ＡＬＧＯ省配線システム

リファレンスマニュアル

『INtime 用 Modbus ライブラリ』

目　次

はじめに

第１章　INtime 用 Modbus ライブラリ

第２章　関数仕様

はじめに

この度は、アルゴシステム製品をお買い上げ頂きありがとうございます。
弊社製品を安全かつ正しく使用していただくために、お使いになる前に本書をお読みいただき、十分に理解していただくようお願い申し上げます。

第１章　INtime 用 Modbus ライブラリ

本章では INtime における Modbus について、基本的な仕様,構成について説明します。

１－１　　Modbus とは

Modbus とは、Modicon 社が同社向け PLC に策定したシリアル通信プロトコルになります。
現在では、産業用機器を接続する際の一般的な手段となっている通信方式の 1 つです。
シリアル通信プロトコルである為、弊社製の薄型パネル PC/組み込み型 PC/VESA 取付パネル PC に実装されているシリアルポートを使用します。
機種ごとにシリアルポート数は異なりますが、最大で 4 ポートまでサポートしています。
シリアルポートは、RS-232C 固定ポートと、RS-232C、RS422/485 に切り替え可能なポートが用意されています。

１－２　　基本設定

シリアルポートを使用するには、実行環境（弊社製薄型パネル PC/組み込み型 PC/VESA 取付パネル PC）側の設定が必要になります。
本項では、必要となる 3 種類の設定について説明します。

１－２－１　ハードウェア設定

弊社薄型パネル PC/組み込み型 PC/VESA 取付パネル PC には、シリアルポートの機能設定を行うための DipSW があります。
DipSW の設定は
　　Mode 設定スイッチ
　　SIO ポート設定スイッチ
の 2 つの設定が必要です。

Mode 設定スイッチでは、ご使用になられるポートの通信仕様設定(OFF：RS232C　/　ON：RS-422/485)を変更できます。
SIO ポート設定スイッチでは、全二重/半二重、TX/RX の終端設定を行います。

設定変更の SW は機種により異なる事がありますので、詳細については「薄型パネル PC/組み込み型 PC/VESA 取付パネル PC ユーザーズマニュアル」を参照してください。
*)上記スイッチは弊社産業用パネル PC である AP を例にしています。

１－２－２　ソフトウェア設定

ハードウェア設定にあわせて、Windows ソフトウェアの設定が必要になります。
コントロールパネル内にインストールされている、ASD Config ツールを使用して、シリアルポートの設定を行います。
設定内容は、
　　ポート毎の通信仕様(RS-232C/RS-422/RS-485)
　　RS-485 使用時の送信 Enable 時間
になります。

詳細については、弊社薄型パネル PC/組み込み型 PC/VESA 取付パネル PC に付属のソフトウェアマニュアルを参照してください。

第２章　関数仕様

本章では、Modbus 通信の関数について説明します。
使用する際のライブラリ名称は「MdbsMst.rsl」になります。

２－１　　機能概要

１）Modbus 通信関数

表 2-1-1. Modbus 通信関数一覧

関数名	機　　能
MBM_CommOpen	Modbus 通信をシリアル通信モードでオープンします。
MBM_CommClose	Modbus 通信をクローズします。
MBM_ReadCoilStatus	Modbus スレーブからコイル情報の取得を開始します。
MBM_ReadCoilStatusAns	MBM_ReadCoilStatus 関数の処理完了を確認します。
MBM_ReadInputStatus	Modbus スレーブから引数ステータス情報の取得を開始します。
MBM_ReadInputStatusAns	MBM_ReadInputStatus 関数の処理完了を確認します。
MBM_ReadHoldingRegister	Modbus スレーブから保持レジスタ情報の取得を開始します。
MBM_ReadHoldingRegisterAns	MBM_ReadHoldingRegister 関数の処理完了を確認します。
MBM_ReadInputRegister	Modbus スレーブから引数レジスタ情報の取得を開始します。
MBM_ReadInputRegisterAns	MBM_ReadInputRegister 関数の処理完了を確認します。
MBM_ForceSingleCoil	Modbus スレーブへ１コイルデータの書き込みを開始します。
MBM_ForceSingleCoilAns	MBM_ForceSingleCoil 関数の処理完了を確認します。
MBM_PresetSingleRegister	Modbus スレーブへ１保持データの書き込みを開始します。
MBM_PresetSingleRegisterAns	MBM_PresetSingleRegister 関数の処理完了を確認します。
MBM_FetchCommEventCounter	Modbus スレーブのイベントカウンタ情報を取得します。
MBM_FetchCommEventCounterAns	MBM_FetchCommEventCounter 関数の処理完了を確認します。
MBM_ForceMultipleCoil	Modbus スレーブへ複数のコイル情報の書き込みを開始します。
MBM_ForceMultipleCoilAns	MBM_ForceMultipleCoil 関数の処理完了を確認します。
MBM_PresetMultipleRegister	Modbus スレーブから保持レジスタ情報の取得を開始します。
MBM_PresetMultipleRegisterAns	MBM_PresetMultipleRegister 関数の処理完了を確認します。

