EPX-64C プログラミングガイド

Rev 1.2

株式会社エルモス

目次

1　はじめに

1.1 概要

パソコンの USB ポートに接続して、デジタル入出力信号を制御する「EPX-64C API 関数」をユーザーアプリケーションから呼び出すことで簡単に **EPX-64C** の I/O 機能、カウンタ機能を制御することができます。下図は全体の構成です。

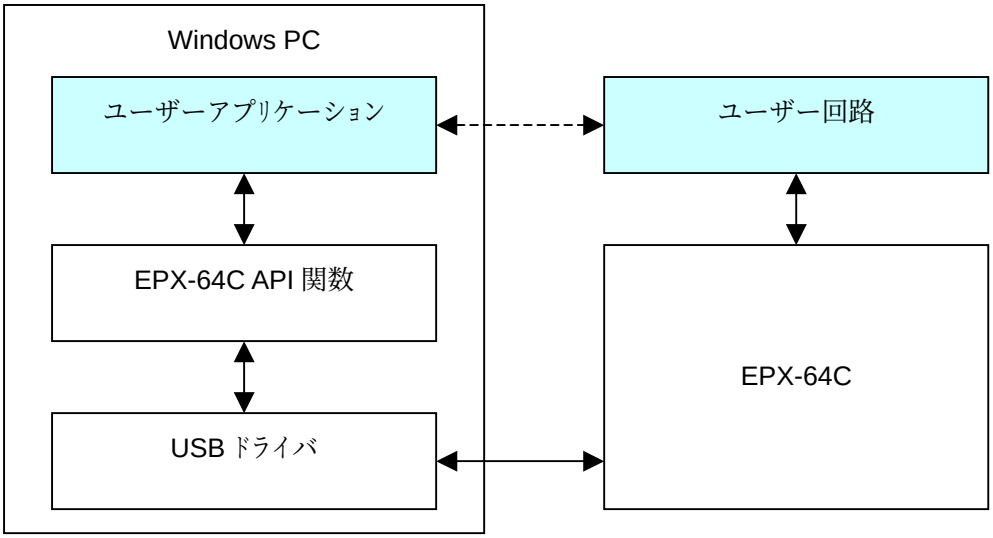

図 1.1 構成

1.2 関数について

「EPX-64C API 関数」は関数群をモジュール化した「EPX64C.dll」で提供されます。
「EPX64C.dll」ファイルは **EPX-64C** をインストールする時にシステムフォルダに入ります。

関数は**デバイス関数**、**I/O 制御関数**、**カウンタ制御関数**に分類されます。
デバイス関数はデバイスの検索、デバイスとの接続、切断等の関数です。
I/O 制御関数は I/O 機能を制御する関数です。
カウンタ制御関数はカウンタ機能を制御する関数です。

1.3 プログラミングの準備

・Visual C++の場合

「EPX64C.h」、「EPX64C.lib」ファイルをプロジェクトに追加してください。

・Visual Basic 6.0 の場合

「EPX64C.bas」ファイルをプロジェクトの標準モジュールに追加してください。

・Visual Basic.NET の場合

「EPX64C.vb」ファイルをプロジェクトに追加してください。

・Visual C#.NET の場合

「EPX64C.cs」ファイルをプロジェクトに追加してください。

※これらのファイルは本製品に付属の CD-ROM「¥library」フォルダにあります。

1.4 関数の戻り値について

関数の戻り値の説明は下表に示します。

定数	値	意味
EPX64C_OK	0	正常終了
EPX64C_INVALID_HANDLE	1	デバイスのハンドルが無効
EPX64C_DEVICE_NOT_FOUND	2	デバイスが見つからない
EPX64C_DEVICE_NOT_OPENED	3	デバイスがオープンできない
EPX64C_OTHER_ERROR	4	その他のエラーが発生した
EPX64C_COMMUNICATION_ERROR	5	通信エラーが発生した
EPX64C_ INVALID _PARAMETER	6	パラメータが無効

表 1.4 関数の戻り値

1.5 注意事項

複数のアプリケーション、またはマルチスレッドによる **EPX-64C** への同時アクセスはできませんので注意してください。

2　関数リファレンス

2.1 デバイス関数

EPX64C_GetNumberOfDevices

現在接続されているデバイスの数を取得します。

int **EPX64C_GetNumberOfDevices** (int ***Number*)

Parameters

Number　　接続デバイス数の格納先へのポインタ

Remarks

デバイスが接続されていない場合は *Number* には 0 が格納されます。

Example

```
int result;
int number;

result = EPX64C_GetNumberOfDevices(&number);
if (result == EPX64C_OK) {
	// EPX64C_GetNumberOfDevices 成功
}
else {
	// EPX64C_GetNumberOfDevices 失敗
}
```

