PCI Bus CPD ボードシリーズ

motionCAT シリーズ

ユーザーズマニュアル

〈ソフトウェア編・別冊〉

INtime　HLS-MCT520/IT 導入版

株式会社ハイバーテック

http://www.hivertec.co.jp/

この説明書は次のボードに適応しています.

PCI	HPCI- MNT520 HPCI- MCAT520
Compact PCI	HCPCI- MNT720
PCI Express	HPCIe- MCAT620

株式会社 ハイバーテック
東京都江東区新大橋 1-8-11
三井生命新大橋ビル
TEL 03-3846-3801
FAX 03-3846-3773
sales@hivertec.co.jp

第 1.01 版 2022 年 11 月 14 日発行

本製品をご使用される前に「注意事項」を必ずご一読の上ご利用をお願い致します.

目　次

図　表　目　次

■ 注意事項

■ 保証範囲

1. 本製品の保証期間は，お買い上げ頂いた日より 3 年間です．保証期間中に弊社の判断により欠陥が判明した場合には，本製品を弊社に引き取り，修理または交換を行います．

2. 保証期間内外に関わらず，弊社製品の使用，供給(納期)または故障に起因する，お客様及び第三者が被った，直接，間接，二次的な損害あるいは，遺失利益の損害に付いて，弊社は本製品の販売価格以上の責任を負わないものとしますので，予めご了承ください．

■ 免責事項

1. 本書に記載された内容に沿わない，製品の取付，接続，設定，運用により生じた損害に対しましては，一切の責任を負いかねますので，予めご了承ください．

2. 本製品は，一般電子機器用(工作機械・計測機器・FA/OA 機器・通信機器等)に製造された半導体製品を使用していますので，その誤作動や故障が直接，生命を脅かしたり，身体・財産等に危害を及ぼしたりする恐れのある装置(医療機器・交通機器・燃焼機器・安全装置等)に適用できるような設計，意図，または，承認，保証もされていません．
   ゆえに本製品の安全性，品質および性能に関しては，本書(またはカタログ)に記載してあること以外は明示的にも黙示的にも一切保証するものではありませんので，予めご了承ください．

3. 保証期間内外に関わらず，お客様が行った弊社の承認しない製品の改造または，修理が原因で生じた損害に対しましては，一切の責任を負いかねますので，予めご了承ください．

4. 本書に記載された内容について，弊社もしくは，第三者の特許権，著作権，商標権，その他の知的所有権の権利に対する保証または実施権の許諾を行うものではありません．
   また本書に記載された情報を使用したことにより第三者の知的所有権等の権利に関わる問題が生じた場合，弊社は，その責任を負いかねますので，予めご了承ください．

■ 安全にお使い頂くために

この度は，弊社 NC ボードシリーズをご採用頂きまして，誠に有り難う御座います．本書は，本製品をご使用して頂く場合の取扱い，留意点に付いて記入してありますので，必ずご一読の上ご利用をお願い致します．
尚，本書は，本書が添付されたNCボード常設箇所付近の分かりやすい場所に常時保管し，必要に応じて適宜参照・確認頂きますよう，お願い致します．

安全上の注意

本製品のご使用前に，必ずこのユーザーズマニュアル及び付属書類を全て熟読し，内容を理解してから正しくご使用下さい．本製品の知識，安全の情報及び注意事項の全てに付いて習熟してからご使用下さい．
本ユーザーズマニュアルでは，安全注意事項のランクを「警告」，「注意」として区分してあります．

⚠ **警 告**	この表示を無視して，誤った取扱いをすると，人が死亡または重傷を負う可能性が想定される内容を示しています．
⚠ **注 意**	この表示を無視して，誤った取扱いをすると，人が傷害を負う可能性または物的損害が想定される内容を示しています．

■ 対象ユーザー

注 意

本製品およびマニュアルは，以下の様な，ユーザーを対象としています．

・拡張用ボードの増設および配線に付いて基本的な知識を有している方．
・制御用電子機器およびパソコン等に付いて基本的な知識を有している方．

■ 添付ソフトウェア適合 OS

注 意

添付ソフトウェアは，INtime および Windows8(32bit/64bit)，Windows7(32bit/64bit)の各エディション，Windows XP SP3(32bit)の何れかの Windows OS が動いているパソコン上においてボードの制御を行う為のソフトウェアです．
上記以外の OS および dRTOS 上でのご使用については非対応です．詳しくは弊社営業までお問合せ下さい．

■ サンプルプログラム

警 告

本製品に添付される「サンプルプログラム」は，ボードが正しく設定・装着されているか，動作環境が正しく設定されているかを理解して頂くとともに，ボードの機能・動作を理解およびボードを制御する手順・制御プログラムの作成方法を理解して頂く為のものです．
故に使用される機器毎に固有な安全対策処理等を含んでいませんので，「動かしてみる」プログラムを定常的に機器運転に使用しないで下さい．

モータや装置を接続して動作させる場合は，モータや装置の特性を考慮した動作条件を設定願います．
特に試運転時は，十分に安全な値で実施し，徐々に所定の値に変更することをお勧めします．

サンプルプログラムを使用し装置を動作させる時，最初は速度の低いところで，また機械系に合った設定を行って動作を確認して下さい．機械系に合わない設定で動作を行うと思わぬ動きをすることがあります．

■ ユーザープログラム

⚠ 警 告
⚠ 本製品を使用し装置を動作させる時は，プログラムのデバッグを充分行ってから動作させて下さい．プログラムに間違いがありますと，思わぬ動きをすることがあります．

■ 試運転・調整

警 告
⚠ 本シリーズ製品を使用し装置を動作させる時は，プログラムのデバッグを充分行ってから動作させてください．プログラムに間違いがありますと，思わぬ動きをすることがあります． ⚠ 本シリーズ製品に添付してあるサンプルプログラムを使用し装置を動作させる時，最初は速度の低いところで，また機械系に合った設定を行って動作を確認してください．機械系に合わない設定で動作を行うと思わぬ動きをすることがあります．

■ motionCAT シリーズのマニュアル構成

motionCAT シリーズ製品のマニュアルは

(1) motionCAT シリーズユーザーズマニュアル <導入編> (INtime 版 別冊あり)

(2) motionCAT シリーズユーザーズマニュアル <運用編>

(3) motionCAT シリーズユーザーズマニュアル <ソフトウェア編> (Windows 版. INtime および DOS 版は別冊)

の 3 部構成です.

各マニュアルの内容は以下の通りです.

motionCAT シリーズユーザーズマニュアル ＜導入編＞

ー全ての開発者向け

- motionCAT 概要と通信の説明
- motionCAT 各マスタのハードウェア説明
- motionCAT 各スレーブのハードウェア説明
- 設置ガイド
- インストール
- 動かしてみる（Windows 版）
- 試運転
- アクセサリ
- 用語解説
- 各社サーボドライバの接続

別冊において以下 INtime HLS- MCT520 導入版

の差異部分を説明

- インストール
- サンプルプログラム（動かしてみるの代替）

motionCAT シリーズユーザーズマニュアル ＜運用編＞

ー主としてソフトウェア開発者向け

- マスタ設定，運用
- DIO モジュールの運用
- アナログモジュールの運用
- 位置決めモジュールの運用
- シリアル（ABS エンコーダ読込）モジュールの運用
- 各 motionCAT 通信デバイスのアクセス情報

各製品ユーザーズマニュアル ＜ソフトウェア編＞

ー主としてソフトウェア開発者向け

- ソフトウェア概要
- ソフトウェア概要
- 各モジュールサンプルプログラム（Windows 版）
- 準備
- ライブラリ関数
- ドライバ関数
- ポート資料

別冊において以下 INtime HLS-MCT520 導入版

の差異部分を説明

- ソフトウェア概要
- ライブラリ関数
- ドライバ関数
- サンプルプログラム

マニュアル更新履歴

版数	日付	更新内容	備考
1.00 版	2015/11/24	HLS- MCT520/IT 導入編・別冊 初版	
1.01 版	2022/11/14	記載事項修正，追記	

1. はじめに

本マニュアルはハイバーテック motionCAT シリーズマスタボード(以下，motionCAT マスターボード)で使用される INtime 版 HLS-MCT520/IT パッケージソフトウェアの API 関数の説明書です．ソフトウェアの基本的な運用方法等につきましては，motionCAT シリーズユーザーズマニュアル＜運用編＞などを併せてご覧ください．

1.1 ソフトウェアの構成

motionCAT マスターボードとそれに繋がるモジュール（=ローカルモジュールまたはスレーブとも表現）を INtime 上で制御するためのアプリケーション開発，動作に必要な最小ソフトウェア構成は以下となります．

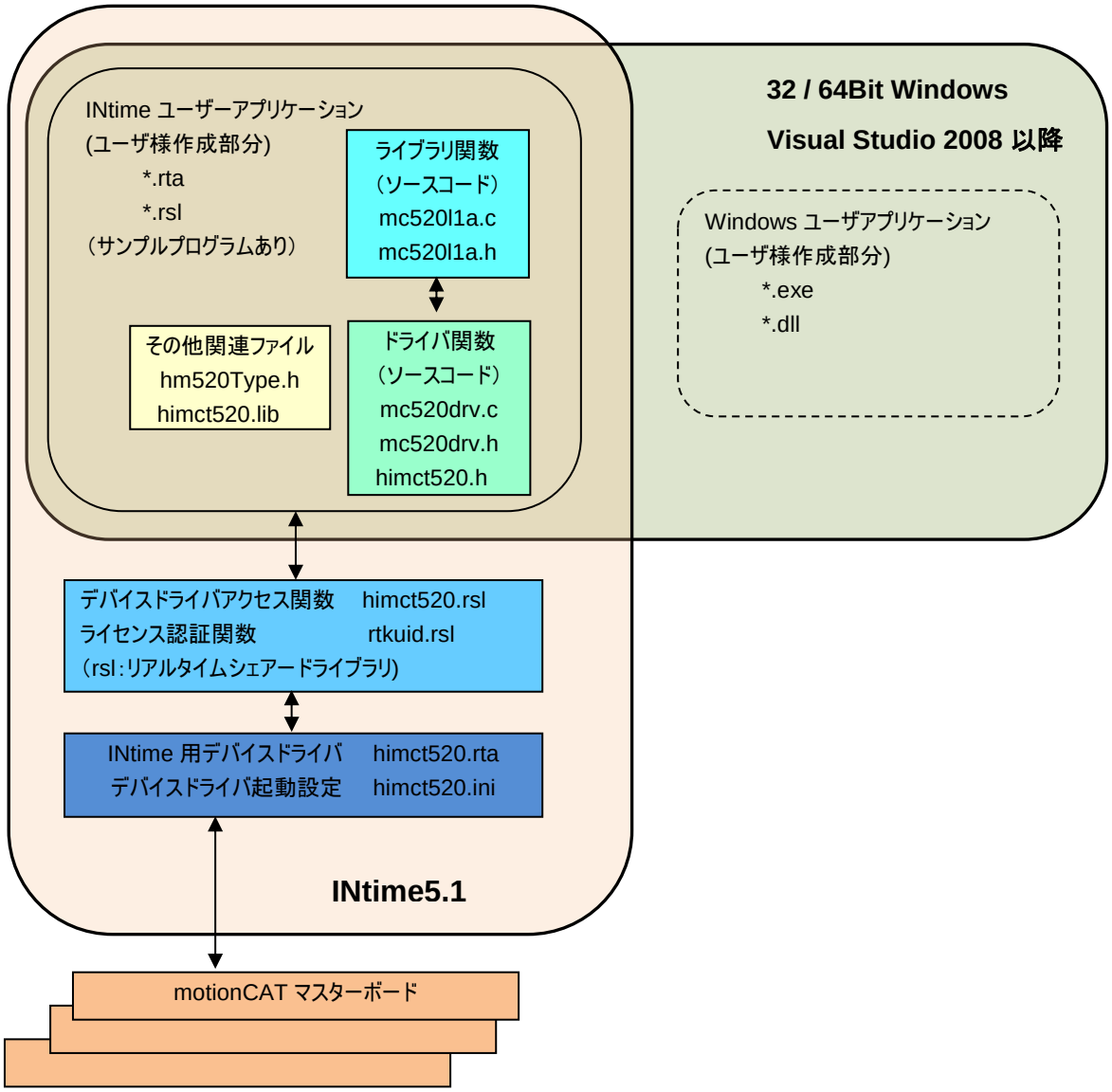


図 1.1-1 ソフトウェアの構成

1.2 Setup.exe によりコピーされるファイル

ソフトウェアパッケージ CD 内の Setup フォルダーにある Setup.exe の実行により以下のファイルが必要な場所にコピーされます.（事前に INtime および Visual Studio のインストールが必要です）

No.	ファイル名	コピー先パス	内容
1	himct520.rta	C:¥Program Files¥INtime¥Drivers¥Mct520	PCI/PCI-e/CPCI BUS motionCAT シリーズ用 INtime デバイスドライバ
2	himct520.ini	C:¥Program Files¥INtime¥Drivers¥Mct520	デバイスドライバ起動設定 ini ファイル
3	himct520.rsl	C:¥Program Files¥INtime¥bin	デバイスドライバアクセス関数 RSL ファイル
4	rtkuid.rsl	C:¥Program Files¥INtime¥bin	ライセンス認証用 INtime RSL ファイル

表 1.2-1 Setup.exe によりコピーされるファイル

サンプルプログラムとドキュメント関連ファイルにつきましては Setup.exe ではインストールされません. ソフトウェアパッケージ CD から任意の場所にコピーしてお使いください.

1.3 関数名

各関数は以下の命名規則によって関数名がつけられています.

No.	種別	関数名	提供形態
1	デバイスドライバアクセス関数	Im500_xxxx	リアルタイムシェアードライブラリ(*.rsl)
2	ドライバ関数	mx500_xxxx	ソースファイル(*.c)
3	ライブラリ関数	hmx500_xxxx	ソースファイル(*.c)

表 1.3-1 関数名

1.4 関数の戻り値

以下はライブラリ関数およびドライバ関数が返す戻り値です. この戻り値は最後に検出したコードを返します.
関数の戻り値が異常値（'0'以外)であった場合は，異常内容に対応した処理を行います. これら異常の内容によっては通常アプリケーションプログラムの続行は困難なものが含まれますので，その場合にはプログラム内容の見直しが必要になります.

No	記号表記	戻り値 16 進数表記	異常内容と確認項目
1	IEOK	0x0000	エラーなし（正常）
主に外部から渡されたデータや設置環境に起因するエラー			
2	IEDEVHDL	0x1001	指定したデバイスハンドルが無効
3	IEADRSERR	0x1002	アドレスの指定エラー
4	IESIZEERR	0x1003	サイズの指定エラー
5	IEAXISERR	0x1004	軸指定が範囲外
6	IELINEERR	0x1005	ライン指定が範囲外
7	IEMODUERR	0x1006	モジュール指定が範囲外
8	IEGROUPERR	0x1007	グループ指定が範囲外
9	IEDATAERR	0x1008	設定値が正しくない
10	IECMDERR	0x1009	コマンドが正しくない
11	IEMDLOVER	0x1101	モジュール接続数が制限を超えている
12	IEMDLNONE	0x1102	モジュールが見つからない
13	IEMDLUNMATCH	0x1103	モジュール数が一致しない
14	IENOTFOUND	0x1104	マスターボードが見つからない
15	IEBRDOVER	0x1105	マスターボードが17枚以上検出
16	IENOTRGBRD	0x1106	ボードが対象ボードでない
17	IENOTREADY	0x1107	デバイスが使用できない

18	IEMEMALLOCERR	0x1201	動的メモリが確保できない
主にプログラム処理に起因するエラー			
19	IEMSGFUNCTION	0x2001	該当する処理が存在しない
20	IEMULTPROC	0x2002	排他が必要な処理で二重起動された
21	IEDELTHREAED	0x2003	割込みタスク削除失敗
22	IECREATETHREAED	0x2004	既に割込みタスク生成済み
23	IEINTEVENTNON	0x2005	割込みイベント情報が無い
24	IETASKCONTROL	0x2006	スレッド生成に失敗
25	IEOBJHANDLE	0x2007	オブジェクトのハンドル取得に失敗
26	IEERRORPROC	0x2008	処理手順が間違っている
27	IECYCSTP	0x2101	サイクリック通信が停止しているのに操作しようとしていた
28	IEACSCOM	0x2102	システム通信中に通信を行った
29	IEPROCERR	0x2103	通信デバイスの操作手順に誤りがある
30	IEUNKNOWN	0x2104	モジュールが識別できない
31	IEBRDALRDYOPEN	0x2105	ボードが既にオープンされている
32	IEBRDOPEN	0x2106	ボードがオープンされていない
33	IEINVALIDMDL	0x2107	操作対象モジュールでない
主にデバイスやOS，ドライバーなどに起因するエラー			
34	IETIMOUT	0x4001	条件成立待ちでタイムアウト発生 */
35	IENONEDATA	0x4002	情報が存在しない(全情報読込み完了) */
36	IEINITERR	0x4003	API の初期化がされてない */
37	IENONEDRV	0x4004	ドライバが起動されてない */
38	IEMSGSEND	0x4005	MSG(メッセージ)送信でエラー */
39	IEMSGRECV	0x4006	MSG 受信でエラー */
40	IERECVTIMEOUT	0x4007	MSG 受信でタイムアウト発生 */
41	IEINTEVTMOUT	0x4008	割込みイベント待ちでタイムアウト */
42	IEINTEVSYSCALL	0x4009	割込みイベント待ちでエラー（システムコールエラー） */
43	IEMHDLCREATE	0x400a	メッセージメモリハンドルの取得失敗 */
44	IEMHDLRELEASE	0x400b	メッセージメモリハンドルの開放失敗 */
45	IERGNCREATE	0x400c	リージョン取得失敗 */
46	IERGNRELEASE	0x400d	リージョン開放失敗 */
47	IEMBOXCREATE	0x400e	メールボックスの取得失敗 */
48	IEMBOXRELEASE	0x400f	メールボックスの開放失敗 */
49	IESEMCREATE	0x4010	セマフォ作成失敗 */
50	IESEMRELEASE	0x4011	セマフォ削除失敗 */
51	IESYSCALL	0x4012	システムコールでエラー発生 */
52	IEUNDEFDEV	0x4013	アクセスしたモジュールが未使用デバイスになっている */
53	IECYCCOM	0x4101	サイクリック通信エラー */
54	IEDATCOM	0x4102	データ通信エラー */
55	IELDEVRCV	0x4103	ローカルデバイス受信エラー */
56	IEOTHERCOM	0x4104	その他通信エラー */

表 1.4-1 関数の戻り値

なお，サーボ装置・メカセンサに起因する異常(サーボアラームやエンドリミットによる停止など)はこの異常報告に含まれません．個々の要因毎に，異常発生内容を明確にすると共に，適切な処置が求められます．

1.5 アプリケーション作成準備

1.5.1 Microsoft Visual C++ (2008 以上)

INtime アプリケーションでプロジェクト作成後，次のファイルをプロジェクトへ追加します．()

No.	ファイル名	内　容	追加先
1	mc520drv.c	ドライバ関数 C ソースコードファイル	プロジェクトファイルと同じフォルダ
2	mc520drv.h	ドライバ関数ヘッダファイル	プロジェクトファイルと同じフォルダ
3	mc520l1a.c	ライブラリ関数 C ソースコードファイル	プロジェクトファイルと同じフォルダ
4	mc520l1a.h	ライブラリ関数ヘッダファイル	プロジェクトファイルと同じフォルダ
5	hm520Type.h	デバイスドライバヘッダーファイル	プロジェクトファイルと同じフォルダ
6	himct520.lib	ドライバ関数リンク用 LIB ファイル	¥Release および¥Debug フォルダ
7	himct520.h	ドライバ関数プロトタイプ定義ファイル	プロジェクトファイルと同じフォルダ

表 1.5-1　プロジェクトへ追加するファイル

上記ファイル以外に，以下のファイルが実行時に必要ですが，HLS-MCT520/IT のセットアップ時にこれらファイルが INtime フォルダーへコピーされます．したがってこれらファイルをユーザーにてプロジェクト内にコピーする必要はありません．

No.	ファイル名	内　容
1	himct520.rsl	ドライバ関数 RSL ファイル
2	rtkuid.rsl	ライセンス認証用関数 RSL ファイル
3	himct520.rta	デバイスドライバ実行ファイル
4	himct520.ini	デバイスドライバ起動設定用 INI ファイル

表 1.5-2　その他ビルドで使用するファイル

ユーザ作成のソースコード中でドライバ関数またはライブラリ関数を使用する場合は以下ファイルを#include 宣言にてヘッダーファイルをインクルードして下さい．

```
例.　　#include　"mc520drv.h"　・・・　ドライバ関数使用の場合
　　#include　"mc520l1a.h"　・・・　ライブラリ関数使用の場合
```

1.5.2　マスタボードを複数枚使用する場合

motionCAT マスターボードを 1 台のコンピュータに複数枚装着し，それぞれのボードを制御する場合について説明します．

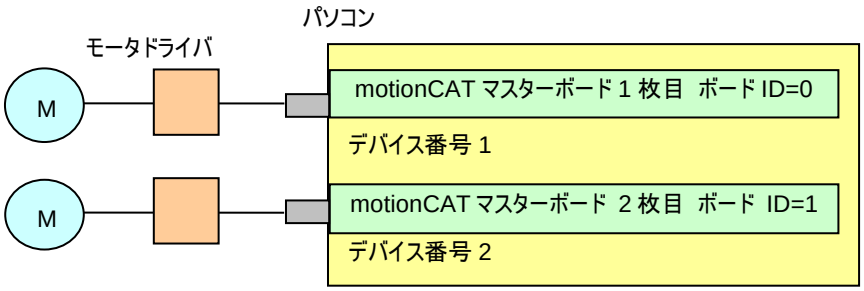


図 1.5-1　ボードを複数枚使用

(1)　ボードのスロット番号とボード ID

PCI ではシステムがボードのアドレス管理をしています．
ボードが装着されるスロットにはシステム側で決めたデバイス番号が割振られます．
しかし，このデバイス番号はシステムによって割振られるため，ボードとスロットの関係が外部から直接認識出来ません．
このために，motionCAT マスターボードには「ボード ID」が設けられています．これにより，ボードとソフトを対応させることが出来ます．

(2)　ボード ID の使用

ボード ID は No.0～15 を設定出来ます．マスターボード同士で重複する ID は使用できません．
これにより motionCAT マスターボードを合計 16 枚まで1台の PC で扱うことが可能です．

1.5.3　ボードアクセス方法

添付される API 関数は，複数の motionCAT マスターボードを制御することができます．あるひとつの motionCAT マスターボードにアクセスするためには，まずこのデバイスをオープンしてアクセスするために必要なデバイスハンドル値を取得します．
デバイスをオープンするためには,どのようなハードウェアリソースを持つデバイスをオープンするのかという情報が必要となります．
この情報をデバイス情報と呼びます．
(I/O ポートアドレスや IRQ 番号等のハードウェアリソースは,システム側によって確定されます．)

