

VERSION

1.0

KNPICC30

WinUSB Library Manual .NET Framework 4

Kiko-Net Co., Ltd.

ご注意

本文書の著作権は(有)キコ・ネットが保有します。

本文書の内容を無断で転載することは一切禁止します。

本文書に記載されているサンプルプログラムの著作権は(有)キコ・ネットが保有します。

本文書の内容およびサンプルプログラムに基づき、アプリケーションを運用した結果、万一損害が発生しても、弊社では一切責任を負いませんのでご了承下さい。

本文書の内容については、万全を期して作成いたしました。が、万一ご不審な点、誤りなどお気付きの点がありましたら弊社までご連絡下さい。

本文書の内容は、将来予告なしに変更されることがあります。

商標について

PIC® microcontroller, MPLAB® IDEは、米国およびその他の国々におけるMicrochip Technology Incの登録商標です。

Windows®の正式名称はMicrosoft®Windows®Operating System です。

Microsoft、Windows、Windows NT は、Microsoft Corporation.の米国およびその他の国における商標または登録商標です。

Windows®7、Windows®Vista、Windows®XP、Windows®2000 Professional、

Windows®Millennium Edition、Windows®98 は、Microsoft Corporation の商品名称です。

その他、記載されている会社名、製品名、アプリケーションは各社の登録商標もしくは商標です。

本文書は®、©、™ を記述していない場合があります。

目次

概要.....	4
ファームウェアの配備	5
ライブラリの配備	5
WinUSB ライブラリ	6
KNPIC30 ファームウェアの構造	6
WinUSB 通信	7
KnWinControl クラス	7
コンストラクター	7
プロパティ	7
初期化.....	8
ReadDllVersion メソッド	9
Open メソッド	9
Close メソッド	9
ReadDeviceType メソッド	10
ReadWinDescriptor メソッド.....	10
リセット.....	11
Reset メソッド	11
LED コントロール	12
WriteLED メソッド	13
WriteLEDCompleted イベント	13
ReadLED メソッド.....	14
ReadLEDCompleted イベント	14
SW コントロール.....	15
ReadSW メソッド.....	16
ReadSWCompleted イベント	16
IO コントロール.....	17
WriteIOConfig メソッド.....	18
WriteIOConfigCompleted イベント	18
ReadIO メソッド	19
ReadIOCompleted イベント.....	19

WriteIO メソッド.....	20
WriteIOCompleted イベント	20
Serial コントロール	21
WriteSerialConfig メソッド	23
WriteSerialConfigCompleted イベント	23
OpenSerial メソッド	24
OpenSerialCompleted イベント	24
CloseSerial メソッド	25
CloseSerialCompleted イベント	25
ReadSerial メソッド.....	26
ReadSerialCompleted イベント	26
WriteSerial メソッド	27
WriteSerialCompleted イベント	27
ADC コントロール.....	28
WriteADCConfig メソッド.....	29
WriteADCConfigCompleted イベント.....	29
ReadADC メソッド	30
ReadADCCompleted イベント	30
ReadADCBuffer メソッド.....	31
ReadADCBufferCompleted イベント.....	31
デバイス設定情報一括取得	32
ReadDeviceConfig メソッド	33
ReadDeviceConfigCompleted イベント.....	33
デバイス統計情報取得	34
ReadDeviceStatistics メソッド.....	35
ReadDeviceStatisticsCompleted イベント	35
ClearDeviceStatistics メソッド	36
ClearDeviceStatisticsCompleted イベント	36
共通クラス	37
WinDescriptor クラス	37
コンストラクター	37
プロパティ	37
GUID クラス.....	38
コンストラクター	38
プロパティ	38
EndpointType クラス	38
コンストラクター	38
プロパティ	38
IOConfigArg クラス.....	39

コンストラクター	39
プロパティ	39
SerialPortArg クラス	40
コンストラクター	40
プロパティ	40
ADCConfigArg クラス.....	41
コンストラクター	41
プロパティ	41
DeviceConfig クラス.....	44
コンストラクター	44
プロパティ	44
ComnStatistics クラス	45
コンストラクター	45
プロパティ	45
メソッド.....	46
Clear メソッド.....	46
Add メソッド.....	46
DeviceStatistics クラス	47
コンストラクター	47
プロパティ	47
IOStatistics クラス.....	47
コンストラクター	47
プロパティ	47
SerialStatistics クラス.....	48
コンストラクター	48
プロパティ	48
ADCStatistics クラス	48
コンストラクター	48
プロパティ	48
エラー コード表.....	49
Bit-Map とコネクタの対応	50
IO ポート コネクタ マップ	50
Rs232C コネクタ マップ	51
ADC コネクタ マップ	51

概要

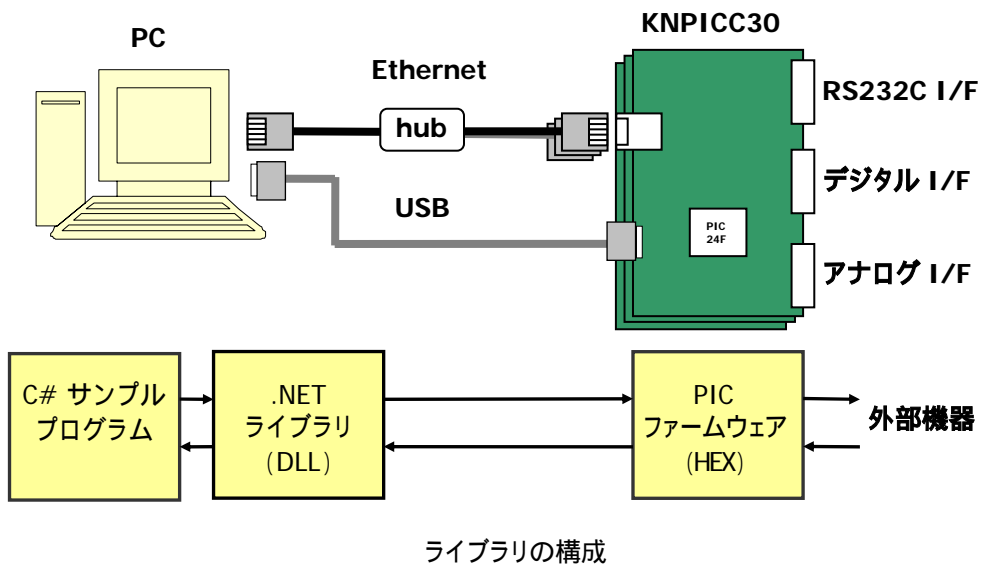
本書は、弊社通信端末CPUボードKNPICC30用WinUSBソフトウェアライブラリについて解説しています。KNPICC30のハードウェア仕様は、『KNPICC30 Hardware Manual』を参照してください。HIDソフトウェアライブラリの仕様は、『KNPICC30 HID Library Manual』を参照してください。UDPソフトウェアライブラリの仕様は、『KNPICC30 UDP Library Manual』を参照してください。