EPX64C_GetSerialNumber

現在接続されているデバイスのシリアル番号を取得します。

Int **EPX64C_GetSerialNumber** (int *Index*, int **SerialNumber*)

Parameters

Index	0 から始まる接続デバイスのインデックス
SerialNumber	シリアル番号の格納先へのポインタ

Remarks

取得したシリアル番号を引数として **EPX64C_OpenBySerialNumber** でオープンすることができます。
尚、このシリアル番号は基板裏面のシリアル番号と同一です。

Example

```
int result;
int index;
int serialNumber;

index = 0;
result = EPX64C_GetSerialNumber(index, &serialNumber);
if (result == EPX64C_OK) {
	// EPX64C_GetSerialNumber 成功
}
else {
	// EPX64C_GetSerialNumber 失敗
}
```

EPX-64C_Open

デバイスをオープンし、デバイスのハンドルを取得します。

Int **EPX64C_Open** (EPX64C_HANDLE **Handle*)

Parameters

Handle　　デバイスのハンドルの格納先へのポインタ

Remarks

デバイスが複数接続されている場合は接続できた最初のデバイスのハンドルを取得します。
取得したハンドルは以後、I/O 制御関数等に引数として渡すことになります。

Example

```
int result;
EPX64C_HANDLE handle;

result = EPX64C_Open(&handle);
if (result == EPX64C_OK) {
	// EPX64C_Open 成功
}
else {
	// EPX64C_Open 失敗
}
```

EPX64C_OpenBySerialNumber

指定したｼﾘｱﾙ番号と一致するデバイスをオープンし、デバイスのハンドルを取得します。

int **EPX64C_OpenBySerialNumber** (int *SerialNumber*, EPX64C_HANDLE **Handle*)

Parameters

SerialNumber　デバイスのｼﾘｱﾙ番号

Handle　デバイスのハンドルの格納先へのポインタ

Remarks

ｼﾘｱﾙ番号は **EPX64C_GetSerialNumber** で取得することができます。

尚、このｼﾘｱﾙ番号は基板裏面のｼﾘｱﾙ番号と同一です。

取得したﾊﾝﾄﾞﾙは以後、I/O 制御関数等に引数として渡すことになります。

Example

```
int result;
EPX64C_HANDLE handle;
int serialNumber;    // 取得したデバイスのｼﾘｱﾙ番号

result = EPX64C_OpenBySerialNumber(serialNumber, &handle);
if (result == EPX64C_OK) {
       // EPX64C_OpenBySerialNumber 成功
}
else {
       // EPX64C_OpenBySerialNumber 失敗
}
```

EPX64C_Close

デバイスをクローズします。

Int **EPX64C_Close** (EPX64C_HANDLE *Handle*)

Parameters

Handle　　デバイスのハンドル

Example

```
int result;
EPX64C_HANDLE handle;       // オープンしたデバイスのハンドル

result = EPX64C_Close(handle);
if (result == EPX64C_OK) {
       // EPX64C_Close 成功
}
else {
       // EPX64C_Close 失敗
}
```

2.2 I/O 制御関数

EPX64C_SetDirection

ポートの入出力方向を設定します。

Int **EPX64C_SetDirection** (EPX64C_HANDLE *Handle*, BYTE *Direction*)

Parameters

Handle　　デバイスのハンドル

Direction　　ポートの入出力方向

Remarks

Direction の各ビットが各ポートの入出力方向に対応しています。
ビットの値が **0** の場合は入力、**1** の場合は出力に設定されます。
初期値（電源投入時）は全て入力（**0x00**）に設定されています。

例）*Direction* = **0xD1**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Direction	1	1	0	1	0	0	0	1
PORT	P7	P6	P5	P4	P3	P2	P1	P0
I/O	OUT	OUT	IN	OUT	IN	IN	IN	OUT

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
BYTE dir;

dir = 0xD1;   // P7,6,4,0 出力  P5,3,2,1 入力
result = EPX64C_SetDirection(handle, dir);
if (result == EPX64C_OK) {
       // EPX64C_SetDirection 成功
}
else {
       // EPX64C_SetDirection 失敗
}
```

EPX64C_GetDirection

現在設定されているポートの入出力方向を取得します。

Int **EPX64C_GetDirection** (EPX64C_HANDLE *Handle*, BYTE **Direction*)

Parameters

Handle　　デバイスのハンドル

Direction　　ポートの入出力方向の格納先へのポインタ

Remarks

Direction の各ビットが各ポートの入出力方向に対応しています。

ビットの値が **0** の場合は入力、**1** の場合は出力に設定されています。

初期値（電源投入時）は全て入力（**0x00**）に設定されています。

例）*Direction* = **0x5A**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Direction	0	1	0	1	1	0	1	0
PORT	P7	P6	P5	P4	P3	P2	P1	P0
I/O	IN	OUT	IN	OUT	OUT	IN	OUT	IN

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
BYTE dir;

result = EPX64C_GetDirection(handle, &dir);
if (result == EPX64C_OK) {
       // EPX64C_GetDirection 成功
}
else {
       // EPX64C_GetDirection 失敗
}
```

