

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。

直列伝送方式における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

`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ー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

<code>bvcmHandle</code>	インターフェースハンドル
<code>uiTarget</code>	ターゲット
<code>pcBuffer</code>	データを受け取るためのバッファへのポインタ
<code>uiBufferSize</code>	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ー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ー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 bvcHandle, unsigned char* pcBuffer,  
unsigned int uiBufferSize)
```

### 引数

bvcHandle 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_STSTATUS` 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、ACK 1 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、ACK 2 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、ACK 3 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、ACK 4 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。