(1)　デバイス情報構造体

```
typedef struct {
        WORD   wIoMapTypePcl;       /*  ﾏｯﾋﾟﾝｸﾞﾀｲﾌﾟ:PCL */
        WORD   wIoMapTypeOpt;       /*  ﾏｯﾋﾟﾝｸﾞﾀｲﾌﾟ:ｵﾌﾟｼｮﾝﾎﾟｰﾄ */
        BYTE*  pbyIoAdrPcl;         /* I/O  ｱﾄﾞﾚｽ:PCL */
        BYTE*  pbyIoAdrOpt;         /* I/O  ｱﾄﾞﾚｽ:ｵﾌﾟｼｮﾝﾎﾟｰﾄ */
        WORD   dwBoardType;         /*  ﾎﾞｰﾄﾞﾀｲﾌﾟ */
        WORD   wLineCnt;            /*  ﾎﾞｰﾄﾞ回線ﾗｲﾝ数 */
        WORD   wBusNumber;          /*  ﾊﾞｽ番号 */
        WORD   wDevNumber;          /*  ﾃﾞﾊﾞｲｽ番号 */
        WORD   wIrqNo;              /* IRQ  番号 */
        WORD   wNumber;             /*  管理番号 */
        WORD   wBoardID;            /*  ﾎﾞｰﾄﾞ ID */
        WORD   wMsiOffset;          /* MSI ｵﾌｾｯﾄ */
} HPC_DEVINFO, *PHPC_DEVINFO, *LPHPC_DEVINFO;
```

この構造体はﾌｧｲﾙ“hm520Type.h”で定義しています．

(2)　関数のリエントラント性および排他について

INtime はマルチスレッドで動作する事が基本となるので，HLS-MCT520/IT に添付されるドライバ関数，およびそれを使用しているライブラリ関数はリエントラント性を持ちます．ドライバ関数はセマフォによりボードおよびライン毎の排他を行っています．
また複数のスレッドにより異なるボードやラインで同時にドライバ関数を使用した場合は，デバイスドライバに先取りされたドライバ関数の処理がまず実行されます．それ以外の関数は待ち行列にキューイングされます．デバイスドライバは，実行中の関数処理が終了するとすぐにキューイングされた関数を実行，待ち行列内のドライバ関数を順番に処理します．
これら処理によってデバイスに対する排他を行っています．

2. ライブラリ関数

2.1 ライブラリ関数について

ライブラリ関数は，Visual Studio2008 Vc／Vc++で作成されていて，ソースコード（mc520l1a.cファイルおよび mc520l1a.h ファイル）の形で提供されます．ご使用時にはユーザー様が作成するアプリケーションに組み込んでビルドします．
このライブラリー関数は，次章のドライバー関数を使用して作られているマスターボードおよびモーションモジュール制御用の応用関数です．

2.1.1 ライン番号，モジュール ID の指定

初期設定，状態読み出し，動作設定，運用設定等の各関数のラインおよびモジュール ID 指定は

・ライン（Line） ・・・ 0：ライン 1， 1：ライン 2

・モジュール ID（Mid） ・・・ 0～63

また複数モジュールの情報をﾋｯﾄ単位で扱う“入力変化割込設定”，“入力変化割込ﾌﾗｸﾞ”，“通信エラーﾌﾗｸﾞｸﾘｱ”，“通信エラーﾌﾗｸﾞ”Read/Write 時には，

・入力変化割込関係 ・・・ 0：モジュール 0～3， 1：モジュール 4～7，・・・・15：モジュール 60～63

・通信エラー関係 ・・・ 0：モジュール 0～15， 1：モジュール 16～31・・・・3:モジュール 48～63

となります．

2.1.2 ライブラリ関数

ライブラリ関数は，ソースコードで提供されます．

(1) 関数一覧

No	関 数 名		機　　能	記載項
L1	デバイス操作	hmx_GetDevInfo	マスターボード枚数，デバイス情報の取得	2.2.1
L2		hmx_DevOpen	マスターボードのオープン，レジスとオプションポートのタ初期化	2.2.2
L3		hmx_DevClose	マスターボードのクローズ	2.2.3
L4	モーションモジュール初期設定	hmx_InitMotionModul	原点復帰モードの設定	2.3.1
L5		hmx_SetOrgMode	原点復帰モードの設定	2.3.2
L6		hmx_SetEls	ELSの設定	2.3.3
L7		hmx_SetOls	OLSの設定	2.3.4
L8		hmx_SetDls	DLS／PCS入力選択，設定	2.3.5
L9		hmx_SetLtc	LATCH入力の設定	2.3.6
L10		hmx_SetClr	CLR入力の設定	2.3.7
L11		hmx_SetCmp	コンパレータ条件の設定	2.3.8
L12		hmx_SetEz	エンコーダZ相の設定	2.3.9
L13		hmx_SetInpos	INPOSの設定	2.3.10
L14		hmx_SetSvAlm	SVALMの設定	2.3.11
L15		hmx_SetSvCtrCl	偏差カウンタクリア出力の設定	2.3.12
L16		hmx_SetSvRdy	SVRDYの設定	2.3.13
L17		hmx_SetCmdPulse	指令パルスの出力形式の設定	2.3.14
L18		hmx_SetAccProfile	加減速形式設定	2.3.15
L19		hmx_SetAutoDec	減速開始点の設定方式	2.3.16
L20		hmx _SetSls	ソフトリミットの設定	2.3.17
L21		hmx_SetCtr3	カウンタ3入力の選択	2.3.18
L22		hmx_SetFHAdj	FH補正機能のON／OFF	2.3.19
L23	モーションモジュール状態読み出し	hmx_ReadMainSts	モーションメインステータス読出し	2.4.1
L24		hmx_ReadErrorSts	エラーステータスの読出し	2.4.2
L25		hmx_ReadEventSts	イベントステータスの読出し	2.4.3
L26		hmx_ReadExSts	拡張ステータスの読出し	2.4.4
L27		hmx_ReadOutp	汎用出力ポート状態読込み	2.4.5
L28		hmx_ReadSpd	指令速度の読出し	2.4.6
L29		hmx_ReadSpdEx	指令速度の読出し(速度倍率含む)	2.4.7
L30		hmx_ReadCtr	カウンタの読出し	2.4.8
L31	モーションモジュール動作設定	hmx_SetFLSpd	ベース速度の設定	2.5.1
L32		hmx_SetAuxSpd	補助速度の設定	2.5.2
L33		hmx_SetAccRate	加速レートの設定	2.5.3
L34		hmx_SetDecRate	減速レートの設定	2.5.4
L35		hmx_SetMult	倍率倍率レジスタ値の設定	2.5.5
L36		hmx_SetEventMask	イベントマスクの設定	2.5.6
L37		hmx_SetDecPoint	減速開始点の設定	2.5.7
L38		hmx_SetCnstIPFeed	2軸補間時の合成速度一定制御のON/OFF	2.5.8
L39		hmx_WritOpeMode	動作モードの設定	2.5.9
L40		hmx_WritSta	STA入力時の動作設定	2.5.10
L41		hmx_WritStp	STP入力時の動作設定と異常停止時のSTP自動出力設定	2.5.11
L42		hmx_WritFHSpd	動作速度の設定	2.5.12
L43		hmx_WritPos	位置決め移動量の設定	2.5.13
L44		hmx_WritCtr	カウンタプリセット	2.5.14
L45		hmx_Writ2AxisLine	2軸直線補間移動量設定	2.5.15
L46		hmx_WritCircl	2軸円弧補間終点，中心位置設定	2.5.16
L47		hmx_DecStop	減速停止	2.6.1
L48		hmx_QuickStop	即停止	2.6.2
L49		hmx_EmgStop	非常停止	2.6.3
L50		hmx_AccStart	加速スタート後減速停止	2.6.4
L51		hmx_CnstStartFH	FH定速スタート	2.6.5
L52		hmx_CnstStartFL	FL定速スタート	2.6.6

L53	モーションモジュール 動作制御	hmx_CnstStartByDec	FH定速スタート後減速停止	2.6.7
L54		hmx_MvAccStart	移動量設定+加速スタート後減速停止	2.6.8
L55		hmx_MvCnstStartFH	移動量設定+FH定速スタート	2.6.9
L56		hmx_MvCnstStartFL	移動量設定+FL定速スタート	2.6.10
L57		hmx_MvCnstStartBy	移動量設定+FH定速スタート後減速停止	2.6.11
L58		hmx_SetGroup	グループ設定（個別）	2.6.12
L59		hmx_SetGroupEx	グループ設定（一括）	2.6.13
L60		hmx_GrpStop	グループ停止	2.6.14
L61		hmx_GrpStart	グループスタート	2.6.15
L62		hmx_SvOn	サーボオン信号オン	2.6.16
L63		hmx_SvOff	サーボオン信号オフ	2.6.17
L64		hmx_SvResetOn	サーボリセット信号オン	2.6.18
L65		hmx_SvResetOff	サーボリセット信号オフ	2.6.19
L66		hmx_SvTlOn	サーボトルク制限信号オン	2.6.20
L67		hmx_SvTlOff	サーボトルク制限信号オフ	2.6.21
L68		hmx_SvGainOn	サーボゲイン切替信号オン	2.6.22
L69		hmx_SvGainOff	サーボゲイン切替信号オフ	2.6.23
L70		hmx_PMOn	パルスモータ励磁オン	2.6.24
L71		hmx_PMOff	パルスモータ励磁オフ	2.6.25
L72		hmx_ResetSEND	モーションモジュールSENDリセット	2.6.26
L73	演算処理	hmx_CalAccRate	加減速レートの計算	2.7.1
L74		hmx_CalDecPoint	減速開始点の計算	2.7.2

表 2.1-1 ライブラリ関数一覧表

(2) ライブラリ関数の戻り値

関数の起動を行った結果は「戻り値」に実行結果が反映されます.
この戻り値はライブラリ関数およびドライバ関数で共通です.
戻り値が '0'の場合は正常終了ですが, '0' 以外の場合には何らかの異常が発生しています. 詳細は「1.4 関数の戻り値」をご覧ください

2.2デバイス操作

2.2.1 hmx_GetDevInfo() マスターボード枚数，デバイス情報の取得

No.	L1
機　能	パソコンに装着され，INtime の管理下にある motionCAT マスターボードの枚数及びデバイス情報を取得します．
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発言語	書　式
VC++	short　hmx_GetDevInfo(WORD* BrdNum, HPC_DEVINFO* BrdInfo);

引　数	説　明
BrdNum	[OUT] motionCAT マスターボードの枚数
BrdInfo	[OUT] motionCAT マスターボードのデバイス情報．motionCAT マスターボードの実装枚数以上の領域を用意する必要があります．(最大で 16 枚まで実装される可能性があります)

VC++
記述例

```
WORD   count;
short     ret;
HPC_DEVINFO   AllBrdInfo[16];          // 実装する全てのボードのデバイス情報を格納する
                                       // エリアの先頭アドレス
ret = hmx_GetDevInfo( &count, &AllBrdInfo[0] );
```

2.2.2 hmx_DevOpen() マスターボードのオープン，レジスタとオプションポートの初期化

No.	L2
機　能	指定したデバイス情報を持つ motionCAT マスターボードをオープンして，他のマスタボードと識別するためのデバイスハンドルを取得します．以降このデバイスハンドルは指定した motionCAT マスターボードにアクセスするために使用します．同時にシステム通信を行い，各ラインに接続する全モジュールのモジュール情報を取得します． これらが正常に終了するとサイクリック通信を開始します．
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発言語	書　式
VC++	short　hmx_DevOpen(DWORD* hDevID, HPC_DEVINFO* BrdInfo , HPC_MODUINF* ModuInfo);

引　数	説　明
hDevID	[OUT] デバイスハンドル
BrdInfo	[IN]　motionCAT マスターボードのデバイス情報
ModuInfo	[OUT] motionCAT マスターボードに接続しているモジュールのデバイス情報

VC++
記述例

```
// パソコンに motionCAT マスターボードが 2 枚装着されていることを想定します．
// デバイス情報格納エリアとしてデバイス情報構造体の配列 BrdInfo[2]を準備し，この中には
// 既に hmx_GetDevInfo 関数により取得したデバイス情報が BrdInfo[2]にコピーされ
// 入っているものとします．
// 2 枚の motionCAT マスターボードのそれぞれのモジュール情報格納用として ModuInfo[2]を用意します
short     ret;                  //関数の戻り値
DWORD   hDevID[2];        //デバイスハンドル取得エリア
HPC_DEVINFO   BrdInfo[2];
HPC_MODUINF   ModuInfo[2];
ret = hmx_DevOpen( hDevID[0],  &DevInfo[0] ,  &DevInfo[0] );
          //1 番目 motionCAT マスターボードのデバイスハンドルとモジュール情報
ret = hmx_DevOpen( hDevID[1],  &DevInfo[1] ,  &DevInfo[1 );
          //2 番目 motionCAT マスターボードのデバイスハンドルとモジュール情報
```

2.2.3 hmx_DevClose()　マスターボードのクローズ

No.	L3
機　能	デバイスハンドルで指定された motionCAT マスターボードをクローズします. 以降このデバイスハンドルは無効となります.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発言語	書　式
VC++	short　hmx_DevClose(DWORD hDevID);

引　数	説　明
hDevID	[IN] デバイスハンドル

備　考	デバイスクローズの前に motinCAT ボードに対する終了処理を行ってください.

2.3 モーションモジュール初期設定

2.3.1 hmx_InitMotion() モーションモジュールの初期化

No.	L4
機　能	MotionCAT マスターボードに接続した指定ライン上の指定モジュールの初期化を行います.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	Short hmx_InitMotionModule(DWORD hDevID, WORD wLine, WORD wMid, WORD wCFunc);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]
wCFunc	[IN] 使用レジスタ(0：P モジュール互換アクセス 1：プリレジスタアクセス) ※C/F/(W)モジュール指定時のみ有効

備　考	モーションモジュールの初期設定値			
	レジスタ	内　容	初期値	補　足
	RMV(, PRMV)	移動量	0	
	RFL(, PRFL)	ベース速度	200	200pps
	RFH(, PRFH)	動作速度	2000	2000pps
	RUR(, PRUR)	加速レート	1364	直線加(減)速時 200→2000pps(2000pps→200pps) 加速(減速)時間：約 0.5 秒
	RDR(, PRDR)	減速レート	0	加速レートと同じ
	RMG(, PRMG)	速度倍率	299	1 倍
	RDP(, PRDP)	減速ポイント	0	
	RFA	補助速度	200	200pps
	RMD(, PRMD)	動作モード	08008000h	
	RENV1	環境設定 1	01434004h	CW/CCW出力,DLS,OLS,SVALM:B接,ELS, SVALM入力時即停止,DLSﾗｯﾁしない, 原点復帰時及び異常終了時SVCTRCL自動出力しない,SVCTRCL出力ﾊﾟﾙｽ幅 ms, INPOS,SVRDY:A接,CLR,LTCH:立下りｴｯｼﾞ
	RENV2	環境設定 2	000004ffh	ｴﾝｺｰﾀﾞ入力逓倍ｶｳﾝﾄ
	RENV3	環境設定3	00700002h	原点復帰ﾓｰﾄﾞ=1,原点復帰時 CTR 自動ｸﾘｱ
	RIRQ	イベントマスク設定	00000001h	正常停止時
	その他		0	
	オプションポート（汎用出力）		初期値	補　足
	ELS 入力極性		0	B 接
	SVON/SVRST/SVIL/SVGAIN		0	OFF
	その他		0	

2.3.2 hmx_SetOrgMode() 原点復帰モードの設定

No.	L5
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの原点復帰モードを設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetOrgMode(DWORD hDevID, WORD wLine, WORD wMid, WORD wMode);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wMode	[IN] 原点復帰モード 0:ORGmode0　OLSoff→on で即停止(加減速動作時は減速停止) 1:ORGmode1　OLSoff→on で即停止(加減速動作時は減速停止)後, 補助速度定速で逆方向へ OLSoff まで動作し, その後補助速度で初めの方向へ動作し OLSoff→on で即停止 2:ORGmode2　定速時は OLSoff→on 後の Z 相カウントアップで即停止 加減速動作時は OLSoff→on で減速, Z 相カウントアップで即停止 3:ORGmode3　定速時は OLSoff→on 後の Z 相カウントアップで即停止 加減速動作時は OLSoff→on で減速, Z 相カウントアップで減速停止 4:ORGmode4　OLSoff→on で即停止(加減速動作時は減速停止)後に補助速度定速で逆転, OLSon→off 後の Z 相カウントアップ時に即停止 5:ORGmode5　OLSoff→on で即停止(加減速動作時は減速停止)後に逆転, OLSon→off 後の Z 相カウントアップ時に即停止(加減速動作時は減速停止) 6:ORGmode6　ELSon で停止後, 補助速度定速で逆転, ELSoffFで即停止 7:ORGmode7　ELSon で停止後, 補助速度定速で逆転、ELSoff 後の Z 相カウントアップ時に即停止 8:ORGmode8　ELSon で停止後に逆転, ELSoff 後の Z 相カウントアップで即停止 (加減速動作時は減速停止) 9:ORGmode9　ORGmode0 の動作後, 機械位置(CTR2)0 点復帰 10:ORGmode　ORGmode3 の動作後, 機械位置(CTR2)0 点復帰 11:ORGmode　ORGmode5 の動作後, 機械位置(CTR2)0 点復帰 12:ORGmode　ORGmode8 の動作後, 機械位置(CTR2)0 点復帰

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 1 モジュール 2 を指定, ORGmode1 (OLS 検出後抜出し再突入原点完了)
	ret = hmx_SetOrgMode(hDevID, 0, 2, 1);

2.3.3 hmx_SetEls() ELS の設定

No.	L6
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの ELS 入力極性と入力時停止方法を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetEls(DWORD hDevID, WORD wLine, WORD wMid, WORD wPol, WORD wStop);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wPol	[IN] 入力極性[0:B 接, 1:A 接]
wStop	[IN] 停止方法[0:即停止, 1:減速停止]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetEls(hDevID, 1, 5, 1, 0); //ライン 2 モジュール 5 を指定, A 接, 即停止

2.3.4 hmx_SetOls() OLS の設定

No.	L7
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの OLS 入力極性を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetOls(DWORD hDevID, WORD wLine, WORD wMid, WORD wPol);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wPol	[IN] 入力極性[0:B 接, 1:A 接]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetOls(hDevID, 0, 3, 1); // ライン 1 モジュール 3 を指定, A 接

2.3.5 hmx_SetDls() DLS の設定

No.	L8
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの DLS 入力極性を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetDls(DWORD hDevID, WORD wLine, WORD wMid, WORD wEnbl, WORD wPol, WORD wStop, WORD wLtc);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wEnbl	[IN] DLS 有効[0:B 無効, 1:有効]
wPol	[IN] 入力極性[0:B 接, 1:A 接]
wStop	[IN] 停止方法[0:即停止, 1:減速停止]
wLtc	[IN] DLS 入力ラッチ有効[0:B 無効, 1:有効]

VC++
记述例
	short ret; //関数の戻り値
	ret = hmx_SetDls(hDevID, 1, 0, 1, 0); // ライン 2 モジュール 0 を指定, DLS 有効 B 接

2.3.6 hmx_SetLtc() LATCH 入力の設定

No.	L9
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのラッチ入力極性を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetLtc(DWORD hDevID, WORD wLine, WORD wMid, WORD wPol);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wPol	[IN] 入力極性[0:B 接, 1:A 接]

VC++
记述例
	short ret; //関数の戻り値
	ret = hmx_SetLtc(hDevID, 0, 4, 1); // ライン 1 モジュール 4 を指定, DLS 有効 A 接

2.3.7 hmx_SetClr() CLR 入力の設定

No.	L10
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのカウンタクリア入力極性とクリアされるカウンタの設定をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetClr(DWORD hDevID, WORD wLine, WORD wMid, WORD wPol, WORD wCtr);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wPol	[IN] 入力極性[0:B 接, 1:A 接]
wCtr	[IN] CLR 入力時にクリアするカウンタ指定(ビット対応)[Bit0:カウンタ 1, Bit1:カウンタ 2, Bit2:カウンタ 3]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetClr(hDevID, 0, 0, 1, 3); // ライン 0 モジュール 0 を指定, A 接,
	// CLR 入力時カウンタ 1(指令),2(ENC 入力)をクリア

2.3.8 hmx_SetCmp() コンパレータ条件の設定

No.	L11
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのコンパレータ比較条件を設定します.(使用するコンパレータはコンパレータ 3 固定)
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetCmp(DWORD hDevID, WORD wLine, WORD wMid, WORD wCtr, WORD wCon, WORD wIndex, long lCmp);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wCtr	[IN] コンパレータと比較するカウンタ[0:カウンタ 1, 1:カウンタ 2, 2:カウンタ 3]
wCon	[IN] コンパレータ条件[0:無効, 1:CMP3=CTR , 2:CMP3=CTR(ｶｳﾝﾄｱｯﾌﾟ), 3:CMP3=CTR(ｶｳﾝﾄﾀﾞｳﾝ), 4:MP3>CTR , 5:MP3<CTR]
wIndex	[IN] 同期(定ﾋﾟｯﾁ出力)有効/無効[0:無効 , 1:有効]
lCmp	[IN] コンパレータ比較値[[パルス数 -134,217,728～134,217,727, ただしカウンタ 3 選択時は 0～32767, 定ピッチ出力有効時は無効]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetCmp(hDevID, 1, 1, 0, 1, 0, 25000);
	// ライン 2 モジュール 1 を指定, カウンタ 1(指令)使用,
	// カウンター致, 等ピッチ出力無効, 比較値 25000

2.3.9 hmx_SetEz() エンコーダZ相の設定

No.	L12
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのエンコーダZ相入力条件を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetEz(DWORD hDevID, WORD wLine, WORD wMid, WORD wCount, WORD wPol);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1，1;Line2]
wMid	[IN] モジュール ID[0～63]
wCount	[IN] 原点復帰で使用する Z 相ｶｳﾝﾄ回数[0～15]
wPol	[IN] 入力極性[0:B 接，1:A 接]

VC++ 記述例	short ret; //関数の戻り値
	ret = hmx_SetEz(hDevID, 0, 5, 1, 1); // ライン 1 モジュール 5 を指定，Z 相カウント回数，A 接

2.3.10 hmx_SetInpos () INPOS の設定

No.	L13
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの INPOS の入力信号処理方法を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	Short hmx_SetInpos(DWORD hDevID, WORD wLine, WORD wMid, WORD wEnbl, WORD wPol);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1，1;Line2]
wMid	[IN] モジュール ID[0～63]
wEnbl	[IN] INPOS 制御[0:OFF,1:ON]
wPol	[IN] 入力極性[0:B 接，1:A 接]

VC++ 記述例	short ret; //関数の戻り値
	ret = hmx_SetInpos(hDevID, 0, 3, 1, 1);
	//ライン 1 モジュール 3 を指定，INPOS 制御 ON，A 接

