェクトを PDO として設定し、変換タイプを設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAO
LabVIEW	HIMC Set Slv AO.vi
Python	SetSlvAO

8.3 スレーブ AIO ステータス

8.3.1 HIMC_GetSlvAIType

目的

スレーブのアナログ入力タイプを取得します。

構文

```
int HIMC_GetSlvAIType(
        int ctrl_id,
        int slv_slot_id,
        int ai_idx,
        int *p_ai_type
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とそのスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ai_idx [in] アナログ入力チャンネル

p_ai_type [out] スレーブのアナログ入力タイプを受信するバッファへのポインタ。
詳細については表 8.1 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAIType
LabVIEW	HIMC Get Slv AI Type.vi
Python	GetSlvAIType

8.3.2 HIMC_GetSlvAOType

目的

スレーブのアナログ入力タイプを取得します。

構文

```
int HIMC_GetSlvAOType(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        int *p_ao_type
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とそのスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ao_idx [in] アナログ出力チャンネル

p_ao_type [out] スレーブのアナログ出力タイプを受信するバッファへのポインタ。
詳細については表 8.1 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAOType
LabVIEW	HIMC Get Slv AO Type.vi
Python	GetSlvAOType

8.3.3 HIMC_GetSlvAIRaw

目的

スレーブのアナログ入力の元の値を取得します。

構文

```
int HIMC_GetSlvAIRaw(
    int ctrl_id,
    int slv_slot_id,
    int ai_idx,
    int *p_ai_raw_val
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
ai_idx [in]	アナログ入力チャンネル
p_ai_raw_val [out]	アナログ入力の生の値を受け取るバッファへのポインタ。 値の表示範囲を調整するには、セクション 1.5 を参照してデータ型の変換を実行してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはアナログ入力オブジェクトを PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAIRaw
LabVIEW	-
Python	GetSlvAIRaw

8.3.4 HIMC_GetSlvAI

目的

スレーブのアナログ入力を取得します。

構文

```
int HIMC_GetSlvAI(
        int ctrl_id,
        int slv_slot_id,
        int ai_idx,
        double *p_ai_val
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
ai_idx [in]	アナログ入力チャンネル
p_ai_val [out]	アナログ入力値を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはアナログ入力オブジェクトを PDO として構成し、変換タイプを設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAI
LabVIEW	HIMC Get Slv AI.vi
Python	GetSlvAI

8.3.5 HIMC_GetSlvAORaw

目的

スレーブのアナログ出力生の値を取得します。

構文

```
int HIMC_GetSlvAORaw(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        int *p_ao_raw_val
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
ao_idx [in]	アナログ出力チャンネル
p_ao_raw_val [out]	アナログ出力の元の値を受け取るバッファへのポインタ。 値の表示範囲を調整するには、セクション 1.5 を参照してデータ型の変換を実行してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはアナログ出力オブジェクトを PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAORaw
LabVIEW	-
Python	GetSlvAORaw

8.3.6 HIMC_GetSlvAO

目的

スレーブのアナログ出力値を取得します。

構文

```
int HIMC_GetSlvAO(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        double *p_ao_val
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます
ao_idx [in]	アナログ出力チャンネル
p_ao_val [out]	アナログ出力値を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはアナログ出力オブジェクトを PDO として設定し、変換タイプを設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAO
LabVIEW	HIMC Get Slv AO.vi
Python	GetSlvAO

8.4 スレーブ AO を HIMC 内部バッファ変数にバインド

8.4.1 HIMC_SetSlvAOMonitor

目的

アナログ出力にバインドされるコントローラー変数を設定します。

構文

```
int HIMC_SetSlvAOMonitor(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        int var_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ao_idx [in] アナログ出力チャンネル

var_id [in] コントローラー変数と軸 ID
次の表は、コントローラー変数と軸 ID の定義を示しています。
コントローラー変数の詳細については、セクション 16.1.2 を参照してください。
例: HMPL_AXIS_0 | HMPL_REF_VEL

コントローラー変数	定義	コントローラー変数	定義
HMPL_REF_POS	軸の基準位置	HMPL_POS_FB	軸の位置フィードバック
HMPL_REF_VEL	軸の基準速度	HMPL_VEL_FB	軸の速度フィードバック
HMPL_REF_ACC	軸の基準加速度	HMPL_ACC_FB	軸の加速度フィードバック
		HMPL_CUR_FB	軸の電流フィードバック

軸 ID	定義
HMPL_AXIS_0	軸 0
HMPL_AXIS_1	軸 1
…	…
HMPL_AXIS_15	軸 15

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはアナログ出力オブジェクトを PDO として設定し、変換タイプを設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAOMonitor
LabVIEW	HIMC Set Slv AO Monitor.vi
Python	SetSlvAOMonitor

8.4.2 HIMC_SetSlvAOParam

目的

スレーブのアナログ出力をコントローラー変数にバインドするように設定します。

構文

```
int HIMC_SetSlvAOParam(
    int ctrl_id,
    int slv_slot_id,
    int ao_idx,
    int ao_en_bind,
    double ao_scale,
    double ao_offset
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_slot_id [in]	スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。
ao_idx [in]	アナログ出力チャンネル
ao_en_bind [in]	0: コントローラー変数にバインドされたアナログ出力の機能は OFF（デフォルト）です 1: コントローラー変数にバインドされたアナログ出力の機能がオンになっています
ao_scale [in]	アナログ出力とコントローラー変数のスケール。
ao_offset [in]	アナログ出力とコントローラー変数のオフセット。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAOParam
LabVIEW	HIMC Set Slv AO Param.vi
Python	SetSlvAOParam

8.4.3 HIMC_GetSlvAOScale

目的

アナログ出力とコントローラー変数のスケールを設定します。

構文

```
int HIMC_SetSlvAOScale(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        double *p_ao_scale
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ao_idx [in] アナログ出力チャンネル

p_ao_scale [in] アナログ出力のスケールとコントローラー変数を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetSlvAOScale
LabVIEW	HIMC Set Slv AO Scale.vi
Python	SetSlvAOScale

8.4.4 HIMC_GetSlvAOOffset

目的

スレーブのアナログ出力とコントローラー変数のオフセットを取得します。

構文

```
int HIMC_SetSlvAOOffset(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        double *p_ao_offset
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ao_idx [in] アナログ出力チャンネル

p_ao_offset [in] アナログ出力とコントローラー変数のオフセットを受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAOOffset
LabVIEW	HIMC Get Slv AO Offset.vi
Python	GetSlvAOOffset

8.4.5 HIMC_IsSlvAOBound

目的

スレーブのアナログ出力がコントローラー変数にバインドされているかどうかを照会します。

構文

```
int HIMC_IsSlvAOBound(
        int ctrl_id,
        int slv_slot_id,
        int ao_idx,
        int *p_is_bound
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ao_idx [in] アナログ出力チャンネル

p_ao_bound [in] スレーブのアナログ出力がコントローラー変数にバインドされているかどうかを受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvAOBound
LabVIEW	HIMC Get Slv AO Bound.vi
Python	GetSlvAO Bound

(このページはブランクになっています)

9. ユーザーテーブル関数

9.1 概要

HIMC は、最大 512,000 個の double 型変数データ（500KB）を保存できる空きメモリ空間を提供します。本章で説明する関数を使用することで、ユーザーはメモリ空間にアクセスできます。書き込まれた値は、コントローラーのランダムアクセスメモリ（RAM）に保存されます。関数「HIMC_SaveUserTable」を使用すると、メモリ空間内のユーザーテーブルのデータが HIMC のハードディスク空間に保存されます。HIMC の電源を入れ直した後、「HIMC_LoadUserTable」を使用すると、保存されたデータがユーザーテーブルのメモリ空間に再度コピーされます。

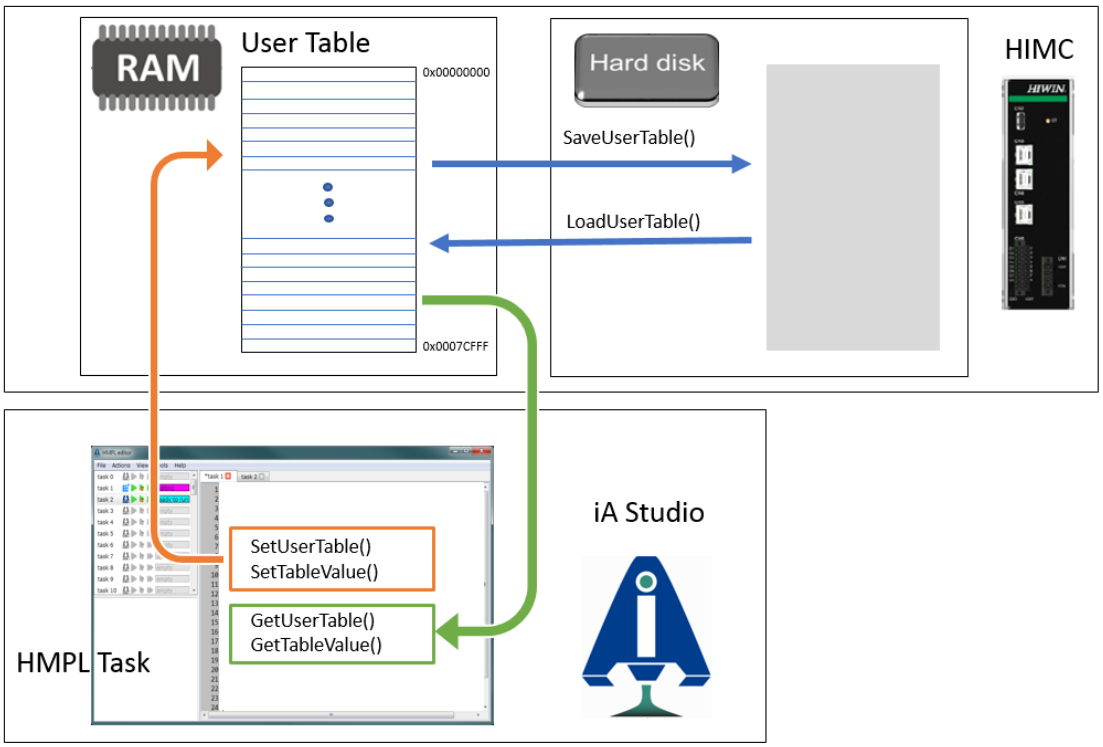


図 9.1.1

注: ユーザーは、iA Studio のテーブル ビューアー (「iA Studio ユーザー ガイド」のセクション 4.11 を参照) を介してユーザー テーブルの変数値にアクセスでき、HIMC のメモリとハード ディスクへの読み込みと保存も行えます。

注意：
動的誤差補正関数で使用される誤差マップは、ユーザーテーブルのメモリ空間に保存されます。動的誤差補正を有効にする場合、他のユーザーテーブル値へのアクセスが、構築された誤差補正値に影響を与えないことを確認する必要があります。

9.2 HIMC_SetUserTable

目的

ユーザー テーブル データをコントローラーに設定します。

構文

```
int HIMC_SetUserTable(
    int     ctrl_id,
    double *p_user_table_data,
    int     start_idx,
    int     number_of_doubles_to_write
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_user_table_data [in] ユーザー テーブルにデータを書き込むためのバッファへのポインタ。

start_idx [in] ユーザー テーブルの開始インデックス

number_of_doubles_to_write [in] 書き込むユーザー テーブルの数

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetUserTable
LabVIEW	HIMC Set User Table.vi
Python	SetUserTable

9.3 HIMC_GetUserTable

目的

コントローラーからユーザー テーブル データを取得します。

構文

```
int HIMC_GetUserTable(
    int     ctrl_id,
    double *p_user_table_data,
    int     start_idx,
    int     number_of_doubles_to_read
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
p_user_table_data [out]	ユーザー テーブルからデータを受信するバッファへのポインタ。
start_idx [in]	ユーザー テーブルの開始インデックス。
number_of_doubles_to_read [in]	読み取るユーザー テーブルの数。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetUserTable
LabVIEW	HIMC Get User Table.vi
Python	GetUserTable

9.4 HIMC_SetTableValue

目的

ユーザー テーブルの特定のインデックスにデータを書き込みます。

構文

```
int HIMC_SetTableValue(
    int     ctrl_id,
    int     index,
    double value
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

index [in] ユーザー テーブルのインデックス

value [in] 入力データ

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetTableValue
LabVIEW	HIMC Set Table Value.vi
Python	SetTableValue

9.5 HIMC_GetTableValue

目的

ユーザー テーブルの特定のインデックスからデータを取得します。

構文

```
int HIMC_GetTableValue(
    int     ctrl_id,
    int     index,
    double *value
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

index [in] ユーザー テーブルのインデックス

value [out] データを受信するバッファへのポインタ

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetTableValue
LabVIEW	HIMC Get Table Value.vi
Python	GetTableValue

9.6 HIMC_SaveUserTable

目的

RAM 内のユーザー テーブル データをメモリに保存します。

構文

```
int HIMC_SaveUserTable(
    int ctrl_id,
    int start_idx,
    int num_data
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

start_idx [in] ユーザー テーブルの開始インデックス

num_data [in] 保存する要素の数

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SaveUserTable
LabVIEW	HIMC Save User Table.vi
Python	SaveUserTable