２）ライブラリフローチャート

ライブラリを使用する際の関数呼び出しのフローチャートを以下に示します。

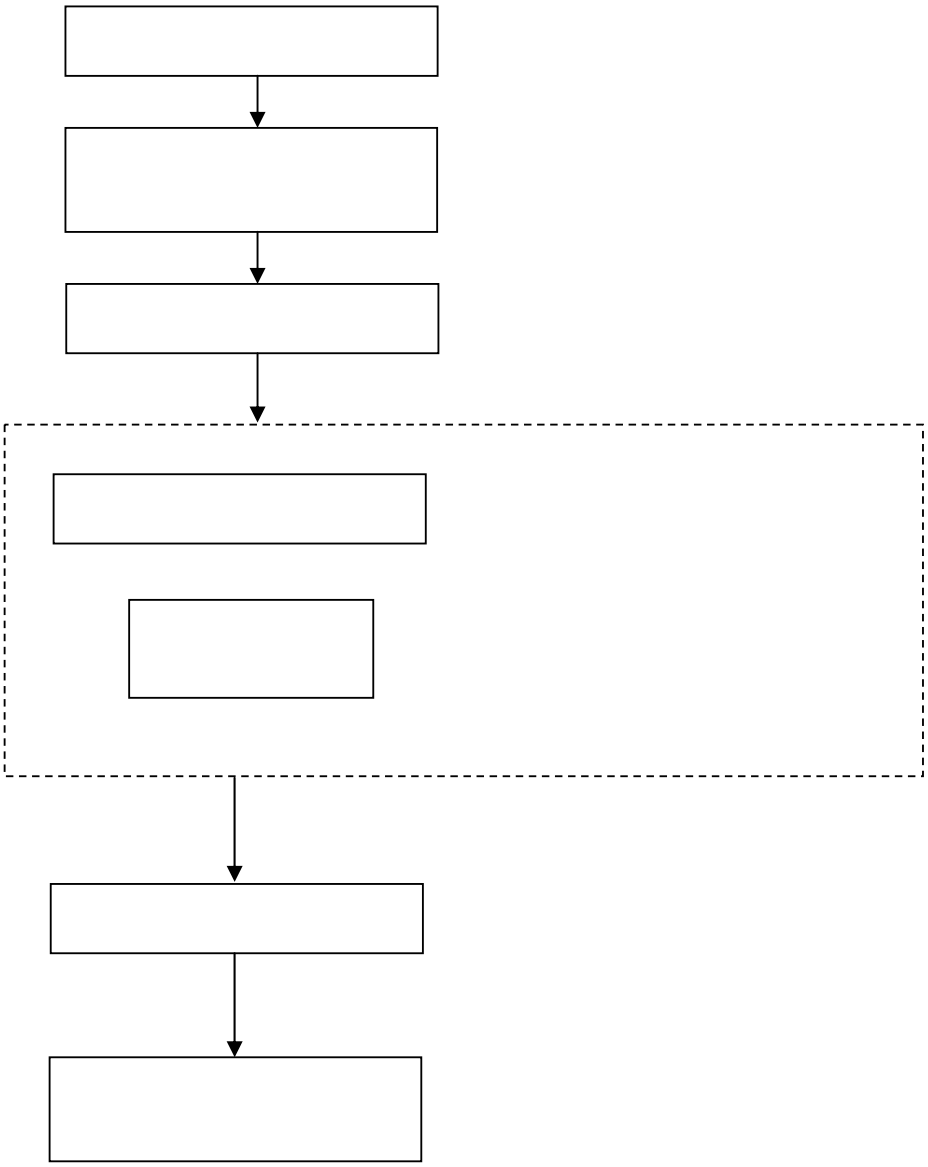

図 2-1-1. ライブラリフローチャート

ライブラリを使用したアプリケーション開始のフローチャートを以下に示します。

APL 開始後、MBM_CommOpen を実行し通信設定・接続を行うことでシリアル通信が可能になります。通信が正常に開始されれば、MBM_Read...などの関数を使用し、データの送受信を行う事が出来ます。
APL 停止時には、MBM_CommClose を実行するようにしてください。

２）FuncID(関数 ID)の使用方法
実行する関数を管理するための ID。例えば MBM_ReadCoilStatus と MBM_ReadCoilStatusAns のような対になる関数では同じ FuncID を使用してください。

①　正常な場合

MBM_ReadCoilStatus(FuncID=1) を実行し、戻り値= MBM_ER_OK で完了
MBM_ReadCoilStatusAns(FuncID=1) を実行し、戻り値=1 で完了
　エラーコード=MBM_ER_OK となっているので
MBM_ReadCoilStatusAns の「コイル情報を格納するポインタ」は正常な結果が読み出せる。

②　異常となる場合(FuncID とファンクションブロックを間違えた場合)

MBM_ReadInputStatus(FuncID=1)を実行し、戻り値= MBM_ER_OK で完了
MBM_ReadCoilStatusAns(FuncID=1) を実行し、戻り値=1 で完了
MBM_ReadCoilStatusAns はエラーとなりエラーコードを確認するとエラー内容が分かる。
　この場合、「MBM_ER_MODBUS_FUNC（エラーコード：0200h)」となります。

③　異常となる場合(Ans 側のファンクションブロックを先に呼び出した場合)

MBM_ReadCoilStatusAns(FuncID=1) を実行し、戻り値=1 で完了
MBM_ReadCoilStatusAns はエラーとなりエラーコードを確認するとエラー内容が分かる。
　この場合、「MBM_ER_NOTINIT（エラーコード：0002h)」となります。

④　補足説明：
正常な場合(複数：呼び出す順番を入れ替えても正常な応答となる)

MBM_ReadCoilStatus(FuncID=1) を実行し、戻り値= MBM_ER_OK で完了