2.3.11 hmx_ SetSvAlm () SVALM の設定

No.	L14
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの SVALM 入力極性と入力時停止方法を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetSvAlm(DWORD hDevID, WORD wLine, WORD wMid, WORD wPol, WORD wStop);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wPol	[IN] 入力極性[0:B 接, 1:A 接]
wStop	[IN] 停止方法[0:即停止, 1:減速停止]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetSvAlm(hDevID, 0, 3, 1, 0); //ライン 1 モジュール 3 を指定, A 接, 即停止

2.3.12 hmx_SetSvCtrCl() 偏差カウンタクリア出力の設定

No.	L15
機　能	デバイスハンドルで指定されたmotionCATマスターボードに接続した指定ライン上の指定モジュールの偏差カウンタクリア自動出力の設定をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetSvCtrCl (DWORD hDevID, WORD wLine, WORD wMid, WORD wEnbl);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wEnbl	[IN] 自動出力設定[0:不使用,1:原点完了時,2:異常停止時,3:原点完了及び異常停止時]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetSvCtrCl(hDevID, 1, 0, 1); // ライン 2 モジュール 0 を指定, 原点完了時出力

2.3.13 hmx_SetSvRdy()　　サーボレディの設定

No.	L16
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの SVRDY 入力極性の設定をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SetSvRdy(DWORD hDevID, WORD wLine, WORD wMid, WORD wPol);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wPol	[IN] 入力極性［0:B 接，1:A 接］

VC++ 記述例	short　ret;　　//関数の戻り値 ret = hmx_SetSvRdy(hDevID, 1, 5, 1);　　//ライン 2 モジュール 5 を指定，A 接

2.3.14 hmx_SetCmdPulse()　　指令パルス出力形式の設定

No.	L17
機　能	デバイスハンドルで指定されたmotionCATマスターボードに接続した指定ライン上の指定モジュールの指令パルスの出力形式を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	Short　hmx_SetCmdPulse(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmdPls);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wCmdPls	[IN] 指令パルスの出力形式[0:個別指令方式,1:共通指令方式]

VC++ 記述例	short　ret;　　//関数の戻り値 ret = hmx_SetCmdPulse(hDevID, 0, 5, 1);　//ライン 1 モジュール 5 を指定，共通指令方式

2.3.15 hmx_SetAccProfile()　加減速形式の設定

No.	L18
機　能	デバイスハンドルで指定されたmotionCATマスターボードに接続した指定ライン上の指定モジュールの加減速形式の設定をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SetAccProfile (DWORD hDevID, WORD wLine, WORD wMid, WORD wPro);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1，1;Line2]
wMid	[IN] モジュール ID[0～63]
wPro	[IN] 加減速形式[0:直線,1:S 字]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SetAccProfile(hDevID, 0, 2, 1);　//ライン 1 モジュール 2 を指定，S 字加減速に設定

2.3.16 hmx_SetAutoDec()　減速開始点設定方式

No.	L19
機　能	デバイスハンドルで指定されたmotionCATマスターボードに接続した指定ライン上の指定モジュールの減速開始点計算方式を手動計算か自動計算か設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SetAutoDec(DWORD hDevID, WORD wLine, WORD wMid, WORD wAuto);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1，1;Line2]
wMid	[IN] モジュール ID[0～63]
wAuto	[IN] 減速開始点の設定方式[0:自動計算設定,1:手動計算設定]

VC++
記述例
	short　ret;　//関数の戻り値
	ret = hmx_SetAutoDec(hDevID, 1, 2, 1);
	//ライン 2 モジュール 2 の減速開始点の設定を手動計算設定

2.3.17 hmx_SetSls()　ソフトリミットの設定

No.	L20
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのソフトリミットの設定をします.

開発言語	書　式
VC++	Short　hmx_SetSls(DWORD hDevID, WORD wLine, WORD wMid, long lPsl, long lMsl, WORD wEnbl, WORD wStop);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1，1;Line2]
wMid	[IN] モジュール ID[0～63]
lPsls	[IN] ＋SLS[パルス数]
lMsls	[IN] －SLS[パルス数]
wEnbl	[IN] 使用/不使用[0:不使用,1:使用]
wStop	[IN] 停止方法[0:即停止,1:減速停止]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 2 モジュール 4, +SLS = 100000, -SLS = -50000，ソフトリミット使用，即停止
	ret = hmx_SetSls (hDevID, 1, 4, 100000, -50000, 1, 0);

備　考	1.ソフトリミット使用時は+SLS は必ず－SLS より大きくして下さい. 2.ソフトリミット不使用時は lMsls = lPsls = wEnbl = 0 とします. 3.スタートコマンド書込み時に SLS が ON 状態の場合，SLS が ON になる方向へはスタートはできません(動きません). 逆方向へはスタートできます.

2.3.18 hmx_SetCtr3()　カウンタ 3 入力の選択

No.	L21
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのカウンタ 3 の設定をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SetCtr3(DWORD hDevID, WORD wLine, WORD wMid, WORD wSrc);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1，1;Line2]
wMid	[IN] モジュール ID[0～63]
wSrc	[IN] カウンタ 3 入力ｿｰｽ[0:指令ﾊﾟﾙｽ,1:ｴﾝｺｰﾀﾞ入力,2:予約,3:予約,4:指令ﾊﾟﾙｽとｴﾝｺｰﾀﾞの偏差ｶｳﾝﾄ]

VC++
記述例
	short　ret;　//関数の戻り値
	ret = hmx_SetCtr3(hDevID, 1, 3, 1);
	//ライン 2 モジュール 3 のカウンタ 3 入力ソースをエンコーダに設定

2.3.19 hmx_SetFHAdj()　FH補正機能の ON／OFF

No.	L22
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの FH 補正機能の設定をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SetFHAdj(DWORD hDevID, WORD wLine, WORD wMid, WORD wEnbl);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wEnbl	[IN] FH 補正機能［0:OFF,1:ON］

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SetFHAdj(hDevID, 1, 5, 0);　//ライン 2 モジュール 5, FH 補正機能 OFF

2.4 モーションモジュール状態読み出し

2.4.1 hmx_ReadMainSts() モーションメインステータスの読出し

No.	L23
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのメインステータスを読出します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_ReadMainSts (DWORD hDevID, WORD wLine, WORD wMid, WORD* wSts);

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1, 1;Line2］		
wMid	[IN] モジュール ID[0～63]		
wSts	[OUT] メインステータス		
	ビット	名称	説明
	0	SINT	'1':割り込み発生. ビット 1,2,3 のいずれかが1
	1	SEND	'1':動作停止. 割り込み(リセットコマンド(0008h)で 0)
	2	SERR	'1':エラー報告あり(エラーステータス(REST)読出しで'0')
	3	SEVT	'1':イベント報告あり(イベントステータス(RIST)読出しで'0')
	8	SBSY	'1':パルス出力開始, '0':動作停止
	*1. 正常終了でのイベント報告は"イベントマスク設定:自動停止"［RIRQ.b0:IREN=1］とします. *2. b1(SEND)は状態を示しているビットです. 電源投入直後は '0' であり, 即(減速)停止指令の実行または一度移動実行後の終了状態で'1'となります. 移動中は '0 'を示します. 通常停止中(= '1')か, 動作中(= '0')かを確認したいときに使用します.		

VC++
記述例
	short ret; //関数の戻り値
	WORD msts; //メインステータス
	ret = hmx_ReadMainSts(hDevID, 1, 1, &msts); //ライン 2 モジュール 1 を指定

2.4.2 hmx_ReadErrorSts() エラーステータスの読出し

No.	L24
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのエラーステータスを読出します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_ReadErrorSts(DWORD hDevID, WORD wLine, WORD wMid, DWORD* dwSts);

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号[0:Line1, 1;Line2]		
wMid	[IN] モジュール ID[0～63]		
dwSts	[OUT] エラーステータス		
	ビット	名　称	説　　明
	0	ESC1	+SLS による停止時
	1	ESC2	-SLS による停止時
	2	ESC3	CMP3 による停止時
	3	ESPL	+ELS による停止時
	4	ESML	-ELS による停止時
	5	ESAL	SVALM による停止時
	6	ESSP	STP 入力による停止時
	7	ESEM	EMG 入力による停止時
	8	ESSD	DLS 入力による停止時
	9	ESPO	手動パルサバッファオーバーフローによる停止時
	10	ESNT	通信エラー発生時による停止時
	12	ESOR	位置のオーバーライド失敗時
	13	ESEE	エンコーダ入力(EA/EB)異常時(停止しない)
	14	ESPE	パルサ入力(PA/PB)異常時(停止しない)
	15	ESDT	(C/F/(W)モジュールのみ) 補間動作データ異常による停止時
	16	ESIP	(C/F/(W)モジュールのみ) 他軸の異常停止による同時停止時
	17	ESAO	(C/F/(W)モジュールのみ) 円弧補間範囲オーバによる停止時
	18	ESAJ	(C/F/(W)モジュールのみ) クロック同期エラー発生(停止しない)
	19	vKM	(C/F/(W)モジュールのみ) クロック同期エラーモニタc設定回数分連続エラー発生による停止
	20	ESPM	(C/F/(W)モジュールのみ) 同時停止用エラーモニタ止設定回数分連続エラー発生による停止
	21	ESWM	(C/F/(W)モジュールのみ) SW 代行用の通信モニタが設定回数分連続発生による停止
	以下の場合に ESDT=1 になります. 円弧補間モードで RIP 設定(円弧中心座標)を(0,0)にしてスタートコマンドを書き込んだ時 または円弧補間モードで RIP 設定(円弧中心座標)が終点座標設定と等しい時		

VC++
記述例
	short ret; //関数の戻り値
	DWORD ests; //エラーステータス
	ret = hmx_ReadErrorSts(hDevID, 1, 2, &ests); //ライン 2 モジュール 2 を指定

2.4.3 hmx_ReadEventSts() イベントステータスの読出し

No.	L25
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのイベントステータスを読出します.

開発言語	書　式
VC++	short hmx_ReadEventSts(DWORD hDevID, WORD wLine, WORD wMid, DWORD* dwSts);

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号[0:Line1, 1;Line2]		
wMid	[IN] モジュール ID[0～63]		
dwSts	[OUT] イベントステータス		
	ビット	名　称	説　　明
	0	ISEN	1:動作完了
	1	ISUS	1:加速開始
	2	ISUE	1:加速終了
	3	ISDS	1:減速開始
	4	ISDE	1:減速終了
	5	ISC1	1:CMP1 比較条件成立(+SLS)
	6	ISC2	1:CMP2 比較条件成立(-SLS)
	7	ISC3	1:CMP3 比較条件成立(脱調検出用途)
	8	ISRCL	1:CLR 信号 ON によるカウンタ値リセット
	9	ISLT	1:LTC 信号 ON によるカウンタ値ラッチ
	10	ISOL	1:OLS 信号 ON によるカウンタ値ラッチ
	11	ISSD	1:DLS 信号 OFF→ON
	12	ISSA	1:STA 信号 OFF→ON
	13	ISNA	(C/F/(W)モジュールのみ) 1:同報通信スタートコマンド(2x01h)によるスタート
	14	ISNP	(C/F/(W)モジュールのみ) 1:同報通信ストップコマンド(2x02h)によるストップ
	15	ISNM	(C/F/(W)モジュールのみ) 1:動作用プリレジスタ書き込み可能(MSTS SPRF ビット変化)
	16	ISBE	(C/F/(W)モジュールのみ) 1:プリレジスタが未設定で現動作完了 (補間の連続実行時に次動作データの設定が間に合わなかったとき)
	イベントステータスを使用するためにはイベントマスクの設定が必要です.		

VC++
記述例
	short ret; //関数の戻り値
	DWORD ists; //サブステータス
	ret = hmx_ReadEventSts(hDevID, 1, 2, &ists); //ライン 2 モジュール 2 を指定

2.4.4 hmx_ReadExSts() 拡張ステータスの読出し

No.	L26
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの拡張ステータスを読出します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_ReadExSts(DWORD hDevID, WORD wLine, WORD wMid, DWORD* dwSts);

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1, 1;Line2］		
wMid	[IN] モジュール ID[0～63]		
dwSts	[OUT] 拡張ステータス		
	ビット	名　称	説　　明
	3-0	CND3-0	動作状態 0000:停止中　1000:加速中 0001:STA 入力待ち　1001:FH 定速動作中 0010:ERC タイマ完了待ち　1010:減速中 0011:方向変化タイマ完了待ち　1011:INPOS ON 待ち 0100:バックラッシュ補正中　1100:未定義 0101:パルサ(PA/PB)入力待ち　1101:未定義 0110:FA 定速動作中　1110:未定義 0111:FL 定速動作中　1111:その他(スタート制御中)
	4	SDIR	動作方向　0:＋方向, 1:－方向
	5	SALM	1:ALM ON
	6	APEL	1:+ELS ON
	7	SMEL	1:-ELS ON
	8	SORG	1:OLS ON
	9	SSD	1:DLS ON（ラッチの状態）
	10	SDIN	1:DLS ON（端子の状態）
	11	SSTA	1:同時スタート信号(STA) on
	12	SSTP	1:同時停止信号(STP) on
	13	SEMG	1:EMG ON
	14	SPCS	1:PCS ON
	15	SERC	1:ERC ON
	16	SEZ	1:EZ ON
	17	SCLR	1:CLR ON
	18	SLTC	1:LTC ON
	19	SINP	1:INPOS ON
	20	SCP1	1:CMP1 条件成立
	21	SCP2	1:CMP2 条件成立
	22	SCP3	1:CMP3 条件成立
	23	SPLS	1:パルス出力 ON
	24	SPH1	1:励磁出力 PH1 H レベル
	25	SPH2	1:励磁出力 PH2 H レベル
	26	SPH3	1:励磁出力 PH3 H レベル
	27	SPH4	1:励磁出力 PH4 H レベル
			次ページに続く

	29,28	PFC1,0	（C/F/(W)モジュールのみ） コンパレータ 3 用プリレジスタ使用状態 00:未確定，01:レジスタ確定，1x: レジスタ確定(レジスタフル)
	31,30	PFM1,0	（C/F/(W)モジュールのみ） 動作用プリレジスタの使用状態 00:未確定，01:レジスタ確定，1x: レジスタ確定(レジスタフル)

VC++
記述例
	short ret; //関数の戻り値
	DWORD exsts; //拡張ステータス
	ret = hmx_ReadExSts(hDevID, 1, 2, &exsts); //ライン 2 モジュール 2 を指定

2.4.5 hmx_ReadOutp() 汎用出力ポート状態読出し

No.	L27
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの汎用出力ポートの状態を読出します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_ReadOutp(DWORD hDevID, WORD wLine, WORD wMid, BYTE* byOutp);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
byOutp	[OUT] 出力ポート状態

VC++
記述例
	short ret; //関数の戻り値
	BYTE out; //出力値
	ret = hmx_ReadOutp(hDevID, 1, 2, &out); //ライン 2 モジュール 2 を指定，格納先

2.4.6 hmx_ReadSpd() 指令速度の読出し

No.	L28
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの指令速度を読出します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_ReadSpd(DWORD hDevID, WORD wLine, WORD wMid, DWORD* dwSpd);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
dwSpd	[OUT] 指令速度（指令速度[pps]＝読出したデータ）

VC++
記述例
	short ret; //関数の戻り値
	DWORDspd; //速度
	ret = hmx_ReadSpd(hDevID, 1, 2, &spd); //ライン 2 モジュール 2 を指定，格納先

2.4.7 hmx_ReadSpdEx() 指令速度の読出し(速度倍率含む)

No.	L29
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの速度倍率を含んだ指令速度を読出します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_ReadSpdEx(DWORD hDevID, WORD wLine, WORD wMid, double* dblSpd);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
dblSpd	[OUT] 指令速度 (指令速度[pps]＝読出したデータ×速度倍率)

VC++
記述例
	short ret; //関数の戻り値
	double spd; //速度
	ret = hmx_ReadSpdEx(hDevID, 1, 2, &spd); //ライン 2 モジュール 2 を指定, 格納先

2.4.8 hmx_ReadCtr() カウンタの読出し

No.	L30
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの指定されたカウンタを読出します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_ReadCtr(DWORD hDevID, WORD wLine, WORD wMid, WORD wCtr, long* lValue);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]
wCtr	[IN] 入力ソース[0:カウンタ(指令), 1:カウンタ 2(エンコーダ入力), 2:カウンタ 3(汎用・偏差)]
lValue	[OUT] カウンタ値

VC++
記述例
	short ret; //関数の戻り値
	long ctr1; //カウンタ 1(指令パルス出力)の値
	//ライン 2 モジュール 2 を指定, カウンタ 1 を指定, 格納先
	ret = hmx_ReadCtr(hDevID, 1, 2, 1, &ctr1);

2.5 モーションモジュール動作設定

2.5.1 hmx_SetFLSpd() ベース速度の設定

No.	L31
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのベース.速度(pps)を速度倍率で除算した値を設定.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetFLSpd(DWORD hDevID, WORD wLine, WORD wMid, DWORD dwRfl);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
dwRfl	[IN] ベース速度レジスタ値(RFL)[1～65535, RFL＜RFH で設定]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetFLSpd(hDevID, 1, 1, 500); //ライン 2 モジュール 1 を指定，RFL=500

備　考	ベース速度・・加減速動作時の立ち上がりの速度(本速度から加速，本速度まで減速して停止)

2.5.2 hmx_SetAuxSpd() 補助速度の設定

No.	L32
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの補助速度(pps)を速度倍率で除算した値を設定.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetAuxSpd(DWORD hDevID, WORD wLine, WORD wMid, DWORD dwRfa);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
dwRfa	[IN] 補助速度レジスタ値(RFA)[1～65535]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetAuxSpd(hDevID,1 , 1, 300); //ライン 2 モジュール 1 を指定，RFA=300

備　考	補助速度・・一部の原点復帰において，原点突入速度等に使用されます.

2.5.3 hmx_SetAccRate() 加速レートの設定

No.	L33
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの加速レート(RUR)を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetAccRate(DWORD hDevID, WORD wLine, WORD wMid, DWORD dwRur);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]
dwRur	[IN] 加速レート(RUR)[1～65535]

VC++ 記述例

```
short       ret;      //関数の戻り値
DWORD       rur;      //RUR
// RUR 計算  RFH=5000, RFL=500,  加速時間=100msec(  直線加速時)
ret = hmx_CalAccRate( &rur, 100, 5000, 500, 0, 0 );
ret = hmx_SetAccRate( hDevID, 1, 1, rur );      //ライン 2 モジュール 1 に設定
```

備　考

加速レート(RUR)と加速時間の関係
RFH:動作速度レジスタ
RFL:ベース速度レジスタ
RUR:加速レートレジスタ

直線加速

$$加速時間[sec] = \frac{(RFH - RFL) \times (RUR + 1) \times 4}{19{,}660{,}800}$$

直線部分のない S 字加速

$$加速時間[sec] = \frac{(RFH - RFL) \times (RUR + 1) \times 8}{19{,}660{,}800}$$

2.5.4 hmx_SetDecRate() 減速レートの設定

No.	L34
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの減速レート(RDR)を設定します. 加速時間と減速時間が異なる場合に RDR の設定を行います.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetDecRate(DWORD hDevID, WORD wLine, WORD wMid, DWORD dwRdr);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]
dwRdr	[IN] 減速レート(RDR)[0～65535] 0 を設定した場合は RUR=RDR として減速レートが決定されます. (加速時間＝減速時間)

VC++
記述例
	short ret; //関数の戻り値
	DWORD rdr; //RDR
	// RDR 計算 RFH=5000, RFL=500, 減速時間=200msec(直線加速時)
	ret = hmx_CalAccRate(&rdr, 200, 5000, 500, 0, 0);
	ret = hmx_SetDecRate(hDevID, 1, 1, rdr); //ライン 2 モジュール 1 に設定

備　考	減速レート(RDR)と減速時間の関係 RFH:動作速度レジスタ RFL:ベース速度レジスタ RDR:減速レートレジスタ 直線減速 $$加速時間[sec]=\frac{(RFH-RFL)\times(RDR+1)\times 4}{19{,}660{,}800}$$ 直線部分のない S 字減速 $$加速時間[sec]=\frac{(RFH-RFL)\times(RDR+1)\times 8}{19{,}660{,}800}$$

2.5.5 hmx_SetMult() 速度倍率レジスタ値の設定

No.	L35
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの速度倍率レジスタ値(RMG)を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetMult(DWORD hDevID, WORD wLine, WORD wMid, DWORD dwRmg);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
dwRmg	[IN] 速度倍率レジスタ値[2～4095]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_SetMult(hDevID, 1, 1, 299); //ライン 2 モジュール 1 を指定，速度倍率 1 倍

備　考

速度と速度倍率の関係及び速度倍率設定値と速度倍率の関係
RFx は速度レジスタ(RFH，RFL，RFA)の値

$$速度[PPS] = RFx \times 速度倍率 = \frac{RFx \times 300}{RMG + 1} \qquad 速度倍率設定値(RMG) = \frac{300}{速度倍率} - 1$$

設定例

RMG(DEC)	RMG(HEX)	速度倍率	出力速度範囲(pps)		
2999	0bb7h	0.1	0.1	～	6,553.5
1499	05dbh	0.2	0.2	～	13,107
599	0257h	0.5	0.5	～	32,767.5
299	012bh	1	1	～	65,535
149	0095h	2	2	～	131,070
59	003bh	5	5	～	327,675
29	001dh	10	10	～	655,350
14	000Eh	20	20	～	1,310,700
11	000Bh	25	25		1,638,375
9	0009h	30	30		1,966,050
5	0005h	50	50	～	3,276,750
4	0004h	60	60		3,932,100
3	0003h	75	75		4,915,125
2	0002h	100	100	～	6,553,500

2.5.6 hmx_SetEventMask() イベントマスクの設定

No.	L36
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのイベントマスクを設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetEventMask(DWORD hDevID, WORD wLine, WORD wMid, DWORD dwMsk);

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1, 1;Line2］		
wMid	[IN] モジュール ID[0～63]		
dwMsk	[IN] イベント割込要因設定 0:イベント無効, 1:イベント発生有効		
	ビット	名　称	説　　明
	0	IREN	動作完了
	1	IRUS	加速開始
	2	IRUE	加速終了
	3	IRDS	減速開始
	4	IRDE	減速終了
	5	IRC1	CMP1 比較条件成立(+SLS)
	6	IRC2	CMP2 比較条件成立(-SLS)
	7	IRC3	CMP3 比較条件成立(脱調検出用途)
	8	IRCL	CLR 信号 ON によるカウンタ値リセット
	9	IRLT	LTC 信号 ON によるカウンタ値ラッチ
	10	IROL	OLS 信号 ON によるカウンタ値ラッチ
	11	IRSD	DLS 信号 OFF→ON
	12	IRSA	STA 信号 OFF→ON
	13	IRNA	（C/F/(W)モジュールのみ） 同報通信スタートコマンド(2x01h)によるスタート
	14	IRNP	（C/F/(W)モジュールのみ） 同報通信ストップコマンド(2x02h)によるストップ
	15	IRNM	（C/F/(W)モジュールのみ） 動作用プリレジスタ書き込み可能（MSTS SPRF ビット変化）
	16	IRBE	（C/F/(W)モジュールのみ） プリレジスタが未設定で現動作完了 （補間の連続実行時に次動作データの設定が間に合わなかったとき）
	上記データの OR したデータを与えることで複数のイベント報告指定.		