9.7 HIMC_LoadUserTable

目的

ユーザー テーブル データをメモリから RAM にロードします。

構文

```
int HIMC_LoadUserTable(
    int ctrl_id,
    int start_idx,
    int num_data
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

start_idx [in] ユーザー テーブルの開始インデックス

num_data [in] 保存する要素の数

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LoadUserTable
LabVIEW	HIMC Load User Table.vi
Python	LoadUserTable

10. ポジショントリガー関数

10.1 概要

HIMC の位置トリガー機能は、HIWIN MIKROSYSTEM ドライバーでのみ使用できます。HMPL コマンドを使用して、PT（位置トリガー）関連機能を操作できます。PT 関連機能を操作する前に、HIWIN MIKROSYSTEM または最寄りの販売代理店に、対応するドライバーについてお問い合わせください。

注 1: PT 関連機能を備えた HIWIN MIKROSYSTEM ドライバーを使用するための要件は次のとおりです。
（1）デジタルエンコーダーであること（2）最初にドライバーの原点復帰手順を完了する必要がある
注 2: PT 関連機能はシミュレータおよび仮想軸ではサポートされていません。

10.1.1 PT 変数

PT 関連の機能は、表 10.1.1.1 に記載されている変数に基づいて操作されます。

表 10.1.1.1

名称	タイプ	単位	説明
Status	int	true/false	PT 機能のステータス。PT がまだ機能しているかどうかを示します。
start position	double	mm または deg	PT 機能の開始位置。PT 出力信号列はこの点から開始されます。
end position	double	mm または deg	PT 機能の終了位置。この時点以降、PT 出力信号は送信されません。
interval	double	mm または deg	連続する PT 出力間の位置間隔。
pulse width	int	ns	各 PT 出力信号の幅。 E1 シリーズドライバーの場合、範囲は 20ns～80,000ns で、最小増分は 20ns です。例：20ns、40ns、… 80,000ns。
position array	double	mm または deg	ランダム間隔 PT 関数のトリガー位置。
status array	int	high/low	ランダム間隔 PT 関数のステータス出力。
start index	int	-	ランダム間隔 PT 関数の開始インデックス。
end index	int	-	ランダム間隔 PT 関数の終了インデックス。

図 10.1.1.1 では、極性が「アクティブハイ」に設定されています。

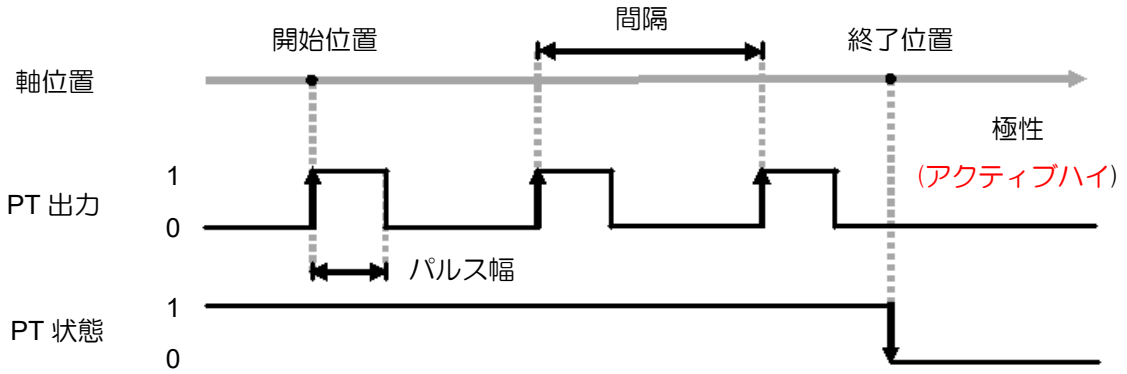


図 10.1.1.1

図 10.1.1.2 では、極性が「アクティブロー」に設定されています。

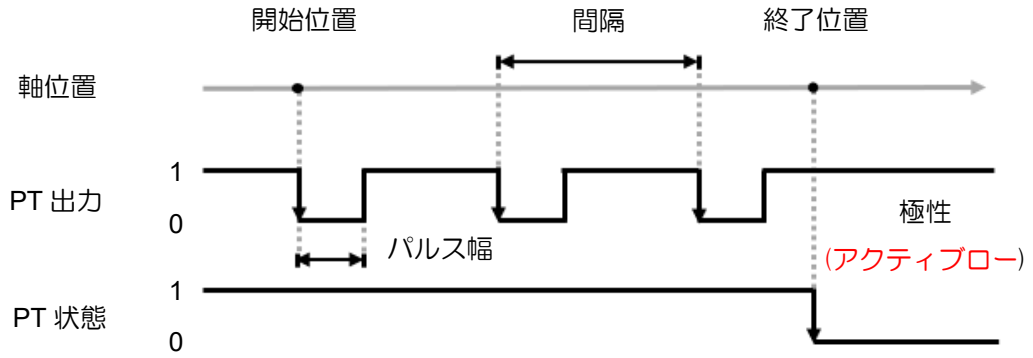


図 10.1.1.2

制限事項：

PT 機能の間隔と軸の速度は、「速度 < 間隔 x 位置サンプリングレート」という式を満たす必要があります。間隔を 100 µm に設定し、位置サンプリングレートを 16 K に設定した場合、速度は 1600 mm/s 未満である必要があります。

注：PT 機能の出力極性（アクティブハイ/ロー）を調整するには、ドライバーの HMI で設定してください。設定を保存した後、ドライの電源を入れ直して出力極性を有効にしてください。

10.1.2 PT 機能の使用フロー

◆ 固定間隔 PT 機能

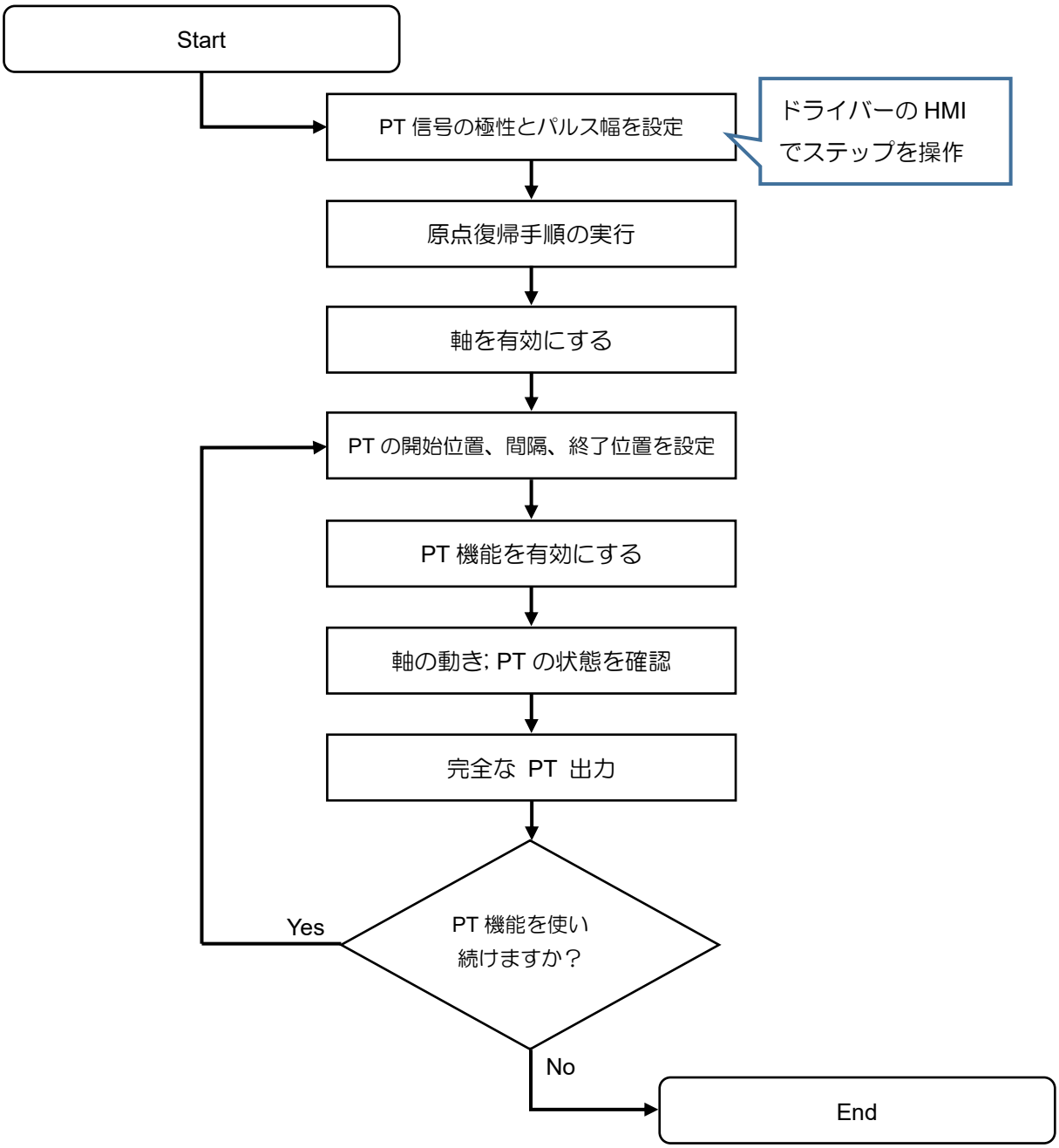


図 10.1.2.1

◆ ランダム間隔 PT 機能

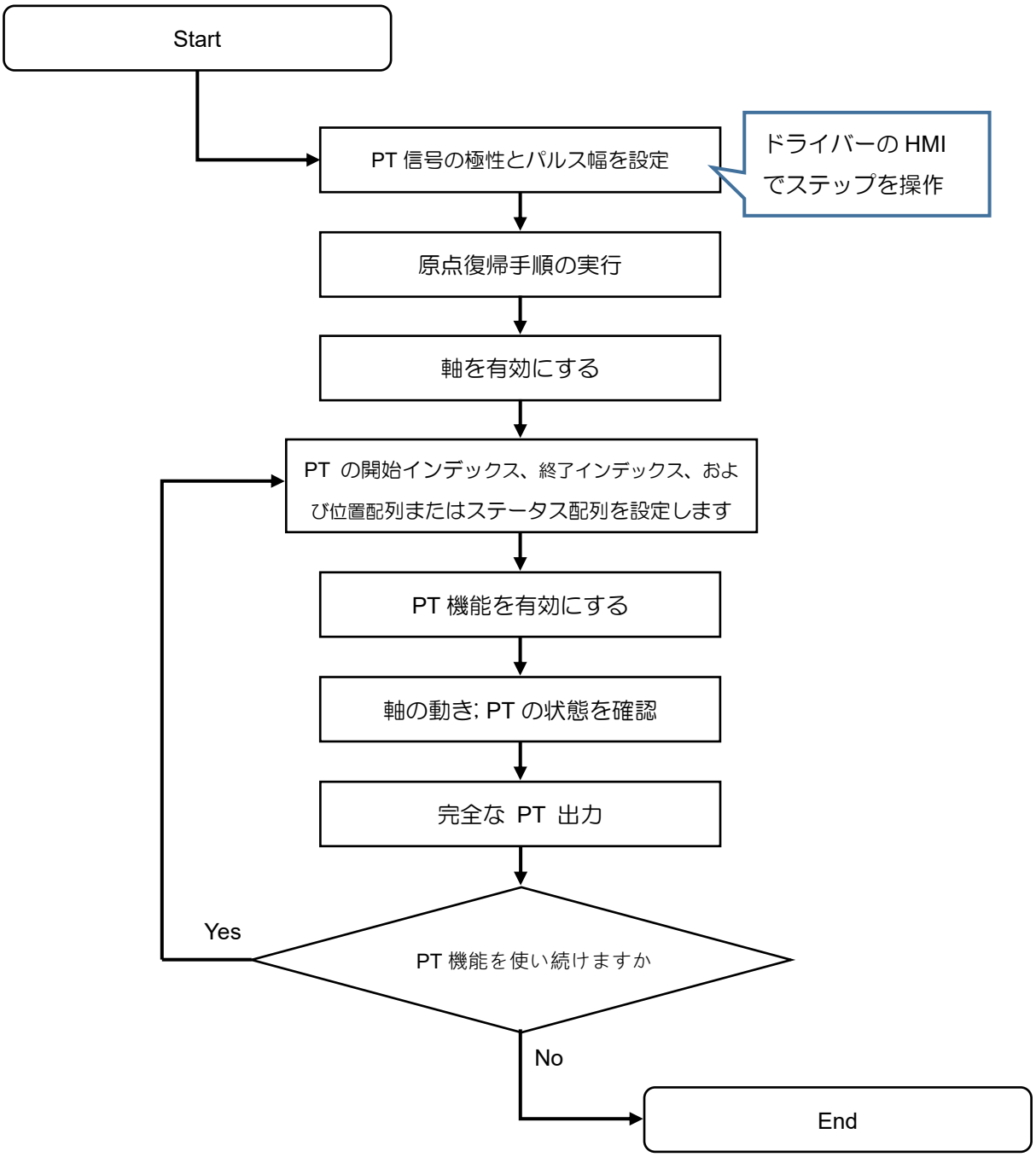


図 10.1.2.2

10.2 HIMC_EnablePT

目的

軸の位置トリガー機能を有効にします。

構文

```
int HIMC_EnablePT(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnablePT
LabVIEW	HIMC Enable PT.vi
Python	EnablePT

10.3 HIMC_DisablePT

目的

軸の位置トリガー機能を無効にします。

構文

```
int HIMC_DisablePT(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisablePT
LabVIEW	HIMC Disable PT.vi
Python	DisablePT

10.4 HIMC_IsPTEnabled

目的

位置トリガー機能が有効かどうかを照会します。

構文

```
int HIMC_IsPTEnabled(
    int ctrl_id,
    int axis_id,
    int *p_is_pt_enabled
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_pt_enabled [out] 位置トリガーの有効状態を受け取るバッファへのポインタ。
軸が「PT 有効」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsPTEnabled
LabVIEW	HIMC Is PT Enabled.vi
Python	IsPTEnabled

10.5 HIMC_SetPosTriggerConfig

目的

軸に位置トリガー構成を設定します。

構文

```
int HIMC_SetPosTriggerConfig(
    int ctrl_id,
    int axis_id,
    PosTriggerPar *pos_trigger_par
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

pos_trigger_par [in] 軸に設定された位置トリガー構成を含むバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPosTriggerConfig
LabVIEW	HIMC Set Pos Trigger Config.vi
Python	SetPosTriggerConfig

10.6 HIMC_SetPT_PosArray

目的

ランダム位置トリガー機能のトリガー位置を設定します。

構文

```
int HIMC_SetPT_PosArray(
    int    ctrl_id,
    int    axis_id,
    int    index,
    double trigger_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
index [in]	トリガー位置のインデックス
trigger_pos [in]	複数のトリガー位置を含む配列 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPT_PosArray
LabVIEW	HIMC Set PT Pos Array.vi
Python	SetPT_PosArray

10.7 HIMC_SetPT_StateArray

目的

ランダム位置トリガー機能のステータス出力を設定します。

構文

```
int HIMC_SetPT_StateArray(
    int ctrl_id,
    int axis_id,
    int index,
    int state
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
index [in]	ステータス出力のインデックス
state [in]	複数のステータス出力を含む配列

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPT_StateArray
LabVIEW	HIMC Set PT State Array.vi
Python	SetPT_StateArray

10.8 HIMC_SetPT_StartIndex

目的

ランダム位置トリガー機能の開始インデックスを設定します。

構文

```
int HIMC_SetPT_StartIndex(
    int ctrl_id,
    int axis_id,
    int index
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
index [in]	開始インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPT_StartIndex
LabVIEW	HIMC Set PT Start Index.vi
Python	SetPT_StartIndex

10.9 HIMC_SetPT_EndIndex

目的

ランダム位置トリガー機能の終了インデックスを設定します。

構文

```
int HIMC_SetPT_EndIndex(
    int ctrl_id,
    int axis_id,
    int index
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
index [in]	終了インデックス.

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetPT_EndIndex
LabVIEW	HIMC Set PT End Index.vi
Python	SetPT_EndIndex

(このページはブランクになっています)

11. タッチプローブ関数

11.1 概要

タッチプローブ機能は、原点復帰手順で使用されるラッチ機能です（AC モーター、ダイレクトドライブモーター、リニアモーターに適用）。エンコーダー入力信号のエッジトリガーにより、エンコーダーの位置フィードバック値を取得します。図 11.1.1 に示すように、ドライバーがエンコーダーのインデックス信号を通過すると、タッチプローブ機能がトリガーされ、インデックス信号の位置が記録されます。タッチプローブ機能を使用すると、ユーザーはコントローラーにタッチプローブ機能がトリガーされたかどうかを問い合わせることができ、コントローラーはラッチによって記録されたインデックス信号の位置を取得できます。

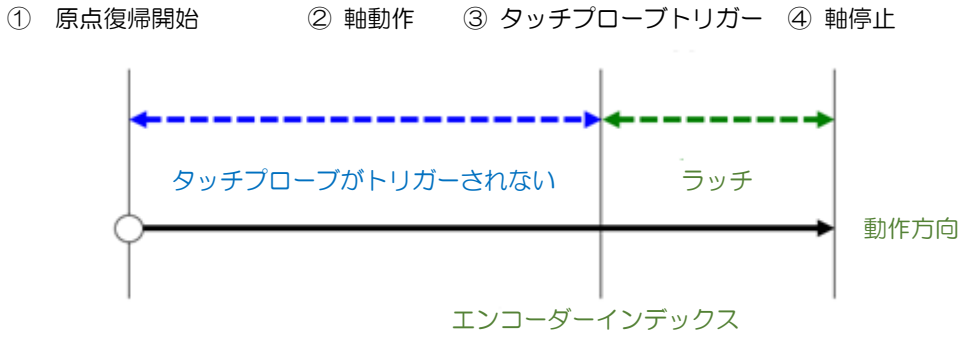


図 11.1.1

注 1：タッチプローブ機能は、ドライバーがサポートするチャンネルとラッチ方向に制限されます。
注 2：タッチプローブ関連の機能は、シミュレータおよび仮想軸ではサポートされていません。

11.2 HIMC_EnableTouchProbe1

目的

軸のタッチプローブ 1 機能を有効にします。

構文

```
int HIMC_EnableTouchProbe1(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableTouchProbe1
LabVIEW	-
Python	EnableTouchProbe1

11.3 HIMC_EnableTouchProbe2

目的

軸のタッチプローブ 2 機能を有効にします。

構文

```
int HIMC_EnableTouchProbe2(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableTouchProbe2
LabVIEW	-
Python	EnableTouchProbe2

11.4 HIMC_DisableTouchProbe1

目的

軸のタッチプローブ 1 機能を無効にします。

構文

```
int HIMC_DisableTouchProbe1(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableTouchProbe1
LabVIEW	-
Python	DisableTouchProbe1

11.5 HIMC_DisableTouchProbe2

目的

軸のタッチプローブ 2 機能を無効にします。

構文

```
int HIMC_DisableTouchProbe2(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableTouchProbe2
LabVIEW	-
Python	DisableTouchProbe2

11.6 HIMC_IsTouchProbe1Enabled

目的

軸のタッチプローブ 1 の有効ステータスを取得します。

構文

```
int HIMC_IsTouchProbe1Enabled(
    int ctrl_id,
    int axis_id,
    int *p_is_probe_enabled
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_probe_enabled [out]	タッチプローブ 1 の有効状態を受け取るバッファへのポインタ。 軸が「タッチプローブ 1 有効」状態の場合、値は 1 になります。 それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsTouchProbe1Enabled
LabVIEW	-
Python	IsTouchProbe1Enabled

11.7 HIMC_IsTouchProbe2Enabled

目的

軸のタッチプローブ 2 の有効ステータスを取得します。

構文

```
int HIMC_IsTouchProbe2Enabled(
    int ctrl_id,
    int axis_id,
    int *p_is_probe_enabled
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_probe_enabled [out]	タッチプローブ 2 の有効状態を受け取るバッファへのポインタ。 軸が「タッチプローブ 2 有効」状態の場合、値は 1 になります。 それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsTouchProbe2Enabled
LabVIEW	-
Python	IsTouchProbe2Enabled

11.8 HIMC_IsTouchProbe1Triggered

目的

軸のタッチプローブ 1 のトリガー ステータスを取得します。

構文

```
int HIMC_IsTouchProbe1Triggered(
    int ctrl_id,
    int axis_id,
    int *p_is_probe_triggered
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_probe_triggered [out]	タッチプローブ 1 のトリガー状態を受け取るバッファへのポインタ。 軸が「タッチプローブ 1 トリガー」状態の場合、値は 1 になります。 それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsTouchProbe1Triggered
LabVIEW	-
Python	IsTouchProbe1Triggered

11.9 HIMC_IsTouchProbe2Triggered

目的

軸のタッチプローブ 2 のトリガー ステータスを取得します。

構文

```
int HIMC_IsTouchProbe2Triggered(
    int ctrl_id,
    int axis_id,
    int *p_is_probe_triggered
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_probe_triggered [out]	タッチプローブ 2 のトリガー状態を受け取るバッファへのポインタ。 軸が「タッチプローブ 2 トリガー」状態の場合、値は 1 になります。 それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsTouchProbe2Triggered
LabVIEW	-
Python	IsTouchProbe2Triggered

11.10 HIMC_GetTouchProbe1Pos

目的

軸のタッチプローブ 1 ラッチ位置を取得します。

構文

```
int HIMC_GetTouchProbe1Pos(
    int     ctrl_id,
    int     axis_id,
    double *p_get_probe_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_get_probe_pos [out]	軸のタッチプローブ 1 ラッチ位置を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetTouchProbe1Pos
LabVIEW	-
Python	GetTouchProbe1Pos

11.11 HIMC_GetTouchProbe2Pos

目的

軸のタッチプローブ 2 ラッチ位置を取得します。

構文

```
int HIMC_GetTouchProbe2Pos(
    int     ctrl_id,
    int     axis_id,
    double *p_get_probe_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_get_probe_pos [out]	軸のタッチプローブ 2 ラッチ位置を受け取るバッファへのポインタ。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetTouchProbe2Pos
LabVIEW	-
Python	GetTouchProbe2Pos

11.12 HIMC_SetTouchProbe1Func

目的

軸のタッチプローブ 1 機能を設定します。

構文

```
int HIMC_SetTouchProbe1Func(
    int ctrl_id,
    int axis_id,
    int tp_input,
    int cont_trigger,
    int detect_edge
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
tp_input [in]	タッチプローブ 1 の信号源 入力範囲: 0 (外部信号)、1 (Z 相)
cont_trigger [in]	タッチプローブ 1 のトリガーモード 入力範囲: 0 (単一トリガー)、1 (連続トリガー)
detect_edge [in]	タッチプローブ 1 のエッジ検出モード 入力範囲: 0 (立ち上がりエッジ検出)、1 (立ち下がりエッジ検出)

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetTouchProbe1Func
LabVIEW	-
Python	SetTouchProbe1Func