KNPICC30は、マイクロチップ社製ハーバードアーキテクチャ16ビットPIC24F CPUを搭載した通信端末用CPUボードです。10/100BASE-TX対応Ethernetコントローラと、USBコントローラ(Full-Speed対応)を搭載しています。RS232Cラインレシーバ/トランシーバを各2チャンネル搭載し、最大2チャンネルのRS232C通信が可能です。デジタルI/Oポートを16本、AD変換器入力を6チャンネル搭載しています。

本ソフトウェアライブラリは、**KNPICC30**用のPICファームウェアを新たに開発することなく、パソコンから**KNPICC30**の機能(LED, SW, RS232C, デジタルI/Oポート, AD変換器)を使用することを目的としています。

本ソフトウェアライブラリは、マイクロソフト社.NET Framework 4ライブラリ(dll)とマイクロチップ社PIC24Fのファームウェアで構成されています。本ソフトウェアライブラリを利用することで、NET Frameworkアプリケーションを作成するだけで、**KNPICC30**の機能を利用できるようになります。

PIC 通信制御ファームウェアは、マイクロチップ社が提供するTCP/IP Stack、USB Frameworkファームウェアを使用しています。



ファームウェアの配備

KNPICC30 には、工場出荷時に、次のプログラムが事前にインストールされています。

- I. **HID ブートローダー**
UDP ブートローダーのインストール/通信パラメータの設定 (USB)
LED の設定, SW の取得, I/O ポートへのアクセス
- II. **UDP ブートローダー**
通信制御ファームウェアのインストール/通信パラメータの設定 (ネットワーク)
- III. **UDP 通信制御プログラム (ユーザプログラム)**
LED, SW, RS232C, デジタル I/O ポート, AD変換器の制御 (ネットワーク)

WinUSBソフトウェア ライブラリを使用するためには、**UDPブートローダー(II)**の領域に、**WinUSB通信制御プログラム**をインストールする必要があります。**WinUSB通信制御プログラム**のインストール方法については、『**KNPICC30 Firmware Manual**』を参照してください。

ライブラリの配備

KNPICC30 を制御する NET Framework アプリケーションを作成するには、KNPICC30 アクセス クラスライブラリ (**KnApiC30.dll**) をプロジェクトの参照に追加する必要があります。

KNPICC30 を HID 経由で制御する NET Framework アプリケーションを作成するには、HID アクセスライブラリ (**KnHidUsb.dll**) を実行ファイルと同じディレクトリに配置する必要があります。

KNPICC30 を WinUSB 経由で制御する NET Framework アプリケーションを作成するには、WinUSB ドライバをインストールし、WinUSB アクセス ライブラリ (**KnWinUsb.dll**) を実行ファイルと同じディレクトリに配置する必要があります。

WinUSBドライバは、Microsoft Corporationが提供する汎用USBドライバです。WinUSBドライバのインストール方法については、『**KNPICC30 Firmware Manual**』を参照してください。

ご注意

KNPICC30 クラスライブラリ、および サンプル プログラム内で、NET Framework 4 以前のNET Framework ライブラリを使用している場合があります。
サンプルを実行するためには、Visual C# 2010 Express(無償)をインストールしてください。実行に必要なNET Frameworkライブラリがすべてインストールされます。

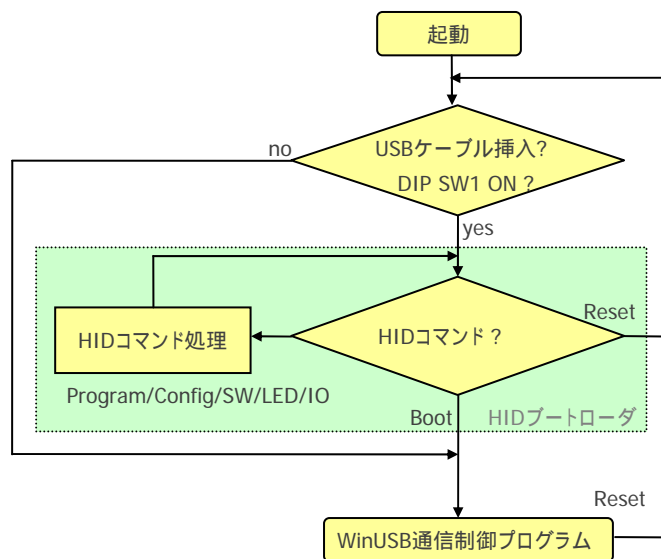
WinUSB ライブラリ

WinUSBライブラリは、WinUSBドライバを使用して、ホスト(パソコン)とデバイス(KNPIC30)間の通信を行います。Microsoft Corporationが提供するWinUSBドライバをインストールする必要があります。WinUSBライブラリは、HIDライブラリと比較してより高速な通信速度が期待できます(パルグ転送)。WinUSBライブラリを使用すると、LEDの設定、SWの取得、およびRS232C、デジタルI/Oポート、AD変換器を制御することが可能です。現行のWinUSBライブラリは、複数のデバイス(KNPIC30)に対応していません。USB上に複数のデバイス(KNPIC30)が検出された場合、最初に見つかったデバイスのみが制御可能です。

(注) 本書では、KNPIC30を制御する装置(主にPC)をホストと記述し、KNPIC30をデバイスと記述しています。

KNPIC30 ファームウェアの構造

WinUSB通信制御プログラムを起動させるためには、HIDブートローダの状態、BootコマンドをKNPIC30に送信する必要があります。また、WinUSB通信制御プログラムが実行している状態で、WinUSB通信制御プログラムを書き換える場合、WinUSB通信制御プログラムに対して、Resetコマンドを送信するか、電源を再投入し、HIDブートローダの状態にしてから、書き換える必要があります。



WinUSB 通信

WinUSB通信は、ホストがデバイスを制御するためにパケットを送信し、その応答を受け取る往復通信です。

KnWinControl クラス

.NET Framework 4

名前空間: KnApis.Win

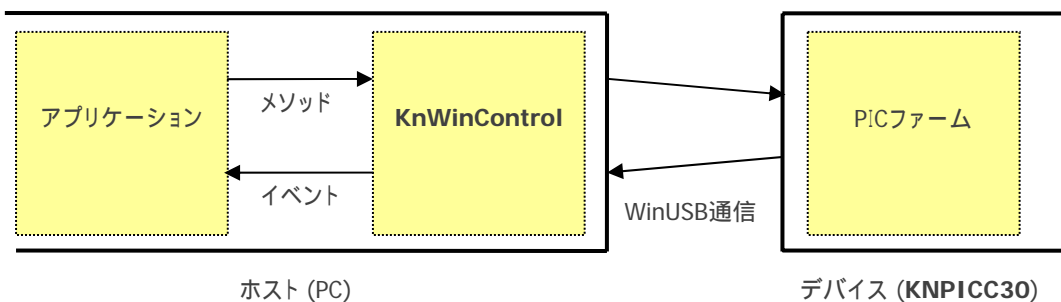
説明: KnWinControlクラスは、WinUSB通信を行うための手段を、メソッドとイベントのペアで提供しています。メソッドでパケットをデバイスに送信し、イベントで、デバイスからの応答結果をアプリケーションに通知します。実行する前に、対応するイベントのイベント ハンドラを登録する必要があります。

