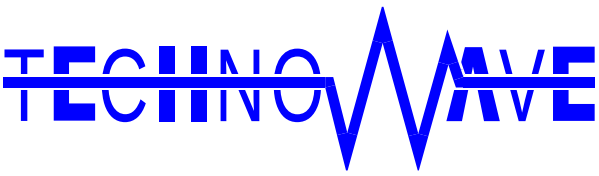
TWXA ライブラリ　関数リファレンス



テクノウェーブ株式会社

目次

1. はじめに

□ 『TWXA ライブラリ』について

『TWXA ライブラリ』は、弊社 I/O ユニット製品の制御用ライブラリです。表 1 は『TWXA ライブラリ』がサポートする製品です。本リファレンスは『TWXA ライブラリ』の個々の関数について使用方法を解説しています。

表 1　対応製品

『USBX-I2219』/『LANX-I2219』/ 『USBX-I0800』/『USBX-I0404』/『USBX-I0008』/『LANX-I0800』/『LANX-I0404』/ 『USBX-A0800』/『LANX-A0800』/ 『USBX-I2424P』/『LANX-I2424P』

□ 本リファレンスの対応ファイルとバージョンについて

本リファレンスは下記のバージョンのファイル内容を元に記載されています。過去のバージョンについては記載内容と異なる場合がございますのでご注意ください。

表 2　対応するライブラリのバージョン

ファイル名	バージョン番号	対応 OS
TWXA.DLL	1.6.x.x [1]	Windows® 7 以降

□ 本リファレンス内の表記について

本リファレンス内では対応製品を「製品」または「デバイス」と表記します。それぞれの製品のホストインタフェースを区別する場合や各関数の対応製品を示す場合には表 3 に従います。

表 3　ホストインタフェースの表記・表示方法

説明文での表記	各関数の対応表示	対応製品
USB デバイス	USB	『USBX-I2219』/『USBX-I0800』/『USBX-I0404』/『USBX-I0008』/ 『USBX-A0800』/『USBX-I2424P』
LAN デバイス	LAN	『LANX-I2219』/『LANX-I0800』/『LANX-I0404』/ 『LANX-A0800』/『LANX-I2424P』

また、製品の持つ機能、端子数などにより使用できる関数が異なるため、表 4 の表示で関数が対応している製品を識別します。ただし、関数によっては全ての機能が利用できない場合もありますので詳細は製品のユーザーズマニュアルでご確認ください。

[1] x にはより細かなバージョンを示す数値が入ります。

表 4　製品タイプの表示方法

説明文での表記		各関数の対応表示	対応製品
I2219 タイプ		I2219	『USBX-I2219』/『LANX-I2219』
I0x0x タイプ	I0800 タイプ	I0800	『USBX-I0800』/『LANX-I0800』
	I0404 タイプ	I0404	『USBX-I0404』/『LANX-I0404』
	I0008 タイプ	I0008	『USBX-I0008』
A0800 タイプ		A0800	『USBX-A0800』/『LANX-A0800』
I2424 タイプ		I2424	『USBX-I2424P』/『LANX-I2424P』

本リファレンス内ではハードウェアの各電気的状態について下記のように表記いたします。

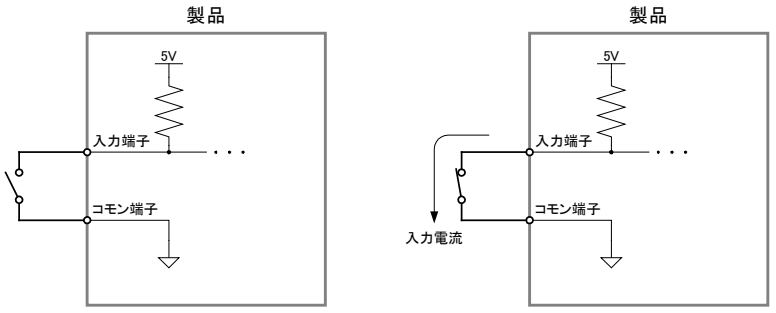
表 5　電気的状態の表記方法

表記	状態
“ON”	電流が流れている状態、スイッチが閉じている状態、オープンコレクタ(オープンドレイン)出力がシンク出力している状態。
“OFF”	電流が流れていない状態、スイッチが開いている状態、オープンコレクタ(オープンドレイン)出力がハイインピーダンスの状態。
“Hi”	電圧がロジックレベルのハイレベルに相当する状態。
“Lo”	電圧がロジックレベルのローレベルに相当する状態。