11.13 HIMC_SetTouchProbe2Func

目的

軸のタッチプローブ 2 機能を設定します。

構文

```
int HIMC_SetTouchProbe2Func(
    int ctrl_id,
    int axis_id,
    int tp_input,
    int cont_trigger,
    int detect_edge
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
tp_input [in]	タッチプローブ 2 の信号源 入力範囲: 0 (外部信号)、1 (Z 相)
cont_trigger [in]	タッチプローブ 2 のトリガーモード。 入力範囲: 0 (単一トリガー)、1 (連続トリガー)
detect_edge [in]	タッチプローブ 2 のエッジ検出モード。 入力範囲: 0 (立ち上がりエッジ検出)、1 (立ち下がりエッジ検出)

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetTouchProbe2Func
LabVIEW	-
Python	SetTouchProbe2Func

12. ダイナミックエラー補償関数

12.1 概要

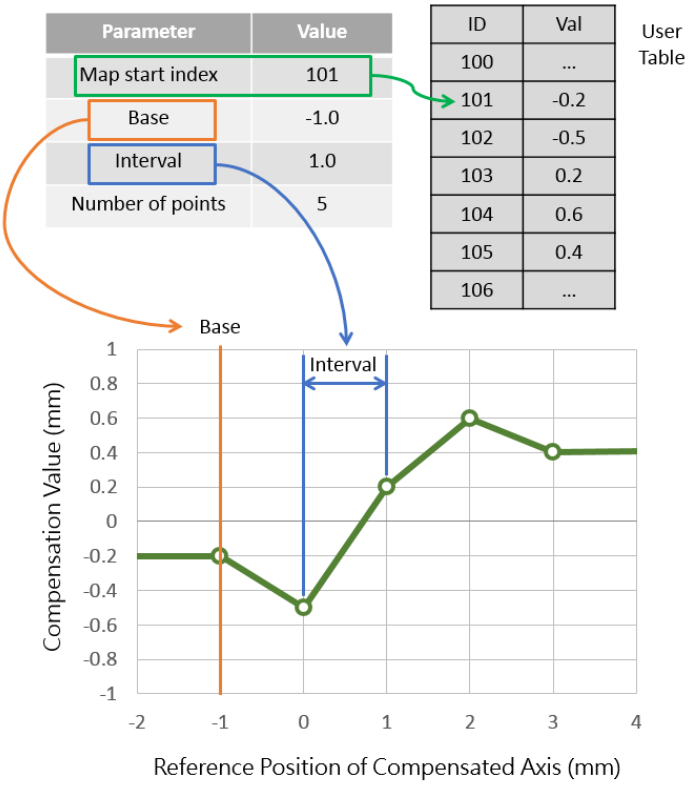
HIMC は、動的な 1D/2D/3D 誤差補正機能を提供します。ユーザーは、関連する誤差測定および計算結果に基づいて、誤差マップを作成し、HIMC 上で設定を行うことができます。設定パラメーターには、補正軸、基準軸、マップポイントの間隔、マップポイントの基準、マップポイントの数、および各マップポイントの補正値が含まれます。補正値の設定には、HIMC ユーザーテーブルのメモリ空間が使用され、誤差マップの最初の ID 位置がユーザーテーブルに提供されます。

注：

(1) ユーザー テーブルの使用方法と説明については、第 9 章を参照してください。

(2) 動的エラー補正を有効にする前に、軸は原点復帰を完了して、補正軸と基準軸の座標位置を固定する必要があります。

ユーザーは、同じ軸を補正軸と基準軸の両方に選択することも、複数の異なる軸を補正軸の基準軸として選択することもできます。たとえば、補正軸を Z 軸とし、基準軸を X 軸と Y 軸とすることができます。補正軸の補正値は、基準軸の移動位置に応じて変化します。軸の移動中、設定されたエラーマップは、補償値が連続的になるように、マップポイント間の補正コマンド値を線形補間で計算します。軸の位置がエラーマップで設定された範囲を超えると、図 12.1.1 に示すように、HIMC は最も近いマップポイントの補正値をその補正コマンドとして使用します。



12.1.1

動的誤差補正を有効にすると、コントローラーの軸制御コマンドにおいて、出力参照位置に補正すべき変位が加算され、既知の測定誤差が除去されます。図 3.1.1 に示すように、この関係は次のように表すことができます：

基準位置 + 位置補正 = 位置出力

動的エラー補正を有効にすると、ユーザーは iA Studio のスコープ マネージャーを介して変数を観察できます。

- 補正値: 軸 → モーション変数 → 位置補正
- 基準位置（補正なし）: 軸 → 動作変数 → 基準位置
- 基準位置（補正あり）: 軸 → 動作変数 → 位置出力

制限：

HIMC では、補正コマンドはプロファイルジェネレータを経由せず、コントローラーによってプリセットされる最大補正値は 1mm です。補正値が 1mm を超える場合、システムはエラーメッセージを表示してユーザーに警告します。

動的誤差補正を有効にする場合、補正対象となる基準座標を固定する必要があります。そのため、ユーザーは軸の原点オフセットを変更できません。

12.2 HIMC_EnableComp

目的

軸の動的誤差補正を有効にします。

構文

```
int HIMC_EnableComp(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

軸が有効な場合、この機能は適用されません。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableComp
LabVIEW	HIMC Enable Comp.vi
Python	EnableComp

12.3 HIMC_DisableComp

目的

軸の動的エラー補正を無効にします。

構文

```
int HIMC_DisableComp(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) 軸の基準位置は電流フィードバックとしてリセットされます。
(2) 軸が有効な場合、この機能は適用されません。
(3) 動的誤差補正の設定がクリアされます。
動的エラー補正を再度有効にするには、設定をリセットする必要があります。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableComp
LabVIEW	HIMC Disable Comp.vi
Python	DisableComp

12.4 HIMC_SetupComp

目的

軸の 1 次元動的誤差補正を設定します。

構文

```
int HIMC_SetupComp(
        int     ctrl_id,
        int     axis_id,
        int     start_idx,
        double  base_val,
        double  interval,
        int     num_pt,
        int     ref_axis_id
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
start_idx [in]	ユーザー テーブル内のマップ ポイントの開始インデックス。
base_val [in]	ベース値（マップ入力の最小値） パラメーター単位: mm または deg
interval [in]	隣接するマップ ポイント間の一定間隔。 パラメーター単位: mm または deg
num_pt [in]	マップ ポイントの数。
ref_axis_id [in]	参照軸のインデックス。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetupComp
LabVIEW	HIMC Setup Comp.vi
Python	SetupComp

12.5 HIMC_SetupComp2D

目的

軸の 2 次元動的誤差補正を設定します。

構文

```
int HIMC_SetupComp2D(
    int     ctrl_id,
    int     axis_id,
    int     start_idx,
    double *base_val,
    double *interval,
    int     *num_pt,
    int     *ref_axis_id
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス.
start_idx [in]	ユーザー テーブル内のマップ ポイントの開始インデックス。
base_val [in]	各次元のベース値（マップ入力の最小値）を含む 2 要素配列へのポインタ。 パラメーター単位: mm または deg
interval [in]	隣接するマップ ポイント間の各次元の一定間隔を含む 2 要素配列へのポインタ。 パラメーター単位: mm または deg
num_pt [in]	各次元のマップ ポイントの数を含む 2 要素配列へのポインタ。
ref_axis_id [in]	参照軸の各次元のインデックスを含む 2 要素配列へのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetupComp2D
LabVIEW	HIMC Setup Comp2D.vi
Python	SetupComp2D

12.6 HIMC_SetupComp3D

目的

軸の 3 次元動的誤差補正を設定します。

構文

```
int HIMC_SetupComp3D(
        int      ctrl_id,
        int      axis_id,
        int      start_idx,
        double *base_val,
        double *interval,
        int      *num_pt,
        int      *ref_axis_id
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
start_idx [in]	ユーザー テーブル内のマップ ポイントの開始インデックス。
base_val [in]	各次元の基本値 (マップ入力の最小値) を含む 3 要素配列へのポインタ。 パラメーター単位: mm または deg
interval [in]	隣接するマップ ポイント間の各次元の一定間隔を含む 3 要素配列へのポインタ。 パラメーター単位: mm または deg
num_pt [in]	各次元のマップ ポイントの数を含む 3 要素配列へのポインタ。
ref_axis_id [in]	参照軸の各次元のインデックスを含む 3 要素配列へのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.4
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetupComp3D
LabVIEW	HIMC Setup Comp3D.vi
Python	SetupComp3D

12.7 HIMC_GetCompPos

目的

コントローラーからドライバーに送信された軸のエラー補正値を取得します。

構文

```
int HIMC_GetCompPos(
    int     ctrl_id,
    int     group_id,
    double *p_comp_pos
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
p_comp_pos [out]	軸の誤差補正値を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.3
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetCompPos
LabVIEW	HIMC Get Comp Pos.vi
Python	GetCompPos

12.8 HIMC_SetCompAlgType

目的

軸の動的誤差補正の補間方法を設定します。

構文

```
int HIMC_SetCompAlgType(
    int ctrl_id,
    int axis_id,
    int alg_type
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

alg_type [in] 動的誤差補正の補間方法。
0: 一次線形補間（初期設定）
1: 3 次スプライン補間

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

3 次元動的誤差補正では、3 次スプライン補間はサポートされません。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetCompAlgType
LabVIEW	HIMC Set Comp Alg Type.vi
Python	SetCompAlgType

(このページはブランクになっています)

13. フィルター機能

13.1 概要

フィルター関数は、プロファイルジェネレータの位置コマンドを修正するために使用されます。現在、HMPL は、スムージング時間、振動抑制フィルター（VSF）、入力シェーピングフィルター（InShape）の 3 種類のフィルターを提供しています。

Smooth Time はモーターを滑らかに加速させ、滑らかな動きを実現します。一方、VSF と InShape は、動作中のモーターの振動（特に機構の負荷が片持ちの場合）を抑制します。「周波数」と「減衰比」を調整することで、振動抑制効果を得ることができます。

VSF と InShape は同時に使用することはできませんが、どちらか一方をスムーズに使用することは可能です。

また、協調動作においては、Axis InShape 機能は役に立ちません。振動を抑制するには、Group InShape 機能を使用する必要があります。

注: フィルターを使用すると、移動時間が長くなり、安定時間が短くなります。

13.2 HIMC_EnableAxisVsf

目的

軸の VSF フィルターを有効にします。

構文

```
int HIMC_EnableAxisVsf(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能はモーターが動作中の場合は使用できません。動作中の場合、モーターが予期しない振動を発生する可能性があります。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableAxisVsf
LabVIEW	HIMC Enable Axis Vsf.vi
Python	EnableAxisVsf

13.3 HIMC_DisableAxisVsf

目的

軸の VSF フィルターを無効にします。

構文

```
int HIMC_DisableAxisVsf(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableAxisVsf
LabVIEW	HIMC Disable Axis Vsf.vi
Python	DisableAxisVsf

13.4 HIMC_SetAxisVsf

目的

VSF フィルターの軸のパラメーターを設定します。

構文

```
int HIMC_SetAxisVsf(
    int ctrl_id,
    int axis_id,
    double frequency,
    double damping_ratio
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
frequency [in]	システム周波数。 パラメーター単位: Hz 入力範囲: 0.1 ~ 200
damping_ratio [in]	減衰比。 入力範囲: 0.7 ~ 1.5 (1.0 が推奨)

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能はモーターが動作中の場合は使用できません。動作中の場合、モーターが予期しない振動を発生する可能性があります。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetAxisVsf
LabVIEW	HIMC Set Axis Vsf.vi
Python	SetAxisVsf

13.5 HIMC_EnableAxisInShape

目的

軸の InShape フィルターを有効にします。

構文

```
int HIMC_EnableAxisInShape(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能はモーターが動作中の場合は使用できません。動作中の場合、モーターが予期しない振動を発生する可能性があります。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableAxisInshape
LabVIEW	HIMC Enable Axis In Shape.vi
Python	EnableAxisInShape

13.6 HIMC_DisableAxisInShape

目的

軸の InShape フィルターを無効にします。

構文

```
int HIMC_DisableAxisInShape(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableAxisInshape
LabVIEW	HIMC Disable Axis In Shape.vi
Python	DisableAxisInShape

13.7 HIMC_SetAxisInShape

目的

InShape フィルターの軸のパラメーターを設定します。

構文

```
int HIMC_SetAxisInShape(
    int ctrl_id,
    int axis_id,
    double frequency,
    double damping_ratio,
    ShaperMode shaper_type
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
frequency [in]	システム周波数。 パラメーター単位: Hz 入力範囲: 3.0 ~ 300
damping_ratio [in]	減衰比。 入力範囲: 0.0 ~ 0.3
shaper_type [in]	シェーパー タイプには Shaper_Normal と Shaper_Robust の 2 つがあります。Shaper_Robust は Shaper_Normal よりも堅牢ですが、Shaper_Normal は振動を抑制するのに十分な強度を持っています。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) この機能はモーターの回転中は使用できません。回転中の場合、モーターが予期せぬ振動を発生する可能性があります。
(2) 周波数と減衰比の初期値はそれぞれ 5.5Hz と 0.03 です。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetAxisInshape
LabVIEW	HIMC Set Axis In Shape.vi
Python	SetAxisInShape

13.8 HIMC_EnableGrpInShape

目的

軸グループの InShape フィルターを有効にします。

構文

```
int HIMC_EnableGrpInShape(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能はモーターが動作中の場合は使用できません。動作中の場合、モーターが予期しない振動を発生する可能性があります。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_EnableGrpInShape
LabVIEW	HIMC Enable Grp In Shape.vi
Python	EnableGrpInShape

13.9 HIMC_DisableGrpInShape

目的

軸グループの InShape フィルターを無効にします。

構文

```
int HIMC_DisableGrpInShape(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_DisableGrpInShape
LabVIEW	HIMC Disable Grp In Shape.vi
Python	DisableGrpInShape

13.10 HIMC_SetGrpInShape

目的

軸グループの InShape フィルターのパラメーターを設定します。

構文

```
int HIMC_SetGrpInShape(
    int ctrl_id,
    int group_id,
    double frequency,
    double damping_ratio,
    ShaperMode shaper_type
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
group_id [in]	軸グループインデックス
frequency [in]	システム周波数。 パラメーター単位: Hz 入力範囲: 3.0 ~ 300
damping_ratio [in]	減衰比。 入力範囲: 0.0 ~ 0.3
shaper_type [in]	シェーパー タイプには Shaper_Normal と Shaper_Robust の 2 つがあります。Shaper_Robust は Shaper_Normal よりも堅牢ですが、Shaper_Normal は振動を抑制するのに十分な強度を持っています。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

(1) この機能はモーターの回転中は使用できません。回転中の場合、モーターが予期せぬ振動を発生する可能性があります。
(2) 周波数と減衰比の初期値はそれぞれ 5.5Hz と 0.03 です。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetGrpInShape
LabVIEW	HIMC Set Grp In Shape.vi
Python	SetGrpInShape

14. HMPL タスク関数

14.1 概要

HIMC には、アプリケーションに基づいたモーションプロファイルコマンドを練習するための 64 個の HMPL タスクが組み込まれています。任意の HMPL タスクにおいて、HMPL タスク機能を使用して他の HMPL タスクを開始または停止できます。HMPL タスクが実行中の場合、再実行を指示することはできません。タスクの実行が完了し、「停止」状態になるまで待つ必要があります。ただし、HMPL タスクが現在実行中かどうかを照会し、アプリケーションに応じて複数の HMPL タスクの順序を制御することは可能です。

14.2 HIMC_StartTask

目的

HMPL タスクの実行を開始します。

構文

```
int HIMC_StartTask(
    int ctrl_id,
    int task_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

task_id [in] HMPL タスク ID

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_StartTask
LabVIEW	HIMC Start Task.vi
Python	StartTask

14.3 HIMC_StartTaskFunc

目的

HMPL タスク内の関数の実行を開始します。

構文

```
int HIMC_StartTaskFunc(
    int ctrl_id,
    int task_id,
    char *func_name
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

task_id [in] HMPL タスク ID

func_name [in] HMPL タスク内の関数名を格納するバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_StartTaskFunc
LabVIEW	HIMC Start Task Func.vi
Python	StartTaskFunc

14.4 HIMC_StopTask

目的

HMPL タスクの実行を停止します。

構文

```
int HIMC_StopTask(
    int ctrl_id,
    int task_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

task_id [in] HMPL タスク ID

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_StopTask
LabVIEW	HIMC Stop Task.vi
Python	StopTask

14.5 HIMC_StopAllTask

目的

すべての HMPL タスク (呼び出し元を含む) の実行を停止します。

構文

```
int HIMC_StopAllTask(
    int ctrl_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_StopAllTask
LabVIEW	HIMC Stop All Task.vi
Python	StopAllTask

14.6 HIMC_IsTaskStop

目的

HMPL タスクの実行が停止されているかどうかを照会します。

構文

```
int HIMC_IsTaskStop(
    int ctrl_id,
    int task_id,
    int *isStop
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

task_id [in] HMPL タスク ID

isStop [out] HMPL タスクのステータスを受け取るバッファへのポインタ。
HMPL タスクが「TaskStop」状態の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsTaskStop
LabVIEW	HIMC Is Task Stop.vi
Python	IsTaskStop

14.7 HIMC_LoadHMPLTask

目的

HMPL ファイルをコントローラーにロードします。

構文

```
int HIMC_LoadHMPLTask(
    int ctrl_id,
    int task_id,
    const char *file_name
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

task_id [in] HMPL タスク ID

file_name [in] ロードする HMPL ファイル パスを格納するバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能は、「iA_Studio」フォルダ内の実行ファイル「HMPL_compiler.exe」と組み合わせて使用する必要があります。この実行ファイルをコピーし、ユーザー開発プログラムの「bin¥Debug」フォルダと「bin¥Release」フォルダに配置してください。

要件

サポートされる最小バージョン	iA Studio 2.0
Executable	HMPL_compiler.exe
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_LoadHMPLTask
LabVIEW	HIMC Load HMPL Task.vi
Python	LoadHMPLTask

15. コールバック関数

15.1 HIMC_SetHmplEvtCallback

目的

HMPL タスクによって送信されたイベントをキャプチャするためのコールバック関数を登録します。

構文

```
int HIMC_SetHmplEvtCallback(
    int ctrl_id,
    HMPLEventCBFuncPtr hmpl_event_cb_func_ptr
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

hmpl_event_cb_func_ptr [in] コールバック関数へのポインタ
関数のプロトタイプは「void func(int arg)」です。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHmplEvtCallback
LabVIEW	-
Python	SetHmplEvtCallback