ご注意

KnWinControl クラスのメソッドには、同じ名前前で引数の最後に object ID が付くものと、付かないものの2種類のメソッドがあります(オーバーロード)。

例: int ReadSW(IPAddress ip) メソッド(1)
 ReadSW(IPAddress ip, object ID) メソッド(2)
 ReadSWCompleted (int result, int sw1, int sw2, object ID) イベント

メソッド(1)を使用した場合、イベントの ID は nullになります。メソッド(2)を使用した場合、イベントのID はメソッド(2)で指定したIDがそのまま返されます。IDは、プログラム上にメソッドを発行する場所が複数ある場合に、イベント内で、このイベントがどちらのメソッドによるものかを識別するために使用できます。



コンストラクター

名前	説明
KnWinControl	KnWinControl クラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
IsBusy	bool	現在の通信状態を取得します。 true: 通信中 false: 通信可能 (通信停止状態)

初期化

機能: DLLのバージョンを取得します。

メソッド:

名前	説明
ReadDllVersion	DLLのバージョンを取得します。

機能: ドライバをオープンします。

メソッド:

名前	説明
Open	ドライバをオープンします。

機能: ドライバをクローズします。

メソッド:

名前	説明
Close	ドライバをクローズします。

機能: デバイス ファームウェアのタイプ/バージョンを取得します。

メソッド:

名前	説明
ReadDeviceType	ファームウェアのタイプ/バージョンを取得します。

機能: デバイスのエンドポイント情報を取得します。

メソッド:

名前	説明
ReadWinDescriptor	デバイスのエンドポイント情報を取得します。

ReadDllVersion メソッド

解説: DLLのバージョンを取得します。このメソッドに応答イベントはありません。

構文: void ReadDllVersion(ref int major, ref int minor)

パラメーター:

名前	型	説明
major	int	DLLのバージョン (Major)
minor	int	DLLのバージョン (Minor)

戻り値: なし

Open メソッド

解説: デバイスをオープンします。このメソッドに応答イベントはありません。

構文: int Open(int vid, int pid, GUID guid)

パラメーター:

名前	型	説明
vid	int	デバイスのVID
pid	int	デバイスのPID
GUID	guid	Device Interface GUID

戻り値:

型	説明
int	デバイスの数

Close メソッド

解説: ドライバをクローズします。このメソッドに応答イベントはありません。

構文: void Close()

パラメーター: なし

戻り値: なし

ReadDeviceType メソッド

解説: ファームウェアのタイプ/バージョンを取得します。このメソッドに応答イベントはありません。

構文: int ReadDeviceType(ref int major, ref int minor)

パラメーター:

名前	型	説明
major	int	ファームウェアのバージョン (Major)
minor	int	ファームウェアのバージョン (Minor)

戻り値:

型	説明
int	ファームウェアのタイプ 0 : HIDブートローダ 1 : UDPブートローダ 2 : UDPユーザープログラム 3 : HIDユーザープログラム 4 : TCPユーザープログラム 5 : WinUSBユーザープログラム

ReadWinDescriptor メソッド

解説: デバイスのエンドポイント情報を取得します。このメソッドに応答イベントはありません。

構文: WinDescriptor ReadWinDescriptor()

パラメーター: なし

戻り値:

型	説明
WinDescriptor	デバイスのエンドポイント情報

リセット

機能: デバイスをリセットします。

メソッド:

名前	説明
Reset	デバイスをリセットします。

Reset メソッド

解説: デバイスをリセットします。この通信に応答イベントはありません。

構文: int Reset()

パラメーター: なし

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

LED コントロール

機能: デバイスのLEDを設定します。

メソッド:

名前	説明
WriteLED	LED設定コマンドを送信し、応答を待ちます。

イベント:

名前	説明
WriteLEDCompleted	WriteLEDの操作の完了時、キャンセル時に発生します。

機能: デバイスのLEDの状態を取得します。

メソッド:

名前	説明
ReadLED	LED状態取得コマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadLEDCompleted	ReadLEDの操作の完了時、キャンセル時に発生します。

WriteLED メソッド

解説: デバイスのLEDを設定します。結果は、WriteLEDCompletedイベントで取得します。

構文: int WriteLED(int led1, int led2)

int WriteLED(int led1, int led2, object ID)

パラメーター:

名前	型	説明
led1	int	LEDの状態: 0: 消灯 1: 点灯
led2	int	LEDの状態: 0: 消灯 1: 点灯
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

WriteLEDCompleted イベント

解説: LED設定コマンドの通信結果を取得します。WriteLEDメソッドの実行後に発生します。

定義: event KnWinWriteLEDCompletedEventHandler WriteLEDCompleted

デリゲート宣言: delegate void KnWinWriteLEDCompletedEventHandler(int result, object ID)

パラメーター:

名前	型	説明
result	int	0: 書き込み成功 負の値: エラー コード表を参照
ID	object	WriteLEDで指定されたID

ReadLED メソッド

解説: デバイスのLEDの状態を取得します。結果は、ReadLEDCompletedイベントで取得します。

構文: int ReadLED()

int ReadLED(object ID)

パラメーター:

名前	型	説明
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadLEDCompleted イベント

解説: LED状態取得コマンドの通信結果を取得します。ReadLEDメソッドの実行後に発生します。

定義: event KnWinReadLEDCompletedEventHandler ReadLEDCompleted

デリゲート宣言: delegate void KnWinReadLEDCompletedEventHandler(int result, int led1, int led2, object ID)

パラメーター:

名前	型	説明
result	int	0: 取得成功 負の値: エラー コード表を参照
led1	int	LED1の状態: 0: 消灯 1: 点灯
led2	int	LED2の状態: 0: 消灯 1: 点灯
ID	object	ReadLEDで指定されたID

SW コントロール

機能: デバイスのSWの状態を取得します。

メソッド:

名前	説明
ReadSW	SWの状態取得コマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadSWCompleted	ReadSWの操作の完了時、キャンセル時に発生します。

ReadSW メソッド

解説: デバイスのSWの状態を取得します。結果は、ReadSWCompletedイベントで取得します。

構文: int ReadSW()

int ReadSW(object ID)

パラメーター:

名前	型	説明
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadSWCompleted イベント

解説: SW状態取得コマンドの通信結果を取得します。ReadSWメソッドの実行後に発生します。

定義: event KnWinReadSWCompletedEventHandler ReadSWCompleted

デリゲート宣言: delegate void KnWinReadSWCompletedEventHandler(int result, int sw1, int sw2, object ID)

