関数リファレンス for CANwacousb.dll

|  |  |  |
| --- | --- | --- |
| Date | Auth | Notice |
| 2023/04/06 | Y.OGAWA | 1st release |
|  |  |  |
|  |  |  |
|  |  |  |

# 目次

[目次 1](#_Toc131665293)

[注意点 2](#_Toc131665294)

[ファイル構成 2](#_Toc131665295)

[関数 3](#_Toc131665296)

[GetInterfaceCount 3](#_Toc131665297)

[SetInterface 3](#_Toc131665298)

[SetBaudrate 4](#_Toc131665299)

# 注意点

・本DLLは、CANabh3.dll と入れ替え利用が可能な、別インターフェース（WacoGiken社製）用となります。

・本書ではCANabh3.dllとの相違点のみ説明されています。基本的な情報は、CANabh3.dll側のプロジェクト

　付属文書を御参照願います。

# ファイル構成

本DLLは、CANabh3と以下の相違点が有ります。

元資料となる、CANabh3側の資料も合わせてご確認下さい。

|  |  |
| --- | --- |
| ファイル名 | 内容 |
| IxxatV2.cpp  IxxatV2.h | 本DLLでは存在しません。  このファイルは、HMS社のUSB-to-CAN V2デバイスを制御する為のファイルです。 |
| WacoCanUsb.cpp  WacoCanUsb.h | このファイルは、WacoGiken社のCANインターフェースを制御する為のファイルです。 |
| Crc.cpp  Crc.h | CCITT-CRCを計算する為のクラスです。  送信用パケット構築、又は受信パケットの検証に必要となります。 |

インターフェースの利用準備

本DLLでは以下のインターフェースのみに対応しています。

|  |  |
| --- | --- |
| メーカー | WacoGiken |
| メーカーURL | https://www.wacogiken.co.jp/ |
| 名称 | CANインターフェース |
| 準備 | インターフェースの入手方法は、要問い合わせとなります。  PCに接続すれば、COMポート扱いで認識します。 |
| 本書更新時の  デバイスドライバ | 不用 |

# 関数

基本的に関数説明は、CANabh3.dllの説明書を御確認下さい。

本DLL専用の関数は有りませんが、一部仕様が異なる関数のみ説明が有ります。

## GetInterfaceCount

|  |  |
| --- | --- |
| 概要 | 使用可能なCANインターフェース数を取得 |
| 詳細 | InitInstanceで指定した「使用したいインターフェース」に対して、  現時点で利用可能な本数(PCに接続されているデバイス数)を取得します。 |
| 構文 | CANABH3API int32\_t GetInterfaceCount() |
| パラメータ | 無し |
| 戻り値 | PCの「COMポート数」が戻ります。 |
| 注意点等 |  |

## SetInterface

|  |  |
| --- | --- |
| 概要 | CAN回線に接続するインターフェースを指定 |
| 詳細 | CAN回線に接続するインターフェースを指定。 |
| 構文 | CANABH3API int32\_t SetInterface(int32\_t nDeviceNum) |
| パラメータ | |  |  | | --- | --- | | 変数名 | 内容 | | nDeviceNum | 開く対象のCOMポート番号を指定して下さい。  ケーブルを接続した時、COM3になるケーブルを選択する  場合は、3を指定します。 | |
| 戻り値 | |  |  | | --- | --- | | 戻り値 | 内容 | | 0 | 正常終了 | | 上記以外 | 異常終了 | |
| 注意点等 | 複数本接続した場合、使われていないCOMポート番号が自動的に割り振られます。  確認するには、Windowsの機能であるデバイスマネージャを御利用下さい。 |

## SetBaudrate

|  |  |
| --- | --- |
| 概要 | CAN回線で使用する通信速度を指定 |
| 詳細 | CAN回線で使用する通信速度を指定します。 |
| 構文 | CANABH3API void SetBaudrate(uint32\_t nBaudrateKbps) |
| パラメータ | |  |  | | --- | --- | | 変数名 | 内容 | | nBaudrateKbps | 通信速度[Kbps]を以下から指定しますが、  実際の通信速度は、CANインターフェース側の  物理的なスイッチで設定する必要が有ります。   |  |  | | --- | --- | | 値 | 通信速度 | | 10 | 10[Kbps] | | 20 | 20[Kbps] | | 50 | 50[Kbps] | | 100 | 100[Kbps] | | 125 | 125[Kbps] | | 250 | 250[Kbps] | | 500 | 500[Kbps] | | 800 | 800[Kbps] | | 1000 | 1000[Kbps] | | |
| 戻り値 | 無し |
| 注意点等 | CAN回線に接続する前に必ず設定する必要が有りますが、実際の通信速度に影響されません。  実際の通信速度は、CANインターフェース側に搭載されているスイッチで指定が必要です。 |

CANインターフェースのプロトコル

構成

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | 開始コード | フラグ | ID | データ | CRC | 終端コード |
| 内容 | STX | フラグ | CAN-ID | データ | CCITT-CRC | ETX |
| 長さ[bytes] | 1 | 2 | 4 | 0 - 8 | 2 | 1 |

解説

|  |  |
| --- | --- |
| 項目名 | STX |
| 内容 | 開始文字となるコード  バイナリで02hとなる | |

|  |  |
| --- | --- |
| 項目名 | フラグ |
| 内容 | 以下の値を16進数2桁のANSI文字列で扱う。  bit7 – bit5 : 常に0  bit4 : 拡張IDフラグ（0..標準ID 1..拡張ID)  bit3 – bit0 : データ長 | |
| 例 | 拡張IDとして8バイトのデータを送信する場合  bit7 – bit5 : 000  bit4 : 1  bit3 – bit0 : 1000  合わせると2進数で 00011000 となり、これを16進数2桁の文字列に変換  最終的な文字は、 18 となる | |

|  |  |
| --- | --- |
| 項目名 | CAN-ID |
| 内容 | CAN-IDを16進数8桁のANSI文字列で扱う。 | |
| 例 | CAN-IDが、0xEF0102 (発信元が02、送信先が01、コードがEFh)の場合  最終的な文字列は、 00EF0102 となる | |

|  |  |
| --- | --- |
| 項目名 | データ |
| 内容 | データ部分を16進数2桁単位のANSI文字列で扱う。  最小はデータ無し（0文字）、最大は8バイト（16文字となる） | |

|  |  |
| --- | --- |
| 項目名 | CCITT-CRC |
| 内容 | フラグ(2文字目)からデータ末尾迄を1バイト単位のバイナリデータとして  扱い、CCITT-CRCで算出して16進数4桁の文字列として扱う。  CCITT-CRC仕様は以下の通り。  長さ（16bit）、コード（0x1021）、値（反転しない）、方向（左回り）  C++で利用する場合は、本プロジェクトに含まれるCRC.cpp/hの利用を推奨。 | |
| 例 | フラグ("18")、CAN-ID("00EF0102")、データ("C309C30900000000")の場合、  算出結果は0xCC4Bとなり、最終的な文字列は CC4B となる。 | |

|  |  |
| --- | --- |
| 項目名 | ETX |
| 内容 | 終端文字となるコード  バイナリで03hとなる | |