15.2 HIMC_SetErrorCallback

目的

コントローラーから送信されたエラー ID をキャプチャするためのコールバック関数を登録します。

構文

```
int HIMC_SetErrorCallback(
    int ctrl_id,
    HimcErrorCBFuncPtr himc_error_cb_func_ptr
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
himc_error_cb_func_ptr [in]	コールバック関数へのポインタ 関数のプロトタイプは「void func(int arg)」です。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

エラー ID については、「ai Studio ユーザーガイド」の第 5 章を参照してください。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetErrorCallback
LabVIEW	-
Python	SetErrorCallback

15.3 HIMC_SetHmplMsgEvtCallback

目的

HMPL タスクによって送信された文字列メッセージをキャプチャするためのコールバック関数を登録します。

構文

```
int HIMC_SetHmplMsgEvtCallback(
    int ctrl_id,
    HMPLMsgEventCBFuncPtr hmpl_msg_cb_func_ptr
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
hmpl_msg_cb_func_ptr [in]	コールバック関数へのポインタ 関数のプロトタイプは「void func(const char *text)」です。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

文字列メッセージの最大長は 128 文字です。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHmplMsgEvtCallback
LabVIEW	-
Python	SetHmplMsgEvtCallback

16. 変数と関数演算機能

16.1 概要

HIMC は、ドライバーおよびコントローラーの変数操作機能を提供します。ドライバーの変数操作では、ドライバーのオブジェクトディクショナリインデックスを指定して変数の値にアクセスし、CoE 通信を介してデータ交換を行う必要があります。コントローラーの変数操作では、コントローラーの変数 ID リストに基づいて、特定の変数のアドレス指定 ID を指定してアクセスする必要があります。コントローラーのパラメーターの定義は、16.1.1 節に記載されています。

注：
特別な用途の要件がない限り、ユーザーは関連する HMI および機能を使用して関連するシステム変数にアクセスすることをお勧めします。変数操作機能を使用する場合は、変数へのアクセスと値の入力の安全性を確保する必要があります。

16.1.1 コントローラー変数リスト

HIMC は、コントローラー変数のアドレス指定 ID として 32 ビットを使用します。その型は 0x□□□□□□□□で、「0x」は値が 16 進数であることを示します。変数操作関数を使用することで、ユーザーは HIMC が提供するシステム変数、軸変数、および軸グループ変数にアクセスできます。アドレス指定 ID の規則は以下のように説明されます：

1. アドレス ID の 1 番目と 2 番目の値は「コントローラー変数のカテゴリ」を示します。0x00□□□□□□はシステム変数に属し、0x83□□□□□□は軸変数に属し、0x82□□□□□□は軸グループ変数に属します。

2. アドレス ID の 3 番目と 4 番目の値は「軸 ID または軸グループ ID」を示します。例えば、軸変数 0x8302□□□□は軸インデックス 02 を格納する変数であり、軸グループ変数 0x8201□□□□は軸インデックス 01 を格納する変数です。

3. アドレス ID の 5 番目から 8 番目の値は、「コントローラーのシステム、軸、または軸グループ変数のアドレス指定位置」を示します。パラメーターリストと説明については、表 16.1.1.1 から表 16.1.1.3 を参照してください。

表 16.1.1.1

システム変数		
アドレス ID	変数名	説明
0x0000012c	HCV_ID_fclk	システム実行クロック（250us ごとに 1 カウント増加）
0x0000012e	HCV_ID_timeInMs	システム実行時間（ms）
0x000007d0	HCV_ID_user_table	ユーザーが自由に使用できる double[512000]配列変数
0x00002328	HCV_ID_ltest0	ユーザーが自由に使用できる int 変数
0x00002329	HCV_ID_ltest1	ユーザーが自由に使用できる int 変数
0x0000232a	HCV_ID_ltest2	ユーザーが自由に使用できる int 変数
0x0000232b	HCV_ID_ltest3	ユーザーが自由に使用できる int 変数
0x0000232c	HCV_ID_ltest4	ユーザーが自由に使用できる int 変数
0x0000232d	HCV_ID_ltest5	ユーザーが自由に使用できる int 変数
0x0000232e	HCV_ID_ltest6	ユーザーが自由に使用できる int 変数
0x0000232f	HCV_ID_ltest7	ユーザーが自由に使用できる int 変数
0x00002330	HCV_ID_ltest8	ユーザーが自由に使用できる int 変数
0x00002331	HCV_ID_ltest9	ユーザーが自由に使用できる int 変数
0x0000235a	HCV_ID_dtest0	ユーザーが自由に使用できる double 変数
0x0000235b	HCV_ID_dtest1	ユーザーが自由に使用できる double 変数
0x0000235c	HCV_ID_dtest2	ユーザーが自由に使用できる double 変数
0x0000235d	HCV_ID_dtest3	ユーザーが自由に使用できる double 変数
0x0000235e	HCV_ID_dtest4	ユーザーが自由に使用できる double 変数

システム変数		
アドレス ID	変数名	説明
0x0000235f	HCV_ID_dtest5	ユーザーが自由に使用できる double 変数
0x00002360	HCV_ID_dtest6	ユーザーが自由に使用できる double 変数
0x00002361	HCV_ID_dtest7	ユーザーが自由に使用できる double 変数
0x00002362	HCV_ID_dtest8	ユーザーが自由に使用できる double 変数
0x00002363	HCV_ID_dtest9	ユーザーが自由に使用できる double 変数
0x0000238c	HCV_ID_mtest	ユーザーが自由に使用できる double[10]配列変数

表 16.1.1.2

軸変数		
アドレス ID	変数名	説明
0x83□□0015	HCV_ID_motion_type	モーションタイプ
0x83□□0033	HCV_ID_pos_tr	ポジションウィンドウ
0x83□□0034	HCV_ID_pos_tr_t	ポジションウィンドウ時間
0x83□□0065	HCV_ID_sw_RL	ソフトウェア右側リミット
0x83□□0066	HCV_ID_sw_LL	ソフトウェア左側リミット
0x83□□0067	HCV_ID_vel_lim	最大速度リミット
0x83□□0068	HCV_ID_acc_lim	最大加速度リミット
0x83□□0069	HCV_ID_dec_lim	最大減速度リミット
0x83□□0079	HCV_ID_max_pos_err	位置誤差リミット
0x83□□007a	HCV_ID_max_comp_lim	ポジション補償リミット
0x83□□00a0	HCV_ID_home_status	原点復帰状態
0x83□□00a1	HCV_ID_home_method	原点復帰方式
0x83□□00a2	HCV_ID_home_fast_vel	高速原点復帰速度
0x83□□00a3	HCV_ID_home_slow_vel	低速原点復帰速度
0x83□□00a4	HCV_ID_home_timeout	原点復帰遅延時間
0x83□□00a5	HCV_ID_home_acc	原点復帰加速度
0x83□□00a6	HCV_ID_home_offset	原点復帰位置オフセット
0x83□□00d3	HCV_ID_max_vel	目標速度
0x83□□00d4	HCV_ID_max_acc	目標加速度
0x83□□00d5	HCV_ID_max_dec	目標減速度
0x83□□00d7	HCV_ID_sm_factor	スムーズタイム
0x83□□00db	HCV_ID_vel_scale	速度スケール (0~100)
0x83□□00dd	HCV_ID_p2p_del	P2P モーション待ち時間
0x83□□00de	HCV_ID_p2p_pos1	P2P ポジション 1
0x83□□00df	HCV_ID_p2p_pos2	P2P ポジション 2
0x83□□00e0	HCV_ID_p2p_repeat	P2P モーションを繰り返す
0x83□□00e1	HCV_ID_rlt_dist	相対移動距離

軸変数		
アドレス ID	変数名	説明
0x83□□00e2	HCV_ID_en_motionManager	モーションマネージャーのモーション軸選択
0x83□□00e3	HCV_ID_acc_time	加速時間
0x83□□00e4	HCV_ID_dec_time	減速時間
0x83□□00e9	HCV_ID_map_io_type	誤差補正型
0x83□□0117	HCV_ID_rollover_turns	ロールオーバーターン
0x83□□0119	HCV_ID_rollover_val	ロールオーバー値
0x83□□0193	HCV_ID_gant_pair	ガントリー構成のガントリーペア ID
0x83□□01f7	HCV_ID_en_delay	軸有効化のタイムアウト
0x83□□01ff	HCV_ID_fb_ratio_pos	ドライバーの位置分解能；長さの単位 (分子)
0x83□□0200	HCV_ID_fb_ratio_cnt	ドライバーの位置分解能。単位：カウント (分母)
0x83□□0209	HCV_ID_fb_curr_ratio_curr	ドライバーの電流分解能；電流単位 (分子)
0x83□□020a	HCV_ID_fb_curr_ratio_cnt	ドライバーの電流分解能；単位：カウント （分母）
0x83□□0213	HCV_ID_rotor_inertia	モーターのローター慣性比
0x83□□0214	HCV_ID_force_constant	モーターのトルク定数
0x83□□0263	HCV_ID_last_err	軸エラーコード
0x83□□03c2	HCV_ID_gear_ratio	ギア比

注: 記号□□は 16 進形式の軸 ID です。例えば、01 は軸インデックス 01、0f は軸インデックス 15 を表します。

表 16.1.1.3

軸グループ変数		
アドレス ID	変数名	説明
0x82□□0002	HCV_ID_grp_num_axis	軸グループの軸数
0x82□□00c9	HCV_ID_grp_lin_vel_lim	軸グループの直線運動の速度リミット
0x82□□00ca	HCV_ID_grp_lin_acc_lim	軸グループの直線運動の加速度リミット
0x82□□00cb	HCV_ID_grp_lin_dec_lim	軸グループの直線運動の減速度リミット
0x82□□00d4	HCV_ID_grp_ang_vel_lim	軸グループの回転運動の速度リミット
0x82□□00d5	HCV_ID_grp_ang_acc_lim	軸グループの回転運動の加速度リミット
0x82□□00d6	HCV_ID_grp_ang_dec_lim	軸グウループの回転運動の減速度リミット
0x82□□00d0	HCV_ID_grp_lin_vel	軸グループの目標速度
0x82□□00d1	HCV_ID_grp_lin_acc	軸グループの目標加速度
0x82□□00d2	HCV_ID_grp_lin_dec	軸グループの目標減速度
0x82□□00d3	HCV_ID_grp_lin_sf	軸グループのスムーズ時間
0x82□□00f0	HCV_ID_grp_lin_acc_time	軸グループの目標加速時間
0x82□□00f1	HCV_ID_grp_lin_dec_time	軸グループの目標減速時間
0x82□□00e7	HCV_ID_grp_ang_vel	軸グループの回転運動の目標速度
0x82□□00e8	HCV_ID_grp_ang_acc	軸グループの回転動作の目標加速度
0x82□□00e9	HCV_ID_grp_ang_dec	軸グループの回転動作の目標減速度

軸グループ変数		
アドレス ID	変数名	説明
0x82□□00ea	HCV_ID_grp_ang_sf	軸グループの回転動作のスムーズ時間
0x82□□00eb	HCV_ID_grp_ang_acc_time	軸グループの回転動作の目標加速時間
0x82□□00ec	HCV_ID_grp_ang_dec_time	軸グループの回転動作の目標減速時間
0x82□□00e4	HCV_ID_grp_coord_sys	軸グループの座標系
0x82□□00e5	HCV_ID_grp_buffer_mode	軸グループのバッファモード
0x82□□00e6	HCV_ID_grp_trans_mode	軸グループの遷移モード
0x82□□00e7	HCV_ID_grp_trans_vel	軸グループの遷移速度
0x82□□00e8	HCV_ID_grp_trans_dis	軸グループの遷移距離
0x82□□00f6	HCV_ID_grp_trans_dev	軸グループの遷移偏差
0x82□□00f7	HCV_ID_grp_trans_curvature	軸グループの遷移曲率
0x82□□00eb	HCV_ID_grp_vel_scale	軸グループの速度スケール（0～100）
0x82□□00da	HCV_ID_grp_shaper_fr	InShape フィルターの軸グループの頻度
0x82□□00db	HCV_ID_grp_shaper_xi	InShape フィルターの軸グループの減衰率
0x82□□038f	HCV_ID_grp_last_err	軸グループエラーコード

注: 記号□□は 16 進形式の軸グループ ID です。例えば、01 は軸グループインデックス 01、0f は軸グループインデックス 15 を表します。

16.2 ドライバーの可変動作

16.2.1 HIMC_ReadSDO

目的

SDO を介してスレーブのオブジェクト値を読み取ります。

構文

```
int HIMC_ReadSDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    int size_bytes,
    int64_t *value
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_id [in]	スレーブインデックス
obj_index [in]	スレーブ オブジェクトのインデックス
obj_subindex [in]	スレーブ オブジェクトのサブインデックス
obj_subindex [in]	スレーブ オブジェクトのバイト長
value [out]	読み取ったオブジェクトの値を格納するバッファへのポインタ。 値の表示範囲を調整するには、セクション 1.5 を参照してデータ型の変換を実行してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ReadSDO
LabVIEW	HIMC Read SDO.vi
Python	ReadSDO

16.2.2 HIMC_WriteSDO

目的

SDO を介してスレーブのオブジェクト値を書き込みます。

構文

```
int HIMC_WriteSDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    int size_bytes,
    int64_t value
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_id [in]	スレーブインデックス
obj_index [in]	スレーブ オブジェクトのインデックス
obj_subindex [in]	スレーブ オブジェクトのサブインデックス
size_bytes [in]	スレーブ オブジェクトのバイト長
value [in]	書き込むオブジェクトの値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_Write SDO.vi
LabVIEW	HIMC Write SDO.vi
Python	WriteSDO

16.2.3 HIMC_ReadPDO

目的

PDO を介してスレーブの PDO オブジェクト値を読み取ります。

構文

```
int HIMC_ReadPDO(
        int ctrl_id,
        int slv_id,
        int obj_index,
        int obj_subindex,
        int64_t *value
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_id [in]	スレーブインデックス
obj_index [in]	スレーブ オブジェクトのインデックス
obj_subindex [in]	スレーブ オブジェクトのサブインデックス
value [out]	読み取ったオブジェクトの値を格納するバッファへのポインタ。 値の表示範囲を調整するには、セクション 1.5 を参照してデータ型の変換を実行してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ReadPDO
LabVIEW	HIMC Read PDO.vis
Python	ReadPDO

16.2.4 HIMC_WritePDO

目的

PDO を介してスレーブの PDO オブジェクト値を書き込みます。

構文

```
int HIMC_WritePDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    int64_t value
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_id [in]	スレーブインデックス
obj_index [in]	スレーブ オブジェクトのインデックス
obj_subindex [in]	スレーブ オブジェクトのサブインデックス
value [in]	書き込むオブジェクトの値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_WritePDO
LabVIEW	HIMC Write PDO.vi
Python	WritePDO

16.2.5 HIMC_ForceWritePDO

目的

PDO を介してスレーブの PDO オブジェクトを強制的に書き込みます。

構文

```
int HIMC_ForceWritePDO(
    int ctrl_id,
    int slv_id,
    int obj_index,
    int obj_subindex,
    int64_t value
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_id [in]	スレーブインデックス
obj_index [in]	スレーブ オブジェクトのインデックス
obj_subindex [in]	スレーブ オブジェクトのサブインデックス
value [in]	書き込むオブジェクトの値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ForceWritePDO
LabVIEW	HIMC Force Write PDO.vi
Python	ForceWritePDO

16.2.6 HIMC_ReleasePDO

目的

強制的に書き込まれた PDO オブジェクトを解放し、HIMC_ForceWritePDO で使用します。

構文

```
int HIMC_ReleasePDO(
        int ctrl_id,
        int slv_id,
        int obj_index,
        int obj_subindex
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
slv_id [in]	スレーブインデックス.
obj_index [in]	スレーブ オブジェクトのインデックス
obj_subindex [in]	スレーブ オブジェクトのサブインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ReleasePDO
LabVIEW	HIMC Release PDO.vi
Python	ReleasePDO

16.3 コントローラー変数操作

16.3.1 HIMC_GetVariableByID

目的

ID によってコントローラーの変数値を取得します。

構文

```
int HIMC_GetVariableByID(
    int     ctrl_id,
    int     var_id,
    double *p_val
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

var_id [in] HIMC コントローラー変数 ID
定義についてはセクション 16.1.1 を参照してください。

p_val [out] 変数の値を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetVariableByID
LabVIEW	HIMC Get Variable By ID.vi
Python	GetVariableByID

16.3.2 HIMC_SetVariableByID

目的

コントローラーの変数値を ID で設定します。

構文

```
int HIMC_SetVariableByID(
    int    ctrl_id,
    int    var_id,
    double val
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

var_id [in] HIMC コントローラー変数 ID
定義についてはセクション 16.1.1 を参照してください。

val [in] 変数の新しい値

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetVariableByID
LabVIEW	HIMC Set Variable By ID.vi
Python	SetVariableByID

16.3.3 HIMC_GetVariableListByID

目的

ID によってコントローラーのリスト変数の値を取得します。

構文

```
int HIMC_GetVariableListByID(
    int     ctrl_id,
    int     *p_var_id,
    int     num,
    double *p_val
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_var_id [in] リスト変数の HIMC コントローラー変数 ID を格納するバッファへのポインタ。
定義についてはセクション 16.1.1 を参照してください。

num [in] 変数の数

p_val [out] リスト変数の値を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetVariableListByID
LabVIEW	HIMC Get Variable List By ID.vi
Python	GetVariableListByID

16.3.4 HIMC_SetVariableListByID

目的

ID 別にコントローラーのリスト変数に値を設定します。

構文

```
int HIMC_SetVariableListByID(
        int     ctrl_id,
        int     *p_var_id,
        int     num,
        double *p_val
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_var_id [in] リスト変数の HIMC コントローラー変数 ID を格納するバッファへのポインタ。
定義についてはセクション 16.1.1 を参照してください。

num [in] 変数の数

p_val [in] リスト変数の新しい値を格納するバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetVariableListByID
LabVIEW	HIMC Set Variable List By ID.vi
Python	SetVariableListByID

16.3.5 HIMC_GetGlobalVariables

目的

コントローラーのリストグローバル変数の値を取得します。

構文

```
int HIMC_GetGlobalVariables(
    int      ctrl_id,
    char     **pp_var_name_array,
    int      length,
    double *p_output_array
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
pp_var_name_array [in]	リストグローバル変数の名前を格納するバッファへのポインタ。
length [in]	グローバル変数の数
p_output_array [out]	リストグローバル変数の値を受け取るバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Executable	HMPL_compiler.exe
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGlobalVariables
LabVIEW	HIMC Get Global Variables.vi
Python	GetGlobalVariables

16.3.6 HIMC_SetGlobalVariables

目的

コントローラーのグローバル変数をリストするための値を設定します。

構文