パラメーター:

名前	型	説明
result	int	0: 読み込み成功 負の値: エラー コード表を参照
sw1	int	SW1の状態: 0: OFF 1: ON
sw2	int	SW2の状態: 0: OFF 1: ON
ID	object	ReadSWで指定されたID

IO コントロール

機能: デバイスのIOポートの構成を設定します。

メソッド:

名前	説明
WriteIOConfig	IOポートの構成設定コマンドを送信し、応答を待ちます。

イベント:

名前	説明
WriteIOConfigCompleted	WriteIOConfigの操作の完了時、キャンセル時に発生します。

機能: デバイスのIOポートからデータを読み取ります。

メソッド:

名前	説明
ReadIO	IOポートの読み込みコマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadIOCompleted	ReadIOの操作の完了時、キャンセル時に発生します。

機能: デバイスのIOポートにデータを書き込みます。

メソッド:

名前	説明
WriteIO	IOポートの書き込みコマンドを送信し、応答を待ちます。

イベント:

名前	説明
WriteIOCompleted	WriteIOの操作の完了時、キャンセル時に発生します。

WriteIOConfig メソッド

解説: デバイスに対してIOポートの構成設定コマンドを送信します。通信結果は、WriteIOConfigCompletedイベントで取得します。

構文: int WriteIOConfig(IOConfigArg param)
 int WriteIOConfig(IOConfigArg param, object ID)

パラメーター:

名前	型	説明
param	IOConfigArg	構成設定データ
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ご注意

WinUSBライブラリには イベント通信はありません。IOConfigArgクラスのTrigger、Interval、Eventプロパティは意味をもちません。

WriteIOConfigCompleted イベント

解説: IOポート設定コマンドの通信結果を取得します。WriteIOConfigメソッドの実行後に発生します。

定義: event KnWinWriteIOConfigCompletedEventHandler WriteIOConfigCompleted

デリゲート宣言: delegate void KnWinWriteIOConfigCompletedEventHandler(int result, object ID)

パラメーター:

名前	型	説明
result	int	0: 設定成功 負の値: エラー コード表を参照
ID	object	WriteIOConfigで指定されたID

ReadIO メソッド

解説: デバイスに対してIOポート読み込みコマンドを送信します。結果は、ReadIOCompletedイベントで取得します。

構文: int ReadIO()

int ReadIO(object ID)

パラメーター:

名前	型	説明
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadIOCompleted イベント

解説: IOポート読み込みコマンドの通信結果を取得します。ReadIOメソッドの実行後に発生します。

定義: event KnWinReadIOCompletedEventHandler ReadIOCompleted

デリゲート宣言: delegate void KnWinReadIOCompletedEventHandler(int result, int data, object ID)

パラメーター:

名前	型	説明
result	int	0: 読み込み成功 負の値: エラー コード表を参照
data	int	読み込みデ - タ
ID	object	ReadIOで指定されたID

WriteIO メソッド

解説: デバイスに対してIOポートへの書き込みコマンドを送信します。結果は、WriteIOCompletedイベントで取得します。

構文: int WriteIO(int data)

int WriteIO(int data, object ID)

パラメーター:

名前	型	説明
data	int	書き込みデータ
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラーコード表を参照

WriteIOCompleted イベント

解説: IOポートの書き込みコマンドの通信結果を取得します。WriteIOメソッドの実行後に発生します。

定義: event KnWinWriteIOCompletedEventHandler WriteIOCompleted

デリゲート宣言: delegate void KnWinWriteIOCompletedEventHandler(int result, object ID)

パラメーター:

名前	型	説明
result	int	0: 書き込み成功 負の値: エラーコード表を参照
ID	object	WriteIOで指定されたID

Serial コントロール

機能: デバイスのシリアル ポートの構成を設定します。

メソッド:

名前	説明
WriteSerialConfig	シリアル ポートの構成設定コマンドを送信し、応答を待ちます。

イベント:

名前	説明
WriteSerialConfigCompleted	WriteSerialConfigの操作の完了時、キャンセル時に発生します。

機能: デバイスのシリアル ポートをオープンします。

メソッド:

名前	説明
OpenSerial	シリアル ポートのオープン コマンドを送信し、応答を待ちます。

イベント:

名前	説明
OpenSerialCompleted	OpenSerialの操作の完了時、キャンセル時に発生します。

機能: デバイスのシリアル ポートをクローズします。

メソッド:

名前	説明
CloseSerial	シリアル ポートのクローズ コマンドを送信し、応答を待ちます。

イベント:

名前	説明
CloseSerialCompleted	CloseSerialの操作の完了時、キャンセル時に発生します。

機能: デバイスのシリアル ポートからデータを送信します。

メソッド:

名前	説明
WriteSerial	シリアル ポート送信コマンドを送信し、応答を待ちます。

イベント:

名前	説明
WriteSerialCompleted	WriteSerialの操作の完了時、キャンセル時に発生します。

機能: デバイスのシリアル ポートで受信したデータを取得します。

メソッド:

名前	説明
ReadSerial	シリアル ポート データ取得コマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadSerialCompleted	ReadSerialの操作の完了時、キャンセル時に発生します。

WriteSerialConfig メソッド

解説: シリアル ポートの構成設定コマンドを送信します。通信結果は、WriteSerialConfigCompleted イベントで取得します。

構文: int WriteSerialConfig(int channel)

int WriteSerialConfig(int channel, object ID)

パラメーター:

名前	型	説明
channel	int	構成設定 2: Rs232C 2チャンネル 1: Rs232C 1チャンネル (Hwフロー制御付)
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

WriteSerialConfigCompleted イベント

解説: シリアル ポートの構成設定コマンドの通信結果を取得します。WriteSerialConfigメソッドの実行後に発生します。

定義: event KnWinWriteSerialConfigCompletedEventHandler WriteSerialConfigCompleted

デリゲート宣言: delegate void KnWinWriteSerialConfigCompletedEventHandler(int result, object ID)

パラメーター:

名前	型	説明
result	int	0: 設定成功 負の値: エラー コード表を参照
ID	object	WriteSerialConfigで指定されたID

OpenSerial メソッド

解説: デバイスのシリアル ポートをオープンします。通信結果は、OpenSerialCompletedイベントで取得します。

構文: int OpenSerial(int port, SerialPortArg param)

int OpenSerial(int port, SerialPortArg param, object ID)

パラメーター:

名前	型	説明
port	int	ポート番号 (1 または 2)
param	SerialPortArg	ポート構成データ
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

OpenSerialCompleted イベント

解説: シリアル ポートのオープン コマンドの通信結果を取得します。OpenSerialメソッドの実行後に発生します。

定義: event KnUdpOpenSerialCompletedEventHandler OpenSerialCompleted

デリゲート宣言: delegate void KnUdpOpenSerialCompletedEventHandler(int result, int port, object ID)

パラメーター:

