ビルバリデータ、コインメックインターフェース

BVCM-MIF-02

Windows API リファレンスマニュアル

本インターフェースは一般的な直列伝送方式（JVMA 方式）のビルバリデータ、コインメックデバイス（以後、単にデバイスと呼びます）をマイクロコンピュータから制御するためのインターフェースです。接続方式としてＵＳＢあるいはシリアル接続が選択可能です。

特徴として USB 接続の場合であっても、CDC クラスとして認識されるため、多くの OS 上でシリアルデバイスとして認識されます。従って接続方式に関わらず本インターフェースはシリアル通信可能なプログラムであれば言語を問わず使用することが可能です。

本マニュアルでは、Windows ユーザ向けに用意されている専用ＡＰＩについて、機能および呼び出し規則を解説します。

１．交信の種類

本インターフェースを用いてＰＣからデバイスと通信する場合、常に交信の主導権はＰＣ側にあり、デバイスがＰＣからのコマンドに答えることで交信が成立します。

直列伝送方式においての交信形式には次にあげる３種類があり、すべてのコマンドはこのうちの適合する交信形式を用いて発行されます。また、交信中に起こった伝送エラーなどに関して、データの再送を促す取り決めなどもありますが､本インターフェースを利用する場合､ユーザはこれを意識する必要はありません。これらタイミング的に厳しく、複雑な部分はインターフェース上のファームウェアが処理するため、ユーザは次章に述べるＡＰＩを通してコマンドを送り、結果を受け取るだけの処理となります。

① 指令交信

デバイスに対して何か特定の操作をする場合の交信（コマンドのみでデータがない場合）で、成功か失敗の情報が返されます。

② 送信データ交信

デバイスに対してデータを送信する場合の交信で、成功か失敗の情報が返されます。

③ 受信データ交信

デバイスからデータを取得する場合の交信で、コマンドを送信するとデータおよび、成功、失敗の情報が返されます。

２．ＡＰＩ関数

インターフェースには専用ＤＬＬ(bvcm_dll.dll)が添付され、ユーザはダイナミックロードあるいは静的リンクすることによってＡＰＩ関数を使用することができます。ＤＬＬの構成ファイルとして以下の３ファイルが供給されます。

- bvcm_dll.dll　　（ＤＬＬ本体）
- bvcm_dll.h　　（ヘッダーファイル）
- bvcm_dll.lib　　（静的リンク用ライブラリ）

ＡＰＩ関数を用いて複数のプログラムから同一デバイスにアクセスした場合、片方の実行が終了するまでは、もう片方のプログラムに対してはデバイス使用中である旨のステータスが返ります。したがってユーザはこのステータスをハンドルしてプロセス間の同期をとる必要があります。

最初にユーザが行うことはインターフェースをオープンすることです。正常にオープンされれば、以後、そのインターフェースに対して、通信を行うことができるようになります。

２－１．関数リファレンス

２－１－１．BVCM_ScanInterface

実装されているインターフェースをスキャンし、そのポート名を返します。

宣言

BVCM_STATUS BVCM_ScanInterface (LPSTR psComPort)

引数

psComPort　ポート名を受け取る文字配列へのポインタ

戻り値

状況に応じたステータスコードが返ります。(関数ステータスの項を参照)

説明

本関数が正常に終了した場合、BVCM_OK が返り、psComPort で示される変数にはポート名（NULL ターミネート文字列）が格納されます。

２－１－２．BVCM_OpenInterface

インターフェースをオープンし、使用可能な状態にします。

宣言

```
BVCM_STATUS BVCM_OpenInterface
(
LPSTR psComPort,
BVCM_HANDLE *bvcmHandle
)
```

引数

psComPort　ポート名を格納した文字配列へのポインタ
bvcmHandle　インターフェースハンドルへのポインタ

戻り値

状況に応じたステータスコードが返ります。（関数ステータスの項を参照）

説明

オープンすべきインターフェースのポート名を指定します。

bvcmHandle は BVCM_HANDLE 型変数へのポインタで、本関数が正常に終了すると BVCM_OK が返り、bvcmHandle で示されるインターフェースハンドルにハンドル値が代入されています。

２－１－３．BVCM_CloseInterface

インターフェースをクローズします。

宣言

BVCM_STATUS BVCM_CloseInterface (BVCM_HANDLE bvcmHandle)

引数

bvcmHandle　インターフェースハンドル

戻り値

状況に応じたステータスコードが返ります。（関数ステータスの項を参照）

説明

bvcmHandle で指定されたインターフェースをクローズします。

本関数が正常に終了すると BVCM_OK が返ってきます。

２－１－４．BVCM_Standby

デバイスを待機状態にします。

宣言

BVCM_STATUS BVCM_Standby (BVCM_HANDLE bvcmHandle, unsigned int uiTarget)

引数

bvcmHandle　インターフェースハンドル
uiTarget　　ターゲット

戻り値

状況に応じたステータスコードが返ります。(関数ステータスの項を参照)

説明

iTarget はビルバリデータとコインメックのどちらにコマンドを送るか、選択します。uiTarget に BVCM_BILL を設定すればビルバリデータに、BVCM_COIN を設定すればコインメックに対し動作します。

本関数が成功すると通常はＡＣＫコードがステータスとして返ってきます。

２－１－５．BVCM_GetAllData

デバイスから全データを取り出します。

宣言

BVCM_STATUS BVCM_GetAllData (BVCM_HANDLE bvcmHandle, unsigned int uiTarget, unsigned char* pcBuffer, unsigned int uiBufferSize)