また、数値について「0x」、「&H」、「H’」はいずれもそれに続く数値が 16 進数であることを表します。“0x10”、“&H1F”、“H’20”などはいずれも 16 進数です。

デジタル入力端子の状態

デジタル入力端子は十分な入力電流が流れている状態を″ON″、入力電流が流れていないか十分でない場合を″OFF″とします。



入力端子の″OFF″状態
(ロジックレベルは″Hi″)

入力端子の″ON″状態
(ロジックレベルは″Lo″)

図 1　デジタル入力端子の"OFF"状態と"ON"状態

デジタル出力端子の状態

デジタル出力端子は出力トランジスタの抵抗値が十分に低くなっている状態、または、リレー接点が閉じている状態を"ON"、出力トランジスタの抵抗値が高い状態、または、リレー接点が開いている状態を"OFF"とします。

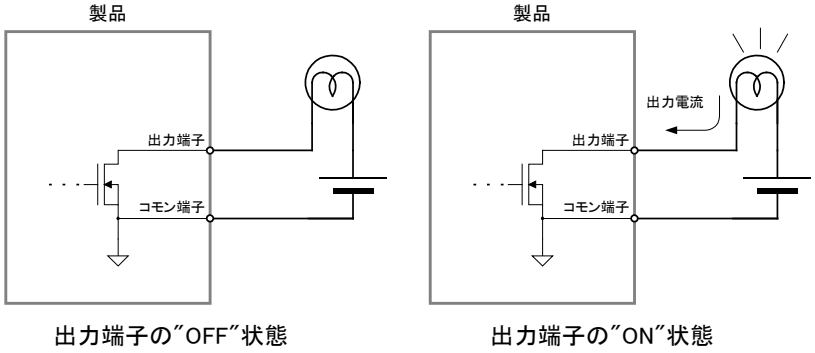


図 2　トランジスタ出力端子の"OFF"状態と"ON"状態

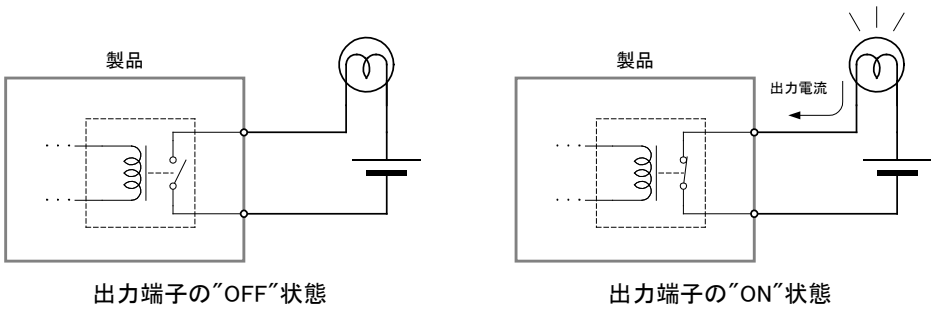


図 3　リレー出力端子の"OFF"状態と"ON"状態

2. 関数リファレンス

□ 関数の説明について

各関数は、対応するホストインタフェースを示すために、関数名を記したタイトルの後に表 3 の対応表示を行っています。また、利用する製品に対して使用できるかどうかを示すために、表 4 に従って対応する製品タイプを表示しています。

説明では最初に C/C++、Visual Basic® (.NET 以後)、Visual Basic for Applications (以下 VBA)、C# それぞれの言語における関数宣言が記載されます(表 6)。次に引数の説明、戻り値の説明(特殊な戻り値を返す場合のみ)、動作説明の順になっています。

C# の各関数は *Techw.IO* 名前空間の *TWXA* クラスのスタティックメンバ関数として宣言されていますが、名前空間とクラス名は省略して記述しています。