```
int HIMC_SetGlobalVariables(
    int      ctrl_id,
    char     **pp_var_name_array,
    int      length,
    double *p_input_array
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
pp_var_name_array [in]	リストグローバル変数の名前を格納するバッファへのポインタ。
length [in]	グローバル変数の数
p_input_array [in]	リストグローバル変数の新しい値を格納するバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Executable	HMPL_compiler.exe
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetGlobalVariables
LabVIEW	HIMC Get Global Variables.vi
Python	GetGlobalVariables

(このページはブランクになっています)

17. HIMC エラー関数

17.1 概要

HIMC は、関連するエラーメッセージを表す 32 ビットのエラーコードを提供します。本章で提供される関数を使用することで、システム、軸、軸グループのエラーコードを取得またはクリアできます。（各カテゴリのエラーコード、名称、および説明は、17.1.1～17.1.3 項に記載されています。）エラーコードのタイプは 0x□□□□□□□□で、「0x」は値が 16 進数であることを示します。その規則は、コントローラー変数のアドレス指定 ID の規則と同じであり、以下のように説明されます：

1. エラーコードの 1 番目と 2 番目の値は「コントローラー変数のカテゴリ」を示します。0x00□□□□□□はシステム変数に属し、0x83□□□□□□は軸変数に属し、0x82□□□□□□は軸グループ変数に属します。
2. エラーコードの 3 番目と 4 番目の値は「軸 ID または軸グループ ID」を示します。例えば、軸変数 0x8302□□□□は軸インデックス 02 を格納する変数であり、軸グループ変数 0x8201□□□□は軸グループインデックス 01 を格納する変数です。
3. エラーコードの 5 番目から 8 番目の値は「変数 ID」を示します。詳細は 17.1.1 から 17.1.3 を参照してください。

注: 関数の戻り値は 10 進数であるため、正しいエラー コードを取得するには、ユーザーが自分で 16 進数に変換する必要があります。

17.1.1 システムエラーメッセージ

表 17.1.1.1

システムエラーコード		
エラーコード	エラー名	説明
0x00000001	eERR_HCV_ID_NOT_FOUND	変数 ID が見つかりませんでした。
0x00000002	eERR_DATA_EXCEEDED	要求されたデータは範囲外です。
0x00000003	eERR_HCV_IS_READ_ONLY	読み取り専用パラメーター
0x00000004	eERR_HCV_VALUE_OUT_OF_RANGE	入力値が範囲外です。
0x00000050	eERR_EWM_CALLBACK_BUSY	EWM コールバック関数はビジーです。
0x00000064	eERR_EMERGENCY_STOP	緊急停止が有効になりました。すべての軸を無効にし、すべてのタスクを停止します。
0x00000107	eERR_NOT_VALID_TASKID	タスク ID が無効です。
0x00000108	eERR_TASK_IS_RUNNING	タスクはすでに実行されています。
0x00000109	eERR_FUNC_NOT_IN_TASK	タスク内に関数が見つかりませんでした。
0x0000010a	eERR_TASK_EMPTY	タスクは空です。
0x0000010b	eERR_TASK_NOT_RUNNING	タスクは実行されていません。
0x0000012c	eERR_NIC_INIT_TOUT	フィールドバスのネットワーク ポートの準備ができていません。
0x0000012d	eERR_HARDWARE_MISMATCH	ハードウェアが認識されません。
0x0000012e	eERR_SLAVE_NUM_MISMATCH	スレーブの数が構成と異なります。
0x00000130	eERR_INVALID_MCK_CNFG	モーションカーネルの構成が無効です。
0x00000138	eERR_HIMC_LOAD_CONFIG_FAIL	SSD からの設定の読み込みに失敗しました。もう一度保存してください。
0x00000139	eERR_HIMC_SAVE_CONFIG_FAIL	HIMC への設定の保存に失敗しました。もう一度保存してください。
0x0000013a	eERR_HIMC_SAVE_CONFIG_COPY_FAIL	HIMC への設定の保存に失敗しました。ファイルを SAVE フォルダに保存できません。
0x0000013c	eERR_ETHERCAT_LICENSE_MISMATCH	EtherCAT ライセンスが一致しません。
0x000001f4	eERR_ISR_NOT_STABLE	中断周期が安定しません。
0x000041f4	eWRN_ISR_NOT_STABLE	中断の周期が安定しない早期警告です。
0x000001f5	eERR_MCK_OVERLOAD	モーション カーネルがオーバーロードされています。
0x0000C1f5	eDBG_MCK_OVERLOAD	モーションカーネルがオーバーロードされています。(警告)
0x000001f6	eERR_ISR_OVERLOAD	CPU が過負荷状態です。
0x0000C1f6	eDBG_ISR_OVERLOAD	CPU が過負荷状態です。(警告)
0x00001388	eERR_HMPL_INVALID_ARG	HMPL では引数が無効です。
0x00001389	eERR_HMPL_INVALID_PTR	HMPL 内のポインタが無効です。
0x0000138a	eERR_HMPL_STACK_OVERFLOW	HMPL でのスタック オーバーフロー。
0x0000138b	eERR_HMPL_ILLEGAL_MEM_OP	HMPL ではメモリの操作は違法です。
0x0000138c	eERR_HMPL_MOTION_NOT_READY	モーション関数は同期状態で呼び出す必要があります。
0x0000138d	eERR_HMPL_STR_TOO_LONG	文字列の長さが範囲外です。
0x0000138e	eERR_HMPL_INVALID_STR_FORMAT	文字列の形式が無効です。
0x0000138f	eERR_HMPL_ARG_OUT_OF_RANGE	引数が範囲外です。

システムエラーコード		
エラーコード	エラー名	説明
0x00001392	eERR_HMPL_ASCII_AGENT_RUNNING	ASCII エージェントはすでに実行されています。複数の ASCII エージェントを同時に実行することはできません。
0x0000139c	eERR_HMPL_CANNOT_RUN_IN_DEBUG	この関数はデバッグモードでは実行できません。
0x000013a6	eERR_HMPL_TOO_MANY_BRK_POINT	タスク内のブレークポイントが多すぎます。
0x000013ec	eERR_HMPL_MUTEX_LOCK_TWICE	同じタスク内で同じミューテックスを 2 回ロックすることはできません。
0x00001450	eERR_HMPL_INVALID_SYS_TIME_MEMORY	バッファが小さすぎます。最小サイズは 30 バイトにする必要があります。
0x00001451	eERR_HMPL_NOT_SUPPORTED	この HMPL 関数はこのプラットフォームではサポートされていません。
0x00001452	eERR_HMPL_CLIENT_NOT_CONNECTED	クライアントが切断されているため送信できません。
0x000014b4	eERR_HMPL_NOT_IN_OP_MODE	この関数は OP モードでのみ実行できます。
0x0000176f	eERR_HMPL_INTERNAL_ERROR	HMPL 内部エラー
0x00001770	eERR_HMPL_EXEC_FAILED	HMPL 関数の実行に失敗しました。
0x00001771	eERR_HMPL_ASM_LOAD_FAILED	HMPL コンパイルに失敗しました。アセンブリ ファイルが空か生成されていません。
0x00001772	eERR_HMPL_STARTTASK_TIMEOUT	HMPL StartTask 関数のタイムアウト。
0x00001773	eERR_HMPL_STOPTASK_TIMEOUT	HMPL StopTask 関数のタイムアウト。
0x000017d4	eERR_ASCII_CONNECT_TIMEOUT	ASCII クライアント接続のタイムアウト。
0x000017d5	eERR_ASCII_CONNECT_FAILED	ASCII クライアント接続に失敗しました。IP とポートを確認してください。
0x000017d6	eERR_ASCII_MULTI_CONNECTING	複数の ASCII クライアントが同時に接続しています。
0x000017d7	eERR_ASCII_MULTI_DISCONNECTING	複数の ASCII クライアントが同時に切断されます。
0x000017d8	eERR_ASCII_DISCONNECT_TIMEOUT	ASCII クライアントの切断タイムアウト。
0x000017de	eERR_ASCII_RECV_TIMEOUT	ASCII クライアントの受信がタイムアウトしました。しばらくしてからもう一度お試しください。
0x000017df	eERR_ASCII_RECV_FAIL	ASCII クライアントの受信に失敗しました。接続がまだ有効かどうかを確認してください。
0x000017e0	eERR_ASCII_MULTI_RECVING	複数の ASCII クライアントが同時に受信しています。
0x000017e8	eERR_ASCII_SEND_TIMEOUT	ASCII クライアントの送信タイムアウトです。しばらくしてからもう一度お試しください。
0x000017e9	eERR_ASCII_SEND_FAIL	ASCII クライアントの送信に失敗しました。接続がまだ有効かどうかを確認してください。
0x000017ea	eERR_ASCII_MULTI_SENDING	複数の ASCII クライアントが同時に送信しています。
0x00001838	eERR_MODBUS_CONNECT_TIMEOUT	Modbus クライアント接続のタイムアウト。
0x00001839	eERR_MODBUS_CONNECT_FAILED	Modbus クライアント接続に失敗しました。IP を確認してください。

システムエラーコード		
エラーコード	エラー名	説明
0x0000183a	eERR_MODBUS_MULTI_CONNECTING	複数の Modbus クライアントが同時に接続します。
0x0000183b	eERR_MODBUS_MULTI_DISCONNECTING	複数の Modbus クライアントが同時に切断されます。
0x0000183c	eERR_MODBUS_DISCONNECT_TIMEOUT	Modbus クライアントの切断タイムアウト。
0x0000183d	eERR_MODBUS_DATALENGTH_ERR	Modbus クライアントの読み取り/書き込みデータ数が制限を超えています。
0x0000183e	eERR_MODBUS_SOCKET_BUSY	Modbus クライアントは 2 つ以上のコマンドを同時に処理します。
0x0000183f	eERR_MODBUS_JOB_TIMEOUT	Modbus クライアントジョブ実行がタイムアウトしました。しばらくしてからもう一度お試しください。
0x00001840	eERR_MODBUS_JOB_FAIL	Modbus クライアントジョブの実行に失敗しました。接続がまだ有効かどうかを確認してください。
0x0000b037	eMSG_HIMC_SET_DEFAULT	コントローラーを工場出荷時のデフォルトに設定します。
0x0000b038	eMSG_HIMC_REBOOT	コントローラーを再起動します。
0x0000b039	eMSG_HIMC_BOOT	コントローラーの電源がオンになっています。
0x0000b03a	eMSG_HIMC_INFO	コントローラー情報。
0x0000b03b	eMSG_HIMC_STORE_CONFIG	ストア構成。
0x0000b03e	eMSG_API_MAIN_ID_CHANGE_GET	API アクセス権限が get により変更されました。
0x0000b03f	eMSG_API_MAIN_ID_CHANGE_RELEASE	リリースにより API アクセス権限が変更になりました。
0x0000b2c8	eMSG_START_HMI_SCOPE	HMI スコープを開始します。
0x0000b2c9	eMSG_STOP_HMI_SCOPE	HMI スコープを停止します。
0x00003fff	eERR_SYS_LOG	このエラーはシステムから送信されました。
0x00007fff	eWRN_SYS_LOG	この警告はシステムから送信されました。
0x0000bfff	eMSG_SYS_LOG	このメッセージはシステムから送信されました。
0x0000ffff	eDBG_SYS_LOG	このデバッグ情報はシステムから送信されます。

17.1.2 軸エラーメッセージ

以下のエラーコードは、軸のエラーまたは無効な操作が原因で表示されます。□□記号は 16 進数の軸 ID です。例えば、01 は軸インデックス 01、0f は軸インデックス 15 を表します。

表 17.1.2.1

軸エラーコード		
エラーコード	エラー名	説明
0x83□□000a	eERR_AXIS_CMD_UNKOWN	コマンド名が不明です。
0x83□□001e	eERR_AXIS_CMD_QUEUE_FULL	軸コマンド キューがいっぱいです。
0x83□□0050	eERR_AXIS_CMD_INVALID_IN_SIM	このコマンドはシミュレータでは許可されません。
0x83□□0060	eERR_AXIS_CMD_INVALID_IN_VIRTUAL	このコマンドは仮想軸では許可されません
0x83□□0064	eERR_AXIS_CMD_INVALID_STATE	軸は現在の動作状態ではコマンドを実行できません
0x83□□006e	eERR_AXIS_CMD_INVALID_ENABLED	有効になっている間のコマンドは許可されません。
0x83□□0078	eERR_AXIS_CMD_INVALID_DISABLED	無効になっている間のコマンドは許可されません。
0x83□□0082	eERR_AXIS_CMD_INVALID_MOVING	軸は移動中にコマンドを実行できません。
0x83□□008c	eERR_AXIS_CMD_INVALID_STOPPING	軸の移動が停止した場合、コマンドは無効になります。
0x83□□0096	eERR_AXIS_CMD_INVALID_ERROR_STATE	軸が ErrorStop 状態の場合、コマンドは無効です。
0x83□□00a0	eERR_AXIS_CMD_INVALID_IN_SYNC	軸が同期動作状態にある場合、コマンドは無効です。
0x83□□00aa	eERR_AXIS_CMD_INVALID_GEAR_MASTER	軸がギアマスター軸の場合、コマンドは無効です。
0x83□□00b4	eERR_AXIS_CMD_INVALID_PP_MODE	軸が PP モードの場合、コマンドは無効です。
0x83□□00be	eERR_AXIS_CMD_INVALID_MAP_SWITCHING	軸が補正マップを切り替えているときはコマンドは無効です。
0x83□□00c8	eERR_AXIS_CMD_INVALID_INPUTSHAPING_ENABLED	位置コマンドシェーピング機能が有効な場合、軸はコマンドを実行できません。
0x83□□00d2	eERR_AXIS_CMD_INVALID_COMP_ENABLED	動的補正が有効な場合、軸はコマンドを実行できません。
0x83□□00dc	eERR_AXIS_CMD_INVALID_GANTRY_MODE	軸はガントリーモードでコマンドを実行できません。
0x83□□00e6	eERR_AXIS_CMD_INVALID_GROUPED	軸が軸グループ内にある場合、コマンドは許可されません。
0x83□□00f0	eERR_AXIS_CMD_INVALID_CONTROL_MODE	現在の制御モードではコマンドは無効です。
0x83□□00fa	eERR_AXIS_CMD_INVALID_OP_MODE	動作モードが無効です。
0x83□□0104	eERR_AXIS_CMD_INVALID_BUFFER_MODE	軸バッファ モードが無効です。
0x83□□0105	eERR_AXIS_CMD_INVALID_SETBUFFERMODE	軸に未完了のコマンドがある場合、コマンドは許可されません。
0x83□□010e	eERR_AXIS_CMD_INVALID_TP_ENABLED	タッチプローブが有効な場合、このコマンドは許可されません。

軸エラーコード		
エラーコード	エラー名	説明
0x83□□012c	eERR_AXIS_CMD_INVALID_PARAMETER	軸コマンドのパラメーターが無効です。
0x83□□0136	eERR_AXIS_CMD_INVALID_POS	軸のターゲット位置が許容範囲外です。
0x83□□0140	eERR_AXIS_CMD_INVALID_VEL	軸速度設定が許容範囲外です。
0x83□□014a	eERR_AXIS_CMD_INVALID_ACC	軸の加速度設定が許容範囲外です。
0x83□□0154	eERR_AXIS_CMD_INVALID_DEC	軸減速設定が許容範囲外です。
0x83□□015e	eERR_AXIS_CMD_INVALID_JERK	軸ジャーク設定が許容範囲外です。
0x83□□0168	eERR_AXIS_CMD_INVALID_SM_TIME	軸のスムーズ時間設定が許容範囲外です。
0x83□□0172	eERR_AXIS_CMD_INVALID_KILL_DEC	軸キル減速設定が許容範囲外です。
0x83□□017c	eERR_AXIS_CMD_INVALID_VEL_SCALE	軸速度スケール設定が許容範囲外です。
0x83□□0190	eERR_AXIS_COMP_NOT_CNFG	軸動的補正設定が正しく構成されていません。
0x83□□01c2	eERR_AXIS_CMD_INVALID_MASTER_SLAVE_CONNECTION	マスター・スレーブ関係の設定が無効です。
0x83□□01cc	eERR_AXIS_CMD_INVALID_SLAVE_ID	スレーブ ID 設定が無効です。
0x83□□01d6	eERR_AXIS_CMD_INVALID_GEAR_RATIO	スレーブ軸のギア比設定が許容範囲外です。
0x83□□01f4	eERR_AXIS_CMD_INVALID_ROLLOVER_POS	軸ロールオーバー位置が無効です。正の値にする必要があります。
0x83□□01fe	eERR_AXIS_CMD_INVALID_ROLLOVER_PP_MODE	プロファイルポジションモードではロールオーバーはサポートされていません。プロファイルポジションモードに切り替える前に、ロールオーバー値を 0 にリセットしてください。
0x83□□0208	eERR_AXIS_CMD_INVALID_FORCECONST_TRQMODE	トルクモードの力定数が無効です。正しい力定数を設定してください。
0x83□□0258	eERR_AXIS_CMD_INVALID_NO_SPECIFIC_PDO	PDO には、コマンドを実行するための特定の CoE オブジェクトが必要です。
0x83□□03f2	eERR_AXIS_DRIVE_FAULT	ドライバーに障害が発生しました。ドライバー内の対応するエラーメッセージを確認してください。
0x83□□03fc	eERR_AXIS_DRIVE_ABNORMAL_DISABLE	ドライバーが異常に無効になっています。
0x83□□0406	eERR_AXIS_DRIVE_ENABLE_TOUT	ドライバーを有効にするのに時間がかかりすぎました。
0x83□□0407	eERR_AXIS_DRIVE_ENABLE_TOUT_RMT	ドライバーの有効化に時間がかかりすぎました。 ドライバーのアクセス設定を確認してください。
0x83□□0410	eERR_AXIS_DRIVE_CLEAR_ERROR_TOUT	ドライバーエラーのクリアに時間がかかりすぎました。
0x83□□041a	eERR_AXIS_DRIVE_DISABLE_TOUT	ドライバーを無効にするのに時間がかかりすぎました。
0x83□□0424	eERR_AXIS_DRIVE_HOME_TOUT	軸を原点位置に戻すのに時間がかかりすぎました。
0x83□□042e	eERR_AXIS_DRIVE_HOME_FAILED	軸原点復帰エラー。ドライバーのエラーコードを確認してください。
0x83□□0456	eERR_AXIS_VEL_LIMIT	基準速度が速度制限を超えました。
0x83□□4456	eWRN_AXIS_VEL_LIMIT	基準速度が速度制限を超えたため、速度はクリップされます。
0x83□□0460	eERR_AXIS_ACC_LIMIT	基準加速度が加速度制限を超えました。
0x83□□046a	eERR_AXIS_CURR_LIMIT	現在のコマンドは現在の制限を超えました。