引数

bvcmHandle	インターフェースハンドル
uiTarget	ターゲット
pcBuffer	データを受け取るためのバッファへのポインタ
uiBufferSize	バッファサイズ

戻り値

状況に応じたステータスコードが返ります。(関数ステータスの項を参照)

説明

uiTarget はビルバリデータとコインメックのどちらにコマンドを送るか、選択します。uiTarget に BVCM_BILL を設定すればビルバリデータに、BVCM_COIN を設定すればコインメックに対し動作します。

本関数実行後、pcBuffer で示されるバッファにはデバイスから取得したデータが入ります。データフォーマットについては、デバイスからのデータがそのまま入りますので、それぞれの機器メーカの通信仕様書を参照して下さい。

本関数が成功すると通常はＡＣＫコードがステータスとして返ってきます。

２－１－６．BVCM_GetData

デバイスから前回のデータ取得以降、変化のあったデータを取り出します。

宣言

BVCM_STATUS BVCM_GetData (BVCM_HANDLE bvcmHandle, unsigned int uiTarget, unsigned char* pcBuffer, unsigned int uiBufferSize)

引数

bvcmHandle	インターフェースハンドル
uiTarget	ターゲット
pcBuffer	データを受け取るためのバッファへのポインタ
uiBufferSize	バッファサイズ

戻り値

状況に応じたステータスコードが返ります。（関数ステータスの項を参照）

説明

uiTarget はビルバリデータとコインメックのどちらにコマンドを送るか、選択します。uiTarget に BVCM_BILL を設定すればビルバリデータに、BVCM_COIN を設定すればコインメックに対し動作します。

本関数実行後、pcBuffer で示されるバッファにはデバイスから取得したデータが入ります。データフォーマットについては、デバイスからのデータがそのまま入りますので、それぞれの機器メーカの通信仕様書を参照して下さい。

本関数が成功すると通常はＡＣＫコードがステータスとして返ってきます。

２－１－７．BVCM_PutData

デバイスにデータを送信します。

宣言

BVCM_STATUS BVCM_PutData (BVCM_HANDLE bvcmHandle, unsigned int uiTarget, unsigned char* pcBuffer)

引数

bvcmHandle	インターフェースハンドル
uiTarget	ターゲット
pcBuffer	送信データを格納したバッファへのポインタ

戻り値

状況に応じたステータスコードが返ります。(関数ステータスの項を参照)

説明

uiTarget はビルバリデータとコインメックのどちらにコマンドを送るか、選択します。uiTarget に BVCM_BILL を設定すればビルバリデータに、BVCM_COIN を設定すればコインメックに対し動作します。

本関数実行前に、pcBuffer で示されるバッファに正しいデータフォーマットでデータフレームを作成しておく必要があります。データフォーマットについてはデバイスによって異なる場合がありますので、それぞれの機器メーカの通信仕様書を参照して下さい。

本関数が成功すると通常はＡＣＫコードがステータスとして返ってきます。

２－１－８．BVCM_LineReset

インターフェースに接続されているデバイスをリセットします。

宣言

BVCM_STATUS BVCM_LineReset (BVCM_HANDLE bvcmHandle)

引数

bvcmHandle　インターフェースハンドル

戻り値

状況に応じたステータスコードが返ります。（関数ステータスの項を参照）

説明

本関数実行後、すぐに制御は戻ってきますが実際のリセットには数秒間かかります。その間は他のＡＰＩによりアクセスすることはできません。

本関数が正常に終了すると、BVCM_OK が返ってきます。

２－１－9. BVCM_GetVersion

インターフェースのバージョン情報を取得します。(DLL のバージョンではありません)

宣言

BVCM_STATUS BVCM_GetVersion (BVCM_HANDLE bvcmHandle, unsigned char* pcBuffer, unsigned int uiBufferSize)

引数

bvcmHandle　インターフェースハンドル
pcBuffer　　バージョン文字列を受け取るバッファーへのポインタ
uiBufferSize　バッファサイズ

戻り値

状況に応じたステータスコードが返ります。(関数ステータスの項を参照)

説明

インターフェースのバージョン文字列が返ります。

本関数が正常に終了すると、BVCM_OK が返ってきます。

２－２．関数ステータス

ＡＰＩ関数の戻り値である BVCM_STSTUS は以下の通り定義されています。

BVCM_OK

デバイスに対するコマンド発行以外の関数の場合、正常に関数が終了するとこの値が返ってきます。

BVCM_INVALID_HANDLE

インターフェースハンドルが間違っています。

BVCM_IO_ERROR

インターフェースとの通信でエラーが発生しました。

BVCM_INVALID_PARAMETER

引数が間違っています。

BVCM_NO_RESPONSE

デバイスが応答しません。

BVCM_OTHER_ERROR

その他のエラー

BVCM_NAK（ＡＣＫコード）

デバイスと正常に交信できましたが、ＮＡＫ応答が返ってきました。

BVCM_ACK1（ＡＣＫコード）

デバイスと正常に交信でき、ＡＣＫ１応答が返ってきました。

BVCM_ACK2（ＡＣＫコード）

デバイスと正常に交信でき、ＡＣＫ２応答が返ってきました。

BVCM_ACK3（ＡＣＫコード）

デバイスと正常に交信でき、ＡＣＫ３応答が返ってきました。

BVCM_ACK4（ＡＣＫコード）

デバイスと正常に交信でき、ＡＣＫ４応答が返ってきました。

BVCM_BUSY

デバイスは現在、コマンドを受けられません。