VC++
記述例
	short ret; //関数の戻り値
	//ライン 2 モジュール 1, 正常停止,コンパレータ 3 条件成立時にイベント発生
	ret = hmx_SetEventMask(hDevID, 1, 1, 0x0081);

2.5.7 hmx_SetDecPoint() 減速開始点の設定

No.	L37
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの減速開始点を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetDecPoint(DWORD hDevID, WORD wLine, WORD wMid, long lRdp);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]
lRdp	[IN] 減速開始点(pulse)

VC++
記述例
	short ret; //関数の戻り値
	//ライン 2 モジュール 1 を指定, 移動残 300pulse で減速開始
	ret = hmx_SetDecPoint(hDevID, 1, 1, 300);

備　考	残移動量が減速開始点以下になると減速を開始します. [減速開始点計算自動時] 自動で計算された減速開始点に対するオフセット値(単位:pulse)となります. ＋の値を設定すると減速が早めに開始され, 減速後ベース速度で動作します. －の値を設定すると減速が遅めに開始され, ベース速度に到達する前に動作完了となります. 0 を設定すると通常の動作になります. [減速開始点計算手動時] 残移動量が設定した値(pulse)以下になると減速を開始します. この場合に設定する値は次式の様になります. (ベース速度到達時に動作完了になる値) 直線減速 $最適値[パルス] = \frac{(RFH^2 - RFL^2) \times (RDR + 1)}{(RMG + 1) \times 32{,}768}$ 直線部分のない S 字減速 $最適値[パルス] = \frac{(RFH^2 - RFL^2) \times (RDR + 1) \times 2}{(RMG + 1) \times 32{,}768}$

2.5.8 hmx_SetCnstIPFeed()　2 軸補間時の合成速度一定制御の ON/OFF

No.	L38
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの合成速度一定制御を設定します.（対象になるモジュールそれぞれに設定が必要です）
対　象	C/F/(W)モジュール (9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SetCnstIPFeed(DWORD hDevID, WORD wLine, WORD wMid, BOOL bOn);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
bOn	[IN] 一定制御有効[0:無効, 1:有効]

VC++
記述例
	short　　ret; //関数の戻り値
	ret = hmx_SetCnstIPFeed(hDevID, 0, 2, 1);　　//ライン 1 モジュール 2 を指定，一定制御有効

2.5.9 hmx_WritOpeMode() 動作モードの設定

No.	L39
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの動作モードを設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_WritOpeMode(DWORD hDevID, WORD wLine, WORD wMid, WORD wMode, WORD wCoord);

引　数	説　明	
hDevID	[IN] デバイスハンドル	
wLine	[IN] ライン番号［0:Line1，1;Line2］	
wMid	[IN] モジュール ID[0～63]	
wMode	[IN] 動作モード	
	00h:コマンド制御による＋方向連続送り	08h:コマンド制御による－方向連続送り
	01h:パルサ入力による連続送り(※1)	02h:±DR 入力による連続送り(※2)
	10h:＋方向原点復帰動作	18h:－方向原点復帰動作
	12h:＋方向原点抜け出し動作	1ah:－方向原点抜け出し動作
	15h:＋方向原点サーチ動作	1dh:－方向原点サーチ動作
	20h:＋ELS 又は＋SLS 位置まで動作	28h:－ELS 又は-SLS 位置まで動作
	22h:＋ELS 又は+SLS 抜け出し動作	2ah:－ELS 又は-SLS 抜け出し動作
	24h:＋方向に Z 相カウント動作	2ch:－方向に Z 相カウント動作
	41h:位置決め動作	42h:PCS 位置決め動作(ライブラリ関数のみ)
	44h:指令位置0点復帰動作	45h:機械位置0点復帰動作(※1)
	46h:＋方向 1 パルス動作	4eh:－方向 1 パルス動作
	47h:タイマ動作	51h:パルサ入力による位置決め動作(※1)
	54h:パルサ入力指令位置 0 点復帰動作(※1)	55h:パルサ入力機械位置 0 点復帰動作
	（以下は C/F/(W)モジュールのみ）	
	60h:連続直線補間動作	61h:直線補間動作
	64h:CW 方向円弧補間動作	65h:CCW 方向円弧補間動作
wCoord	指定モード ABS/INC 切替[0:相対位置指定,1:指令座標位置指定,2:機械座標位置指定]	

VC++
記述例
	short ret; //関数の戻り値
	//ライン 2 モジュール 1 を指定，位置決め動作，相対位置指定
	ret = hmx_WritOpeMode(hDevID, 1, 1, 0x41, 0);

2.5.10 hmx_WritSta()　STA 入力時の動作設定

No.	L40
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの STA 入力時の動作を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_WritSta(DWORD hDevID, WORD wLine, WORD wMid, WORD wEnbl, WORD wTrg);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wEnbl	[IN] STA 入力有効設定[0:無効, 1:有効]
wTrg	[IN] 入力仕様設定[0:レベル, 1:エッジ]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_WritSta(hDevID, 0, 3, 1, 1); //ライン 1 モジュール 3 を指定，STA 有効，エッジトリガ

2.5.11 hmx_WritStp()　STP 入力時の動作設定と異常停止時の STP 自動出力設定

No.	L41
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの STP 入力時の動作と異常停止時の STP 自動出力を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_WritStp(DWORD hDevID, WORD wLine, WORD wMid, WORD wEnbl, WORD wStop, WORD wOut);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wEnbl	[IN] STP 入力有効設定[0:無効, 1:有効]
wStop	[IN] STP 入力時の停止方法[0:即停止, 1:減速停止]
wOut	[IN] 異常停止時 STP 出力設定[0:しない, 1:する]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 1 モジュール 3 を指定,STP 有効，減速停止，出力なし
	ret = hmx_WritSta(hDevID, 0, 3, 1, 1);

2.5.12 hmx_WritFHSpd()　動作速度の設定

No.	L42
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの動作速度(pps)を速度倍率で除算した値(RFH)を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_WritFHSpd(DWORD hDevID, WORD wLine, WORD wMid, DWORD dwRfh);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
dwRfh	[IN] 動作速度レジスタ値(RFH)[2～65535, RFL＜RFH で設定]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_WritFHSpd(hDevID, 1, 1, 5000); //ライン 2 モジュール 1 を指定，RFH=5000

2.5.13 hmx_WritPos()　位置決め移動量の設定

No.	L43
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの位置決め動作の移動量を設定します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_WritPos(DWORD hDevID, WORD wLine, WORD wMid, long lDst);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
Dst	[IN] 移動量(pulse)[-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_WritPos(hDevID, 1, 1, 15000);　//ライン 2 モジュール 1 を指定，+方向 15000pulse

2.5.14 hmx_WritCtr()　カウンタプリセット

No.	L44
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの指定されたカウンタへプリセット(座標値の書込み)をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_WritCtr(DWORD hDevID, WORD wLine, WORD wMid, long lPre, WORD wCtr);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
lPre	[IN] プリセット値[-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]
wCtr	[IN] カウンタ選択[1:カウンタ 1, 2:カウンタ 2, 3:カウンタ 3]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_WritCtr(hDevID, 1, 1, 5000, 1);　//ライン 2 モジュール 1，プリセット値=5000，カウンタ 1

2.5.15 hmx_Writ2AxisLine()　2 軸直線補間の移動量の設定

No.	L45
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定 2 モジュールによる 2 軸直線補間の移動量をそれぞれに設定します.
対　象	C/F/(W)モジュール (G9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_Writ2AxisLine(DWORD hDevID, WORD wLine, WORD wMidX, WORD wMidY, long lDstX, long lDstY);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidX	[IN] モジュール 1 ID[0～63]
wMidY	[IN] モジュール 2 ID[0～63]
lDstX	[IN] モジュール 1　移動量(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]
lDstX	[IN] モジュール 2　移動量(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 2 モジュール 6 とモジュール 8,直線補間，それぞれ+15000 および-30000pulse 移動
	ret = hmx_Writ2AxisLine(hDevID, 1, 6, 8, 15000, -30000);

2.5.16 hmx_WritCircl()　2 軸円弧補間の移動量の設定

No.	L46
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールの円弧補間の移動量を設定します. 終点位置及び中心位置 1, 2 の順番は X 軸に近い軸の順番になります. データの設定方法については「ユーザーズマニュアル<運用編>」を参照して下さい.
対　象	C/F/(W)モジュール (G9103 搭載モジュール)

開発言語	書　式
VC++	shmx_WritCircl(DWORD hDevID, WORD wLine, WORD wMidX, WORD wMidY, long lDstX, long lDstY, long lCenX, long lCenY);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidX	[IN] モジュール 1 ID[0～63]
wMidY	[IN] モジュール 2 ID[0～63]
lDstX	[IN] モジュール 1 終点位置(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]
lDstX	[IN] モジュール 2 終点位置(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]
lCenX	[IN] モジュール 1 中心位置(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]
lCenX	[IN] モジュール 2 中心位置(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 2 モジュール 5 とモジュール 7 円弧補間，それぞれ終点位置 0 および 0，中心位置 1000 および 0
	ret = hmx_WritCircl(hDevID, 1, 5, 7, 0, 0, 1000, 0);

2.6 モーションモジュール動作制御指令

2.6.1 hmx_DecStop() 減速停止

No.	L47
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを減速停止します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	hmx_DecStop(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_DecStop(hDevID, 0, 0); //ライン 1 モジュール 0, 減速停止

2.6.2 hmx_QuickStop() 即停止

No.	L48
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを即停止します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_QuickStop(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_QuickStop(hDevID, 0, 0); //ライン 1 モジュール 0, 即停止

2.6.3 hmx_EmgStop() 非常停止

No.	L49
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを非常停止します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_EmgStop(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_EmgStop(hDevID, 0, 0); //ライン 1 モジュール 0, 非常停止

2.6.4 hmx_AccStart() 加速スタート後減速停止

No.	L50
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを加速スタートして FH 定速動作をします. 位置決め動作では位置到達時 FH 定速から減速停止します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_AccStart(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_AccStart(hDevID, 0, 0); //ライン 1 モジュール 0, 加速スタート

2.6.5 hmx_ CnstStartFH() FH 定速スタート

No.	L51
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールをFH定速スタートします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_CnstStartFH(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_ CnstStartFH (hDevID, 0, 0); //ライン 1 モジュール 0, FH 定速スタート

2.6.6 hmx_ CnstStartFL() FL 定速スタート

No.	L52
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールをFL 定速スタートします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_CnstStartFL(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号[0:Line1, 1;Line2]
wMid	[IN] モジュール ID[0～63]

VC++
記述例
	short ret; //関数の戻り値
	ret = hmx_ CnstStartFL (hDevID, 0, 0); //ライン 1 モジュール 0, FL 定速スタート

2.6.7 hmx_ CnstStartByDec () FH定速スタート後減速停止

No.	L53
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールをFH定速スタートします. 位置決め動作では位置到達時 FH 定速から減速停止します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_CnstStartByDec(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++
記述例
	short ret; //関数の戻り値
	//ライン 1 モジュール 0, FH定速スタート後減速停止
	ret = hmx_ CnstStartByDec(hDevID, 0, 0);

2.6.8 hmx_MvAccStart() 移動量設定+加速スタート後減速停止

No.	L54
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを移動量設定後加速スタートして FH 定速動作をします. 位置決め動作では位置到達時 FH 定速から減速停止します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_MvAccStart(DWORD hDevID, WORD wLine, WORD wMid, long lDst);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]
lDst	[IN] 移動量(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]

VC++
記述例
	short ret; //関数の戻り値
	//ライン 1 モジュール 0, 移動量 20000puls, 加速スタート
	ret = hmx_MvAccStart(hDevID, 0, 0, 20000);

2.6.9 hmx_ MvCnstStartFH() 移動量設定+FH 定速スタート

No.	L55
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを移動量設定後FH定速スタートします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_MvCnstStartFH(DWORD hDevID, WORD wLine, WORD wMid, long lDst);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
lDst	[IN] 移動量(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]

VC++
記述例
	short ret; //関数の戻り値
	//ライン 1 モジュール 0，移動量 20000puls，FH 定速スタート
	ret = hmx_ MvCnstStartFH (hDevID, 0, 0, 20000);

2.6.10 hmx_ MvCnstStartFL() 移動量設定+FL 定速スタート

No.	L56
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを移動量設定後FL 定速スタートします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_CnstStartFL(DWORD hDevID, WORD wLine, WORD wMid, long lDst);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
lDst	[IN] 移動量(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]

VC++
記述例
	short ret; //関数の戻り値
	//ライン 1 モジュール 0，移動量 20000puls，FL 定速スタート
	ret = hmx_ MvCnstStartFL (hDevID, 0, 0, 20000);

2.6.11 hmx_ CnstStartByDec ()　移動量設定+FH定速スタート後減速停止

No.	L57
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定モジュールを移動量設定後FH定速スタートします．位置決め動作では位置到達時 FH 定速から減速停止します．
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_MvCnstStartByDec(DWORD hDevID, WORD wLine, WORD wMid, long lDst);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
lDst	移動量(pulse) [-134,217,728～134,217,727(28 ﾋﾞｯﾄ)]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 1 モジュール 0，移動量 20000puls，，FH定速スタート後減速停止
	ret = hmx_ CnstStartByDec(hDevID, 0, 0, 20000);

2.6.12 hmx_ SetGroup()　グループ設定（個別）

No.	L58
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールをグループ設定します．
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SetGroup(DWORD hDevID, WORD wLine, WORD wMid, WORD wGroup, WORD wSync);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wGroup	[IN] 設定するグループ番号[0～7] ※0 はグループ設定解除
wSync	[IN] 同時スタート／ストップ設定 [0:同期ｸﾘｱ, 1:STA 同期ｾｯﾄ, 2:STP 同期ｾｯﾄ, 3:STA/STP 同期ｾｯﾄ，その他:変更なし]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 2 モジュール 5，グループ 2，STA/STP 設定変更なし
	ret = hmx_ SetGroup(hDevID, 1, 5, 2, 0xff);

備　考	P ﾓｼﾞｭｰﾙ互換のｸﾞﾙｰﾌﾟｺﾏﾝﾄﾞを使用する場合，"RMD.MSPE"(STP ｺﾏﾝﾄﾞ入力で停止する設定) または"RMD.MSY"(STA ｺﾏﾝﾄﾞ入力で開始する設定)を設定する必要があります

2.6.13 hmx_ SetGroupEx()　グループ設定（一括）

No.	L59
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の複数指定モジュールを一括でグループ設定および同時スタート／ストップ設定をします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_SetGroupEx(DWORD hDevID, WORD wLine, WORD wGroup, GRPSET_DATA* tGrpSet);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wGroup	[IN] 設定するグループ番号[0～7] ※0 はグループ設定解除

tGrpSet	[IN] 設定するモジュールの指定および STA/STP 機能設定（ビット対応）
	GRPSET_DATA 構造体
	struct {
	DWORD grpset1; // ｸﾞﾙｰﾌﾟ設定要求ﾌﾗｸﾞ(ﾓｼﾞｭｰﾙ 0～31/Bit0～31)
	DWORD grpset2; // ｸﾞﾙｰﾌﾟ設定要求ﾌﾗｸﾞ(ﾓｼﾞｭｰﾙ 32～63/Bit0～31)
	DWORD taset1; // STA 同期要求ﾌﾗｸﾞ(ﾓｼﾞｭｰﾙ 0～31/Bit0～31)
	DWORD staset2; // STA 同期要求ﾌﾗｸﾞ(ﾓｼﾞｭｰﾙ 32～63/Bit0～31)
	DWORD stpset1; // STP 同期要求ﾌﾗｸﾞ(ﾓｼﾞｭｰﾙ 0～31/Bit0～31)
	DWORD stpset2; // STP 同期要求ﾌﾗｸﾞ(ﾓｼﾞｭｰﾙ 32～63/Bit0～31)
	} GRPSET_DATA;
	
	grpset1: ｸﾞﾙｰﾌﾟ設定要求ﾌﾗｸﾞ ･･･ ﾓｼﾞｭｰﾙ 0～31 [0:設定しない 1:設定する]
	grpset2: ｸﾞﾙｰﾌﾟ設定要求ﾌﾗｸﾞ ･･･ ﾓｼﾞｭｰﾙ 32～63 [0:設定しない 1:設定する]
	grpset1: STA 同期設定要求ﾌﾗｸﾞ
	･･･ ﾓｼﾞｭｰﾙ 0～31 [0:ｸﾘｱ 1:ｾｯﾄ ｸﾞﾙｰﾌﾟ設定が無効なﾓｼﾞｭｰﾙは変更しない]
	Grpset2: STA 同期設定要求ﾌﾗｸﾞ
	･･･ ﾓｼﾞｭｰﾙ 32～63 [0:ｸﾘｱ 1:ｾｯﾄ ｸﾞﾙｰﾌﾟ設定が無効なﾓｼﾞｭｰﾙは変更しない)
	grpset1: STP 同期設定要求ﾌﾗｸﾞ
	･･･ ﾓｼﾞｭｰﾙ 0～31 [0:ｸﾘｱ 1:ｾｯﾄ ｸﾞﾙｰﾌﾟ設定が無効なﾓｼﾞｭｰﾙは変更しない]
	grpset12 STP 同期設定要求ﾌﾗｸﾞ
	･･･ ﾓｼﾞｭｰﾙ 32～63 [0:ｸﾘｱ 1:ｾｯﾄ ｸﾞﾙｰﾌﾟ設定が無効なﾓｼﾞｭｰﾙは変更し]
	※モジュール番号が小さいほうがビット下位側(Bit0 側)になります

VC++
記述例
	short ret; //関数の戻り値
	//ライン 2 モジュール 5，グループ 2，STA/STP 設定変更なし
	ret = hmx_ SetGroup(hDevID, 1, 5, 2, 0xff);

備　考	P ﾓｼﾞｭｰﾙ互換のｸﾞﾙｰﾌﾟｺﾏﾝﾄﾞを使用する場合，"RMD.MSPE"(STP ｺﾏﾝﾄﾞ入力で停止する設定) または"RMD.MSY"(STA ｺﾏﾝﾄﾞ入力で開始する設定)を設定する必要があります

2.6.14 hmx_ GrpStop()　グループ停止

No.	L60
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定したモジュールをグループ停止します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_GrpStop(DWORD hDevID, WORD wLine, WORD wGroup, WORD wCmdNo);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]
wGroup	[IN] 設定するグループ番号[0～7] ※0 はグループ設定解除
wCmdNo	[IN] 停止機能番号[0:即停止, 1:減速停止, 2:非常停止]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 2 モジュール 5, グループ 2, 即停止
	ret = hmx_ GrpStop(hDevID, 1, 5, 2, 0);

備　考	停止ｺﾏﾝﾄﾞはﾓｼﾞｭｰﾙに合わせて適切な同報通信ｺﾏﾝﾄﾞ(0x200x)を関数内で切り替えて使用しています. (GRPSTP/GRPSDSTP/GRPQSTP/GRPESTP ｺﾏﾝﾄﾞを使用 ただし P ﾓｼﾞｭｰﾙが指定ｸﾞﾙｰﾌﾟ内に混在している場合は GRPESTP は使用できません. さらに 使用する同報ｺﾏﾝﾄﾞは P ﾓｼﾞｭｰﾙ互換ｺﾏﾝﾄﾞになります. P ﾓｼﾞｭｰﾙ互換ｺﾏﾝﾄﾞを使用する場合, 動作に必要な設定は関数内で自動に行ないます. 注意: ・この関数で GRPSTP を使用して同時停止させる場合, "RMD.MSPE"(STP ｺﾏﾝﾄﾞ入力で停止する設定), RENV1.STPM(停止方法の設定)の設定が必要です. "RMD.MSPE"の設定はｸﾞﾙｰﾌﾟ設定関数内で設定を指定するか, 手動で設定を行う必要があります. ・本関数内では"RMD.MSPE"設定および設定のﾁｪｯｸは行っていません. 設定が正しく行われていない場合, ﾓｼﾞｭｰﾙは停止しません. RENV1.STPM(停止方法)は CmdNo の指定により関数内で設定できます.

2.6.15 hmx_ GrpStart()　グループスタート

No.	L61
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定したモジュールをグループスタートします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_GrpStart(DWORD hDevID, WORD wLine, WORD wGroup, WORD wCmdNo);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wGroup	[IN] 設定するグループ番号[0～7] ※0 はグループ設定解除
wCmdNo	[IN] スタート機能番号 [0:FL 定速ｽﾀｰﾄｾｯﾄ，1:FH 定速ｽﾀｰﾄｾｯﾄ，2:FH 定速ｽﾀｰﾄ後減速停止ｾｯﾄ(FH->減速)，3:加速ｽﾀｰﾄ後減速停止ｾｯﾄ(加速->定速->減速)]

VC++
記述例
	short　ret;　//関数の戻り値
	//ライン 2 モジュール 5，グループ 2，加速ｽﾀｰﾄ後減速停止
	ret = hmx_ GrpStart(hDevID, 1, 5, 2, 3);

備　考	ｽﾀｰﾄｺﾏﾝﾄﾞは指定したｺﾏﾝﾄﾞを書込み，同報通信ｺﾏﾝﾄﾞ GRPSTA でｽﾀｰﾄします．. (STAFL/STAFH/STAD/STAUD ｺﾏﾝﾄﾞを使用) 注意： ・この関数では GRPSTA を使用して同時ｽﾀｰﾄさせるため，"RMD.MSY"(STA ｺﾏﾝﾄﾞ入力でｽﾀｰﾄする設定)の設定が必要です．"RMD.MSY"の設定はｸﾞﾙｰﾌﾟ設定関数内で設定を指定するか，手動で設定を行う必要があります． ・本関数内では"RMD.MSY"設定のﾁｪｯｸは行っていません． 設定が正しく行われていない場合，ﾓｼﾞｭｰﾙは同時ｽﾀｰﾄせず，順次ｽﾀｰﾄします．

2.6.16 hmx_SvOn()　サーボオン信号オン

No.	L62
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボオン信号(SVON)をオンします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvOn(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvOn(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボオン

2.6.17 hmx_SvOff()　サーボオン信号オフ

No.	L63
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボオン信号(SVON)をオフします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvOff(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvOff(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボオフ

2.6.18 hmx_SvResetOn()　サーボリセット信号オン

No.	L64
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボリセット信号(SVRST)をオンします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvResetOn(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvResetOn(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボリセットオン

2.6.19 hmx_SvResetOff()　サーボリセット信号オフ

No.	L65
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボリセット信号(SVRST)をオフします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvResetOff(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvResetOff(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボリセットオフ

2.6.20 hmx_SvTrqOn()　サーボトルク制限信号オン

No.	L66
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボトルク制限信号をオンします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvTrqOn(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvTrqOn(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボトルクオン

2.6.21 hmx_SvTrqOff()　サーボトルク制限信号オフ

No.	L67
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボトルク制限信号をオフします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvTrqOff(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvTrqOff(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボトルクオフ

2.6.22 hmx_SvGainOn()　サーボゲイン切替信号オン

No.	L68
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボゲイン切替信号をオンします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvGainOn(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvGainOn(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボゲインオン

2.6.23 hmx_SvGainOff()　サーボゲイン切替信号オフ

No.	L69
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのサーボゲイン切替信号をオフします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_SvGainOff(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_SvGainOff(hDevID, 0, 0);　//ライン 1 モジュール 0, サーボゲインオフ

2.6.24 hmx_PMOn()　パルスモータ励磁オン

No.	L70
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのパルスモータ励磁をオンします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_PMOn(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_PMOn(hDevID, 0, 0);　//ライン 1 モジュール 0，パルスモータ励磁オン

備　考	SVON をパルスモータドライバのモーターフリー(出力電流オフ)信号に接続した場合に使用可能.