名前	型	説明
result	int	0: オープ成功 負の値: エラー コード表を参照
port	int	ポート番号 (1 または 2)
ID	object	OpenSerialで指定されたID

ご注意

WinUSBライブラリには イベント通信はありません。SerialConfigArgクラスのEvent、Threshold、Timeoutプロパティは意味をもちません。

CloseSerial メソッド

解説: デバイスのシリアル ポートをクローズします。通信結果は、CloseSerialCompletedイベントで取得します。

構文: int CloseSerial(int port)

int CloseSerial(int port, object ID)

パラメーター:

名前	型	説明
port	int	ポート番号 (1 または 2)
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

CloseSerialCompleted イベント

解説: シリアル ポートのクローズ コマンドの通信結果を取得します。CloseSerialメソッドの実行後に発生します。

定義: event KnWinCloseSerialCompletedEventHandler CloseSerialCompleted

デリゲート宣言: delegate void KnWinCloseSerialCompletedEventHandler(int result, int port, object ID)

パラメーター:

名前	型	説明
result	int	0: クローズ成功 負の値: エラー コード表を参照
port	int	ポート番号
ID	object	CloseSerialで指定されたID

ReadSerial メソッド

解説: デバイスのシリアル ポートで受信したデータを取得します。通信結果は、ReadSerialCompleted イベントで取得します。

構文: int ReadSerial(int port)

int ReadSerial(int port, object ID)

パラメーター:

名前	型	説明
port	int	ポート番号 (1 または 2)
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadSerialCompleted イベント

解説: シリアル ポート読み込み コマンドの通信結果を取得します。ReadSerialメソッドの実行後に発生します。

定義: event KnWinReadSerialCompletedEventHandler ReadSerialCompleted

デリゲート宣言: delegate void KnWinReadSerialCompletedEventHandler(int result, int port, int len, byte[] data, object ID)

パラメーター:

名前	型	説明
result	int	0: 受信成功 負の値: エラー コード表を参照
port	int	ポート番号
len	int	取得したバイト数
data	byte[]	取得したデータ (Max 512バイト)
ID	object	ReadSerialで指定されたID

WriteSerial メソッド

解説: デバイスのシリアル ポートからデータを送信します。通信結果は、WriteSerialCompletedイベントで取得します。

構文: int WriteSerial(int port, int len, byte[] data)

int WriteSerial(int port, int len, byte[] data, object ID)

パラメーター:

名前	型	説明
port	int	ポート番号 (1 または 2)
len	int	送信するバイト数 (Max 2048バイト)
data	byte[]	送信データ
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

WriteSerialCompleted イベント

解説: シリアル ポートへの送信 コマンドの通信結果を取得します。WriteSerialメソッドの実行後に発生します。

定義: event KnWinWriteSerialCompletedEventHandler WriteSerialCompleted

デリゲート宣言: delegate void KnWinWriteSerialCompletedEventHandler(int result, int port, int len, object ID)

パラメーター:

名前	型	説明
result	int	0: 送信成功 負の値: エラー コード表を参照
port	int	ポート番号
len	int	送信したバイト数
ID	object	WriteSerialで指定されたID

ご注意

このメソッドは、デバイスがデータを送信完了してからイベントを返します。

ADC コントロール

機能: デバイスのAD変換器の構成を設定します。

メソッド:

名前	説明
WriteADCConfig	AD変換器の構成設定コマンドを送信し、応答を待ちます。

イベント:

名前	説明
WriteADCConfigCompleted	WriteADCConfigの操作の完了時、キャンセル時に発生します。

機能: デバイスのAD変換データをワンショットで取得します。

メソッド:

名前	説明
ReadADC	ADCデータ取得コマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadADCCompleted	ReadADCの操作の完了時、キャンセル時に発生します。

機能: デバイスのAD変換データをAD変換データバッファから取得します。

メソッド:

名前	説明
ReadADCBuffer	ADCデータ取得コマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadADCBufferCompleted	ReadADCBufferの操作の完了時、キャンセル時に発生します。

WriteADCConfig メソッド

解説: デバイスに対してAD変換器の構成設定コマンドを送信します。通信結果は、WriteADCConfigCompletedイベントで取得します。

構文: int WriteADCConfig(ADCConfigArg param)

int WriteADCConfig(ADCConfigArg param, object ID)

パラメーター:

名前	型	説明
param	ADCConfigArg	AD変換器構成設定データ
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

WriteADCConfigCompleted イベント

解説: AD変換器の構成設定コマンドの通信結果を取得します。WriteADCConfigメソッドの実行後に発生します。

定義: event KnWinWriteADCConfigCompletedEventHandler WriteADCConfigCompleted

デリゲート宣言: delegate void KnWinWriteADCConfigCompletedEventHandler(int result, object ID)

パラメーター:

名前	型	説明
result	int	0: 設定成功 負の値: エラー コード表を参照
ID	object	WriteADCConfigで指定されたID

ご注意

WinUSBライブラリには イベント通信はありません。ADCConfigArgクラスのThresholdプロパティは意味をもちません。

ReadADC メソッド

解説: デバイスのAD変換データをワンショットで取得します。通信結果は、ReadADCCompletedイベントで取得します。

構文: int ReadADC(int pchannel, int nchannel)

int ReadADC(int pchannel, int nchannel, object ID)

パラメーター:

名前	型	説明
pchannel	int	正入力チャンネル指定 (0 – 5: AN0 – AN5)
nchannel,	int	負入力チャンネル指定 (0: VR- 1: AN1)
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadADCCompleted イベント

解説: AD変換読み込み コマンドの通信結果を取得します。ReadADCメソッドの実行後に発生します。

定義: event KnWinReadADCCompletedEventHandler ReadADCCompleted

デリゲート宣言: delegate void KnWinReadADCCompletedEventHandler(int result, int data, object ID)

パラメーター:

名前	型	説明
result	int	0: 取得成功 負の値: エラー コード表を参照
data	int	取得したデータ
ID	object	ReadADCCompletedで指定されたID

ReadADCBuffer メソッド

解説: デバイスのAD変換データを取得します。通信結果は、ReadADCBufferCompletedイベントで取得します。

構文: int ReadADC(I)

int ReadADC(object ID)

パラメーター:

名前	型	説明
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadADCBufferCompleted イベント

解説: AD変換読み込み コマンドの通信結果を取得します。ReadADCBufferメソッドの実行後に発生します。

定義: event KnWinReadADCBufferCompletedEventHandler ReadADCBufferCompleted

デリゲート宣言: delegate void KnWinReadADCBufferCompletedEventHandler(int result, int len, ushort[] data, object ID)

パラメーター:

名前	型	説明
result	int	0: 取得成功 負の値: エラー コード表を参照
len	int	取得したデータ数
data	ushort[]	取得したデータ
ID	object	ReadADCCompletedで指定されたID

ADCデータ形式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
チャンネルID					AD変換データ (000 - 3FF)										