表 6 関数宣言の記述例

言語	関数宣言
C/C++	TW_STATUS TWXA_Open(TW_HANDLE *phDev, long Number, long Opt)
VB	Function TWXA_Open(ByRef phDev As System.IntPtr, ByVal Number As Integer, ByVal Opt As TWXA_OPEN_OPT) As Integer
VBA	Function TWXA_Open(ByRef phDev As Long, ByVal Number As Long, ByVal Opt As TWXA_OPEN_OPT) As Long
C#	STATUS Open(out System.IntPtr phDev, int Number, OPEN_OPT Opt)

各関数の引数の中には、入力できる値が限定されていて、ある定数を入力することが適当なものがあります。そのような場合、各開発環境の入力支援機能(インテリセンス)を十分活用できるよう、言語毎に異なった定数や列挙型を定義しています。

表 7 は *TWXA_Open()* 関数の *Opt* 引数の入力候補の一部です。引数の入力候補は表のように各言語別に記述方法が記載されます。

"C/C++"と書かれた行は C および C++で使用できる記述方法です。この値は*#define* で定義された定数です。

"C++"と書かれた行は C++で使用できる記述方法です。定数専用に宣言されたクラスのスタティックメンバになっています。Visual Studio® でこの定数を入力する場合、最初に"TWXA::"と入力すると画面に入力候補が表示されますので、定数を選択して入力を行ってください。

"VB/VBA"と書かれた行は Visual Basic と VBA で使用可能な記述方法です。この場合、関数の引数自体が列挙型となっており定数は列挙子です。

"C#"と書かれた行は C#で使用可能な記述方法です。この場合も Visual Basic 同様に関数の引数が列挙型となっています。名前空間は省略して記述しています。

表 7　引数の入力候補の例

言語	値	説明
C/C++	TWXA_ANY_DEVICE	制御できるデバイスであればインタフェースや製品タイプを問わずに接続します。
C++	TWXA::OPEN_OPT::ANY_DEVICE	
VB/VBA	TWXA_OPEN_OPT.ANY_DEVICE	
C#	TWXA.OPEN_OPT.ANY_DEVICE	
C/C++	TWXA_IF_USB	ホストインタフェースが USB のデバイスに接続します。
C++	TWXA::OPEN_OPT::IF_USB	
VB/VBA	TWXA_OPEN_OPT.IF_USB	
C#	TWXA.OPEN_OPT.IF_USB	
C/C++	TWXA_IF_LAN	ホストインタフェースが LAN のデバイスに接続します。
C++	TWXA::OPEN_OPT::IF_LAN	
VB/VBA	TWXA_OPEN_OPT.IF_LAN	
C#	TWXA.OPEN_OPT.IF_LAN	

ポインタや参照渡しの引数の場合、[入力]または[出力]という記述でデータの方向を示しています。[入力]の場合、関数の入力パラメータとして呼び出し側が予め変数に値をセットする必要があることを示します。[出力]の場合、関数側が実行結果を変数にセットして戻ります。変数によっては両方向に意味があるものもあります。

ほとんどの関数の戻り値は 32 ビットの整数で関数の実行結果を表します(次項参照)。関数がそれ以外の特別な戻り値を返す場合は、引数の説明の後に戻り値についての説明が記載されます。

- VBA のインテリセンスでは入力候補として列挙子のみが表示され型が表示されませんが、複数の型で同じ列挙子名を使用している場合があるため、列挙子のみ記述した場合にエラーとなる場合があります。例えば POd という列挙子は TWXA_WPORT.POd と TWXA_RPORT.POd が定義されているため、POd のみの記述ではエラーとなります。

□ 関数の戻り値の意味

以下に主な戻り値と、予想される原因、対処方法を示します。括弧内の数値が関数から返される数値です。尚、戻り値を示す各定数は各言語用の定義ファイル（拡張子が「.h」、「.vb」、「.bas」、「.cs」のファイル）中で定義されています。

TW_OK (H'00000000)

正常終了。

TW_INVALID_HANDLE (H'00000001)

デバイスのハンドルが無効であることを示します。既に切断されたハンドルや、接続されていないハンドルを関数に渡した可能性があります。プログラムの内容を確認してください。

TW_DEVICE_NOT_FOUND (H'00000002)

デバイスが見つからなかった場合に返ります。デバイスの電源が入っていない、正しいネットワークに接続されていない、他のプログラムがデバイスを使用しているなどの理由が考えられます。デバイスの状態を確認してください。

TW_IO_ERROR (H'00000004)