2.6.25 hmx_PMOff()　パルスモータ励磁オフ

No.	L71
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのパルスモータ励磁オフ(モーターフリー)にします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_PMOff(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_PMOff(hDevID, 0, 0);　//ライン 1 モジュール 0，パルスモータ励磁オフ

備　考	SVON をパルスモータドライバのモーターフリー(出力電流オフ)信号に接続した場合に使用可能.

2.6.26 hmx_ResetSEND()　モーションモジュールSENDリセット

No.	L72
機　能	デバイスハンドルで指定された motionCAT マスターボードに接続した指定ライン上の指定モジュールのモーションモメインステータスの SEND ビット（Bit1）をクリアします.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short　hmx_ResetSEND(DWORD hDevID, WORD wLine, WORD wMid);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]

VC++ 記述例	short　ret;　//関数の戻り値 ret = hmx_ResetSEND(hDevID, 0, 0);　//ライン 1 モジュール 0 の SEND ビットクリア

備　考	SEND ビットはリードクリアあるいは自動リセットされませんのでこのコマンドの発行が必要です.

2.7 加減速レートの計算

2.7.1 hmx_CalAccRate() 加減速レートの計算

No.	L73
機　能	加減速時間，RFH，RFL 等より加減速レート(RUR, RDR)を計算します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	short hmx_CalAccRate(DWORD* dwRur, DWORD dwTim, DWORD dwRfh, DWORD dwRfl, WORD wPro, DWORD dwRus);

引　数	説　明
dwRur	[OUT] 加減速レート計算値 (加減速レート[1～65535])
dwTim	[IN] 加減速時間(msec)
dwRfh	[IN] FH レジスタ値[1～65535]
dwRfl	[IN] FL レジスタ値[1～65535]
wPro	[IN] 加減速形式［0:直線,1:S 字］
dwRus	[IN] 加減速形式を S 字にした場合の加速時 S 字速度区間[0~50000]，0 を指定した場合は直線部分なし

VC++
記述例
	short ret; //関数の戻り値
	DWORD rate; //加減速レート
	//加減速時間 200ms, RFH=10000, RFL=500，直線加減速
	ret = hmx_CalAccRate(&rate, 200, 10000, 500, 0, 0);
	//ライン 1 モジュール 0，加速レート設定
	ret = hmx_SetAccRate(hDevID, 0, 0, rate);

2.7.2 hmx_CalDecPoint() 減速開始点の計算

No.	L74
機　能	加減速時間，RFH，RFL 等より加減速レート(RUR, RDR)を計算します.
対　象	P/W/M/C/F モジュール (G9003/9103 搭載モジュール)

開発言語	書　式
VC++	Short CalDecPoint(DWORD* dwRdp, DWORD dwRfh, DWORD dwRfl, DWORD dwRmg, DWORD dwRdr, WORD wPro, DWORD dwRds);

引　数	説　明
dwRdp	[OUT] 減速開始点計算値 (減速開始点[0～16777215])
dwRfh	[IN] FH レジスタ値[1～65535]
dwRfl	[IN] FL レジスタ値[1～65535]
dwRmg	[IN] RMG レジスタ値[2～4095]
wPro	[IN] 加減速形式［0:直線,1:S 字］
dwRds	[IN] 加減速形式を S 字にした場合の減速時 S 字速度区間[0~50000]，0 を指定した場合は直線部分なし

VC++
記述例
	short ret; //関数の戻り値
	DWORD sdp; //減速開始点
	//加減速時間 200ms, RFH=10000, RFL=500, RMG=299(1 倍)，直線加減速
	ret = hmx_CalDecPoint(&sdp, 10000, 500, 299, 0);
	//ライン 1 モジュール 0 減速開始点設定
	ret = hmx_SetDecPoint(hDevID, 0, 0, sdp);

3. ドライバー関数

ドライバー関数はmotionCATマスターボードあるいはモジュール(=ローカルモジュールまたはスレーブとも表現)に対する基本的な制御を行うための関数群をソースコード(mc520drv*.c ファイルおよび mc520drv.h ファイル)で提供したものです.
各関数は Visual Studio2008 Vc／Vc++で作成されています. ご使用時にはライブラリ関数と同様に, ユーザー様が作成するアプリケーションに組み込んでビルドします.
このドライバー関数は, 次章の RSL(リアルタイムシェアードライブラリ)で提供されるデバイスドライバーアクセス関数を使用して作られている, マスターボードおよび各モジュール制御用の基本機能関数です.

3.1 関数の種類

No.	関数名	機 能	記載項
D1	mx500_ApiStartup()	API関数を活性化(初期化処理)	3.4.1
D2	mx500_ApiCleanup()	API関数の使用を完了(資源解放)	3.4.2
D3	mx500_GetMstBrdCount()	マスターボード枚数の取得	3.4.3
D4	mx500_GetDeviceInfo()	マスターボード情報の取得	3.4.4
D5	mx500_OpenDevice()	マスターボードオープン	3.4.5
D6	mx500_CloseDevice()	マスターボードクローズ	3.4.6
D7	mx500_rOptPortB()	マスターボード オプションポート 1バイト読出し	3.4.7
D8	mx500_wOptPortB()	マスターボード オプションポート 1バイト書込み	3.4.8
D9	mx500_rOptPortW()	マスターボード オプションポート 2バイト読出し	3.4.9
D10	mx500_wOptPortW()	マスターボード オプションポート 2バイト書込み	3.4.10
D11	mx500_rCenMsts()	センターデバイス メインステータス読出し	3.4.11
D12	mx500_wCenCmd()	センターデバイス コマンド書込み	3.4.12
D13	mx500_rCenIsts()	センターデバイス 割込みステータス読出し	3.4.13
D14	mx500_rCenBuf()	センターデバイス 入力バッファ読出し	3.4.14
D15	mx500_wCenBuf()	センターデバイス 出力バッファ書込み	3.4.15
D16	mx500_rCenRFiFo()	センターデバイス 受信用FIFO読出し	3.4.16
D17	mx500_wCenSFiFo()	センターデバイス 送信用FIFO書込み	3.4.17
D18	mx500_rLclInfo()	ローカルデバイス情報読出し	3.4.18
D19	mx500_wLclInfo()	ローカルデバイス情報書込み	3.4.19
D20	mx500_rLclCycErr()	サイクリック(I／O)通信エラーフラグ読出し	3.4.20
D21	mx500_wLclCycErr()	サイクリック(I／O)通信エラーフラグリセット	3.4.21
D22	mx500_rLclSetInt()	入力変化割込設定読出し	3.4.22
D23	mx500_wLclSetInt()	入力変化割込設定書込み	3.4.23
D24	mx500_rLclInt()	入力変化割込フラグ読出し	3.4.24
D25	mx500_wLclInt()	入力変化割込フラグリセット	3.4.25
D26	mx500_rPortDatB()	I/O ポートデータ 1バイト読出し	3.4.26
D27	mx500_wPortDatB()	I/O ポートデータ 1バイト書込み	3.4.27
D28	mx500_rPortDatW()	I/O ポートデータ 2バイト読出し	3.4.28
D29	mx500_wPortDatW()	I/O ポートデータ 2バイト書込み(DIO／モーションデバイス)	3.4.29
D30	mx500_rCenPortW()	センターデバイス 指定アドレス2バイト読出し	3.4.30
D31	mx500_wCenPortW()	センターデバイス 指定アドレス2バイト書込み	3.4.31
D32	mx500_rCenReg()	センターデバイス レジスタ読出し	3.4.32
D33	mx500_wCenReg()	センターデバイス レジスタ書込み	3.4.33
D34	mx500_rPclPort()	モーションデバイス 汎用出力ポート 出力状態読出し	3.4.34
D35	mx500_wPclPort()	モーションデバイス 汎用出力ポート 出力設定書込み	3.4.35
D36	mx500_rPclMsts()	モーションデバイス メインステータス読出し	3.4.36
D37	mx500_wPclCmd()	モーションデバイス 制御コマンド書込み	3.4.37
D38	mx500_rPclReg()	モーションデバイス レジスタ読出し	3.4.38
D39	mx500_wPclReg()	モーションデバイス レジスタ書込み	3.4.39

D40	mx500_rPclMltReg()	モーションデバイス 複数レジスター括読出し(C ﾓｼﾞｭｰﾙのみ)	3.4.40
D41	mx500_wPclMltReg()	モーションデバイス 複数レジスター括書込み(C ﾓｼﾞｭｰﾙのみ)	3.4.41
D42	mx500_PclMltWriteRead()	モーションデバイス 複数コマンド一括書込みと読出し(C ﾓｼﾞｭｰﾙのみ)	3.4.42
D43	mx500_rAmodAin()	アナログモジュール アナログ入力データ読出し	3.4.43
D44	mx500_rAmodAout()	アナログモジュール アナログ出力データ読出し	3.4.44
D45	mx500_wAmodAout()	アナログモジュール アナログ出力データ書込み	3.4.45
D46	mx500_wAmodCmd()	アナログモジュール 制御コマンド書込み	3.4.46
D47	mx500_rAmodReg()	アナログモジュール レジスタ設定データ読出し	3.4.47
D48	mx500_wAmodReg()	アナログモジュール レジスタ設定データ書込み	3.4.48
D49	mx500_rAmodStat()	アナログモジュール ステータス読出し	3.4.49
D50	mx500_rSmodResp()	シリアルモジュール コマンド応答受信	3.4.50
D51	mx500_wSmodCmd()	シリアルモジュール コマンド送信	3.4.51
D52	mx500_rSmodMessage()	シリアルモジュール メッセージ受信	3.4.52
D53	mx500_wSmodMessage()	シリアルモジュール 情報コマンド・メッセージ送信	3.4.53
D54	mx500_SetIntCall()	割込処理の登録	3.4.54
D55	mx500_ResetIntCall()	割込処理の削除	3.4.55
D56	mx500_WaitInt()	割込ｲﾍﾞﾝﾄ待ち	3.4.56
D57	mx500_GetIntData()	割込ｲﾍﾞﾝﾄ発生時情報取得	3.4.57
D58	mx500_GetSftVerNo()	ソフトウェアバージョンの取得	3.4.58

表 3.1-1 ドライバ関数一覧

3.2 プリレジスタ

CモジュールおよびFモジュール，Wモジュールの一部にはRMV，RFL，RFH，RUR，RDR，RMG，RDP，RMD，RIP，RUS，RDS，RCP3，RCI，RMVY，RYPIの各レジスタとスタートコマンドにプリレジスタがあります．プリレジスタとは，動作中に次の動作用データをセットしておくレジスタで，G9103のプリレジスタは下図のような構成になっています．プリレジスタを使用する場合は，まず「プリレジスタ」に対してデータを書き込みます．書き込まれた後は，前回指令したコマンドが終了して「レジスタ」内が空き次第自動的に「プリレジスタ」の内容が「レジスタ」へシフトされ，「プリレジスタ」は次のコマンドを受け入れられる状態になります．なお「プリレジスタ」にデータが残っているにも関わらず次のデータを書き込むと上書きされてしまいますので，空きを確認しながら書き込みを行う必要があります．
「プリレジスタ」の内容が「レジスタに送られるタイミングは，スタートコマンドを書き込んだタイミングになります．この時点で各「プリレジスタ」のデータは各「レジスタ」へ一斉転送されます．（書き込みを行っていないプリレジスタの内容は最後に書かれた値がレジスタに転送されます）

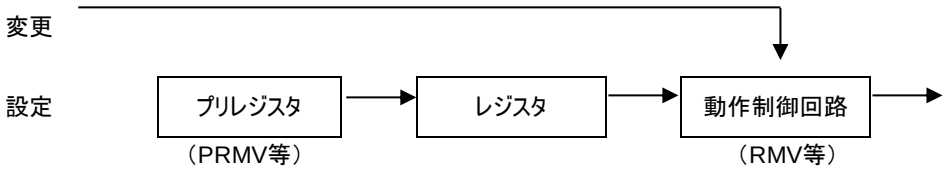


表 3.2-1 プリレジスタ

一度プリレジスタを使った運用を始めたら，その後の運用は必ずプリレジスタを経由した書き込みを行ってください．
これを守らなかった場合，その後の動作が正常に機能しない場合があります．
（運用でどうしても途中で切り替えが必要な場合は，そのモジュールに対してソフトウェアリセットを含む初期化処理を行なう必要があります）
ただし，速度および位置のオーバーライド等，現在の動作を変更する場合は「レジスタ」に直接新データを書込んで使用します．
なおライブラリーおよびドライバー関数を使った運用を行っている場合，ドライバー内で「レジスタ」／「プリレジスタ」を自動的に使い分けているため，ユーザー様は意識しないでご利用できます．（初期化関数を実施した時点でレジスタ使用方法が決定されます）

3.3 ドライバ関数の戻り値

関数の起動を行った結果は「戻り値」に実行結果が反映されます．
この戻り値はライブラリ関数と同じです．

3.4 関数の詳細

3.4.1 mx500_ApiStartup()　API 関数を活性化（初期化処理）

No.	D1
機　能	HLS-MCT520 用 API 関数を活性化（API の初期化処理）
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_ApiStartup(void);

引　数	説　明
	なし

VC++ 記述例	short　　ret; ret = mx500_ApiStartup();

3.4.2 mx500_ApiCleanup()　API 関数の使用を完了（資源解放）

No.	D2
機　能	HLS-MCT520 用 API 関数の使用を完了（API の資源を解放）
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_ApiCleanup(void);

引　数	説　明
	なし

VC++ 記述例	short　　ret; ret = mx500_ApiCleanup();

3.4.3 mx500_GetDeviceCount()　マスターボード枚数の取得

No.	D3
機　能	パソコンに装着されている motionCAT マスターボードで INtime に割付けられている枚数を取得します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_GetMstBrdCount(WORD *psDevCnt);

引　数	説　明
psDevCnt	[OUT] motionCAT マスタボードの枚数

VC++ 記述例	short　ret;　　//関数の戻り値 short　count;　//ボードの枚数 ret = mx500_GetMstBrdCount (&count);

3.4.4 It500_GetDeviceInfo()　マスターボード情報の取得

No.	D4
機　能	現在パソコンに装着されている motionCAT マスターボードの枚数とデバイス情報を取得します. デバイス情報は，デバイスオープン時に利用します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_GetDeviceInfo(WORD *psDevCnt, HPC_DEVINFO *pstDevInfo);

引　数	説　明
psDevCnt,	[OUT] motionCAT マスターボード枚数
pstDevInfo	[OUT] motionCAT マスタボードのデバイス情報

VC++
記述例
	short　ret;　　//関数の戻り値
	DWORD　count;　　//motionCAT マスターボードの枚数
	HPC_DEVINFO　AllDevInfo[16];　　//この例では 16 枚 b 分の motionCAT マスターボードの
	　　//デバイス情報の格納領域
	ret = mx500_GetDeviceInfo(&count, &AllDevInfo[0]);

備　考	デバイス情報を格納する領域のサイズについて Windows 版同名関数では，取得する枚数の上限をセットして渡していますが，INtime 版ではボード枚数を返す変数としてのみ使用します．したがってデバイス情報の領域も実装されているボード枚数以上の領域を確保する必要があります．特に問題でなければボード実装が可能な上限である 16 を推奨します.

3.4.5 mx500_OpenDevice()　マスターボードオープン

No.	D5
機　能	引数で渡したボード情報を持つ motionCAT マスターボードをオープンし，他ボードと識別するためのデバイスハンドルを取得します．以降このデバイスハンドルは目的の motionCAT マスターボードにアクセスするためのハンドルとなります.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_OpenDevice(DWORD *phDevID, HPC_DEVINFO *pstDevInfo);

引　数	説　明
phDevID	[OUT]　デバイスハンドル
pstDevInfo	[IN]　オープンする motionCAT マスターボードのデバイス情報の格納先（対象ボードの先頭アドレス）

VC++ 記述例	short　ret;　　//関数の戻り値 DWORD　count;　　//motionCAT マスターボードの枚数 ret = mx500_GetDeviceInfo(&count, &HpcDevInfo[1]);　//2 枚目のﾏｽﾀﾎﾞｰﾄﾞ情報が 　　//格納された領域の先頭ｱﾄﾞﾚｽ

3.4.6 mx500_CloseDevice()　　マスタボードクローズ

No.	D6
機　能	指定したデバイスハンドルを持つ motionCAT マスタボードをクローズします. 以降このデバイスハンドルは無効となり, このボードに対するアクセスはできません.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_CloseDevice(DWORD hDevID);

引　数	説　明
hDevID	[IN] デバイスハンドル

VC++ 記述例	short　ret;　　//関数の戻り値 ret = It500_CloseDevice(hDevID);

備　考	デバイスクローズの前に必ずパルス停止などの終了処理をしてください.

3.4.7 mx500_rOptPortB() マスターボード オプションポート 1 バイト読出し

No.	D7
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定オプションポートの内容を 1Byte 読出します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_rOptPortB (DWORD hDevID, WORD wOfsPort, BYTE* bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wOfsPort,	[IN] 読み出すポートのオフセットアドレス
bData	[OUT] オプションポートデータ

VC++
記述例
	Short ret; //関数の戻り値
	BYTE pdt;
	ret = mx500_rOptPortB(hDevID, 8, &pdt); // motionCAT マスターボード PCI 割込み状態を読出し

備考	オプションポートはワードアクセスのみしかできないため，内部的には 2Byte 読み込みをして要求のあった上位または下位バイトのみ返します.

3.4.8 mx500_wOptPortB() マスターボード オプションポート 1バイト書込み

No.	D8
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定オプションポートにデータを 1Byte 書込みます.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_wOptPortB (DWORD hDevID, WORD wOfsPort, BYTE bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wOfsPort,	[IN] 読み出すポートのオフセットアドレス
bData	[IN] オプションポートデータ

VC++
記述例
	Short ret; //関数の戻り値
	BYTE pdt;
	ret = mx500_wOptPortB(hDevID, 8, 1); // motionCAT マスターボード PCI 割込みを有効に設定.

備考	オプションポートはワードアクセスのみしかできないため，内部的には一度 2Byte 読み込みをしたのち，書き込む側の Byte データを元のデータと入れ替えてから書き戻しています.

3.4.9 mx500_rOptPortW() マスターボード オプションポート 2 バイト読出し

No.	D9
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定オプションポートの内容を 2Byte 読出します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_rOptPortW (DWORD hDevID, WORD wOfsPort, WORD* wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wOfsPort,	[IN] 読み出すポートのオフセットアドレス
wData	[OUT] オプションポートデータ

VC++
記述例
	Short ret; //関数の戻り値
	WORD pdt;
	ret = mx500_rOptPortW(hDevID, 8, &pdt); // motionCAT マスターボード PCI 割込み状態を読出し

3.4.10 mx500_wOptPortW() マスターボード オプションポート 2 バイト書込み

No.	D10
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定オプションポートにデータを 2Byte 書込みます.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_wOptPortB (DWORD hDevID, WORD wOfsPort, WORD wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wOfsPort,	[IN] 読み出すポートのオフセットアドレス
wData	[IN] オプションポートデータ

VC++
記述例
	Short ret; //関数の戻り値
	WORD pdt;
	ret = mx500_wOptPortW(hDevID, 8, 1); // motionCAT マスターボード PCI 割込みを有効に設定.

3.4.11 mx500_rCenMsts()　センターデバイス メインステータスの読出し

No.	D11
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインのセンターデバイス，メインステータスを読出します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_rCenMsts(DWORD hDevID, WORD wLine, WORD* wCSts);

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
wCSts	[OUT] センターデバイスメインステータス		
	ビット	名　称	説　　明
	0	CEND	データ送信用 FIFO 書き込み可能時に 1 になります．システム通信,またはデータ通信が完了して，データ送信用 FIFO に次データの書き込みが可能になった時に 1 になります．本ビットのクリア方法は RENV0.bit9 の状態によります.
	1	BRKF	ブレークフレーム受信時に 1 になります．本ビットのクリア方法は，RENV0.bit9 の状態によります.
	2	IOPC	「入力変化割り込み設定」を 1 にセットした入力ポートの状態が変化した時に 1 になります．「入力ポート変化フラグ」の全 256 ビットの OR 信号です．全ビットが 0 になると，このビットは 0 に戻ります.
	3	EIOE	サイクリック通信エラー発生時に 1 になります．「サイクリック通信エラーフラグ」の全 64 ビットの OR 信号です.
	4	EDTE	データ通信エラー発生時に 1 になります．本ビットのクリア方法は，RENV0.bit9 の状態によります.
	5	ERAE	ローカルデバイス側受信処理エラー発生時に 1 になります．本ビットのクリア方法は，RENV0.bit9 の状態によります.
	6	CAER	アプリケーションのアクセスエラーです．送信データが空のままでデータ送信コマンドを書き込む等，アプリケーションから不適切なアクセスがあると 1 になります．本ビットのクリア方法は，RENV0.bit9 の状態によります.
	8	REF	未送信の出力ポートデータがある時 1 になります．出力ポートエリアにデータを書き込むと 1 になり，全ポートへのサイクリック通信を 2 回以上エラー無しで行った後に 0 に戻ります.
	9	TDBB	データ送信用 FIFO に送信データがある時に 1 になります． データ送信用 FIFO に書き込むと 1 になり，データ送信コマンド，または送信用 FIFO リセットコマンドを書き込んだ時に 0 に戻ります.
	10	RDBB	データ受信用 FIFO に受信データがある時に 1 になります．モーションモジュール等のデータデバイスからデータを受信すると 1 になり，アプリケーションが受信データを全て読み出すと 0 に戻ります.
	12	SBSY	サイクリック通信スタート中に 1 になります.
	13	RBSY	リセット処理中に 1 になります.
	14	DBSY	システム通信中，またはデータ通信中に 1 になります.