MBM_ReadCoilStatus(FuncID=2) を実行し、戻り値= MBM_ER_OK で完了
MBM_ReadCoilStatusAns(FuncID=1) を実行し、戻り値=1 で完了
　エラーコード=MBM_ER_OK となっているので
MBM_ReadCoilStatusAns の Read は正常な結果が読み出せる。
MBM_ReadCoilStatusAns(FuncID=2) を実行し、戻り値=1 で完了
　エラーコード=MBM_ER_OK となっているので
MBM_ReadCoilStatusAns の Read は正常な結果が読み出せる。

２－２　　Modbus 通信関数

MBM_CommOpen

機能　　Modbus 通信をシリアル通信モードで開始します。

書式

```
int   MBM_CommOpen(
    int   Type,
    int   ComNo,
    int   Baudrate,
    int   DataLen,
    int   Parity,
    int   StopBit,
    int   *pCommNo,
    int   Timeout,
    int   Retry
);
```

引数

int	Type	: Modbus 通信タイプ（0：ASCII モード 1：RTU モード）。
int	ComNo	: シリアルポート番号。
int	Baudrate	: 通信速度。
int	DataLen	: データ長。
int	Parity	: パリティ。
int	StopBit	: ストップビット。
int	pCommNo	: 通信番号取得ポインタ（0 から連番）。
int	Timeout	: タイムアウト時間設定。（単位 msec,0 はタイムアウトしない）
int	Retry	: リトライ回数設定。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_ALREADYINIT	: 既に Open されている。
MBM_ER_OPENOVER	: 同時オープンできる数を超えた。
MBM_ER_COMOPEN	: シリアルポートオープン失敗。

説明　　Modbus 通信をシリアル通信で通信開始します。
通信番号（0 から連番）を pCommNo に格納します。
本関数の正常完了後、Modbus を使った送受信が可能になります。本ライブラリを使用して Modbus 通信を行う前に、必ずコールする必要があります。