軸エラーコード		
エラーコード	エラー名	説明
0x83□□0474	eERR_AXIS_DAMPINGRATIO_LIMIT	軸の減衰比設定が許容範囲外です。
0x83□□047e	eERR_AXIS_FREQUENCY_LIMIT	軸の周波数設定が許容範囲外です。
0x83□□07da	eERR_AXIS_SWRL	軸参照位置が右のソフトウェア制限に達しました。
0x83□□07e4	eERR_AXIS_SWLL	軸参照位置が左のソフトウェア制限に達しました。
0x83□□07ee	eERR_AXIS_HWRL	軸の右ハードウェア制限信号がトリガーされました。
0x83□□07f8	eERR_AXIS_HWLL	軸の左ハードウェア制限信号がトリガーされました。
0x83□□0802	eERR_AXIS_COMP_LIMIT	軸補正位置が最大補正限界を超えました。
0x83□□083e	eERR_AXIS_PERR	軸位置誤差が保護限界を超えました。まず、モーターの動作に機械的な干渉がないか確認してください。
0x83□□0848	eERR_AXIS_VERR	軸速度誤差が保護限界を超えました。まず、モーターの動作に機械的な干渉がないか確認してください。
0x83□□08a2	eERR_AXIS_PVT_MOTION_VEL_LIMIT	軸 PVT モーションの速度が保護制限を超えました。まず、指定されたパラメーターが有効かどうかを確認してください。
0x83□□08ac	eERR_AXIS_PVT_MOTION_ACC_LIMIT	軸 PVT モーションの加速度が保護限界を超えました。まず、指定されたパラメーターが有効かどうかを確認してください。
0x83□□08b6	eERR_AXIS_PVT_MOTION_INVALID_TIME	軸 PVT モーションの時間シーケンスが無効です。まず、指定されたパラメーターが有効かどうかを確認してください。
0x83□□0bb8	eERR_AXIS_CTRL_ERR	軸内部制御エラー。
0x83□□0fa0	eERR_AXIS_CMD_GEAR_DISABLED	ギアが無効になっている間は、ギア コマンドは許可されません。
0x83□□0fa1	eERR_AXIS_CMD_INVALID_AXIS_IN_CAM	軸がカム内にある場合、ギア コマンドは無効です。
0x83□□1388	eERR_CAM_CMD_INVALID_ENGAGE_WINDOW	カム作動ウィンドウが許容範囲外です。
0x83□□1389	eERR_CAM_CMD_INVALID_ENGAGE_POSITION	カム篏合位置がカム テーブル領域外です。
0x83□□138a	eERR_CAM_CMD_INVALID_MASTER_SCALE_FACTOR	カムマスターのスケール係数が許容範囲外です。
0x83□□138b	eERR_CAM_CMD_INVALID_CAM_SCALE_FACTOR	カムのスケール係数が許容範囲外です。
0x83□□138c	eERR_CAM_CMD_INVALID_CAMTABLE_ID	カムテーブル ID が許容範囲外です。
0x83□□138d	eERR_CAM_CMD_INVALID_DISENGAGE_WINDOW	カム解除ウィンドウが許容範囲外です。
0x83□□138e	eERR_CAM_CMD_INVALID_DISENGAGE_POSITION	カムの解除位置が許容範囲外です。
0x83□□138f	eERR_CAM_CMD_INVALID_OPERATION_IN_ENGAGED_STATE	カム コマンドはエンゲージ状態では無効です。
0x83□□1390	eERR_CAM_CMD_INVALID_START_MODE	カム開始モードが有効な列挙値に対応していません
0x83□□1391	eERR_CAM_CMD_INVALID_MOVE_MODE	カム移動モードが有効な列挙値に対応していません。

軸エラーコード		
エラーコード	エラー名	説明
0x83□□1392	eERR_CAM_ENGAGED_FAILED	エンゲージ ウィンドウが小さすぎるため、CamMaster がエンゲージ ウィンドウを通過する可能性があります。
0x83□□1393	eERR_CAM_CMD_NOTINUSE	プロフィール終了モードは現在使用されていません
0x83□□1394	eERR_CAM_CMD_CAM_DISABLED	カムが無効になっている間はカム コマンドは許可されません。
0x83□□1395	eERR_CAM_CMD_INVALID_AXIS_NOT_INCAM	軸がカム内にない場合はカム コマンドは無効です。
0x83□□1396	eERR_CAM_CMD_INVALID_AXIS_NOT_IN_DISENGAGED	軸が解除されていない場合、カム コマンドは無効です。
0x83□□1397	eERR_CAM_CMD_INVALID_AXIS_IN_GEAR	軸がギアに入っている場合、カム コマンドは無効です。

17.1.3 グループエラーメッセージ

以下のエラーコードは、軸グループにおけるエラーまたは無効な操作が原因で表示されます。□□記号は 16 進数の軸グループ ID です。例えば、01 は軸グループインデックス 01、0f は軸グループインデックス 15 を表します。

表 17.1.3.1

軸グループエラーコード		
エラーコード	エラー名	説明
0x82□□000a	eERR_CRD_CMD_UNKNOWN	軸グループコマンドが不明です。
0x82□□0014	eERR_CRD_CMD_REACH_MAX_NUM_AXIS	軸グループの軸の最大数に達しました。
0x82□□001e	eERR_CRD_CMD_INVALID_KIN_SETTING	運動学タイプの設定が無効です。
0x82□□001f	eERR_CRD_CMD_INVALID_SPECIFIC_KIN	軸グループが特定の運動学タイプの場合、コマンドは無効です。
0x82□□0028	eERR_CRD_CMD_AXIS_DUPLICATED	軸は既にグループ内にあるため、追加できませんでした。
0x82□□0032	eERR_CRD_CMD_GRP_SIZE_EMPTY	軸グループが空です。
0x82□□003c	eERR_CRD_CMD_GRP_SIZE_FULL	軸グループがいっぱいなので、これ以上軸を保持できません。
0x82□□0046	eERR_CRD_CMD_INVALID_MOVING	軸グループが移動中はコマンドは無効です。
0x82□□0050	eERR_CRD_CMD_INVALID_DISABLED	軸グループが無効の場合、コマンドは無効です。
0x82□□005a	eERR_CRD_CMD_INVALID_INPUTSHAPING_PARAMETER_INCOMPLETE	軸グループ inshape 関数のパラメーターが不完全です。
0x82□□006e	eERR_CRD_CMD_INVALID_STATE	軸グループは、現在のモーション状態ではコマンドを実行できません。
0x82□□0078	eERR_CRD_CMD_QUEUE_FULL	最後のコマンドが完了するまでお待ちください。
0x82□□0082	eERR_CRD_CMD_GRP_AXIS_INVALID	グループ軸が無効です。
0x82□□008c	eERR_CRD_CMD_QUEUE_IS_NOT_EMPTY	コマンド キューが空ではありません。
0x82□□0096	eERR_CRD_CMD_INVALID_QUEUE_SIZE	コマンド キューのサイズが無効です。
0x82□□00d2	eERR_CRD_CMD_INVALID_POS	軸グループのターゲット位置または方向が許容範囲外です。
0x82□□00dc	eERR_CRD_CMD_INVALID_LIN_VEL	軸グループの線速度設定が許容範囲外です。
0x82□□00e6	eERR_CRD_CMD_INVALID_LIN_ACC	軸グループの直線加速度設定が許容範囲外です。
0x82□□00f0	eERR_CRD_CMD_INVALID_LIN_DEC	軸グループの直線減速設定が許容範囲外です。
0x82□□00fa	eERR_CRD_CMD_INVALID_LIN_JERK	軸グループの直線ジャーク設定が許容範囲外です。
0x82□□0104	eERR_CRD_CMD_INVALID_LIN_SM_TIME	軸グループの線形スムーズ時間設定が許容範囲外です。
0x82□□010e	eERR_CRD_CMD_INVALID_DAMPINGRATIO	軸グループの減衰比設定が許容範囲外です。
0x82□□0118	eERR CRD_CMD_INVALID_FREQUENCY	軸グループの周波数設定が許容範囲外です。
0x82□□0140	eERR_CRD_CMD_INVALID_ANG_VEL	軸グループの角速度設定が許容範囲外です。
0x82□□014a	eERR_CRD_CMD_INVALID_ANG_ACC	軸グループの角加速度設定が許容範囲外です。
0x82□□0154	eERR_CRD_CMD_INVALID_ANG_DEC	軸グループの角減速設定が許容範囲外です。

軸グループエラーコード		
エラーコード	エラー名	説明
0x82□□015e	eERR_CRD_CMD_INVALID_ANG_JERK	軸グループの角度ジャーク設定が許容範囲外です。
0x82□□0168	eERR_CRD_CMD_INVALID_ANG_SM_TIME	軸グループの角度スムーズ時間設定が許容範囲外です。
0x82□□0190	eERR_CRD_CMD_INVALID_VEL_SCALE	軸グループの速度スケールが許容範囲外です。
0x82□□019a	eERR_CRD_CMD_INVALID_TRANS_VEL	軸グループの遷移速度が無効です。
0x82□□01a4	eERR_CRD_CMD_INVALID_TRANS_DIS	軸グループの遷移距離が無効です。
0x82□□01a5	eERR_CRD_CMD_INVALID_TRANS_DEV	軸グループの遷移偏差が無効です。
0x82□□01a6	eERR_CRD_CMD_INVALID_TRANS_CURVE	軸グループの遷移曲率が無効です。
0x82□□01b8	eERR_CRD_CMD_TRANS_MODE_UNKNOWN	パス遷移モード名が不明です。
0x82□□01c2	eERR_CRD_CMD_COORD_SYS_UNKNOWN	座標系が不明です。
0x82□□01cc	eERR_CRD_CMD_BLEND_MODE_UNKNOWN	パスのブレンドモード名が不明です。
0x82□□01fe	eERR_CRD_CMD_LIN_INVALID_PARAM	パラメーターは線形パス計画には無効です。
0x82□□0262	eERR_CRD_CMD_CIRC_INVALID_PARAM	円形パス計画ではパラメーターが無効です。
0x82□□026c	eERR_CRD_CMD_CIRC_INVALID_CENTER	円形パスの中心位置が開始点/終了点に近すぎます。
0x82□□0276	eERR_CRD_CMD_CIRC_ANGLE_SMALL	円形パスの中心角が小さすぎます。
0x82□□0280	eERR_CRD_CMD_CIRC_INVALID_RADIUS	円パスの半径が無効です。
0x82□□028a	eERR_CRD_CMD_CIRC_INVALID_COORD	円形パスの座標系が無効です。
0x82□□02c6	eERR_CRD_CMD_BEZIER_INVALID_PARAM	パラメーターはベジェ曲線パス計画には無効です。
0x82□□02d0	eERR_CRD_CMD_BSPLINE_INVALID_PARAM	パラメーターは、B スプライン曲線パス計画には無効です。
0x82□□02da	eERR_CRD_CMD_CURVE_INVALID_STARTPOS	曲線パス計画の開始位置が無効です。
0x82□□02e4	eERR_CRD_CMD_COORD_INVALID_PARAM	座標変換のパラメーターが無効です。
0x82□□02ee	eERR_CRD_CMD_NURBS_INVALID_PARAM	パラメーターは NURBS 曲線パスプランニングには無効です。
0x82□□02f8	eERR_CRD_CMD_LOOKAHEAD_INVALID_PARAM	先読み動作のパラメーターは無効です。
0x82□□03f2	eERR_CRD_AXIS_ABNORMALLY_DISABLED	軸グループ内の 1 つ以上の軸が異常に無効になっています。
0x82□□03fc	eERR_CRD_AXIS_SWL	軸グループ内の軸の 1 つがソフトウェア制限に達しました。

17.2 HIMC_GetLastError

目的

コントローラーの最新のエラーコードを取得します。

構文

```
int HIMC_GetLastError(
    int ctrl_id,
    int *p_error_code
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

p_error_ code [out] コントローラーの最新のエラー コードを受信するためのバッファへのポインタ。
定義についてはセクション 17.1.1 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetLastError
LabVIEW	HIMC Get Last Error.vi
Python	GetLastError

17.3 HIMC_GetAxisLastErr

目的

軸の最新のエラーコードを取得します。

構文

```
int HIMC_GetAxisLastErr(
    int ctrl_id,
    int axis_id,
    int *err_code
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

err_code [out] 軸の最新のエラー コードを受信するためのバッファへのポインタ。
定義についてはセクション 17.1.2 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetAxisLastErr
LabVIEW	HIMC Get Axis Last Err.vi
Python	GetAxisLastErr

17.4 HIMC_ClearAxisLastErr

目的

軸の最新のエラー コードをクリアします。

構文

```
int HIMC_ClearAxisLastErr(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ClearAxisLastErr
LabVIEW	HIMC Clear Axis Last Err.vi
Python	ClearAxisLastErr

17.5 HIMC_GetGrpLastErr

目的

軸グループの最新のエラー コードを取得します。

構文

```
int HIMC_GetGrpLastErr(
    int ctrl_id,
    int group_id,
    int *err_code
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

err_code [out] 軸グループの最新のエラー コードを受信するバッファへのポインタ。
定義についてはセクション 17.1.3 を参照してください。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ClearAxisLastErr
LabVIEW	HIMC Clear Axis Last Err.vi
Python	ClearAxisLastErr

17.6 HIMC_ClearGrpLastErr

目的

軸グループの最新のエラー コードをクリアします。

構文

```
int HIMC_ClearGrpLastErr(
    int ctrl_id,
    int group_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

group_id [in] 軸グループインデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_ClearGrpLastErr
LabVIEW	HIMC Clear Grp Last Err.vi
Python	ClearGrpLastErr

17.7 HIMC_GetDriveErr

目的

ドライバーのエラーコードを取得します。

構文

```
int HIMC_GetDriveErr(
    int ctrl_id,
    int axis_id,
    int *err_code
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

err_code [in] ドライバーのエラー コードを受信するためのバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetDriveErr
LabVIEW	HIMC Get Drive Err.vi
Python	GetDriveErr

17.8 HIMC_GetErrorInformation

目的

指定されたエラー ID の情報を取得します。

構文

```
int HIMC_GetErrorInfomation(
    int error_id,
    char *p_name,
    int name_buff_len,
    int *p_name_actual_len,
    char *p_description,
    int description_buff_len,
    int *p_description_actual_len
);
```

パラメーター

error_id [in]	エラー ID を指定します。
p_name [out]	指定されたエラー ID のエラー名を受け取るバッファへのポインタ。
name_buff_len [in]	エラー名を受け取るバッファの最大文字数を指定します。
p_name_actual_len [out]	エラー名の実際の文字数を受け取るバッファへのポインタ。（終端のヌル文字は除く。）
p_description [out]	指定されたエラー ID のエラーの説明を受け取るバッファへのポインタ。
description_buff_len [in]	エラーの説明を受け取るバッファの最大文字数を指定します。
p_description_actual_len [out]	エラー記述の実際の文字数を受け取るバッファへのポインタ。（終端のヌル文字は除く。）

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetErrorInformation
LabVIEW	HIMC Get Error Information.vi
Python	GetErrorInformation

(このページはブランクになっています)

18. 原点復帰機能

18.1 概要

HIMC は CiA 402 原点復帰モードをサポートしており、ステージ構成に基づいて各軸の原点復帰方法を設定できます。CiA 402 原点復帰モードで定義されている原点復帰方法は表 18.1.1 に示され、詳細な図と説明は表 18.1.2 に示されています。

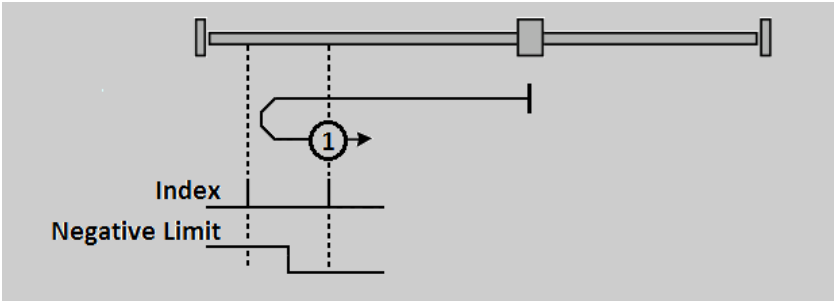
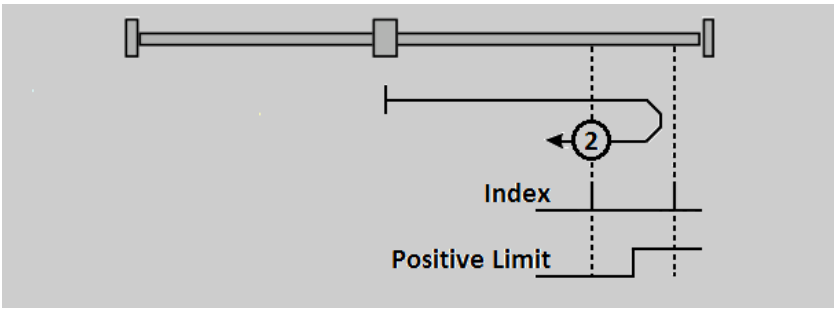
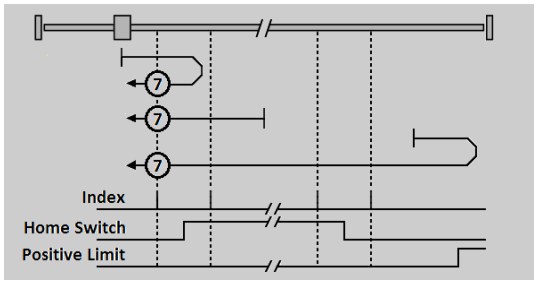
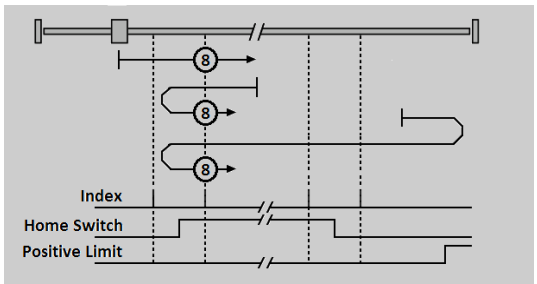
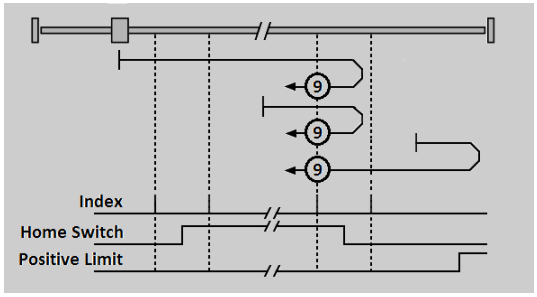
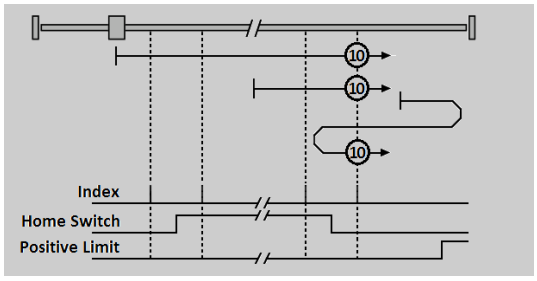
注 1: 原点復帰方法は、ドライバーでサポートされている方法に制限されます。