VC++ 記述例	short ret;　　//関数の戻り値 WORD csts;　　//センターデバイスメインステータス ret = mx500_rCenMsts(hDevID, 0, &csts);　//ライン 1，センターデバイスメインステータス読出し

3.4.12 mx500_wCenCmd()　センターデバイス 制御コマンド書込み

No.	D12
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインのコマンドバッファへ制御コマンドデータを書込みます.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_wCenCmd(DWORD hDevID, WORD wLine, WORD wCmd);

引　数	説　明			
hDevID	[IN] デバイスハンドル			
wLine	[IN] ライン番号［0:Line1，1;Line2］			
wCmd	[IN] センターデバイスコマンド （g はグループ番号(0~7)，mm はモジュール ID(00~3f)，xx は各情報エリア(00~7f) ）			
	No.	コマンド	返答および付随データの有無と場所	名　称
	1	0000h	なし	無効コマンド
	2	0100h	なし	ソフトウェアリセット
	3	0200h	なし	送信用 FIFO リセット
	4	0300h	なし	受信用 FIFO リセット
	5	04mmh	なし	センター割込ステータスのクリア
	6	0600h	なし	エラーカウンタクリア
	7	0610h	なし	ブレーク通信コマンド．RENV0(8)＝“1”と設定して自動ブレーク機能を無効としたとき、本コマンドで任意のタイミングでブレーク通信を発行可能，RENV0(8)＝“0”の時は無効
	8	1000h	なし	全モジュールへのシステム通信
	9	1100h	なし	サイクリック通信除外中の全モジュールへのシステム通信
	10	12mmh	なし	指定したモジュールへのシステム通信
	11	13mmh	受信 FIFO	指定したモジュールの属性情報の取得
	12	2gmmh	なし	指定したグループへのコマンド送信
	13	3000h	なし	サイクリック通信の開始
	14	3100h	なし	サイクリック通信の停止
	15	40mmh	受信／送信 FIFO	データ通信
	16	41mmh	なし	データ通信キャンセル
	17	50xxh	出力バッファ	デバイス情報」エリアへの書き込み
	18	51xxh	出力バッファ	I/O 通信エラーフラグエリアへの書き込み
	17	52xxh	出力バッファ	入力変化割り込み設定エリアへの書き込み
	19	53xxh	出力バッファ	入力変化割り込みフラグ」エリアへの書き込み
	19	54xxh	出力バッファ	ポートデータエリアへの書き込み
	20	5500h	出力バッファ	RENV0 書込みコマンド
	21	60xxh	入力バッファ	デバイス情報エリアの読み出し
	22	61xxh	入力バッファ	I／O通信エラーフラグエリアの読み出し
	23	62xxh	入力バッファ	入力変化割り込み設定エリアの読み出し
	24	63xxh	入力バッファ	入力変化割り込みフラグエリアの読み出し
	25	64xxh	入力バッファ	ポートデータエリアの読み出し
	26	6500h	入力バッファ	RENV0 読出しコマンド
	27	6501h	入力バッファ	エラーカウンタリードコマンド
	28	6502h	入力バッファ	サイクリック周期レジスタリードコマンド
	29	6503h	入力バッファ	受信アドレスレジスタリードコマンド
	30	6504h	入力バッファ	バージョン情報レジスタ リードコマン

VC++ 記述例	short　ret;　　//関数の戻り値 ret = mxt500_wCenCmd(hDevID, 1, 0x0100);　//ライン 2，ソフトウェアリセット

3.4.13 mx500_rCenIsts()　センターデバイス 割込ステータス読出し

No.	D13
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインの割込みステータスを読出します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_rCenIsts(DWORD hDevID, WORD wLine, WORD* wISts);

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1, 1;Line2］		
wISts	[OUT] センターデバイス割込ステータス		
	ビット	名　称	説　　明
	0	EDN0	ステータスのEDTE=1または、ERAE=1エラー発生時のデバイス番号. 次回のエラー発生まで記憶されます
	1	EDN1	
	2	EDN2	
	3	EDN3	
	4	EDN4	
	5	EDN5	
	7	LNRV	ローカル側データ未受信時に1になります. データ通信がエラーで終了した（EDTE=1）場合, ローカル側がセンターからのデータを受信出来なかった時（ローカル側は無応答）は1になり, 受信出来た時（ローカル側からの通信が通信障害などで破壊され, センタ側が正常に受信できなかった場合. この場合ローカル側が正常にデータを受け取れたかどうかを確認する作業が必要となります）は0になります. 次回のエラー発生まで記憶されます。
	8	ERA0	ローカル側が正常受信したにもかかわらず, そのパケットの内容がローカルデバイスの種類にマッチしていない場合に発生. このとき以下のようなコードを本ビット部分に記憶します。このコードは次回のエラー発生まで記憶されます。 ■コード発生条件 0001 ･･･ センターデバイス内の“デバイス情報エリア”のI／Oの設定情報とローカル側のI/Oポートの組合せ（PMD端子にて行います）が異なっている場合 0010 ･･･ I/Oデバイスがデータ通信を受信した時 0011 ･･･ データデバイスが自己の持っている受信ﾊﾞｯﾌｧ容量以上のデータ通信を受信したとき
	9	ERA1	
	10	ERA2	
	11	ERA3	
	12	CAE0	CPU が G9001(A)対して不正なアクセスを行った場合, 以下のようなコードを本ビット部分に記憶. このコードは次回のエラー発生まで記憶されます. ■コード発生条件 0001 ･･･ 使用デバイス数がゼロで I/O 通信スタートコマンドを書き込んだ時 0010 ･･･ 送信データを送信用 FIFO に設定しないでデータ送信スタートコマンド書き込んだ時 0011 ･･･ DBSY=1 の時に、以下のアクセスを行った時 ・受信用 FIFO からの読み出し, または送信用 FIFO への書き込みを行ったとき ・システム通信, またはデータ通信スタートコマンドを書き込んだ時 0100 ･･･ デバイス情報エリアのビット7が“0”となっている（未使用デバイス扱いとなっている）デバイスに対して、データ通信を行った時
	13	CAE1	
	14	CAE2	
	15	CAE3	

VC++ 記述例	short　ret;　　//関数の戻り値 WORD　 ists;　　//センターデバイス割込ステータス ret = mx500_rCenIsts(hDevID, 0, &csts);　//ライン 1, センターデバイス割込ステータス読出し

3.4.14 mx500_rCenBuf()　センターデバイス 入力バッファ読出し

No.	D14
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインの入力バッファを読出します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_rCenBuf(DWORD hDevID, WORD wLine, WORD* wInBuf);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wInBuf	[OUT] 入力バッファデータ

VC++ 記述例	short　ret;　　//関数の戻り値 WORD bfdt;　　//入力バッファデータ ret = mx500_wCenCmd(hDevID, 1, 0x6500);　　//ライン 1，RENV0 レジスタ読み出し ret = mx500_rCenBuf(hDevID, 1, &bfdt);
備考	センターデバイスの各エリアおよびレジスタからのデータ読出し手順は (1)センターデバイスコマンドを書込む(60xx～64xx,6500,6501～6504) (2)入力バッファを読み出す となります. なお，入力バッファの内容を読み出す前段において，データの読出しコマンドが発行される処理については，これら一連の動作を 1 つの関数内で行っている mx500_rCenReg()関数があります．通常はこちらをご使用ください.

3.4.15 mx500_wCenBuf()　センターデバイス 出力バッファ書込み

No.	D15
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインの出力バッファにデータを書込みます.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_wCenBuf(DWORD hDevID, WORD wLine, WORD wOutBuf);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wData	[IN] 出力バッファデータ

VC++ 記述例	short　ret;　　//関数の戻り値 ret = mx500_wCenBuf(hDevID, 1, 0x017c);　//ライン 1，RENV0 レジスタへ 017ch 書き込み ret = mx500_wCenCmd(hDevID, 1, 0x5500);
備考	センターデバイスの各エリアおよびレジスタへのデータ書込み手順は (1)出力バッファにデータを書き込む (2)センターデバイスコマンドを書込む(50xx～54xx,5500) となります. なお，出力バッファにデータを書き込んだ後にそのデータをレジスタ等に書き込む処理を行う場合は，一連の書き込み動作を 1 つの関数内で行っている mx500_wCenReg()関数をご使用ください.

3.4.16 mx500_rCenRFIFO()　センターデバイス 受信用 FIFO 読出し

No.	D16
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインの受信用 FIFO を読出します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_rCenRFiFo(DWORD hDevID, WORD wLine, WORD* wReceive);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wReceive	[OUT] 受信用 FIFO データ

VC++
記述例
	short　ret;　　//関数の戻り値
	WORD dt;　　//入力 FIFO 読出しデータ格納先
	//ライン 1，モジュール 5 の属性情報を読出し
	ret = mx500_wCenCmd(hDevID, 0, 0x1305);　//属性情報取得コマンド送信
	***応答データ待ち処理
	ret = mx500_rCenRFiFo(hDevID, 0, &dt);　　//取り出し

備考	各モジュールからのデータを読み出す場合の一般的な手順は (1)送信用 FIFO にコマンド書き込み（mx500_wCenSFIFO 使用） （データ読出し時は通常 1 ワード） (2)データ通信コマンド(4000h～403f)送信 (3)応答受信があるまで待つ (4)データ受信用 FIFO からデータを読み出し （コマンドによって返ってくるデータサイズは異なります） となります. 属性情報取得のコマンド（コマンド 1300h～133fh）は(1)の手順がありません またモーションモジュールのレジスタ読出しを行う場合は，通常は mx500_rPclReg をご使用ください. この関数は上記 1～4 の手順を 1 つの関数内で行います.

3.4.17 mx500_wCenSFIFO()　センターデバイス 送信用 FIFO 書込み

No.	D17
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインの送信用 FIFO にデータを書込みます.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_wCenSFiFo(DWORD hDevID, WORD wLine, WORD wSend);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1, 1;Line2］
wSend	[IN] 送信用 FIFO データ

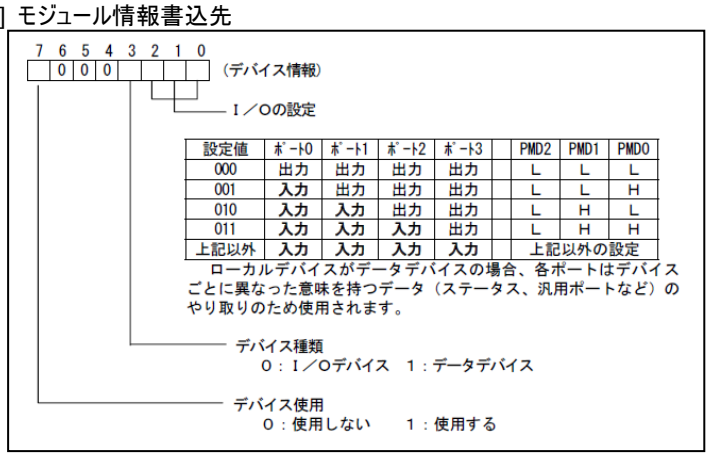
VC++
記述例
	short ret; //関数の戻り値
	//ライン 1, モジュール 5 の RMV レジスタに 30000 を書込み
	//送信 FIFO へ書込み
	ret = mx500_wCenSFiFo(hDevID, 0, 0x00d0); //RMV レジスタ読出しコマンド
	ret = mx500_wCenSFiFo(hDevID, 0, 0x93e0); //下位データ
	ret = mx500_wCenSFiFo(hDevID, 0, 0x0004); //上位データ
	ret = mx500_wCenCmd(hDevID, 0, 0x4005); //データ通信（送信 FIFO の内容を送信）
	～データ通信完了待ち処理 → 完了

備考	各モジュールへデータを書き込む場合の一般的な手順は (1)送信用 FIFO にコマンドおよびデータを指定された順番で書き込み（mx500_wCenSFIFO 使用） （コマンドによってデータサイズは異なります） (2)データ通信コマンド(4000h～403f)送信 (3)データ通信完了まで待つ となります. またモーションモジュールのレジスタ書込みを行う場合は, 通常は mx500_wPclReg をご使用ください. この関数は上記 1～3 の手順を 1 つの関数内で行います.

3.4.18 mx500_rLclInfo()　　ローカルデバイス情報読出し

No.	D18
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのモジュール情報を読出します.
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_rLclInfo(DWORD hDevID, WORD wLine, WORD wMid, BYTE* bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
bData	[OUT] モジュール情報書込先  ※ I/O デバイス･･･G9002 搭載モジュール（I/O/D/T/A/B/S/R） ※ データデバイス･･･G9003/G9103 搭載モジュール（P/M/W/C/F モジュール）

VC++ 記述例	short　ret;　　//関数の戻り値 WORD info;　　//モジュール情報データ ret = mx500_rLclInfo(hDevID, 0, 1, &info);　//ライン 1，モジュール 1 のモジュール情報

3.4.19 mx500_wLclInfo()　　ローカルデバイス情報書込み

No.	D19
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのモジュール情報を書込みます.
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_wLclInfo(DWORD hDevID, WORD wLine, WORD wMid, BYTE bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
bData	[IN] モジュール情報

VC++ 記述例	short　ret;　　//関数の戻り値 WORD info;　　//モジュール情報データ ret = mx500_wLclInfo(hDevID, 0, 1, 0x8f);　//ライン 1，モジュール 1，モジュール I/O を全入力で設定

3.4.20 mx500_rLclCycErr()　　サイクリック通信(I/O)エラーフラグ読出し

No.	D20
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのサイクリック通信エラーフラグを読出します．モジュールの指定は単一のモジュールを指定するのではなく，1 データに 16 モジュール分のエラー情報を持つデータ単位でのモジュールブロック番号の指定(0～3)になります
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_rLclCycErr(DWORD hDevID, WORD wLine, WORD wMidPrm, WORD* wFlag);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidPrm	[IN] モジュールブロック番号[0～3]
wFlag	[OUT] 通信エラーフラグの状態（1 データは 16 モジュール分） (0:エラーなし　1:エラー発生) モジュール ID 先頭　tMid=wMidPrm×16

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
tMid+7 エラーフラグ	tMid+6 エラーフラグ	tMid+5 エラーフラグ	tMid+4 エラーフラグ	tMid+3 エラーフラグ	tMid+2 エラーフラグ	tMid+1 エラーフラグ	tMid+0 エラーフラグ
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
tMid+15 エラーフラグ	tMid+14 エラーフラグ	tMid+13 エラーフラグ	tMid+12 エラーフラグ	tMid+11 エラーフラグ	tMid+10 エラーフラグ	tMid+9 エラーフラグ	tMid+8 エラーフラグ

VC++ 記述例

```
short   ret;            //関数の戻り値
WORD edt;       //通信エラーデータ
ret = mx500_rLclCycErr( hDevID, 0, 1, &edt );    //ライン 1，モジュール 20(モジュールブロック番号 1)
                                        → 取得したデータの Bit4 がモジュール 20 のエラー状態
```

3.4.21 mx500_wLclCycErr()　　サイクリック通信(I/O)エラーフラグリセット

No.	D21
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのサイクリック通信エラーフラグをクリアします．クリアはクリアするモジュールに対応するビットを 1 にして書き込みます． モジュールの指定は単一のモジュールを指定するのではなく，1 データに 16 モジュール分のエラー情報を持つデータ単位でのモジュールブロック番号の指定(0～3)になります
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_wLclCycErr(DWORD　hDevID，WORD　wLine，WORD　wMidPrm，WORD　wFlag);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidPrm	[IN] モジュールブロック番号[0～3]
wFlag	[IN] 通信エラーフラグのクリア要求（1 データは 16 モジュール分） (0:処理しない　1:エラーリセット要求) モジュール ID 先頭 tMid=wMidPrm×16

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
tMid+7 ｴﾗｰ ﾘｾｯﾄ	tMid+6 ｴﾗｰ ﾘｾｯﾄ	tMid+5 ｴﾗｰ ﾘｾｯﾄ	tMid+4 ｴﾗｰ ﾘｾｯﾄ	tMid+3 ｴﾗｰ ﾘｾｯﾄ	tMid+2 ｴﾗｰ ﾘｾｯﾄ	tMid+1 ｴﾗｰ ﾘｾｯﾄ	tMid+0 ｴﾗｰ ﾘｾｯﾄ
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
tMid+15 ｴﾗｰ ﾘｾｯﾄ	tMid+14 ｴﾗｰ ﾘｾｯﾄ	tMid+13 ｴﾗｰ ﾘｾｯﾄ	tMid+12 ｴﾗｰ ﾘｾｯﾄ	tMid+11 ｴﾗｰ ﾘｾｯﾄ	tMid+10 ｴﾗｰ ﾘｾｯﾄ	tMid+9 ｴﾗｰ ﾘｾｯﾄ	tMid+8 ｴﾗｰ ﾘｾｯﾄ

```
VC++
記述例
short   ret;            //関数の戻り値
//ライン 1，モジュール 20(モジュールブロック番号 1)のエラーリセット
ret = mx500_rLclCycErr( hDevID, 0, 1, 0x0010 );
```

3.4.22 mx500_rLclSetInt()　　入力変化割込設定読出し

No.	D22
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの入力変化割込の設定を読出します．モジュールの指定は単一のモジュールを指定するのではなく，1 データに 4 モジュール分のエラー情報を持つデータ単位でのモジュールブロック番号の指定(0～15)になります
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_rLclSetInt(DWORD　hDevID,　WORD　wLine,　WORD　wMidPrm,　WORD* wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidPrm	[IN] モジュールブロック番号[0～15]
wData	[OUT] 入力変化フラグの設定状態（1 データは 4 モジュール分）（0:割込無効 1:割込み有効） モジュール ID 先頭 tMid=wMidPrm×4

wData

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
モジュール ID tMid+1				モジュール ID tMid+0			
ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効	ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
モジュール ID tMid+3				モジュール ID tMid+2			
ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効	ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効

```
VC++
記述例
short   ret;          //関数の戻り値
WORD idt;             //入力変化フラグ割込み設定データ
ret = mx500_rLclSetInt( hDevID, 0, 5, &idt );   //ライン 1，モジュール 20(モジュールブロック番号 5)
                → 取得したデータの Bit0～3 がモジュール 20，ポート 0～3 の設定状態
```

3.4.23 mx500_wLclSetInt()　　入力変化割込設定書込み

No.	D23
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの入力変化割込の設定を書込みます．モジュールの指定は単一のモジュールを指定するのではなく，1 データに 4 モジュール分のエラー情報を持つデータ単位でのモジュールブロック番号の指定(0～15)になります
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_wLclSetInt(DWORD　hDevID,　WORD　wLine,　WORD　wMidPrm,　WORD　wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidPrm	[IN] モジュールブロック番号[0～15]
wData	[IN] 入力変化フラグの設定（1 データは 4 モジュール分）（0:割込無効 1:割込み有効） モジュール ID 先頭 tMid=wMidPrm×4

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
モジュール ID tMid+1				モジュール ID tMid+0			
ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効	ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
モジュール ID tMid+3				モジュール ID tMid+2			
ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効	ポート 3 割込 有効	ポート 2 割込 有効	ポート 1 割込 有効	ポート 0 割込 有効

VC++ 記述例	short　ret;　　//関数の戻り値 ret = mx500_wLclSetInt(hDevID, 0, 5, 0x0003);　//ライン 1，モジュール 20(モジュールブロック番号 5) → モジュール 20 のポート 0,1 を割込み有効に設定

3.4.24 mx500_rLclInt()　　入力変化割込フラグ読出し

No.	D24
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの入力変化割込フラグの状態を読出します．モジュールの指定は単一のモジュールを指定するのではなく，1 データに 4 モジュール分のエラー情報を持つデータ単位でのモジュールブロック番号の指定(0～15)になります
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_rLclInt(DWORD hDevID, WORD wLine, WORD wMidPrm, WORD* wFlag);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidPrm	[IN] モジュールブロック番号[0～15]
wFlag	[OUT] 入力変化フラグの入力状態（1 データは 4 モジュール分）（0:変化なし 1:入力変化あり） モジュール ID 先頭 tMid=wMidPrm×4

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
モジュール ID tMid+1				モジュール ID tMid+0			
ポート 3 変化あり	ポート 2 変化あり	ポート 1 変化あり	ポート 0 変化あり	ポート 3 変化あり	ポート 2 変化あり	ポート 1 変化あり	ポート 0 変化あり
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
モジュール ID tMid+3				モジュール ID tMid+2			
ポート 3 変化あり	ポート 2 変化あり	ポート 1 変化あり	ポート 0 変化あり	ポート 3 変化あり	ポート 2 変化あり	ポート 1 変化あり	ポート 0 変化あり

VC++ 記述例

```
short   ret;            //関数の戻り値
WORD idt;           //入力変化フラグ入力状態データ
ret = mx500_rLclInt( hDevID, 0, 5, &idt );    //ライン 1，モジュール 20(モジュールブロック番号 5)
                → 取得したデータの Bit0～3 がモジュール 20，ポート 0～3 の入力状態
```

3.4.25 mx500_wLclInt()　　入力変化割込フラグリセット

No.	D25
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの入力変化割込フラグの状態をクリアします．モジュールの指定は単一のモジュールを指定するのではなく，1 データに 4 モジュール分のエラー情報を持つデータ単位でのモジュールブロック番号の指定(0～15)になります
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_wLclInt(DWORD hDevID, WORD wLine, WORD wMidPrm, WORD wFlag);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMidPrm	[IN] モジュールブロック番号[0～15]
wFlag	[IN] 入力変化フラグの状態（1 データは 4 モジュール分）（0:処理なし 1:フラグリセット要求） モジュール ID 先頭 tMid=wMidPrm×4

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
モジュール ID tMid+1				モジュール ID tMid+0			
ポート 3 フラグクリア	ポート 2 フラグクリア	ポート 1 フラグクリア	ポート 0 フラグクリア	ポート 3 フラグクリア	ポート 2 フラグクリア	ポート 1 フラグクリア	ポート 0 フラグクリア
Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
モジュール ID tMid+3				モジュール ID tMid+2			
ポート 3 フラグクリア	ポート 2 フラグクリア	ポート 1 フラグクリア	ポート 0 フラグクリア	ポート 3 フラグクリア	ポート 2 フラグクリア	ポート 1 フラグクリア	ポート 0 フラグクリア

VC++ 記述例	short　ret;　　//関数の戻り値 ret = mx500_wLclInt(hDevID, 0, 5, 0x0003);　//ライン 1，モジュール 20(モジュールブロック番号 5) → モジュール 20 のポート 0,1 のフラグをクリア

3.4.26 mx500_rPortDatB()　　I/O ポートデータ 1 バイト読出し

No.	D26
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの指定 I/O ポートデータを 1Byte 読出します.
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_rPortDatB(DWORD hDevID, WORD wLine, WORD wMid, WORD wPortNo, BYTE* bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wPortNo	[IN] ポート番号[0～3]
bData	[OUT] I/O ポートデータ

VC++
記述例
	short　ret;　　//関数の戻り値
	BYTE　pdt;　　//ポートデータ
	// ライン 1，モジュール 0，ポート 2 の I/O ポート読出し
	ret = mx500_rPortDatB(hDevID, 0, 0, 2, &pdt);

備考	ポートデータはワードアクセスのみしかできないため，内部的には 2Byte 読み込みをして要求のあった上位または下位バイトのみ返します.