デバイスとの通信中にエラーが発生したことを示します。ケーブルが抜かれた場合などに発生します。接続を確認してください。

TW_INSUFFICIENT_RESOURCES (H'00000005)

1 つのプロセスから接続可能なデバイス数(デフォルト値 256)を超えたことを示します。別のプロセスから制御する必要があります。

TW_EEPROM_READ_FAILED (H'0000000B)

EEPROM/FRAM からのデータ読出しに失敗したことを示します。お手数ですが製品サポートにお問い合わせください。

TW_EEPROM_WRITE_FAILED (H'0000000C)

EEPROM/FRAM への書込みに失敗したことを示します。お手数ですが製品サポートにお問い合わせください。

TW_EEPROM_NOT_PRESENT (H'0000000E)

EEPROM/FRAM を搭載していないデバイスに対して EEPROM/FRAM 操作関数を呼び出した場合に返されます。ハンドルの接続先が目的の製品と異なることが考えられます。接続やデバイスの仕様を確認してください。

TW_INVALID_ARGS (H'00000010)

関数に与えられた引数が無効であることを示します。引数の値が誤っている可能性がありますので、プログラムの内容を確認してください。

TW_NOT_SUPPORTED (H'00000011)

サポートされない機能が呼び出されたことを示します。接続中のデバイスがその機能をサポートしない場合、デバイスのファームウェアバージョンが古い場合、動作モードが誤っている場合などに発生します。デバイスの仕様や状態を確認してください。

TW_OTHER_ERROR (H'00000012)

予期しないエラーを示します。お手数ですが製品サポートにお問い合わせください。

TW_TIMEOUT (H'FFFF0001)

送信または受信処理がタイムアウトしたことを示します。何らかの原因でデバイスからの応答がない場合に発生します。電源や接続、動作モードを確認してください。

TW_FILE_ERROR (H'FFFF0002)

ファイル操作に関するエラーが発生したことを示します。ファイルパスが間違っていないか、他のプロセスにより排他オープンされていないか確認してください。

TW_MEMORY_ERROR (H'FFFF0003)

操作に必要なメモリ確保に失敗したことを示します。システムの状態を確認してください。

TW_DATA_NOT_FOUND (H'FFFF0004)

必要な設定データが見つからないことを示します。お手数ですが製品サポートにお問い合わせください。

TW_SOCKET_ERROR (H'FFFF0005)

ネットワークライブラリのエラーです。ネットワークの制御においてエラーが発生したことを示します。多くの場合 Winsock の WSAGetLastError() を呼び出すとさらに詳しい情報を得ることができます。

TW_ACCESS_DENIED (H'FFFF0006)

デバイスとの認証作業に失敗したことを示します。パスワードが間違っている可能性があります。デバイスに設定したパスワードとライブラリに設定したパスワードが一致していることを確認してください。

TW_NOT_SUPPORTED_MODE (H'FFFF0007)

呼び出された機能をサポートしないモードであることを示します。デバイスのモード設定を確認してください。

TW_FLASH_MODE_DEVICE (H'FFFF0008)

フラッシュ書換えモードのデバイスのため制御ができないことを示します。デバイスのモード設定を確認してください。

TW_BUSY (H'FFFF0009)

指定の機能が既に使用されていることを示します。

TW_NOT_INITIALIZED (H'FFFF000A)

指定の機能が初期化されていないことを示します。使用する機能の動作モードを設定する前に、動作の開始処理や停止処理を行っていないか確認してください。

TW_ATF_ERR_FILE_VERSION (H'FFFF0101)

ATF ファイルのバージョンエラーです。ATF のファイルバージョンがライブラリで扱えないものであるときに発生します。新しいライブラリを入手してください。

TW_ATF_ERR_ILLEGAL_FILE (H'FFFF0102)

ATF ファイルの内容が不正であることを示します。ファイルが壊れているか、何らかの理由で正しくない ATF ファイルが作成されたことが考えられます。ファイルに異常が無いと思われる場合には、お手数ですが製品サポートにお問い合わせください。

TW_ATF_ERR_SERVICE_VERSION (H'FFFF0103)

システムファームのバージョンが ATF ファイルの要求に合わないことを示します。システムファームのバージョンが古いか、ATF ファイルの作成時に誤ったバージョン番号を記述した可能性があります。システムファームのバージョン、および、ATF ファイルの要求バージョンを確認してください。