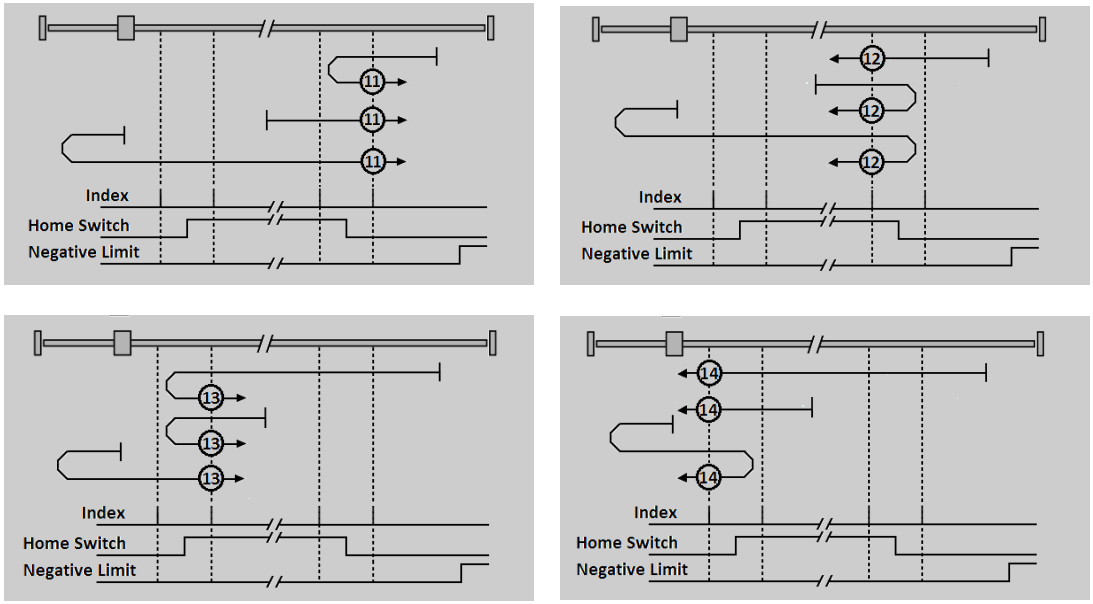
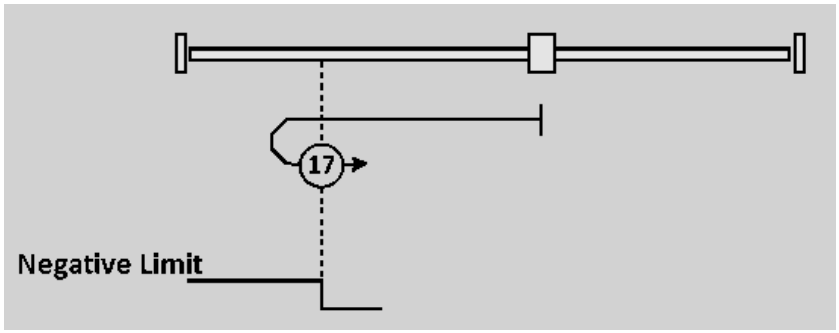
注 2：関数「HIMC_MoveHome」はシミュレータおよび仮想軸ではサポートされていません。

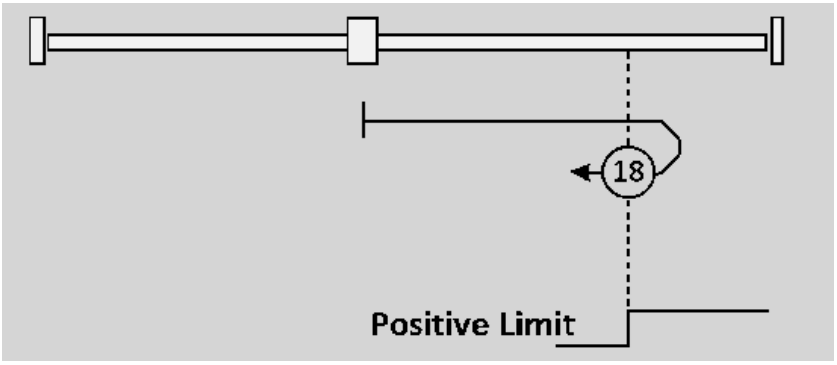
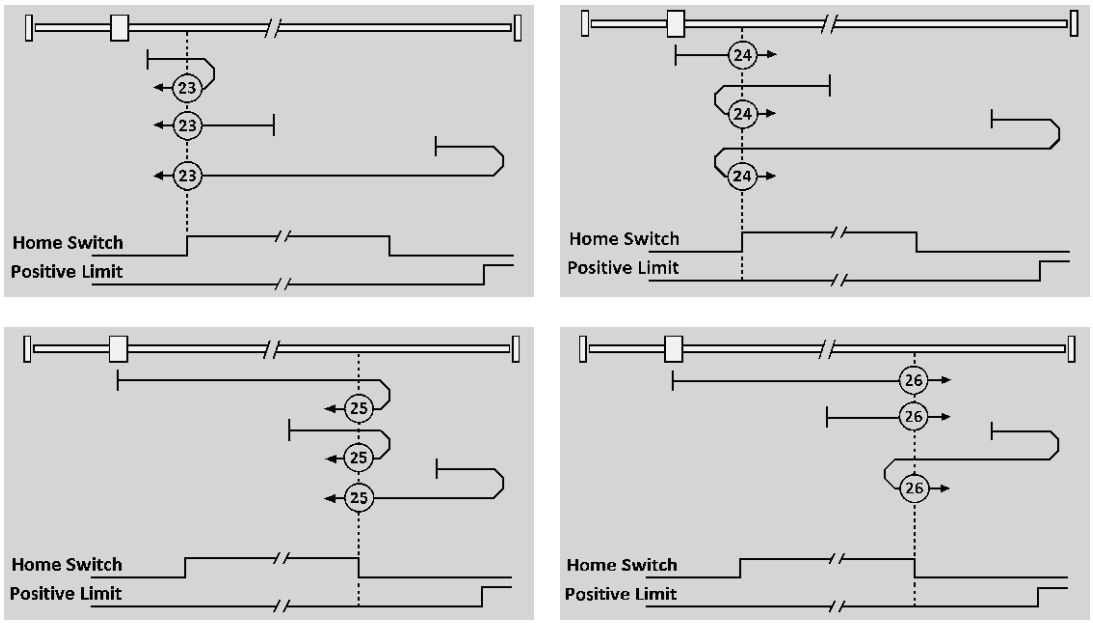
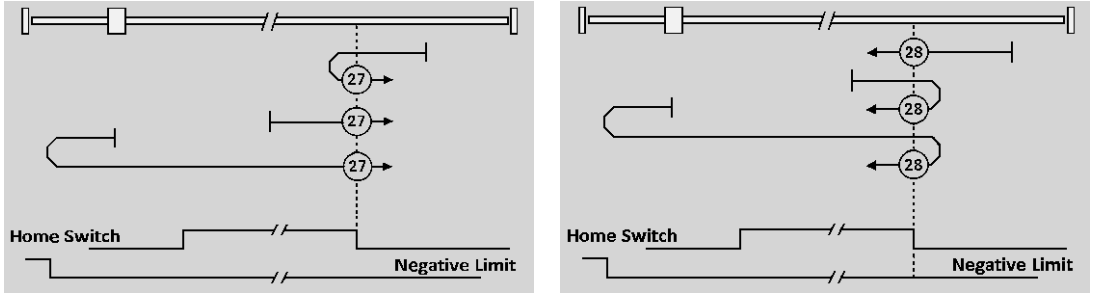
表 18.1.1

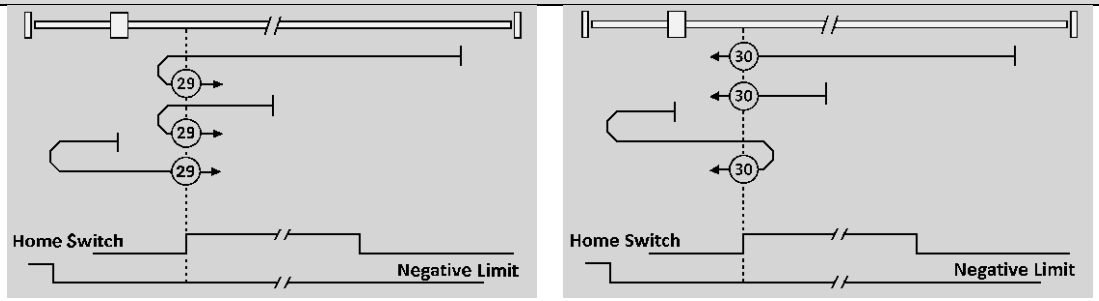
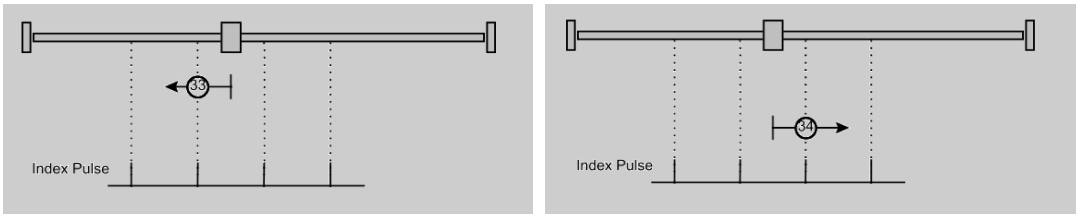
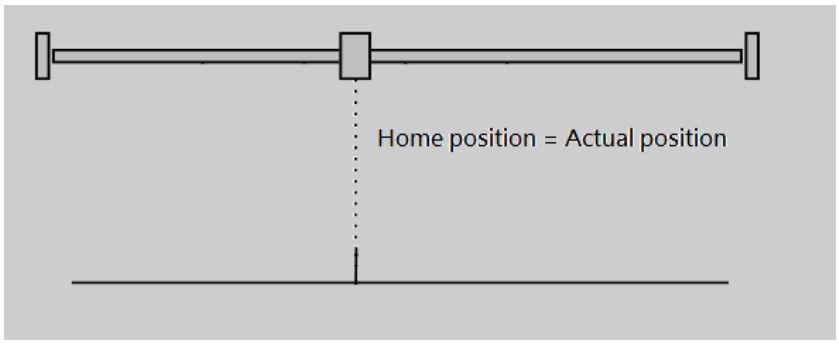
原点復帰方式	説明
1	負のリミットスイッチとインデックスパルスによる原点復帰
2	正リミットスイッチとインデックスパルスによる原点復帰
7~10	ホームスイッチとインデックスパルスによる原点復帰- 正の初期方向
11~14	ホームスイッチとインデックスパルスによる原点復帰- 負の初期方向
17	負のリミットスイッチで原点復帰
18	正リミットスイッチによる原点復帰
23~26	ホームスイッチによる原点復帰- 正の初期方向
27~30	ホームスイッチによる原点復帰- 負の初期方向
33~34	インデックスパルスによる原点復帰
37	現在の位置に原点設定

表 18.1.2

原点復帰方法	原点復帰手順
1	
	負リミットスイッチが非アクティブの場合、移動の初期方向は左方向です。原点位置は、負リミットスイッチが非アクティブになった位置の右側の最初のインデックスパルスです。 負リミットが設定されていない場合、原点復帰は失敗します。
2	
	正リミットスイッチが非アクティブの場合、移動の初期方向は右方向です。原点位置は、正リミットスイッチが非アクティブになった位置の左側の最初のインデックスパルスです。 正リミットが設定されていない場合、原点復帰は失敗します。
7~10	   

<table>
<tr><th>原点復帰方法</th><th>原点復帰手順</th></tr>
<tr><td></td><td>動作の初期方向は、探索する原点スイッチのエッジに依存します。原点スイッチが開始時にアクティブである場合、方法7と8の初期方向は負方向になります。それ以外の場合の初期方向は正方向になります。
原点スイッチと正方向のリミットが指定されていない場合、原点復帰は失敗します。</td></tr>
<tr><td rowspan="2">11~14</td><td> 
</td></tr>
<tr><td>動作の初期方向は、探索する原点スイッチエッジに依存します。原点スイッチが開始時にアクティブである場合、方法11および12の初期方向は正方向です。それ以外の場合の初期方向は負方向です。
原点スイッチと負方向リミットが指定されていない場合、原点復帰は失敗します。</td></tr>
<tr><td rowspan="2">17</td><td> 
</td></tr>
<tr><td>負リミットスイッチが非アクティブの場合、動作の初期方向は左方向です。原点位置は、負リミットスイッチが非アクティブになった位置の右側になります。
負リミットが設定されていない場合、原点復帰は失敗します。</td></tr>
</table>

<table>
<tr><th>原点復帰方法</th><th>原点復帰手順</th></tr>
<tr><td rowspan="2">18</td><td> 
</td></tr>
<tr><td>正リミットスイッチが非アクティブの場合、動作の初期方向は右方向になります。原点位置は、正リミットスイッチが非アクティブになった位置の左側になります。
正リミットが設定されていない場合、原点復帰は失敗します。</td></tr>
<tr><td rowspan="2">23~26</td><td> 
</td></tr>
<tr><td>動作の初期方向は、探索する原点スイッチのエッジに依存します。原点スイッチが開始時にアクティブである場合、方法23および24の初期方向は負方向になります。それ以外の場合の初期方向は正方向になります。
原点スイッチと正方向のリミットが指定されていない場合、原点復帰は失敗します。</td></tr>
<tr><td>27~30</td><td> 
</td></tr>
</table>

<table>
<tr><th>原点復帰方法</th><th>原点復帰手順</th></tr>
<tr><td rowspan="2"></td><td> 
</td></tr>
<tr><td>動作の初期方向は、探索する原点スイッチエッジに依存します。原点スイッチが開始時にアクティブである場合、方法27および28の初期方向は正方向です。それ以外の場合の初期方向は負方向です。
原点スイッチと負方向のリミットが指定されていない場合、原点復帰は失敗します。</td></tr>
<tr><td rowspan="2">33~34</td><td> 
</td></tr>
<tr><td>原点復帰方向はそれぞれ負方向（33）または正方向（34）です。原点位置は、選択された方向で検出されたインデックスパルスの位置です。</td></tr>
<tr><td rowspan="2">37</td><td> 
</td></tr>
<tr><td>モーターの現在位置が原点位置として定義されます。このメソッドでは、ドライバーが操作可能状態である必要はありません。オブジェクトは以下のように初期化されます。

062h（位置要求値）= 6064h（位置実値）= 607Ch（原点オフセット）
6063h（位置実値内部値）= 60FCh（位置要求内部値）= 0</td></tr>
</table>

18.2 HIMC_MoveHome

目的

軸の原点復帰手順を実行します。

構文

```
int HIMC_MoveHome(
    int ctrl_id,
    int axis_id
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

備考

この機能を使用する場合、ユーザーはオブジェクト 0x6060 (操作モード) とオブジェクト 0x6061 (操作モード表示) を PDO として設定する必要があります。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_MoveHome
LabVIEW	HIMC Move Home.vi
Python	MoveHome

18.3 HIMC_SetHomeMethod

目的

原点復帰手順の原点復帰方法を設定します。

構文

```
int HIMC_SetHomeMethod(
        int ctrl_id,
        int axis_id,
        int method
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

method [in] 原点復帰方法の詳細については、表 18.1.1 および表 18.1.2 を参照してください。
初期設定値は 33 です。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHomeMethod
LabVIEW	HIMC Set Home Method.vi
Python	SetHomeMethod

18.4 HIMC_SetHomeSwitchVel

目的

原点復帰手順の高速原点復帰速度を設定します。

構文

```
int HIMC_SetHomeSwitchVel(
    int    ctrl_id,
    int    axis_id,
    double fast_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

fast_vel [in] 高速原点復帰速度、初期値は 20 です。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHomeSwitchVel
LabVIEW	HIMC Set Home Switch Vel.vi
Python	SetHomeSwitchVel

18.5 HIMC_SetHomeZeroVel

目的

原点復帰手順の低速原点復帰速度を設定します。

構文

```
int HIMC_SetHomeZeroVel(
    int    ctrl_id,
    int    axis_id,
    double slow_vel
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

slow_vel [in] 低速原点復帰速度、初期値は 5 です。
パラメーター単位: mm/s または deg/s

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHomeZeroVel
LabVIEW	HIMC Set Home Zero Vel.vi
Python	SetHomeZeroVel

18.6 HIMC_SetHomeAcc

目的

原点復帰手順の原点復帰加速度を設定します。

構文

```
int HIMC_SetHomeAcc(
    int     ctrl_id,
    int     axis_id,
    double acc
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

acc [in] 原点復帰加速度、初期値は 2000 です。
パラメーター単位: mm/s^2 または deg/s^2

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHomeAcc
LabVIEW	HIMC Set Home Acc.vi
Python	SetHomeAcc

18.7 HIMC_SetHomeOffset

目的

原点復帰手順の原点オフセットを設定します。

構文