3.4.27 mx500_wPortDatB()　　I/O ポートデータ 1バイト書込み

No.	D27
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの指定 I/O ポートデータを 1Byte 書込みます.
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_wPortDatB(DWORD hDevID, WORD wLine, WORD wMid, WORD wPortNo, BYTE bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wPortNo	[IN] ポート番号[0～3]
bData	[IN] I/O ポートデータ

VC++
記述例
	short　ret;　　//関数の戻り値
	// ライン 1,モジュール 0,ポート 2 の I/O ポートへ 5ah を書込み
	ret = mx500_rPortDatB(hDevID, 0, 0, 2, 0x5a);

備考	ポートデータはワードアクセスのみしかできないため，内部的には一度 2Byte 読み込みをしたのち，書き込む側の Byte データを元のデータと入れ替えてから書き戻しています.

3.4.28 mx500_rPortDatW()　　I/O ポートデータ 2 バイト読出し

No.	D28
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの指定 I/O ポートデータを 2Byte 読出します.
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_rPortDatW(DWORD hDevID, WORD wLine, WORD wMid, WORD wPortNo, WORD* wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wPortNo	[IN] ポート番号[0,2] ※奇数ポート番号は指定不可
wData	[OUT] I/O ポートデータ

VC++
記述例
	short　ret;　　//関数の戻り値
	WORD　pdt;　　//ポートデータ
	// ライン 1,モジュール 0,ポート 2,3 の I/O ポート読出し
	ret = mx500_rPortDatW(hDevID, 0, 0, 2, &pdt);

3.4.29 mx500_wPortDatW()　　I/O ポートデータ 2 バイト書込み

No.	D29
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールの指定 I/O ポートデータを 2Byte 書込みます.
対　象	全モジュール

開発環境	書　式
VC++	short　mx500_wPortDatB(DWORD hDevID, WORD wLine, WORD wMid, WORD wPortNo, WORD wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wPortNo	[IN] ポート番号[0,2] ※奇数ポート番号は指定不可
wData	[IN] I/O ポートデータ

VC++
記述例
	short　ret;　　//関数の戻り値
	// ライン 1，モジュール 0，ポート 2,3 に 305ah を書込み
	ret = mx500_rPortDatW(hDevID, 0, 0, 2, 0x305a);

3.4.30 mx500_rCenPortDatW()　　センターデバイス 指定アドレス 2 バイト読出し

No.	D30
機　能	デバイスハンドルで指定された motionCAT マスターボードのマッピングエリア先頭から指定されたオフセットアドレスのデータ 2Byte を読出します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_rCenPortW(DWORD hDevID, WORD wOfsPort, WORD* wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wOfsPort,	[IN] センターデバイスのマッピングエリア先頭からのオフセット ※奇数オフセットは指定不可
wData	[OUT] ポートデータ

VC++
記述例
	short　ret;　　//関数の戻り値
	WORD　dat;　　//データ
	// オフセット 200h(ライン 2 のセンターデバイスメインステータス)読出し
	（→通常は mx500_rCenMsts()関数を使用します）
	ret = mx500_rCenPortDatW(hDevID, 0x200, &dat);

3.4.31 mx500_wCenPortDatW()　　センターデバイス 指定アドレス 2 バイトデータ書込み

No.	D31
機　能	デバイスハンドルで指定された motionCAT マスターボードのマッピングエリア先頭から指定されたオフセットアドレスへ 2Byte のデータを書込みます.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_wCenPortW(DWORD hDevID, WORD wOfsPort, WORD wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wOfsPort,	[IN] センターデバイスのマッピングエリア先頭からのオフセット ※奇数オフセットは指定不可
wData	[IN] ポートデータ

VC++
記述例
	short　ret;　　//関数の戻り値
	// オフセット 302h(ライン 2，モジュール 0，ポート 2,3)にポートデータ 0x1234 を書込み
	ret = mx500_wCenPortDatW(hDevID, 0x302, 0x1234);

3.4.32 mx500_rCenReg()　　センターデバイス レジスタ読出し

No.	D32
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインに対してデータを読み出すためにセンターデバイスコマンド発行，送信完了待ち，入力ﾊﾞｯﾌｧからのデータ取り出し等の一連の処理を行います.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_rCenReg(DWORD hDevID, WORD wLine, WORD wCmd, WORD* wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wCmd	[IN] センターデバイスコマンド[60xxh～64xxh, 6500h, 6501h～6504h] xx は 0～7fh
wData	[OUT] レジスタデータ

VC++ 記述例	short　ret;　　//関数の戻り値 WORD dat;　　//レジスタデータ ret = mx500_rCenReg(hDevID, 0, 0x6500, &dat);　//ライン 1，RENV0 読出し

備考	この関数はセンターデバイス内のデータを読出すため，mx500_wCenCmd()，mx500_rCenBuf()等を使い，一つの関数にまとめたものです.

3.4.33 mx500_wCenReg()　　センターデバイス レジスタ書込み

No.	D33
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ラインに対して，出力ﾊﾞｯﾌｧへのデータ書込み，センターデバイスコマンド発行，送信完了待ち等の一連の処理を行います.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short　mx500_wCenReg(DWORD hDevID, WORD wLine, WORD wCmd, WORD wData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wCmd	[IN] センターデバイスコマンド[50xxh～54xxh, 5500h] xx は 0～7fh
wData	[IN] レジスタデータ

VC++ 記述例	short　ret;　　//関数の戻り値 ret = mx500_wCenReg(hDevID, 0, 0x5500, 0x01fc);　//ライン 1，RENV01fch 書込み

備考	この関数はセンターデバイスにデータを書込むため，mx500_wCenBuf()，mx500_wCenCmd()等を使い，一つの関数にまとめたものです.

3.4.34 mx500_rPclPort()　　モーションデバイス 汎用出力ポート 出力状態読出し

No.	D34
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールに対して汎用出力ポートの状態を読出します.
対　象	P/M/C/F/V(旧 W) モジュール (G9003/9103 搭載モジュール)

開発環境	書　式
VC++	short　mx500_rPclPort(DWORD hDevID, WORD wLine, WORD wMid, BYTE *bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
bData	[OUT] 出力状態

VC++ 記述例	short　ret;　　//関数の戻り値 BYTE dat;　　//出力状態データ ret = mx500_rPclPort(hDevID, 0, 0, &dat);　//ライン 1，モジュール 0，出力状態読出し

3.4.35 mx500_wPclPort()　　モーションデバイス 汎用出力ポート 出力設定書込み

No.	D35
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールに対して汎用出力ポートに出力設定を書込みます.
対　象	P/M/C/F/V(旧 W)モジュール (G9003/9103 搭載モジュール)

開発環境	書　式
VC++	short　mx500_wPclPort(DWORD hDevID, WORD wLine, WORD wMid, BYTE bData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
bData	[IN] 出力設定データ

VC++ 記述例	short　ret;　　//関数の戻り値 ret = mx500_wPclPort(hDevID, 0, 0, 0x5a);　//ライン 1，モジュール 0，汎用出力 5ah 書込み

3.4.36 mx500_rPclMStst()　　モーションデバイス メインステータスの読出し

No.	D36
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールに対してモーションメインステータスを読出します.
対　象	P/M/C/F/V(旧 W)モジュール (G9003/9103 搭載モジュール)

開発環境	書　式
VC++	short　mx500_rPclMsts(DWORD hDevID, WORD wLine, WORD wMid, WORD* wMSts);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wMSts	[OUT] モーションメインステータス

VC++
記述例
	short　ret;　　//関数の戻り値
	WORD sts;　　//メインステータス
	//ライン 1，モジュール 0，モーションメインステータス読出し
	ret = mx500_rPclMsts(hDevID, 0, 0, &sts);

3.4.37 mx500_wPclCmd()　　モーションデバイス コマンド書込み

No.	D37
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールに対してモーション制御コマンドを書込みます.

開発環境	書　式
VC++	short　mx500_wPclCmd(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmd);
対　象	P/M/C/F/V(旧 W)モジュール (G9003/9103 搭載モジュール)

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wCmd	[IN] モーション制御コマンド

VC++ 記述例	short　ret;　　//関数の戻り値 ret = mx500_wPclCmd(hDevID, 0, 0, 0x4a);　//ライン 1，モジュール 0，減速停止ｺﾏﾝﾄﾞ書込み

3.4.38 mx500_rPclReg()　　モーションデバイス レジスタ読出し

No.	D38
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのレジスタ内容を読み出します.
対　象	P/M/C/F/V(旧 W)モジュール (G9003/9103 搭載モジュール)

開発環境	書　式
VC++	short mx500_rPclReg(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmd, DWORD* dwData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wCmd	[IN] モーションレジスタ操作コマンド
dwData	[OUT] レジスタデータ

VC++
記述例
	short ret; //関数の戻り値
	WORD reg; //レジスタデータ
	//ライン 1,モジュール 0,RMV レジスタ読出し
	ret = mx500_rPclReg(hDevID, 0, 0, 0x00d0, &reg);

3.4.39 mx500_wPclReg()　　モーションデバイス レジスタ書込み

No.	D39
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのレジスタにデータを書込みます.
対　象	P/M/C/F/V(旧 W)モジュール (G9003/9103 搭載モジュール)

開発環境	書　式
VC++	short mx500_wPclReg(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmd, DWORD dwData);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wCmd	[IN] モーションレジスタ操作コマンド
dwData	[IN] レジスタデータ

VC++
記述例
	short ret; //関数の戻り値
	//ライン 1,モジュール 0,RFH レジスタ 10000 を設定
	ret = mx500_wPclReg(hDevID, 0, 0, 0x0092, 10000);

3.4.40 mx500_rPclMultReg()　　モーションデバイス 複数レジスター括読出し

No.	D40
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールに FIFO を使って複数のレジスタ読出しコマンドを一括で書込み，書き込んだコマンドに対する応答データを一括読出しします.
対　象	C/F　モジュール (G9103 搭載モジュール)

開発環境	書　式
VC++	Short　mx500_rPclMltReg(DWORD hDevID, WORD wLine, WORD wMid, WORD* wCmd, DWORD* dwReg, WORD wRegNum);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wCmd	[IN] レジスタ読出しコマンド（複数コマンド可能）
dwReg	[OUT] 各レジスタデータ（コマンド個数分のデータ）
wRegNum	[IN] ﾚｼﾞｽﾀ数

VC++
記述例
	short　ret;　　//関数の戻り値
	WORD cmd[4];
	DWORD reg[4]
	//ライン 1,モジュール 0 に 4 つの複数読出しコマンドが書き込まれた cmd を指定し，reg に
	//その応答データを取得.
	ret = mx500_rPclMultReg(hDevID, 0, 0, &cmd[0], &reg[0], 4);

備考	レジスタ読出しコマンドのみが対象 ただし送受信するデータサイズおよび格納先の管理はユーザー様において行ってください

3.4.41 mx500_wPclMultReg()　　モーションデバイス　複数レジスター括書込み

No.	D41
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールに FIFO を使って複数のレジスタ書込みコマンドを一括で書込みます.
対　象	C/F モジュール (G9103 搭載モジュール)

開発環境	書　式
VC++	Short　mx500_wPclMltReg(DWORD hDevID, WORD wLine, WORD wMid, WORD* wCmd, DWORD* dwReg, WORD wRegNum);

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wCmd	[IN] レジスタ書込みコマンド（複数コマンド可能）
dwReg	[IN] 各レジスタデータ（コマンド個数分のデータ）
wRegNum	[IN] ﾚｼﾞｽﾀ数

VC++
記述例
	short　ret;　　//関数の戻り値
	WORD cmd[4];
	DWORD reg[4];
	//ライン 1,モジュール 0 に 4 つの複数書込みコマンドとそのデータが書き込まれた cmd と reg を
	//指定して書込み.
	ret = et = mx500_wPclMultReg(hDevID, 0, 0, &cmd[0], &reg[0], 4);;

備考	レジスタ読出しコマンドのみが対象

3.4.42 mx500_PclMltSndRcv()　モーションデバイス 複数コマンド送受信

No.	D42
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールに FIFO を使って複数のレジスタ読出しコマンド，レジスタ書込みコマンド，制御コマンドを一括送信し，読出しコマンドに対する応答データがある場合はそれらのデータを一括で受信します.
対　象	C/F モジュール (G9103 搭載モジュール)

開発環境	書　式
VC++	Short　mx500_PclMltSndRcv(DWORD hDevID, WORD wLine, WORD wMid, WORD* wWriteData, WORD wWriteNum, WORD* wReadData, WORD* wReadNum)

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wWriteData	[IN] 送信データ
wSendNum	[IN] 送信データ数(WORD 単位)
wRecvData	[OUT] 受信データ 読出しコマンド1コマンドあたり以下のデータ(6Byte/1 コマンド)が受信されます. データは送信データで送られた順番，かつデータを返すコマンドの応答データは詰めてセットされます. +0:読出しコマンドコード +2:読出しデータ下位 +4:読出しデータ上位
wRecvNum	[OUT] 受信データ数(WORD 単位)

VC++
記述例
	short　ret;　　//関数の戻り値
	WORD wdat[5];
	WORD rdat[6];
	WORD rsiz;
	//ライン 1,モジュール 0 に 4 つの複数コマンド（書込み 1,制御コマンド 1,読出しコマンド 2）が
	//書き込まれた wdat を指定し，読出しコマンドの応答データとサイズを rdat，rsiz に取得.
	ret = mx500_PclMltSndRcv(hDevID, 0, 0, &wdat, 5, &rdat, &rsiz);

備考	レジスタ読出しコマンド，レジスタ書込みコマンド，制御コマンドが対象（混在可能） ただし送受信するデータサイズとその格納先の管理はユーザー様において行ってください

3.4.43 mx500_rAmodAin アナログモジュール アナログ入力データ読出し

No.	D43
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュール，指定チャンネルからアナログモジュールのアナログ入力データを読み出します.
対　象	A/B/HMS-A モジュール (G9002 搭載モジュールの一部)

言語	書　式
VC++	short mx500_rAmodAin(DWORD hDevID, WORD wLine, WORD wMid, WORD wCh, WORD* wData, WORD* wSts)

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
wMid	[IN] モジュール ID[0～63]		
wCh	[IN] チャンネル番号[0:CH1 / 1:CH2 / 2:CH3 / 3:CH4 / 4:CH5 / 5:CH6 / 6:CH7 /7:CH8] ※HMS-A8001 以外は CH4 まで		
wData	[OUT] アナログ入力データ[0～4095／0～10V]		
wSts	[OUT] ステータス情報		
	ビット	名　称	説　　明
	11-0	(DATA)	0 固定
	12	12/8	読み出したデータのデータ長（0:8Bit / 1:12Bit）
	13	(D/S)	1 固定
	14	OOR	各チャンネルのいずれかのコンパレータが範囲外を検出した場合に 1
	15	ACK	コマンドを受け付ける毎に反転. 実行結果の成否には関係ありません.

VC++
記述例
	short ret; //関数の戻り値
	WORD wdat;
	WORD st;
	//ライン 2,モジュール 7 のアナログモジュールからチャンネル CH4 のアナログを入力
	ret = mx500_rAmodAin(hDevID, 1, 7, 3, &wdat, &st);

3.4.44 mx500_rAmodAout アナログモジュール アナログ出力データ読出し

No.	D45
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュール，指定チャンネルからアナログモジュールの出力データを読出します.
対　象	A/B/HMS-A モジュール (G9002 搭載モジュールの一部)

言語	書　式
VC++	short mx500_rAmodAout(DWORD hDevID, WORD wLine, WORD wMid, WORD wCh, WORD* wData, WORD* wSts)

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
wMid	[IN] モジュール ID[0～63]		
wCh	[IN] チャンネル番号[0:CH(チャンネル)1 / 1:CH2 / 2:CH3 / 3:CH4] ※B モジュールは CH1 のみ. B 以外は CH1～CH4		
wData	[OUT] アナログ出力データ[0～4095／0～10V]		
wSts	[OUT] ステータス情報		
	ビット	名　称	説　　明
	11-0	(DATA)	0 固定
	12	12/8	読み出したデータのデータ長（0:8Bit / 1:12Bit）
	13	(D/S)	1 固定
	14	OOR	各チャンネルのいずれかのコンパレータが範囲外を検出した場合に 1
	15	ACK	コマンドを受け付ける毎に反転. 実行結果の成否には関係ありません.

VC++
記述例
	short ret; //関数の戻り値
	WORD wdat;
	WORD st;
	//ライン 2,モジュール 7,チャンネル 3 のアナログ出力値を読出し
	ret = mx500_rAmodAout(hDevID, 1, 7, 2, &wdat, &st);

3.4.45 mx500_wAmodAout アナログモジュール アナログ出力データ書込み

No.	D44
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュール，指定チャンネルへアナログモジュールのアナログ出力データを書き込みます.
対　象	A/B/HMS-A モジュール (G9002 搭載モジュールの一部)

言語	書　式
VC++	short mx500_wAmodAout(DWORD hDevID, WORD wLine, WORD wMid, WORD wCh, WORD wData, WORD* wSts)

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
wMid	[IN] モジュール ID[0～63]		
wCh	[IN] チャンネル番号[0:CH(チャンネル)1 / 1:CH2 / 2:CH3 / 3:CH4] ※B モジュールは CH1 のみ. B 以外は CH1～CH4		
wData	[IN] アナログ出力データ[0～4095／0～10V]		
wSts	[OUT] ステータス情報		
	ビット	名　称	説　　明
	7-0	CHx-L/Hbit	コンパレータ状態(0:コンパレータ条件不成立，1:CHx-Low または High コンパレータが条件成立中). ラッチクリアされるまで状態を保持します.
	11-8	DONExx	アナログ出力完了. (0:アナログ出力中，1:補間出力または連続出力動作完了) 次回 CHx D/A に出力するまで保持します.
	12	ERR	不正なコマンドまたはパラメータによってエラー(1 でエラー発生). 次のコマンド発行までエラーを保持します.
	13	D/S	0 固定
	14	BUSY	各チャンネルのいずれかが D/A 出力中の場合に 1
	15	ACK	コマンドを受け付ける毎に反転. 実行結果の成否には関係ありません.

VC++
記述例
	short ret; //関数の戻り値
	WORD st;
	//ライン 2,モジュール 7,チャンネル 2 へ 5V 出力
	ret = mx500_wAmodAout(hDevID, 1, 7, 2, 0x800, &st);

3.4.46 mx500_wAmodCmd アナログモジュール 制御コマンド書込み

No.	D46
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュール，指定チャンネルへアナログモジュールの制御コマンドを書込みます.
対　象	A/B/HMS-A モジュール (G9002 搭載モジュールの一部)

言語	書　式
VC++	short mx500_wAmodCmd(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmdNo, WORD wData, WORD* wSts)

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
wMid	[IN] モジュール ID[0～63]		
wCmdNo	[IN] 制御コマンド番号 (0:プリセット手動切り替え / 1:コンパレータラッチクリア，NOP / 2:D/A 0V 出力，連続切替実行中止) ■0：プリセット手動切り替え		
	ビット	名　称	説　　明
	3-0	CH1PNO	チャンネル CH1 プリセット番号[0,1～15] ※0 は切り替え無効
	7-4	CH2PNO	チャンネル CH2 プリセット番号[0,1～15] ※0 は切り替え無効
	9,8	CH3PNO	チャンネル CH3 プリセット番号[0,1～3] ※0 は切り替え無効
	11,10	CH4PNO	チャンネル CH4 プリセット番号[0,1～3] ※0 は切り替え無効
	■1：コンパレータラッチクリア，NOP　※NOP で使用する場合は下記各ビットを 0 で実行		
	ビット	名　称	説　　明
	0	CH1L	チャンネル CH1 下限値ラッチクリア
	1	CH1H	チャンネル CH1 上限値ラッチクリア
	2	CH2L	チャンネル CH2 下限値ラッチクリア
	3	CH2H	チャンネル CH2 上限値ラッチクリア
	4	CH3L	チャンネル CH3 下限値ラッチクリア
	5	CH3H	チャンネル CH3 上限値ラッチクリア
	6	CH4L	チャンネル CH4 下限値ラッチクリア
	7	CH4H	チャンネル CH4 上限値ラッチクリア
	■2：D/A 0V 出力，連続切替実行中止		
	0	CH1ABRT	CH1 連続出力中止
	1	CH2ABRT	CH21 連続出力中止
	2	CH3ABRT	CH3 連続出力中止
	3	CH4ABRT	CH4 連続出力中止
	8	CH1-0V	CH1 を連続出力中止&0V 出力
	9	CH2-0V	CH2 を連続出力中止&0V 出力
	10	CH3-0V	CH3 を連続出力中止&0V 出力
	11	CH4-0V	CH4 を連続出力中止&0V 出力

wData	[IN] 書込みデータ		
wSts	[OUT] ステータス情報		
	ビット	名　称	説　　明
	7-0	CHx-L/Hbit	コンパレータ状態(0:コンパレータ条件不成立, 1:CHx-Low または High コンパレータが条件成立中). ラッチクリアされるまで状態を保持します.
	11-8	DONExx	アナログ出力完了. (0:アナログ出力中, 1:補間出力または連続出力動作完了) 次回 CHx D/A に出力するまで保持します.
	12	ERR	不正なコマンドまたはパラメータによってエラー(1 でエラー発生). 次のコマンド発行までエラーを保持します.
	13	D/S	0 固定
	14	BUSY	各チャンネルのいずれかが D/A 出力中の場合に 1
	15	ACK	コマンドを受け付ける毎に反転. 実行結果の成否には関係ありません.