シリアル通信設定

Baudrate　：通信速度

通信速度	内容
0	1200 bps
1	2400 bps
2	4800 bps
3	9600 bps
4	19200 bps
5	38400 bps
6	57600 bps
7	115200 bps

DataLen　：データビット長

データビット長	内容
0	7 ビット
1	8 ビット

Parity　：パリティビット

パリティビット	内容
0	NON　：ノンパリティ
1	ODD　：奇数パリティ
2	EVEN　：偶数パリティ

StopBit　：ストップビット長

ストップビット長	内容
0	1 ビット
1	1.5 ビット
2	2 ビット

Timeout　：タイムアウト時間(msec)
　　[0 ～ 65535]　　0:タイムアウトしない

Retry　：リトライ回数
　　[0 ～ 65535]

MBM_CommClose

機能　Modbus 通信を終了します。

書式
```
void MBM_CommClose(
    int CommNo
);
```

引数　int　CommNo　：通信番号。

戻り値　なし

説明　Modbus 通信を切断します。
本 FB をコール後は、MBM_CommOpen 以外の FB の実行は不可になります。
本ライブラリの使用を終了する場合は、必ずコールする必要があります。

MBM_ReadCoilStatus

機能　Modbus スレーブにコイル情報の取得を指令します。

書式

```
int MBM_ReadCoilStatus(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned short Count
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: コイル読み出し開始アドレス。
unsigned short	Count	: 読み出しコイル数（bit 単位）。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブからコイル情報読み出しを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBC_ReadCoilStatusAns 関数に、同じ関数 ID を指定することで確認します。

MBM_ReadCoilStatusAns

機能　MBM_ReadCoilStatus 関数の処理完了を確認します。

書式

```
int MBM_ReadCoilStatusAns(
    int FuncID,
    int *pECode,
    unsigned char *pBValue
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。
unsigned char	pBValue	: コイル情報を格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBC_ReadCoilStatus 関数の処理完了を確認します。
正常終了の場合、pBValue に読み出したデータが格納されます。
読み出しバイト数は読み出しコイル数/8 です。

読出しコイル数の設定についての説明
読出しコイル数が 10 の場合

１バイト	8	7	6	5	4	3	2	1
２バイト	リザーブ						10	9

MBM_ReadInputStatus

機能　Modbus スレーブからインプットステータス情報の取得を開始します。

書式

```
int MBM_ReadInputStatus(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned short Count
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: インプットステータス読み出し開始アドレス。
unsigned short	Count	: 読み出しインプットステータス数(Bit 単位)。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブからインプットステータス情報読み出しを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_ReadInputStatusAns 関数に、同じ関数 ID を指定することで確認します。

MBM_ReadInputStatusAns

機能　MBM_ReadInputStatus 関数の処理完了を確認します。

書式
```
int MBM_ReadInputStatusAns(
    int FuncID,
    int *pECode,
    unsigned char *pBValue
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。
unsigned char	pBValue	: インプットステータスを格納するポインタ。

戻り値

値１	: 処理完了。
値０	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBM_ReadInputStatus 関数の処理完了を確認します。
正常終了の場合は、pBValue に読み出したデータが格納されます。
読み出しバイト数は読み出しインプットステータス数/8 です。

読み出しインプットステータス数の設定についての説明
読み出しインプットステータス数が 10 の場合

１バイト	8	7	6	5	4	3	2	1
２バイト	リザーブ						10	9

MBM_ReadHoldingRegister

機能　Modbus スレーブから保持レジスタ情報の取得を開始します。

書式

```
int MBM_ReadHoldingRegister(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned short Count
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: 保持レジスタ読み出し開始アドレス。
unsigned short	Count	: 読み出し保持レジスタ数（Word 単位）。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブから保持レジスタ読み出しを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_ReadHoldingRegisterAns 関数に、同じ関数 ID を指定することで確認します。

MBM_ReadHoldingRegisterAns

機能　MBM_ ReadHoldingRegister 関数の処理完了を確認します。

書式