デバイス設定情報一括取得

機能: デバイスの設定情報 (LED, SW, RS232C, デジタルI/Oポート, AD変換器の設定状態) を一括取得します。

メソッド:

名前	説明
ReadDeviceConfig	設定情報一括取得コマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadDeviceConfigCompleted	ReadDeviceConfigの操作の完了時、キャンセル時に発生します。

ReadDeviceConfig メソッド

解説: デバイスの設定情報一括取得コマンドを送信します。通信結果は、ReadDeviceConfigCompletedイベントで取得します。

構文: int ReadDeviceConfig()

int ReadDeviceConfig(object ID)

パラメーター:

名前	型	説明
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadDeviceConfigCompleted イベント

解説: 設定情報一括取得コマンドの通信結果を取得します。ReadDeviceConfigメソッドの実行後に発生します。

定義: event KnWinReadDeviceConfigCompletedEventHandler ReadDeviceConfigCompleted

デリゲート宣言: delegate void KnWinReadDeviceConfigCompletedEventHandler(int result, DeviceConfigArg param, object ID)

パラメーター:

名前	型	説明
result	int	0: 取得成功 負の値: エラー コード表を参照
param	DeviceConfigArg	設定情報
ID	object	ReadDeviceConfigで指定されたID

ご注意

WinUSBライブラリには ハートビート通信はありません。DeviceConfigクラスのHeartBeatプロパティは意味をもちません。

デバイス統計情報取得

機能: デバイスの統計情報を取得します。

メソッド:

名前	説明
ReadDeviceStatistics	統計情報取得コマンドを送信し、応答を待ちます。

イベント:

名前	説明
ReadDeviceStatisticsCompleted	ReadDeviceStatisticsの操作の完了時、キャンセル時に発生します。

機能: デバイスの統計情報をクリアします。

メソッド:

名前	説明
ClearDeviceStatistics	統計情報クリアコマンドを送信し、応答を待ちます。

イベント:

名前	説明
ClearDeviceStatisticsCompleted	ClearDeviceStatisticsの操作の完了時、キャンセル時に発生します。

ReadDeviceStatistics メソッド

解説: デバイスの統計情報取得コマンドを送信します。通信結果は、ReadDeviceStatisticsCompleted イベントで取得します。

構文: int ReadDeviceStatistics()

int ReadDeviceStatistics(object ID)

パラメーター:

名前	型	説明
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ReadDeviceStatisticsCompleted イベント

解説: 統計情報取得コマンドの通信結果を取得します。ReadDeviceStatisticsメソッドの実行後に発生します。

定義: event KnWinReadDeviceStatisticsCompletedEventHandler ReadDeviceStatisticsCompleted

デリゲート宣言: delegate void KnWinReadDeviceStatisticsCompletedEventHandler(int result, DeviceStatisticsArg param, object ID)

パラメーター:

名前	型	説明
result	int	0: 取得成功 負の値: エラー コード表を参照
param	DeviceStatistics	デバイス統計情報
ID	object	ReadDeviceConfigで指定されたID

ご注意

WinUSBライブラリでは、DeviceStatisticsクラスのControlプロパティとEventプロパティは意味をもちません。

ClearDeviceStatistics メソッド

解説: デバイスに統計情報クリアコマンドを送信します。通信結果は、ClearDeviceStatisticsCompletedイベントで取得します。

構文: int ClearDeviceStatistics()

int ClearDeviceStatistics(object ID)

パラメーター:

名前	型	説明
ID	object	メソッドの識別子

戻り値:

型	説明
int	0: メソッド受付成功 負の値: エラー コード表を参照

ClearDeviceStatisticsCompleted イベント

解説: 統計情報クリアコマンドの通信結果を取得します。ClearDeviceStatisticsメソッドの実行後に発生します。

定義: event KnWinClearDeviceStatisticsCompletedEventHandler ClearDeviceStatisticsCompleted

デリゲート宣言: delegate void KnWinClearDeviceStatisticsCompletedEventHandler(int result, object ID)

パラメーター:

名前	型	説明
result	int	0: 取得成功 負の値: エラー コード表を参照
ID	object	ClearDeviceStatisticsで指定されたID

共通クラス

この項には、各通信で使用される汎用的なクラスが含まれます。

WinDescriptor クラス

.NET Framework 4

名前空間: KnApis.Common

説明: WinDescriptor クラスは、デバイスのデバイス/エンドポイント情報を表現するクラスです。

コンストラクター

名前	説明
WinDescriptor	WinDescriptor クラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Vid	int	デバイスのVID (ReadOnly)
Pid	int	デバイスのPID (ReadOnly)
DeviceSpeed	int	デバイススピード (ReadOnly) 03 : high-speed 01 : full-speed
EndpointCount	int	デバイスのエンドポイント数 (ReadOnly)
Guid	GUID	ドライバのGUID (ReadOnly)
Endpoint	List<EndpointType>	エンドポイント情報リスト (ReadOnly)

GUID クラス

.NET Framework 4

名前空間: KnApis.Common

説明: GUID クラスは、ドライバのGUID情報を表現するクラスです。

コンストラクター

名前	説明
GUID	GUID クラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
guid1	int	Guid情報1
guid2	int	Guid情報2
Guid3	int	Guid情報3
Guid4	int[8]	Guid情報4

EndpointType クラス

.NET Framework 4

名前空間: KnApis.Common

説明: EndpointType クラスは、デバイスのエンドポイント情報を表現するクラスです。

コンストラクター

名前	説明
EndpointType	EndpointType クラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
PipeType	int	パイプタイプ (ReadOnly)
PipeId	int	パイプID (ReadOnly)
Interval	int	インターバル (ReadOnly)
MaxPktSize	int	最大パケットサイズ (ReadOnly)

IOConfigArg クラス

.NET Framework 4

名前空間: KnApis.Common

説明: IOConfigArgクラスは、デバイス(KNPIC30)のIOポート構成を指定するクラスです。

コンストラクター

名前	説明
IOConfigArg	IOConfigArgクラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Direction	int	IOポート入出力をbit単位で指定します。 0: 出力 1: 入力
OpenDrain	int	IOポート出力形式をbit単位で指定します。 0: トーテン ポール 1: オープン ドレイン (Directionで、出力を指定した場合の意味をもちます)
Trigger	int	イベント通信のトリガーをbit単位で指定します。 0: 状態変化した場合にイベント通信を発生しない 1: 状態変化した場合にイベント通信を発生する
Interval	int	イベント通信 を発生させるためのIOポートのポーリング周期を指定します。(mS: 1 < Interval < 65534)
Event	int	イベント通信の有効化/無効化。 0: イベント通信無効 1: イベント通信有効
Readed	int	IOポートの読み込み値 (ReadOnly)