EPX64C_OutputPort

ポートに出力します。

Int **EPX64C_OutputPort** (EPX64C_HANDLE *Handle*, BYTE *Port*, BYTE *Value*)

Parameters

Handle	デバイスのハンドル
Port	ポートの指定
Value	ポート出力値

Remarks

ポートが入力に設定されている場合は出力されません。

Port の各ビットが各ポートの指定に対応しています。複数ポートの指定はできません。

	P7	P6	P5	P4	P3	P2	P1	P0
Port	0x80	0x40	0x20	0x10	0x08	0x04	0x02	0x01

Value の各ビットがポートの各ピンに対応しています。

ビットの値が **0** の場合は **LOW** レベル、**1** の場合は **HIGH** レベルになります。

例) *Port* = **0x04**, *Value* = **0xA5**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Value	1	0	1	0	0	1	0	1
PIN	P27	P26	P25	P24	P23	P22	P21	P20
LEVEL	H	L	H	L	L	H	L	H

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
BYTE port, value;

port = 0x04;  // Port2
value = 0xA5; // P27,25,22,20 “HIGH”  P26,24,23,21 “LOW”
result = EPX64C_OutputPort(handle, port, value);
if (result == EPX64C_OK) {
       // EPX64C_OutputPort 成功
}
else {
       // EPX64C_OutputPort 失敗
}
```

EPX64C_InputPort

ポートから入力します。

int **EPX64C_InputPort** (EPX64C_HANDLE *Handle*, BYTE *Port*, BYTE **Value*)

Parameters

Handle　　デバイスのハンドル

Port　　ポートの指定

Value　　ポート入力値の格納先へのポインタ

Remarks

Port の各ビットが各ポートの指定に対応しています。複数ポートの指定はできません。

	P7	P6	P5	P4	P3	P2	P1	P0
Port	0x80	0x40	0x20	0x10	0x08	0x04	0x02	0x01

Value の各ビットがポートの各ピンに対応しています。
ビットの値が **0** の場合は **LOW** レベル、**1** の場合は **HIGH** レベルになります。

例) *Port* = **0x20**, *Value* = **0x3C**

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Value	0	0	1	1	1	1	0	0
PIN	P57	P56	P55	P54	P53	P52	P51	P50
LEVEL	L	L	H	H	H	H	L	L

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
BYTE port, value;

port = 0x20;  // Port5
result = EPX64C_InputPort(handle, port, &value);
if (result == EPX64C_OK) {
       // EPX64C_InputPort 成功
}
else {
       // EPX64C_InputPort 失敗
}
```

2.3 カウンタ制御関数

EPX64C_SetCounterMode

カウンタの動作モードを設定します。

Int **EPX64C_SetCounterMode** (EPX64C_HANDLE *Handle*, int *Mode*)

Parameters

Handle　　デバイスのハンドル

Mode　　カウンタ動作モード

Remarks

Mode	説明
0	単相パルス入力モード　（初期値）
1	2 相パルス入力（位相差パルス入力）モード

単相パルス入力、2 相パルス入力については「**EPX-64C 取扱説明書**」をご参照ください。

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
int mode;

mode = 0;      // 単相パルス入力モード
result = EPX64C_SetCounterMode(handle, mode);
if (result == EPX64C_OK) {
       // EPX64C_SetCounterMode 成功
}
else {
       // EPX64C_SetCounterMode 失敗
}
```

EPX64C_GetCounterMode

カウンタの動作モードを取得します。

Int **EPX64C_GetCounterMode** (EPX64C_HANDLE *Handle*, int **Mode*)

Parameters

Handle	デバイスのハンドル
Mode	カウンタ動作モードの格納先へのポインタ

Example

```
int result;
EPX64C_HANDLE handle;       // オープンしたデバイスのハンドル
int mode;

result = EPX64C_GetCounterMode(handle, &mode);
if (result == EPX64C_OK) {
       // EPX64C_GetCounterMode 成功
}
else {
       // EPX64C_GetCounterMode 失敗
}
```

EPX64C_SetCounterDirection

カウンタのカウント方向を設定します。

Int **EPX64C_SetCounterDirection** (EPX64C_HANDLE *Handle*, int *Direction*)

Parameters

Handle　　デバイスのハンドル

Direction　　カウント方向

Remarks

Direction	説明
0	カウント方向はアップカウント　（初期値）
1	カウント方向はダウンカウント

2 相パルス入力時のカウント方向については「**EPX-64C 取扱説明書**」をご参照ください。

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
int dir;

dir = 0;       // カウント方向はアップカウント
result = EPX64C_SetCounterDirection(handle, dir);
if (result == EPX64C_OK) {
       // EPX64C_SetCounterDirection 成功
}
else {
       // EPX64C_SetCounterDirection 失敗
}
```