```
int MBM_ReadHoldingRegisterAns(
    int FuncID,
    int *pECode,
    unsigned short *pWValue
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。
unsigned short	pWValue	: 保持レジスタ情報を格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBM_ReadHoldingRegister 関数の処理完了を確認します。
正常終了の場合は、pWValue に読み出したデータが格納されます。

MBM_ReadInputRegister

機能　Modbus スレーブから引数レジスタ情報の取得を開始します。

書式

```
int MBM_ReadInputRegister(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned short Count
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: 引数レジスタ読み出し開始アドレス。
unsigned short	Count	: 読み出し引数レジスタ数 (Word 単位)。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブから引数レジスタ読み出しを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_ReadInputRegisterAns 関数に、同じ関数 ID を指定することで確認します。

MBM_ReadInputRegisterAns

機能　MBM_ReadInputRegister 関数の処理完了を確認します。

書式

```
int MBM_ReadInputRegisterAns(
    int FuncID,
    int *pECode,
    unsigned short *pWValue
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。
unsigned short	pWValue	: 引数レジスタ情報を格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBM_ReadInputRegister 関数の処理完了を確認します。
正常終了の場合は、pWValue に読み出したデータが格納されます。

MBM_ForceSingleCoil

機能　Modbus スレーブへ１コイルデータの書き込みを開始します。

書式

```
int MBM_ForceSingleCoil(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned char OnOff
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: コイル書き込みアドレス。
unsigned char	OnOff	: 書き込みデータ （0 : OFF　1 : ON)。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブへコイルデータ 1bit の書き込みを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_ForceSingleCoilAns 関数に、同じ関数 ID を指定することで確認します。

MBM_ForceSingleCoilAns

機能　MBM_ForceSingleCoil 関数の処理完了を確認します。

書式

```
int MBM_ForceSingleCoilAns(
    int FuncID,
    int *pECode
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBM_ForceSingleCoil 関数の処理完了を確認します。

MBM_PresetSingleRegister

機能　Modbus スレーブへ１保持データの書き込みを開始します。

書式

```
int MBM_PresetSingleRegister(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned short WData
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: 保持レジスタ書き込みアドレス。
unsigned short	WData	: 書き込みワードデータ。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブへ保持レジスタ１ワード分の書き込みを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_PresetSingleRegisterAns 関数に、同じ関数 ID を指定することで確認します。

MBM_PresetSingleRegisterAns

機能　MBM_ PresetSingleRegister 関数の処理完了を確認します。

書式

```
int MBM_PresetSingleRegisterAns(
    int FuncID,
    int *pECode
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBM_PresetSingleRegister 関数の処理完了を確認します。

MBM_FetchCommEventCounter

機能　Modbus スレーブのイベントカウンタ情報を取得します。

書式

```
int MBM_FetchCommEventCounter(
    int FuncID,
    int CommNo,
    int NodeNo
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブのイベント情報を取得します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_FetchCommEventCounterAns 関数に、同じ関数 ID を指定することで確認します。

MBM_FetchCommEventCounterAns

機能　MBM_ FetchCommEventCounter 関数の処理完了を確認します。

書式

```
int MBM_FetchCommEventCounterAns(
    int FuncID,
    int *pECode,
    unsigned short *pStatus,
    unsigned short *pEveCnt
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。
unsigned short	pStatus	: ステータス情報を格納するポインタ。
unsigned short	pEveCnt	: イベントカウンタ情報を格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBM_FetchCommEventCounter 関数の処理完了を確認します。
正常終了した場合は、pStatus にステータス情報が、pEveCnt にイベントカウンタ情報が格納されます。

MBM_ForceMultipleCoil

機能　Modbus スレーブへ複数のコイル情報の書き込みを開始します。

書式

```
int MBM_ForceMultipleCoil(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned char *pBData,
    unsigned short Count
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: コイル書き込み開始アドレス。
unsigned char	pBData	: 書き込みデータを格納するポインタ。
unsigned short	Count	: 書き込みコイル数(Bit 単位) 。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブへコイルデータ複数 bit 分の書き込みを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_ForceMultipleCoilAns 関数に、同じ関数 ID を指定することで確認します。

書き込みコイル数の設定についての説明
書き込みコイル数が 10 の場合

１バイト	8	7	6	5	4	3	2	1
２バイト	リザーブ						10	9

MBM_ForceMultipleCoilAns

機能　MBM_ ForceMultipleCoil 関数の処理完了を確認します。

書式