```
int HIMC_SetHomeOffset(
    int     ctrl_id,
    int     axis_id,
    double offset
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
offset [in]	原点オフセット。初期値は 0 です。 パラメーター単位: mm または deg

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHomeOffset
LabVIEW	HIMC Set Home Offset.vi
Python	SetHomeOffset

18.8 HIMC_SetHomeTimeout

目的

原点復帰手順のタイムアウトを設定します。

構文

```
int HIMC_SetHomeTimeout(
    int ctrl_id,
    int axis_id,
    int timeout
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

timeout [in] タイムアウト。初期値は 120,000 です。
パラメーター単位: ms

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHomeTimeout
LabVIEW	HIMC Set Home Timeout.vi
Python	SetHomeTimeout

18.9 HIMC_SetHomedStatus

目的

軸の原点復帰状態を設定します。

構文

```
int HIMC_SetHomedStatus(
        int ctrl_id,
        int axis_id,
        int is_homed
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
is_homed [in]	原点復帰が完了したことを示すには「1」に設定します。 原点復帰が完了していないことを示すには「0」に設定します（初期値）。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_SetHomedStatus
LabVIEW	-
Python	-

18.10 HIMC_IsHomed

目的

軸が原点復帰手順を完了したかどうかを照会します。

構文

```
int HIMC_IsHomed(
        int ctrl_id,
        int axis_id,
        int *p_is_homed
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_homed [out] 軸が原点復帰手順を完了したかどうかを受け取るバッファへのポインタ。
軸が原点復帰手順を完了している場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsHomed
LabVIEW	HIMC Is Homed.vi
Python	IsHomed

18.11 HIMC_IsHoming

目的

軸が原点復帰手順を実行しているかどうかを照会します。

構文

```
int HIMC_IsHoming(
        int ctrl_id,
        int axis_id,
        int *p_is_homing
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

axis_id [in] 軸インデックス

p_is_homing [out] 軸が原点復帰手順が実行中かどうかを受け取るバッファへのポインタ。
軸が原点復帰動作中の場合、値は 1 になります。それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsHoming
LabVIEW	HIMC Is Homing.vi
Python	IsHoming

18.12 HIMC_IsHomeSwitch

目的

軸の原点スイッチのステータスを取得します。

構文

```
int HIMC_IsHomeSwitch(
        int ctrl_id,
        int axis_id,
        int *p_is_home_switch
);
```

パラメーター

ctrl_id [in]	HIWIN モーション コントローラーのコントローラー ID。 HIMC_ConnectCtrl を呼び出して取得する必要があります。
axis_id [in]	軸インデックス
p_is_home_switch [out]	軸の原点スイッチの状態を受け取るバッファへのポインタ。 軸の原点スイッチの状態がオンの場合、値は 1 になります。 それ以外の場合は 0 になります。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_IsHomeSwitch
LabVIEW	-
Python	IsHomeSwitch

(このページはブランクになっています)

19. スレーブ関数

19.1 HIMC_GetSlvChNum

目的

チャネル タイプに基づいてスレーブの対応するチャネル番号を取得します。

構文

```
int HIMC_GetSlvChNum(
    int ctrl_id,
    int slv_slot_id,
    int ch_type,
    int *p_ch_num
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_slot_id [in] スレーブ ID とスロット ID。スレーブにスロットがない場合、スロット ID は無視されます。

ch_type [in] チャネルの種類
1: 汎用入力、2: 汎用出力、3: アナログ入力、4: アナログ出力。

p_ch_num [out] スレーブのチャネル番号を受信するためのバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvChNum
LabVIEW	-
Python	GetSlvChNum

19.2 HIMC_GetSlvSlotNum

目的

スレーブのスロット番号を取得します。

構文

```
int HIMC_GetSlvSlotNum(
    int ctrl_id,
    int slv_id,
    int *p_slot_num
);
```

パラメーター

ctrl_id [in] HIWIN モーション コントローラーのコントローラー ID。
HIMC_ConnectCtrl を呼び出して取得する必要があります。

slv_id [in] スレーブ ID

p_slot_num [out] スレーブのスロット番号を受信するバッファへのポインタ。

戻りの値

関数が成功した場合は int 値 0 を返し、関数が失敗した場合は 0 以外の値を返します。

要件

サポートされる最小バージョン	iA Studio 3.3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	HIMC_GetSlvSlotNum
LabVIEW	-
Python	GetSlvSlotNum

(このページはブランクになっています)

20. データ構造

20.1 コム情報

目的

接続タイプとその情報を定義します。

構文

```
typedef struct {
    ComType type;
    struct {
        char ip[20];
        char port[12];
    } TCP_IP;

    struct {
        char com_port_name[80];
        int  baud_rate;
    } RS232;

    struct {
        char autoExecExe;
    } Simulator;
} ComInfo;
```

パラメーター

ComType	接続タイプ
TCP_IP	ネットワーク接続パラメーター、IP、ポートを含む構造体。
RS232	RS232 接続パラメーター、COM ポート名、ボー レートを含む構造体。
Simulator	シミュレータ接続パラメーターと autoExecExe を含む構造体。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	ComInfo
LabVIEW	-
Python	ComInfo

20.2 CoordPosition

目的

座標移動の位置または距離を定義します。

構文

```
typedef struct {
    double x_pos;
    double y_pos;
    double z_pos;
    double a_pos;
    double b_pos;
    double c_pos;
} CoordPosition, *PCoordPosition;
```

パラメーター

x_pos	終点の直線位置 X。単位：mm
y_pos	終点の直線位置 Y。単位：mm
z_pos	終点の直線位置 Z。単位：mm
a_pos	終点の方向角 A。単位：度
b_pos	終点の方向角 B。単位：度
c_pos	終点の方向角 C。単位：度

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	CoordPosition
LabVIEW	-
Python	CoordPosition

20.3 MotionProfile

目的

モーション プロファイルの設定を定義します。

構文

```
typedef struct {
    double max_vel;
    double max_acc;
    double max_dec;
    double smooth_time;
} MotionProfile, *PMotionProfile;
```

パラメーター

max_vel　直線運動の最大接線速度。
単位: mm/s または deg/s
範囲: 0 ~ 5000

max_acc　直線運動の最大接線加速度
単位: mm/s^2 または deg/s^2
範囲: >0 ~ 50000 (加速度は 0 にできません)

max_dec　直線運動の最大接線減速度
単位: mm/s^2 または deg/s^2
範囲: >0 ~ 50000 (減速度は 0 にできません)

smooth_time　直線運動のスムーズな時間
単位: ms
範囲: 0 ~ 500

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	MotionProfile
LabVIEW	-
Python	MotionProfile

20.4 CenterPosition

目的

中心位置の設定を定義します。

構文

```
typedef struct {
    double x_pos;
    double y_pos;
    double z_pos;
} CenterPosition, *PCenterPosition;
```

パラメーター

x_pos　　X 軸の位置。単位：mm
y_pos　　Y 軸の位置。単位：mm
z_pos　　Z 軸の位置。単位：mm

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	CenterPosition
LabVIEW	-
Python	CenterPosition

20.5 NormalVector

目的

法線ベクトルの設定を定義します。

構文

```
typedef struct {
    double x_vector;
    double y_vector;
    double z_vector;
} NormalVector, *PNormalVector;
```

パラメーター

x_vector	X 方向ベクトル
y_vector	Y 方向ベクトル
z_vector	Z 方向ベクトル

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	NormalVector
LabVIEW	-
Python	NormalVector

20.6 TransPrm

目的

位置トリガー設定を定義します。

構文

```
typedef struct {
    double trans_vel,
    double trans_dis,
    double trans_dev;
    double trans_curv;
} TransPrm, *PTransPrm;
```

パラメーター

trans_vel　軸グループの新しい遷移モードの速度パラメーター。
単位: mm/s

trans_dis　軸グループの新しい遷移モードの距離パラメーター。
単位: mm

trans_dev　軸グループの新しい遷移モードの最大偏差パラメーター。
単位: mm

trans_curv　軸グループの新しい遷移モードの最大曲率パラメーター。
単位: mm^{-1}

要件

サポートされる最小バージョン	iA Studio 3.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	TransPrm
LabVIEW	-
Python	TransPrm

20.7 PosTriggerPar

目的

位置トリガー設定を定義します。

構文

```
typedef struct {
    double start_pos;
    double end_pos;
    double interval;
    int pulse_width;
} PosTriggerPar, *PPosTriggerPar;
```

パラメーター

start_pos　PT 関数の開始位置
単位: mm または deg

end_pos　PT 関数の終了位置
単位: mm または deg

interval　連続する PT 出力間の位置間隔
単位: mm または deg

pulse_width　各 PT 出力信号の幅
単位: ns

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	PosTriggerPar
LabVIEW	-
Python	PosTriggerPar

20.8 Data_t

目的

CoE オブジェクトの値の型を定義します。

構文

```
typedef union {
    unsigned char bytes[65];
    unsigned char uint8_;
    unsigned short uint16_;
    unsigned int uint32_;
    unsigned uint64_t uint64_;
    signed char int8_;
    short int16_;
    int int32_;
    int64_t int64_;
    float float_;
    double double_;
} Data_t;
```

パラメーター

bytes	値の型は文字列です。(注: C# ではサポートされていません)
uint8_	値の型は unsigned char です。
uint16_	値の型は符号なし短整数です。
uint32_	値の型は符号なし整数です。
uint64_	値の型は符号なし長整数です。
int8_	値の型は char です。
int16_	値の型は短整数です。
int32_	値の型は整数です。
int64_	値の型は長整数です。
float_	値の型は float です。
double_	値の型は double です。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	Data_t
LabVIEW	-
Python	Data_t

21. 列挙

21.1 ComType

定義

接続タイプの列挙

構文

```
typedef enum {
    COM_TYPE_TCPIP,
    COM_TYPE_RS232,
    COM_TYPE_SIMULATOR
} ComType;
```

パラメーター

COM_TYPE_TCPIP	接続タイプは TCPIP です。
COM_TYPE_RS232	接続タイプは RS232 です。
COM_TYPE_SIMULATOR	接続タイプはシミュレータです。

要件

サポートされる最小バージョン	iA Studio 0.23
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	ComType
LabVIEW	-
Python	ComType

21.2 CoordSystem

定義

座標系

構文

```
typedef enum {
    kCoord_ACS = 0,
    kCoord_MCS = 1,
    kCoord_PCS = 2,
    kCoord_GLOBAL = 3,
    kCoord_WCS1 = 1 << 8,
    kCoord_WCS2 = 2 << 8,
    kCoord_WCS3 = 3 << 8,
    kCoord_WCS4 = 4 << 8,
    kCoord_WCS5 = 5 << 8,
    kCoord_WCS6 = 6 << 8,
    kCoord_WCS7 = 7 << 8,
    kCoord_WCS8 = 8 << 8,
    kCoord_WCS9 = 9 << 8,
    kCoord_WCS10 = 10 << 8,
    kCoord_WCS11 = 11 << 8,
    kCoord_WCS12 = 12 << 8,
    kCoord_WCS13 = 13 << 8,
    kCoord_WCS14 = 14 << 8,
    kCoord_WCS15 = 15 << 8,
    kCoord_OFFSET = 1 << 15
} CoordSystem;
```

説明

セクション 6.1.2 を参照してください。

要件

サポートされる最小バージョン	iA Studio 2.0
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	CoordSystem
LabVIEW	-
Python	CoordSystem

21.3 MotionBufferMode

定義
隣接する座標移動セグメント間のバッファモード

構文

```
typedef enum {
    kBM_Aborting = 0,
    kBM_Buffered = 1,
    kBM_BlendingLow = 2,
    kBM_BlendingPrevious = 3,
    kBM_BlendingNext = 4,
    kBM_BlendingHigh = 5,
    kBM_LookAhead = 6
} MotionBufferMode;
```

説明
セクション 6.1.4 を参照してください。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	MotionBufferMode
LabVIEW	-
Python	MotionBufferMode

21.4 MotionTransitionMode

定義

隣接する座標運動セグメント間の遷移モード

構文

```
typedef enum {
    kTM_NONE = 0,
    kTM_StartVelocity = 1,
    kTM_ConstantVelocity = 2,
    kTM_CornerDistance = 3,
    kTM_MaxCornerDeviation = 4,
    kTM_PLCOpenReserved_05 = 5,
    kTM_PLCOpenReserved_06 = 6,
    kTM_PLCOpenReserved_07 = 7,
    kTM_PLCOpenReserved_08 = 8,
    kTM_PLCOpenReserved_09 = 9,
    kTM_PLCOpenReserved_10 = 10,
    kTM_MaxCornerCurvature = 11
} MotionTransitionMode;
```

説明

セクション 6.1.5 を参照してください。

要件

サポートされる最小バージョン	iA Studio 3.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	MotionTransitionMode
LabVIEW	-
Python	MotionTransitionMode

21.5 ShaperMode

定義

入力シェーピングフィルター（InShape）のフィルターモード

構文

```
typedef enum {
    Shaper_Normal = 0,
    Shaper_Robust
} ShaperMode;
```

パラメーター

Shaper_Normal	通常の入力シェーピング フィルター
Shaper_Robust	堅牢な入力シェーピング フィルター

要件

サポートされる最小バージョン	iA Studio 1.1
Header	HIMC_API.h
Library	HIMC_API.lib
DLL	HIMC_API.dll

他の API 環境の対応名称

C#	ShaperMode
LabVIEW	-
Python	ShaperMode

(このページはブランクになっています)

22. 付録

22.1 API エラーコード

API でコントローラーにアクセスすると、次のエラー コードが表示されます。

表 22.1.1

API エラーコード		
エラーコード	エラー名	説明
0x01000000	eERR_API_COMM_ERR	コントローラーとの通信中にエラーが発生しました。
0x0100000a	eERR_API_CONNECT_FAIL	コントローラーに接続できません。
0x01000014	eERR_API_TOUT	タイムアウト期間が経過したため、この操作は戻りました。
0x0100001e	eERR_API_ACCESS_REJECT	リクエストは拒否されました。
0x01000028	eERR_API_FIFO_MISMATCH	致命的な API エラー
0x01000032	eERR_API_FIFO_FULL	ネットワークがビジー状態です。
0x0100003c	eERR_API_HIMC_NOT_READY	HIMC はまだ準備ができていません。
0x01000046	eERR_API_PROTOCOL_MISMATCH	致命的な API エラー
0x01000050	eERR_API_INPUT_ARG_ERR	引数が無効です。
0x0100005a	eERR_API_NOT_SUPPORT	このバージョンでは API はサポートされていません。
0x01000064	eERR_API_BUSY	API がビジー状態です。
0x0100006e	eERR_API_FILE_TRANS_FAIL	ファイルの転送に失敗しました。
0x01000073	eERR_API_ZIP_CORRUPT	ZIP ファイルが破損しています。
0x01000078	eERR_API_ID_NOT_FOUND	接続 ID が見つかりませんでした。まだ接続されていない可能性があります。
0x01000082	eERR_API_SLV_DB_NOT_READY	スレーブは準備ができていません。
0x0100008c	eERR_API_SLV_ID_INVALID	スレーブ ID が無効です。
0x01000096	eERR_API_INVALID_VAR_ID	変数 ID が無効です。
0x010000a0	eERR_API_VAR_VAL_OUT_OF_RANGE	値が範囲外です。
0x010000a5	eERR_API_VAR_IS_READ_ONLY	変数は読み取り専用です。
0x010000aa	eERR_API_FS_ACCESS_DENIED	ファイル システムにアクセスできません。権限を確認してください。
0x010000b4	eERR_API_TASK_ID_INVALID	タスク ID が無効です。
0x010000be	eERR_API_TASK_EMPTY	タスクは空です。
0x010000c3	eERR_API_TASK_FUNC_NOT_FOUND	関数が見つかりません。
0x010000c8	eERR_API_TASK_NOT_RUNNING	タスクは実行されていません。
0x010000d2	eERR_API_TASK_IS_RUNNING	タスクはすでに実行されています。
0x010000d7	eERR_API_TOO_MANY_BRK_POINT	タスク内のブレークポイントが多すぎます。
0x010000dc	eERR_API_INVALID_ERROR_ID	エラーID が無効です。
0x010000e6	eERR_API_INSUFFICIENT_BUFFER	バッファが不足しています。
0x010000f0	eERR_API_STR_TOO_LONG	文字列の長さが範囲外です。
0x010000fa	eERR_API_HIMC_VERSION_MISMATCH	API はこのコントローラーのバージョンと互換性がありません。
0x010003e8	eERR_API_MOTION_ERROR	モーションコントロールエラーです。エラーログを確認してください。
0x010087d0	eMSG_API_PACKET_SYS_CALL	パケットからのシステムコールメッセージ

API エラーコード		
エラーコード	エラー名	説明
0x010047da	eWRN_API_PACKET_BUSY	API データ パケットの使用率が高くなっています。
0x010047e4	eWRN_API_PACKET_INVALID_SYS_CALL_ID	システムコール ID が無効です。
0x01008bb8	eMSG_API_CONNECT	API クライアントが接続されました。
0x01008bb9	eMSG_API_DISCONNECT	API クライアントが切断されました。
0x01008bba	eMSG_API_VERSION	API クライアントのバージョン。
0x0100270f	eERR_API_FATAL	致命的な API エラー。
0x01003fff	eERR_API_LOG	このエラーは API から送信されます。
0x01007fff	eWRN_API_LOG	この警告は API から送信されます。
0x0100bfff	eMSG_API_LOG	このメッセージは API から送信されます。
0x0100ffff	eDBG_API_LOG	このデバッグ情報は API から送信されます。

（このページはブランクになっています）

HIMC シリーズ
API ユーザーマニュアル
バージョン：V1.2　　2025 年 12 月改訂

---

1. HIWIN は HIWIN Mikrosystem Corp., HIWIN Technologies Corp., ハイウィン株式会社の登録商標です。ご自身の権利を保護するため、模倣品を購入することは避けてください。
2. 実際の製品は、製品改良等に対応するため、このカタログの仕様や写真と異なる場合があります。
3. HIWIN は「貿易法」および関連規制の下で制限された技術や製品を販売・輸出しません。制限された HIWIN 製品を輸出する際には、関連する法律に従って、所管当局によって承認を受けます。また、核・生物・化学兵器やミサイルの製造または開発に使用することは禁じます。