EPX64C_GetCounterDirection

カウンタのカウント方向を取得します。

Int **EPX64C_GetCounterDirection** (EPX64C_HANDLE *Handle*, int **Direction*)

Parameters

Handle	デバイスのハンドル
Direction	カウント方向の格納先へのポインタ

Example

```
int result;
EPX64C_HANDLE handle;       // オープンしたデバイスのハンドル
int dir;

result = EPX64C_GetCounterDirection(handle, &dir);
if (result == EPX64C_OK) {
       // EPX64C_GetCounterDirection 成功
}
else {
       // EPX64C_GetCounterDirection 失敗
}
```

EPX64C_SetCounterControl

カウンタの動作を設定します。

int **EPX64C_SetCounterControl** (EPX64C_HANDLE *Handle*, int *Control*)

Parameters

Handle　　デバイスのハンドル

Control　　カウンタ動作

Remarks

Control	説明
0	カウンタのカウント動作は停止　（初期値）
1	カウンタはカウント動作

外部からのカウンタ制御入力信号（$\overline{\text{EX\_CTEN}}$ ）でも制御が可能です。

カウンタ制御入力信号については「**EPX-64C 取扱説明書**」をご参照ください。

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
int control;

control = 1;  // カウント動作
result = EPX64C_SetCounterControl(handle, control);
if (result == EPX64C_OK) {
       // EPX64C_SetCounterControl 成功
}
else {
       // EPX64C_SetCounterControl 失敗
}
```

EPX64C_GetCounterControl

カウンタの動作状態を取得します。

int **EPX64C_GetCounterControl** (EPX64C_HANDLE *Handle*, int **Control*)

Parameters

Handle	デバイスのハンドル
State	カウンタ動作状態の格納先へのポインタ

Example

```
int result;
EPX64C_HANDLE handle;       // オープンしたデバイスのハンドル
int control;

result = EPX64C_GetCounterControl(handle, &control);
if (result == EPX64C_OK) {
       // EPX64C_GetCounterControl 成功
}
else {
       // EPX64C_GetCounterControl 失敗
}
```

EPX64C_ResetCounter

カウンタをリセット(0 クリア) します。

Int **EPX64C_ResetCounter** (EPX64C_HANDLE *Handle*)

Parameters

Handle　　デバイスのハンドル

Example

```
int result;
EPX64C_HANDLE handle;       // オープンしたデバイスのハンドル

result = EPX64C_ResetCounter(handle);
if (result == EPX64C_OK) {
       // EPX64C_ResetCounter 成功
}
else {
       // EPX64C_ResetCounter 失敗
}
```

EPX64C_GetCounterStatus

カウンタの状態を取得します。

int **EPX64C_GetCounterStatus** (EPX64C_HANDLE *Handle*, int **Status*)

Parameters

Handle　　デバイスのハンドル

Status　　カウンタ動作状態の格納先へのポインタ

Remarks

Status	説明
0	カウンタは正常状態
1	カウンタはオーバーフロー（カウンタ停止状態）
2	カウンタはアンダーフロー（カウンタ停止状態）

オーバーフロー（カウンタ値：0x7FFFFFFF→0x80000000）またはアンダーフロー（カウンタ値：0x80000000→0x7FFFFFFF）が発生すると、カウンタは自動的に停止します。
再度、カウンタを動作させるには **EPX64C_ResetCounter** でカウンタをリセットしてください。

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
int status;

result = EPX64C_GetCounterStatus(handle, &status);
if (result == EPX64C_OK) {
       // EPX64C_GetCounterStatus 成功
}
else {
       // EPX64C_GetCounterStatus 失敗
}
```

EPX64C_GetCounterValue

カウンタの値を取得します。

int **EPX64C_GetCounterValue** (EPX64C_HANDLE *Handle*, int **Value*)

Parameters

Handle	デバイスのハンドル
Value	カウンタ値の格納先へのポインタ

Remarks

カウンタ値は符号付き 32 ビット（-2,147,483,648 ~ 2,147,483,647）です。

Example

```
int result;
EPX64C_HANDLE handle;        // オープンしたデバイスのハンドル
int value;

result = EPX64C_GetCounterValue(handle, &value);
if (result == EPX64C_OK) {
        // EPX64C_GetCounterValue 成功
}
else {
        // EPX64C_GetCounterValue 失敗
}
```