```
int MBM_ForceMultipleCoilAns(
    int FuncID,
    int *pECode
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。

説明　MBM_ForceMltCoil 関数の処理完了を確認します。

MBM_PresetMultipleRegister

機能　Modbus スレーブから保持レジスタ情報の取得を開始します。

書式

```
int MBM_PresetMultipleRegister(
    int FuncID,
    int CommNo,
    int NodeNo,
    unsigned short Address,
    unsigned short *pWData,
    unsigned short Count
);
```

引数

int	FuncID	: 関数 ID。
int	CommNo	: 通信番号。
int	NodeNo	: Modbus ノード番号。
unsigned short	Address	: 保持レジスタ書き込み開始アドレス。
unsigned short	pWData	: 保持レジスタ書き込みワードデータを格納するポインタ。
unsigned short	Count	: 書き込み保持レジスタワード数。

戻り値

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。

説明　Modbus スレーブへ保持レジスタ 1 ワード分の書き込みを開始します。
実行処理を監視するために関数 ID を指定します。
処理完了は MBM_PresetMultipleRegisterAns 関数に、同じ関数 ID を指定することで確認します。

MBM_PresetMultipleRegisterAns

機能　MBM_ PresetMultipleRegister 関数の処理完了を確認します。

書式

```
int MBM_PresetMultipleRegisterAns(
    int FuncID,
    int *pECode
);
```

引数

int	FuncID	: 関数 ID。
int	pECode	: エラーコードを格納するポインタ。

戻り値

値 1	: 処理完了。
値 0	: 実行中。

エラーコード

MBM_ER_OK	: 正常。
MBM_ER_NOTINIT	: Open されていない。
MBM_ER_INVALID	: 無効な引数。
MBM_ER_COMSEND	: シリアルポート送信失敗。
MBM_ER_COMRECV	: シリアルポート受信失敗。
MBM_ER_COMSEND_TIMEOUT	: シリアルポート送信タイムアウト。
MBM_ER_COMRECV_TIMEOUT	: シリアルポート受信タイムアウト。
MBM_ER_COMRECV_FORMAT	: シリアルポート受信データフォーマット異常。
MBM_ER_MODBUS_FUNC	: Modbus ファンクションコードエラー。
MBM_ER_MODBUS_ADDR	: Modbus アドレスエラー。
MBM_ER_MODBUS_DATA	: Modbus データエラー。 s

説明　MBM_PresetMultipleRegiste 関数の処理完了を確認します。

２－３　　エラーコード

エラーコード一覧

表 2-3-1. エラーコード一覧

定義名	コード	内容
MBM_ER_OK	0000h	正常
MBM_ER_INVALID	0001h	無効な引数
MBM_ER_NOTINIT	0002h	初期化されていない
MBM_ER_ALREADYINIT	0003h	既に初期化されている
MBM_ER_OPENOVER	0004h	同時オープンできる数を超えた
MBM_ER_COMOPEN	0100h	シリアルポートオープン失敗
MBM_ER_COMSEND	0101h	シリアルポート送信失敗
MBM_ER_COMRECV	0102h	シリアルポート受信失敗
MBM_ER_COMSEND_TIMEOUT	0103h	シリアルポート送信タイムアウト
MBM_ER_COMRECV_TIMEOUT	0104h	シリアルポート受信タイムアウト
MBM_ER_COMRECV_FORMAT	0105h	シリアルポート受信データフォーマット異常
MBM_ER_MODBUS_FUNC	0200h	Modbus ファンクションコードエラー
MBM_ER_MODBUS_ADDR	0201h	Modbus アドレスエラー
MBM_ER_MODBUS_DATA	0202h	Modbus データエラー
MBM_ER_FUNC_FUNCID	0300h	無効な FuncID
MBM_ER_FUNC_RUNNING	0301h	実行中

７６ＩＴ１０００３Ａ　　２０１７年　０６月　　初版

株式会社アルゴシステム

本社
〒５８７－００２１　大阪府堺市美原区小平尾６５６番地
ＴＥＬ(０７２)３６２－５０６７
ＦＡＸ(０７２)３６２－４８５６

ホームページ　http://www.algosystem.co.jp