*IOポートのbitとコネクタの対応は、『Bit-Mapとコネクタの対応』の「IOポート コネクタ マップ」を参照してください。

SerialPortArg クラス

.NET Framework 4

名前空間: KnApis.Common

説明: SerialPortArg クラスは、シリアル ポートの各ポート構成を指定するクラスです。

コンストラクター

名前	説明
SerialPortArg	SerialPortArg クラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
BaudRate	int	ボーレートを指定します。 1: 2.4k 2: 9.6k 3: 19.2k 4: 38.4k 5: 57.6k 6: 115.2k
Parity	int	パリティ/データ長を指定します。 1: 8-bit データ, パリティなし 2: 8-bit データ, パリティ EVEN 3: 8-bit データ, パリティ ODD 4: 9-bit データ, パリティなし
StopBit	int	ストップ ビットを指定します。 1: 1ストップ ビット 2: 2ストップ ビット
Event	int	イベント通信の有効化/無効化。 0: イベント通信無効 1: イベント通信有効
Threshold	int	イベント通信 のトリガーとなる受信バイト数の閾値を指定します(0 < Threshold <= 1024)。デバイスの各シリアル ポートで指定したバイト数以上のデータを受信した場合、EventSerialReceived イベントが発生します。(ここで指定したバイト数以上のデータがイベント通信で送信されます。)
Timeout	int	デバイスのシリアル ポートの受信バッファ内に、Thresholdで指定したバイト数未満のデータが存在している場合に、ここで指定したタイムアウト時間内に新たなデータを受信しない場合、EventSerialReceived イベントが発生します。(mS: Max 65534mS)
IsOpen	int	現在のポートの状態 (ReadOnly) 0: クローズ 1: オープン
Count	int	受信バッファに存在するデータ数 (ReadOnly)

ADCConfigArg クラス

.NET Framework 4

名前空間: KnApis.Common

説明: ADCConfigArgクラスは、デバイス(KNPIC30)のAD変換器の構成を指定するクラスです。

ご注意

AD変換の設定を柔軟にするために、ADCConfigArgクラスは、PIC24Fの内蔵ADCのレジスタを直接設定する構成になっています。ADCConfigArgクラスを設定するためには、PIC24Fの内蔵ADCの機能に関する知識が必要です。内蔵ADCの機能詳細は **PIC24FJ256GB110 Family Data Sheet**を参照してください。

コンストラクター

名前	説明
ADCConfigArg	ConfigADCConfigArg クラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Threshold	int	各イベント通信のトリガーとなる閾値を指定します(0 < Threshold <= 512)。指定した数のAD変換が完了した場合、イベント通信が発生します。 ここで指定したデータ数のデータがイベント通信で送信されます。(1データ = 2バイト)
AD1PCFG	int	AN0 – AN5をAD変換器入力として使用するか、IOポートとして使用するかをBit単位で選択します(bit6以上は意味がありません)。 0 : AD変換器入力 1 : IOポート 例: 0x00 : すべてAD変換器入力 0x01 : AN0のみIOポート、その他はAD変換器入力 0x3F : すべてIOポート 0x3E : AN0のみAD変換器入力、その他はIOポート
IEC0AD1IE	int	AD変換割り込み許可 0 : 割り込み禁止 1 : 割り込み許可 割り込み許可するとイベント通信が発生します。

AD1CSSL	int	AD変換をスキャンする入力 (AN0 – AN5)をBit単位で選択します(bit6以上は意味がありません)。 0 : スキャンしない 1 : スキャンする 例: 0x3F : AN0 – AN5すべてスキャンする 0x01 : AN0のみスキャンする 0x03 : AN0、AN1をスキャンする
AD1CON1SSRC	int	AD変換トリガ要因選択 0x03 : Timer3 0x07 : 内蔵カウンタ
AD1CON1ASAM	int	自動サンプリング選択 0x00 : 手動サンプリング 0x01 : 自動サンプリング
AD1CON2VCFG	int	基準電圧選択 0x00 : Vref+ = AVDD Vref- = AVSS 0x01 : Vref+ = 外部 Vref- = AVSS 0x02 : Vref+ = AVDD Vref- = 外部 0x03 : Vref+ =外部 Vref- = 外部
AD1CON2CSCNA	int	スキャン選択 0x00 : スキャンしない 0x01 : スキャンする
AD1CON2SMPI	int	割り込み周期選択 0x00 : 1 AD変換毎 0x01 : 2 AD変換毎 ... 0x0F : 16 AD変換毎
AD1CON3ADCS	int	AD変換クロック選択 0x00 : 0.5TCY 0x01 : 1.0TCY ... 0xFF : 128TCY TCY = 16MHz
AD1CON3SAMC	int	サンプリング時間選択 0x00 : 0TAD 0x01 : 1TAD ... 0x0F : 31TAD TAD = AD変換クロック

PR3	int	Timer3周期選択 0x0000 – 0xFFFF
T3CONTCKPS	int	Timer3プレスケーラ選択 0x00 : 1/1 0x01 : 1/8 0x02 : 1/64 0x03 : 1/256 Timer3クロック = 16MHz
T3CONTON	int	Timer3動作 0x00 : 停止 0x01 : 動作
Count	int	AD変換バッファに存在するデータ数 (ReadOnly)

DeviceConfig クラス

.NET Framework 4

名前空間: KnApis.Common

説明: DeviceConfigクラスは、デバイス(KNPIC30)の設定情報を一括表示するためのクラスです。すべてプロパティは、読み込み専用です。

コンストラクター

名前	説明
DeviceConfig	DeviceConfigクラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
LED1	int	LED1の設定状態
LED2	int	LED2の設定状態
SW1	int	SW1の設定状態
SW2	int	SW2の設定状態
IOConfig	IOConfigArg	IOポートの設定状態
SerialConfig	int	シリアルポート構成の設定状態
Serial1Port	SerialPortArg	シリアルポート1の設定状態
Serial2Port	SerialPortArg	シリアルポート2の設定状態
ADCCConfig	ADCCConfigArg	AD変換器の設定状態
HeartBeat	int	ハートビート通信の設定状態

ComnStatistics クラス

.NET Framework 4

名前空間: KnApis.Common

説明: ComnStatisticsクラスのプロパティは、「エラー コード表」に対応しています。ComnStatistics クラスは、通信エラー情報を返します。

コンストラクター

名前	説明
ComnStatistics	ComnStatisticsクラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Try	int	試行回数
Success	int	成功回数
Failure	int	失敗回数
Timeout	int	タイムアウト
Cancel	int	キャンセル
Exception	int	例外発生
Busy	int	スレッドがビジー
NotBusy	int	スレッドがビジーでない
NoCallback	int	コールバックが未登録
Parameter	int	パラメータ エラー
Unknown	int	未知のエラー
DeviceNotSupport	int	デバイスがサポートしていない
DeviceDeny	int	デバイスが拒否
DeviceInternal	int	デバイス内部エラー
DeviceErase	int	フラッシュ イレース エラー
DeviceSend	int	送信エラー

