汎用Ｉ／Ｏインターフェース

GPIO-MIF-01

本ボードはＵＳＢ、シリアル接続の汎用入出力インターフェースです（以後、単にインターフェースと呼びます。）。ユーザは専用ＤＬＬによるＡＰＩ関数を用いて簡単にデジタル信号の入出力が行えます。

インターフェース１枚につき１６入力、１６出力の合計３２入出力を備えており、複数のインターフェースを併用することにより、より多くの入出力を行うことができます。

１．ＡＰＩ関数

インターフェースには専用ＤＬＬ(bvcm.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。

- gpio_dll.dll　　（ＤＬＬ本体）
- gpio_dll.h　　（ヘッダーファイル）
- gpio_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．GPIO_ScanInterface

実装されているインターフェースをスキャンし、その台数を返します。

宣言

GPIO_STATUS GPIO_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、GPIO_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．GPIO_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

```
GPIO_STATUS GPIO_OpenInterface
(
LPSTR psComPort,
GPIO_HANDLE *gpioHandle
)
```

引数

psComPort	ポート名を格納した文字配列へのポインタ
gpioHandle	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。

gpioHandle は GPIO_HANDLE 型変数へのポインタで、本関数が正常に終了すると GPIO_OK が返り、gpioHandle 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．GPIO_CloseInterface

インターフェースをクローズします。

宣言

GPIO_STATUS GPIO_CloseInterface (GPIO_HANDLE gpioHandle)

引数

gpioHandle　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

gpioHandle 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 GPIO_OK が返ってきます。

２－１－４．GPIO_Input

指定したポートをバイト単位で読み取ります。

宣言

GPIO_STATUS GPIO_Input (GPIO_HANDLE gpioHandle, char cPort, unsigned char* pucData)

引数

gpioHandle	インターフェースハンドル
cPort	ポート名
pucData	入力データを受取る変数へのポインタ

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

cPort は読み取るポート名をアスキー文字１バイトで指定します。(‘A’、‘B’等)

pucData は読み取ったデータを格納する変数へのポインタです。関数が正常終了したとき、ここで指定した変数にデータが格納されています。

２－１－５．GPIO_Output

指定したポートにバイト単位でデータを出力します。

宣言

GPIO_STATUS GPIO_Output (GPIO_HANDLE gpioHandle, char cPort, unsigned char ucData)

引数

gpioHandle	インターフェースハンドル
cPort	ポート名
ucData	出力データ

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

cPort は出力するポート名をアスキー文字１バイトで指定します。（‘C’、‘D’等）

ucData は出力するデータを指定します。

２－１－６．GPIO_BitTest

入力ポートの特定ビットの状態を調べます。

宣言

GPIO_STATUS GPIO_BitTest (GPIO_HANDLE gpioHandle, char cPort, unsigned char ucBit)

引数

gpioHandle	インターフェースハンドル
cPort	ポート名
ucBit	ビット番号

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

cPort は読み取るポート名をアスキー文字１バイトで指定します。（‘A’、‘B’等）

ucBit はビット番号を０～７のバイナリ値で指定します。

２－１－７．GPIO_BitSet

出力ポートの特定ビットをセットします。

宣言

GPIO_STATUS GPIO_BitSet (GPIO_HANDLE gpioHandle, char cPort, unsigned char ucBit)

引数

gpioHandle	インターフェースハンドル
cPort	ポート名
ucBit	ビット番号

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

cPort は出力するポート名をアスキー文字１バイトで指定します。（‘C’、‘D’等）

ucBit はビット番号を０～７のバイナリ値で指定します。

２－１－８．GPIO_BitReset

出力ポートの特定ビットをクリアします。

宣言

GPIO_STATUS GPIO_BitReset (GPIO_HANDLE gpioHandle, char cPort, unsigned char ucBit)

引数

gpioHandle インターフェースハンドル
cPort ポート名
ucBit ビット番号

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

cPort は出力するポート名をアスキー文字１バイトで指定します。（‘C’、‘D’等）

ucBit はビット番号を０～７のバイナリ値で指定します。

２－１－９．GPIO_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

```
GPIO_STATUS GPIO_GetVersion
(
GPIO_HANDLE gpioHandle,
unsigned char* pcBuffer,
unsigned int uiBufferSize
)
```

引数

gpioHandle	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 GPIO_STSTUS は以下の通り定義されています。

GPIO_OK

正常に関数が終了するとこの値が返ってきます。

GPIO_INVALID_HANDLE

インターフェースハンドルが間違っています。

GPIO_COMMAND_ERROR

コマンド形式が間違っています。

GPIO_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。

GPIO_TIMEOUT

通信がタイムアウトしました。

GPIO_BUSY

デバイスは現在、コマンドを受けられません。

GPIO_OTHER_ERROR

その他のエラー

GPIO_H_LEVEL

ビットテストの結果、指定ポートはＨレベルです。

GPIO_Ｌ_LEVEL

ビットテストの結果、指定ポートはＬレベルです。