VC++
記述例
	short ret; //関数の戻り値
	WORD st;
	//ライン 2,モジュール 7,チャンネル 3 と 4 の連続出力を中止
	ret = mx500_wAmodCmd(hDevID, 1, 7, 2, 0x000c, &st);

3.4.47 mx500_rAmodReg アナログモジュール レジスタ読出し

No.	D48
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュール，指定チャンネルのアナログモジュールレジスタからデータを読出します.
対　象	A/B/HMS-A モジュール (G9002 搭載モジュールの一部)

言語	書　式
VC++	short mx500_rAmodReg(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmd, WORD* wData, WORD* wSts)

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
wMid	[IN] モジュール ID[0～63]		
wCmd	[IN] レジスタ読出しコマンド		
	3000h:システム動作条件設定,		
	3100h:全コンパレータ集約動作設定,		
	320xh:各チャンネルコンパレータ動作設定(x=0:設定 1,1:設定 2)		
	33xyh:プリセットデータイベント切り替え設定(x=チャンネル番号 0-3,y=0:コンパレータ,1:DI)		
	34xxh, 35xxh, 36xxh:未定義		
	37xxh:予約(使用不可)		
	38xyh:プリセットデータ D/A 値登録 (x=チャンネル番号 0-3,y=プリセットデータ番号 1~15)		
	39xyh:A/D コンパレータ基準値登録(x=チャンネル番号 0-3,y=0:LOW 設定,1:High 設定)		
	3Axyh:プリセットデータ到達時間(x=チャンネル番号 0~3,y=プリセットデータ番号 1~15)		
	3Bxyh:プリセットデータ保持時間(x=チャンネル番号 0~3,y=プリセットデータ番号 1~15)		
	3Cx0h:プリセットデータ連続出力条件設定(x=チャンネル番号 0~3)		
	3Dx0h:D/A オフセット値登録(x=チャンネル番号 0~3)		
	3Ex0h:A/D オフセット値登録(x=チャンネル番号 0~3)		
	3Fxxh:未定義		
wData	[OUT] レジスタデータ		
wSts	[OUT] ステータス情報		
	ビット	名　称	説　　明
	11-0	(DATA)	0 固定
	12	12/8	読み出したデータのデータ長（0:8Bit / 1:12Bit）
	13	(D/S)	1 固定
	14	OOR	各チャンネルのいずれかのコンパレータが範囲外を検出した場合に 1
	15	ACK	コマンドを受け付ける毎に反転. 実行結果の成否には関係ありません.

VC++ 記述例

```
short   ret;            //関数の戻り値
WORD st;
WORD wdt;
//ライン 2,モジュール 3,チャンネル 2 の A/D コンパレータ基準値 HIGH 側を読出し
ret = mx500_rAmodReg( hDevID, 1, 3,   0x3911, &wdat, &st );
```

3.4.48 mx500_wAmodReg アナログモジュール レジスタ書込み

No.	D47
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュール，指定チャンネルのアナログモジュールレジスタへデータを書込みます.
対　象	A/B/HMS-A モジュール (G9002 搭載モジュールの一部)

言語	書　式
VC++	short mx500_wAmodReg(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmd, WORD wData, WORD* wSts)

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
wMid	[IN] モジュール ID[0～63]		
wCmd	[IN] レジスタ書込みコマンド		
	1000h:システム動作条件設定,		
	1100h:全コンパレータ集約動作設定,		
	120xh:各チャンネルコンパレータ動作設定(x=0:設定 1,1:設定 2)		
	13x0h:プリセットデータイベント切り替え設定(x=チャンネル番号 0~3, y=0:コンパレータ,1:DI)		
	14xxh, 15xxh, 16xxh:未定義		
	18xyh:プリセットデータ D/A 値登録(x=チャンネル番号 0~3,y=プリセットデータ番号 1~15)		
	19xyh:A/D コンパレータ基準値登録(x=チャンネル番号 0~3,y=0:LOW 設定,1:High 設定)		
	1Axyh:プリセットデータ到達時間設定(x=チャンネル番号 0~3,y=プリセットデータ番号 1~15)		
	1Bxyh:プリセットデータ保持時間設定(x=チャンネル番号 0~3,y=プリセットデータ番号 1~15)		
	1Cx0h:D/A 連続切り替え実行条件設定(x=チャンネル番号 0~3)		
	1Dx0h:D/A オフセット値登録(x=チャンネル番号 0~3)		
	1Ex0h:A/D オフセット値登録(x=チャンネル番号 0~3)		
	1Fxxh:未定義		
wData	[IN] レジスタデータ		
wSts	[OUT] ステータス情報		
	ビット	名　称	説　　明
	7-0	CHx-L/Hbit	コンパレータ状態(0:コンパレータ条件不成立，1:CHx-Low または High コンパレータが条件成立中). ラッチクリアされるまで状態を保持します.
	11-8	DONExx	アナログ出力完了. (0:アナログ出力中，1:補間出力または連続出力動作完了) 次回 CHx D/A に出力するまで保持します.
	12	ERR	不正なコマンドまたはパラメータによってエラー(1 でエラー発生). 次のコマンド発行までエラーを保持します.
	13	D/S	0 固定
	14	BUSY	各チャンネルのいずれかが D/A 出力中の場合に 1
	15	ACK	コマンドを受け付ける毎に反転. 実行結果の成否には関係ありません.

VC++
記述例
	short ret; //関数の戻り値
	WORD st;
	//ライン 2,モジュール 3,チャンネル 4 の A/D オフセット値に 0x333 を書込み
	ret = mx500_rAmodReg(hDevID, 1, 3, 0x1e30, &st);

3.4.49 mx500_rAmodStat アナログモジュール ステータス読出し

No.	D49
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュール，指定チャンネルのアナログモジュールレジスタからデータを読出します.
対　象	A/B/HMS-A モジュール (G9002 搭載モジュールの一部)

言語	書　式
VC++	short mx500_rAmodStat(DWORD hDevID, WORD wLine, WORD wMid, WORD* wSts)

引　数	説　明		
hDevID	[IN] デバイスハンドル		
wLine	[IN] ライン番号［0:Line1，1;Line2］		
id	[IN] モジュール ID[0～63]		
wSts	[OUT] ステータス情報		
	ビット	名　称	説　　明
	11-0	(DATA)	0 固定
	12	12/8	読み出したデータのデータ長（0:8Bit / 1:12Bit）
	13	(D/S)	1 固定
	14	OOR	各チャンネルのいずれかのコンパレータが範囲外を検出した場合に 1
	15	ACK	コマンドを受け付ける毎に反転. 実行結果の成否には関係ありません.

VC++
記述例
	short ret; //関数の戻り値
	WORD st;
	//ライン 2,モジュール 7 のアナログモジュールのステータス取得
	ret = mx500_rAmodStat((hDevID, 1, 7, &st);

3.4.50 mx500_rSmodResp シリアルモジュール コマンド応答受信

No.	D50
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのシリアル通信モジュールへ発行したコマンドに対するコマンド応答データ（ヘッダーブロックおよび付随するデータブロック）を読出します．終了時は送信処理中フラグをクリアします．
対　象	S/R モジュール (G9004 搭載モジュール)

言語	書　式
VC++	short mx500_rSmodResp(DWORD hDevID, WORD wLine, WORD wMid, BYTE* byResp)

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
byResp	[OUT] 応答受信ﾃﾞｰﾀ

VC++
記述例
	short ret; //関数の戻り値
	WORD st;
	BYTE Data[21]
	//ライン 2,モジュール 3 からアブソリュートエンコーダデータ 13Byte を Data に受信(ﾍｯﾀﾞﾌﾞﾛｯｸ含め 21Byte)
	ret = mx500_rSmodResp(hDevID, 1, 3, Data);

3.4.51 mx500_wSmodCmd シリアルモジュール コマンド送信

No.	D51
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのシリアル通信モジュールへコマンドデータ（ヘッダーブロックおよび付随するデータブロック）を送信します．
対　象	S/R モジュール (G9004 搭載モジュール)

言語	書　式
VC++	short mx500_wSmodCmd(DWORD hDevID, WORD wLine, WORD wMid, BYTE* byCmd)

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
byCmd	[IN] コマンドデータ

VC++
記述例
	short ret; //関数の戻り値
	WORD st;
	BYTE Data[8]
	//ライン 2,モジュール 3,Data に入っているアブソリュートエンコーダデータ読出しコマンド 8Byte を送信
	ret = mx500_wSmodCmd(hDevID, 1, 3, Data);

3.4.52 mx500_rSmodMessage シリアルモジュール メッセージ受信

No.	D50
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのシリアル通信モジュールからメッセージデータを読出し，送信処理中フラグをクリアします.
対　象	S/R モジュール (G9004 搭載モジュール)

言語	書　式
VC++	short mx500_rSmodMessage(DWORD hDevID, WORD wLine, WORD wMid, WORD wNum, WORD* wRMsg)

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wNum	[IN] メッセージ受信データ数（ﾜｰﾄﾞ単位）
wRMsg	[OUT] 受信メッセージ

VC++
記述例
	hort ret; //関数の戻り値
	WORD st;
	WORD Data[6]
	//ライン 2,モジュール 3 から FIFO データ 11Byte を Data に受信
	Data[11] = 0;
	ret = mx500_rSmodMessage(hDevID, 1, 3, 0x0001, 6, Data);

備考	・実際に受信が必要なデータサイズが奇数バイトの場合でも，受信データを格納するデータは＋1した ワード単位の領域を確保してパラメータに指定します. 付加した最後の 1Byte には不定なデータが入りますので，データ処理時に除外してご使用ください.

3.4.53 mx500_wSmodMessage シリアルモジュール 情報コマンド･メッセージ送信

No.	D53
機　能	デバイスハンドルで指定された motionCAT マスターボードの指定ライン，指定モジュールのシリアル通信モジュールへ情報コマンドおよびメッセージデータを書込みます.
対　象	S/R モジュール (G9004 搭載モジュール)

言語	書　式
VC++	short mx500_wSmodMessage(DWORD hDevID, WORD wLine, WORD wMid, WORD wCmd, WORD wNum, WORD* wSMsg)

引　数	説　明
hDevID	[IN] デバイスハンドル
wLine	[IN] ライン番号［0:Line1，1;Line2］
wMid	[IN] モジュール ID[0～63]
wCmd	[IN] 情報コマンド
	0001:,ﾒｯｾｰｼﾞ送信
	0002:送信処理中フラグ（ステータス1 Bit6）のリセット
	0003:再送要求
wNum	[IN] ﾒｯｾｰｼﾞ送信データ数（ﾜｰﾄﾞ単位）
wSMsg	[IN] 送信ﾒｯｾｰｼﾞ

VC++
記述例

```
short   ret;            //関数の戻り値
WORD st;
WORD Data[8]
//ライン 2,モジュール 3,ﾒｯｾｰｼﾞ送信ｺﾏﾝﾄﾞおよび Data に入っている 15Byte のﾒｯｾｰｼﾞを送信
Data[15]=0;
ret = mx500_wSmodMessage( hDevID, 1, 3,  0x0001, 8, Data );

//ライン 2,モジュール 3,に送信中ﾌﾗｸﾞｸﾘｱｺﾏﾝﾄﾞを送信（ﾒｯｾｰｼﾞなし）
ret = mx500_wSmodMessage( hDevID, 1, 3,  0x0002, 0, NULL );
```

備考

・情報コマンドのみ送信（ﾒｯｾｰｼﾞが無い）する場合は，wSize=0 および wMsg=Null で指定します.
・実際に送信が必要なデータサイズが奇数バイトの場合でも，送信データが格納された領域には＋1したワード単位の領域を確保してパラメータに指定します.
　付加した最後の 1Byte には 0 を書込んでください.

3.4.54 mx500_SetIntCall()　　割込処理の登録

No.	D54
機　能	デバイスハンドルで指定されたmotionCATマスターボードから発生した割込を受ける割込処理の設定および登録を行い割込機能を有効にします．ここでいう割込処理とはデバイスドライバー内で動作している割込みハンドラを指します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_SetIntCall(DWORD hDevID, RTHANDLE *prhSem);

引　数	説　明
hDevID	デバイスハンドル
prhSem	割込を受け取る割込処理タスク（デバイスドライバー内）へのハンドル

VC++ 記述例	short　　ret;　　//関数の戻り値 RTHANDLE　intHndl;　　//割込処理タスクのハンドル ret = mx500_SetIntCall(hDevID, &intHndl);
備　考	割込みイベントの受信は、mx500_WaitInt()で行ないます

3.4.55 mx500_ResetIntCall()　　割込処理の削除

No.	D55
機　能	デバイスハンドルで指定された motionCAT マスターボードの割込み設定の解除および割込処理の削除を行い割込機能を無効にします.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_ResetIntCall(DWORD hDevID);

.0 引　数	説　明
hDevID	デバイスハンドル

VC++ 記述例	short　　ret;　　//関数の戻り値 ret = mx500_ResetIntCall(hDevID);
備　考	割込みイベント受信用のハンドルは削除されます

3.4.56 mx500_WaitInt()　　割込イベント待ち

No.	D56
機　能	割込ハンドラから、割込み発生時の情報取得の完了を受信します。 タイムアウト指定で指定時間内に割込が発生しない場合はエラーを返します
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short mx500_WaitInt(RTHANDLE rhSem, WORD wTout, WORD wFlg, DWORD *pdEvCnt);

引　数	説　明
rhSem	割込みイベント受信用のハンドル
wTout	10mSec 単位のタイムアウト指定（0～FFFEh,　永久待ち:FFFFh, ）
wFlg	1:残りの割込みイベントを削除
pdEvCnt	残りの割込イベント数を取得（割込みが連続して発生した場合）

VC++
記述例
	short　　ret;
	RTHANDLE　intHndl;
	DWORD　　EvtCnt
	//取得した intHndl のハンドルを持つボードからの割込み待ち. タイムアウト 5sec,
	//残り割込イベントはそのまま残す
	ret = mx500_WaitInt(intHndl, 500, 0, &EvtCnt);

3.4.57 mx500_GetIntData　　割込イベント要因の取得

No.	D57
機　能	割込み発生時の各ライン，モジュールの以下情報を読込みます。 ① 各ラインセンターメインステータス ② 各ライン割込みステータス ③ 各ライン RENV0 割込み有効設定の状態 ④ 各ライン全モジュールの入力変化割込み設定 ⑤ 各ライン全モジュールの入力変化割込み発生状況 ⑥ 各ライン全モジュールの通信エラーフラグ発生状況 ⑦ 各ライン全モジュールのポートデータ ※割込み発生の無いラインはすべてのデータを 0 セット ※最大 31 イベントまでの割込情報を保持します.
対　象	HLS-MCT520 に適応している全ての motionCAT マスターボード

開発環境	書　式
VC++	short It500_GetIntData(DWORD hDevID, WORD *rem, INTEVDATA *intEvData);

引　数	説　明
hDevID	デバイスハンドル
VC++ 記述例	short　　ret; WORD　　rem; INTEVDATA　intEvData; ret = It500_GetIntData(hDevID, &rem, &intEvData);

① 割込機能有効化手順

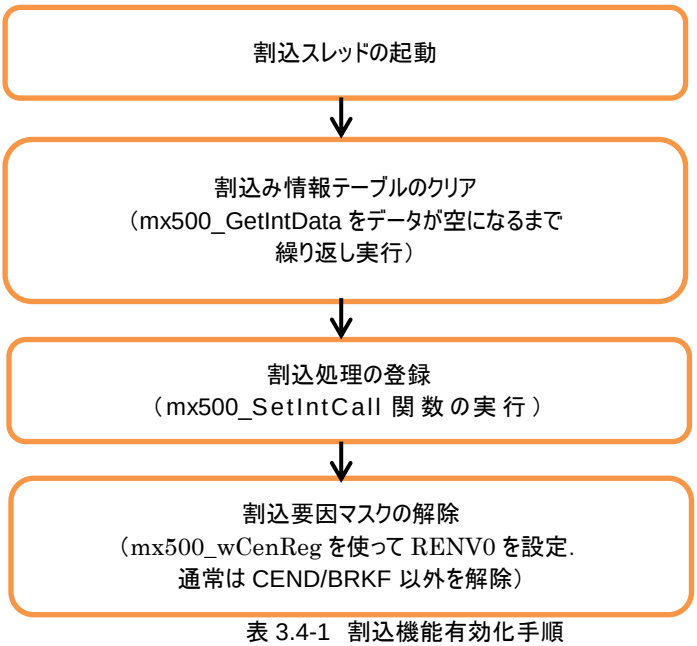


表 3.4-1 割込機能有効化手順

② 割込スレッド内処理手順

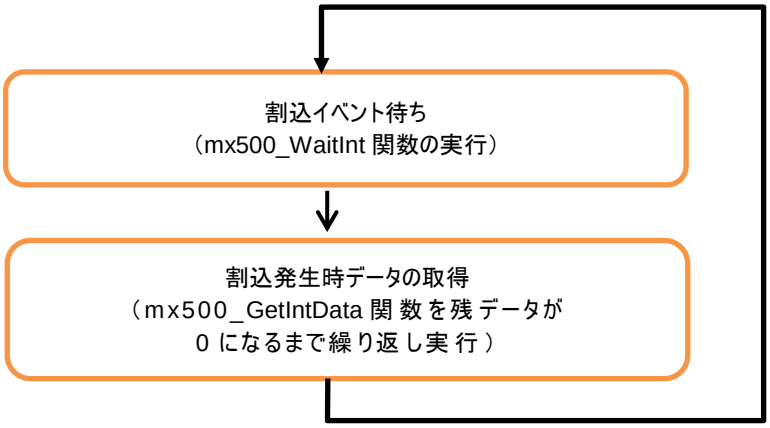


表 3.4-2 割込スレッド内処理手順

③ 割込機能無効化手順

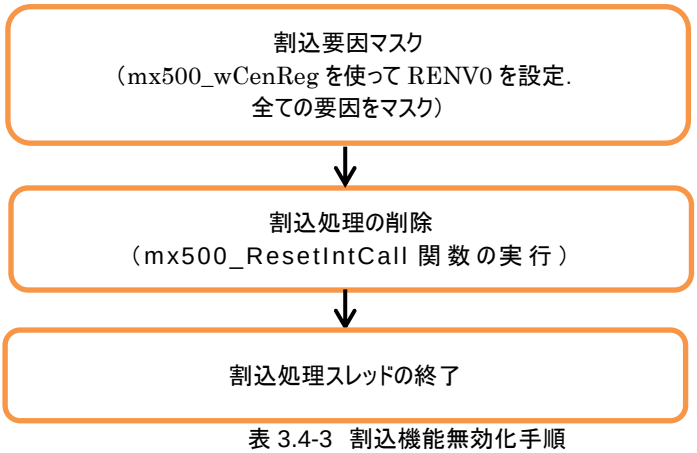


表 3.4-3 割込機能無効化手順

3.4.58 It500_GetDevVerNo　　バージョン番号の取得

No.	D58
機　能	現在PCにインストールされているデバイスドライバと，アプリケーションにリンクしたドライバアクセス関数のバージョン番号を取得します

開発環境	書　式
VC++	short　mx500_GetSftVerNo(DWORD *pdVerNo);

引　数	説　明
pdVerNo	バージョン番号（デバイスドライバ+ドライバアクセス関数）
VC++ 記述例	short　　ret; DWORD　verno; ret = mx500_GetSftVerNo(&verno);
備　考	31　　24 23　　16 15　　8　7　　0 ドライバアクセス関数バージョン番号(x.xx) / デバイスドライババージョン番号(x.xx) メジャーバージョン / マイナー以下バージョン / メジャーバージョン / マイナー以下バージョン ※　ハイバーテックで管理の番号が設定されます。

4. デバイスドライバーアクセス関数

デバイスドライバーアクセス関数は RSL（リアルタイムシェアードライブラリ）ファイル himct520.rsl で提供され，motionnet 通信でデバイスへアクセスするために必要な関数です．
これら関数のパラメータおよび戻り値，呼び出しシーケンス等は前述したアンダースコア以降が同名の関数と同じです．
ただしこのデバイスドライバーアクセス関数をユーザー様が直接使用することは極力お避け下さい．
デバイスドライバー関数内はデバイスドライバーアクセス関数を使用して作成されていますが，複数スレッドやタスクからの同時アクセスを可能にするため，内部では必要な他処理が加えられています．
やむを得ずアプリケーション作成でデバイスドライバーアクセス関数を直接使用する場合は，関数同士の排他やパラメータの管理等に注意してご使用ください．

4.1 関数の種類

No.	関数名	機　能	記載項
MD1	Im500_ApiStartup()	API関数を活性化（初期化処理）	
MD2	Im500_ApiCleanup()	API関数の使用を完了（資源解放）	
MD3	Im500_GetMstBrdCount()	マスターボード枚数の取得	
MD4	Im500_GetDeviceInfo()	マスターボード情報の取得	
MD5	Im500_OpenDevice()	マスターボードオープン	
MD6	Im500_CloseDevice()	マスターボードクローズ	
MD7	Im500_rCenMsts()	センターデバイス メインステータス読出し	
MD8	Im500_wCenCmd()	センターデバイス コマンド書込み	
MD9	Im500_rCenIsts()	センターデバイス 割込みステータス読出し	
MD10	Im500_rCenBuf()	センターデバイス 入力ﾊﾞｯﾌｧ読出し	
MD11	Im500_wCenBuf()	センターデバイス 出力ﾊﾞｯﾌｧ書込み	
MD12	Im500_rCenRFiFo()	センターデバイス 受信用FIFO読出し	
MD13	Im500_wCenSFiFo()	センターデバイス 送信用FIFO書込み	
MD14	Im500_rCenReg()	センターデバイス レジスタ読出し	
MD15	Im500_wCenReg()	センターデバイス レジスタ書込み	
MD16	Im500_rCenPortW()	センターデバイス 指定アドレス2ﾊﾞｲﾄ読出し	
MD17	Im500_wCenPortW()	センターデバイス 指定アドレス2ﾊﾞｲﾄ書込み	
MD18	Im500_rLclCycErr()	サイクリック（I／O）通信エラーフラグ読出し	
MD19	Im500_wLclCycErr()	サイクリック（I／O）通信エラーフラグリセット	
MD20	Im500_rLclInfo()	ローカルデバイス情報読出し	
MD21	Im500_wLclInfo()	ローカルデバイス情報書込み	
MD22	Im500_rLclSetInt()	入力変化割込設定読出し	
MD23	Im500_wLclSetInt()	入力変化割込設定書込み	
MD24	Im500_rLclInt()	入力変化割込フラグ読出し	
MD25	Im500_wLclInt()	入力変化割込フラグリセット	
MD26	Im500_rPortDatB()	I/O ポートデータ 1ﾊﾞｲﾄ読出し	
MD27	Im500_wPortDatB()	I/O ポートデータ 1ﾊﾞｲﾄ書込み	
MD28	Im500_rPortDatW()	I/O ポートデータ 2ﾊﾞｲﾄ読出し	
MD29	Im500_wPortDatW()	I/O ポートデータ 2ﾊﾞｲﾄ書込み(DIO／モーションデバイス)	
MD30	Im500_rOptPortB()	マスターボード オプションポート 1ﾊﾞｲﾄ読出し	
MD31	Im500_wOptPortB()	マスターボード オプションポート 1ﾊﾞｲﾄ書込み	
MD32	Im500_rOptPortW()	マスターボード オプションポート 2ﾊﾞｲﾄ読出し	

MD33	Im500_wOptPortW()	マスターボード オプションポート 2バイト書込み	
MD34	Im500_SetIntCall()	割込処理の登録	
MD35	Im500_ResetIntCall()	割込処理の削除	
MD36	Im500_WaitInt()	割込ｲﾍﾞﾝﾄ待ち	
MD37	Im500_GetIntData()	割込ｲﾍﾞﾝﾄ発生時情報取得	
MD38	Im500_GetBoardCode()	ボードコードの取得（不使用）	
MD39	Im500_GetSftVerNo()	ソフトウェアバージョンの取得	