Command	int	コマンド エラー
Length	int	レングス エラー
SeqNumber	int	シーケンス番号 エラー
DeviceMode	int	モード エラー
CtrlCode	int	コントロールコード エラー
ErrorCode	int	エラーコード エラー
HeaderSize	int	ヘッダーサイズ エラー
DataSize	int	データサイズ エラー
AccessTime	int	通信が成功した場合の通信時間

メソッド

名前	説明
Clear	すべてのプロパティを 0 にします。
Add	ComnStatistics クラスの各プロパティに数値を加算します。

Clear メソッド

解説: すべてのプロパティを 0 にします。

構文: void Clear()

パラメーター: なし

戻り値: なし

Add メソッド

解説: プロパティに数値を加算します。

構文: void Add(ComnStatistics statistics)

パラメーター:

名前	型	説明
statistics	ComnStatistics	加算するComnStatistics クラス

戻り値: なし

DeviceStatistics クラス

.NET Framework 4

名前空間: KnApis.Common

説明: DeviceStatisticsクラスは、デバイスのエラー情報を返します。

コンストラクター

名前	説明
DeviceStatistics	DeviceStatisticsクラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Control	ComnStatistics	コントロール通信のエラー情報
Event	ComnStatistics	イベント通信のエラー情報
IO	IOStatistics	IOポートのエラー情報
Serial1	SerialStatistics	シリアルポート1のエラー情報
Serial2	SerialStatistics	シリアルポート2のエラー情報
ADC	ADCStatistics	AD変換器のエラー情報

IOStatistics クラス

.NET Framework 4

名前空間: KnApis.Common

説明: IOStatistics クラスは、IOポートのエラー情報を返します。

コンストラクター

名前	説明
IOStatistics	IOStatisticsクラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Update	int	ポート状態変化回数 (イベント通信)
Stall	int	イベント通信FIFOオーバーフロー

SerialStatistics クラス

.NET Framework 4

名前空間: KnApis.Common

説明: SerialStatistics クラスは、シリアルポートのエラー情報を返します。

コンストラクター

名前	説明
SerialStatistics	SerialStatisticsクラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Sent	int	送信バイト数
Received	int	受信バイト数
Parity	int	パリティエラー回数
Framing	int	フレーミングエラー回数
OverFlow	int	バッファオーバーフロー回数
OverRun	int	バッファオーバーラン回数

ADCStatistics クラス

.NET Framework 4

名前空間: KnApis.Common

説明: ADCStatisticsクラスは、AD変換器のエラー情報を返します。

コンストラクター

名前	説明
ADCStatistics	ADCStatisticsクラスの新しいインスタンスを初期化します。

プロパティ

名前	型	説明
Convert	int	変換回数
OverRun	int	バッファオーバーラン回数

エラーコード表

名前空間: KnApis.Common

列挙子宣言 enum ERRORTYPE

列挙子	数値	説明
SUCCESS	0	成功
TIMEOUT	-1	タイムアウト
CANCEL	-2	キャンセル
EXCEPTION	-3	例外発生
BUSY	-4	スレッドがビジー
NOT_BUSY	-5	スレッドがビジーでない
NOCALLBACK	-6	コールバックが未登録
PARAMETER	-7	パラメータ エラー
UNKNOWN	-8	未知のエラー
DEVICE_NOT_SUPPORT	-20	デバイスがサポートしていない
DEVICE_DENY	-21	デバイスが拒否
DEVICE_INTERNAL	-22	デバイス内部エラー
DEVICE_ERASE	-23	フラッシュ イレース エラー
DEVICE_SEND	-24	送信エラー
COMMAND	-30	コマンド エラー
LENGTH	-31	レングス エラー
SEQUENCENUM	-32	シーケンス番号 エラー
DEVICEMODE	-33	モード エラー
CTRLCODE	-34	コントロールコード エラー
ERRORCODE	-35	エラーコード エラー
HEADERSIZE	-36	ヘッダーサイズ エラー
DATASIZE	-37	データサイズ エラー

Bit-Map とコネクタの対応

IOポートコネクタマップ

Bit-Map	コネクタ	ピン番号	CPU ピン番号	CPUピン機能
bit23	CN7	1	24	VREF+/PMA6/CN42/RA10
bit22	CN7	2	23	VREF-/PMA7/CN41/RA9
bit21	CN7	5	20	PGED1/AN0/RP0/CN2/RB0
bit20	CN7	6	19	PGEC1/AN1/RP1/CN3/RB1
bit19	CN7	9	18	AN2/C2INB/VMIO/ RP13 /CN4/RB2
bit18	CN7	10	17	AN3/C2INA/VPIO/CN5/RB3
bit17	CN7	13	16	PGED3/AN4/C1INB/USBOEN/RP28/CN6/RB4
bit16	CN7	14	15	PGEC3/AN5/C1INA/VBUSON/RP18/CN7/RB5
bit15	CN5	1	38	RP5/CN21/RD15
bit14	CN5	2	37	RPI43/CN20/RD14
bit13	CN5	3	65	CN19/RD13
bit12	CN5	4	64	RPI42/CN57/RD12
bit11	CN5	5	56	SCL1/RP3/PMCS2/CN55/RD10
bit10	CN5	6	55	SDA1/DPLN/RP4/CN54/RD9
bit9	CN5	7	54	DMLN/RTCC/RP2/CN53/RD8
bit8	CN5	8	63	RP22/PMBE/CN52/RD3
bit7	CN5	9	62	DPH/RP23/CN51/RD2
bit6	CN5	10	61	VCPCON/RP24/CN50/RD1
bit5	CN5	11	58	DMH/RP11/INT0/CN49/RD0
bit4	CN5	12	53	SDA2/RPI35/CN44/RA15
bit3	CN5	13	52	SCL2/RPI36/CN43/RA14
bit2	CN5	14	73	VCMPST2/CN69/RF1
bit1	CN5	15	74	CN78/RG1
bit0	CN5	16	75	CN77/RG0

*bit23 – bit16 は、ADCを無効化した場合に使用できます。

Rs232C コネクタ マップ

Mode 2 (Rs232Cを2チャンネル使用)

Rs232Cポート	Rs232C機能	コネクタ	ピン番号
1	TXD	CN1	1
1	RXD	CN1	2
2	TXD	CN1	5
2	RXD	CN1	6

Mode 1 (ハードウェア フロー制御Rs232cを2チャンネル使用)

Rs232Cポート	Rs232C機能	コネクタ	ピン番号
1	TXD	CN1	1
1	RXD	CN1	2
1	RTS	CN1	5
1	CTS	CN1	6

ADC コネクタ マップ

ADCチャンネル	ADC機能	コネクタ	ピン番号
0	AN0入力	CN7	5
1	AN1入力	CN7	6
2	AN2入力	CN7	9
3	AN3入力	CN7	10
4	AN4入力	CN7	13
5	AN5入力	CN7	14
-	ADC基準電圧	CN7	1
-	ADC基準電圧	CN7	2