TW_FLASH_ERR_ERASE (H'FFFF0203)

フラッシュメモリの消去に失敗したことを示します。ハードウェアトラブルの可能性があります。お手数ですが製品サポートにお問い合わせください。

TW_FLASH_ERR_WRITE (H'FFFF0204)

フラッシュメモリの書込みに失敗したことを示します。ハードウェアトラブルの可能性があります。お手数ですが製品サポートにお問い合わせください。

TW_FLASH_ERR_SIZE (H'FFFF0210)

フラッシュメモリへの書込みバイト数が正しくないことを示します。128 バイトの倍数になっていない場合、4096 バイトより大きい指定になっている場合などに発生します。書込みバイト数の指定を確認してください。

TW_FLASH_ERR_ADDRESS (H'FFFF0211)

フラッシュメモリの書込み指定アドレスが正しくないことを示します。書込みが許されない領域、または、128 バイト境界になっていないことが考えられます。アドレス指定を確認してください。

TW_FLASH_ERR_NOT_ERASED (H'FFFF0212)

未消去のフラッシュメモリ領域に書込みを行おうとしたことを示します。書込みを行う前に消去を行ってください。

□ 構造体

一部の関数では構造体を使用します。構造体は関連する関数の前に説明を記述しています。関数同様 C/C++、Visual Basic、VBA、C# それぞれの言語における宣言が記載され、次にメンバの意味、構造体の説明の順になっています。

□ マルチスレッドプログラムからの呼び出しについて

ライブラリでは複数のスレッドからの関数呼び出しをサポートしていますが、デバイスとの通信仕様により、1 つのデバイスを複数のスレッドから同時に制御することはできません。何らかの理由により、複数のスレッドから 1 つのデバイスにアクセスする必要がある場合には、クリティカルセクションなどを使用することにより、ライブラリ関数の呼び出しをシリアル化し、複数のスレッドが同時に 1 つのデバイスにアクセスしないようにプログラムしてください。

1 つのスレッドが 1 つのデバイスのみを制御する場合は、複数のスレッドから同時にライブラリ関数を呼び出しても問題ありません。

□ デバイスへの接続／切断に関する関数

TWXA_Open() USB LAN I2219 IO800 IO404 IO008 AO800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_Open(TW_HANDLE *phDev, long Number, long Opt)
VB	Function TWXA_Open(ByRef phDev As System.IntPtr, ByVal Number As Integer, ByVal Opt As TWXA_OPEN_OPT) As Integer
VBA	Function TWXA_Open(ByRef phDev As Long, ByVal Number As Long, ByVal Opt As TWXA_OPEN_OPT) As Long
C#	STATUS Open(out System.IntPtr phDev, int Number, OPEN_OPT Opt)

phDev ： [出力]取得したハンドルの格納先
Number ： 接続する製品の装置番号。0 を指定した場合は最初に見つかったデバイスに接続します
Opt ： 接続オプション。以下を OR で結合

言語	値	説明
C/C++	TWXA_ANY_DEVICE	制御できるデバイスであればインタフェースや製品タイプを問わずに接続します。
C++	TWXA::OPEN_OPT::ANY_DEVICE	
VB/VBA	TWXA_OPEN_OPT.ANY_DEVICE	
C#	TWXA.OPEN_OPT.ANY_DEVICE	
C/C++	TWXA_IF_USB	ホストインタフェースが USB のデバイスに接続します。
C++	TWXA::OPEN_OPT::IF_USB	
VB/VBA	TWXA_OPEN_OPT.IF_USB	
C#	TWXA.OPEN_OPT.IF_USB	
C/C++	TWXA_IF_LAN	ホストインタフェースが LAN のデバイスに接続します。
C++	TWXA::OPEN_OPT::IF_LAN	
VB/VBA	TWXA_OPEN_OPT.IF_LAN	
C#	TWXA.OPEN_OPT.IF_LAN	
C/C++	TWXA_IF_ANY	インタフェースの種類(USB または LAN)を問わずに接続します。USB デバイス、LAN デバイスの順に接続可能なデバイスを検索します。
C++	TWXA::OPEN_OPT::IF_ANY	
VB/VBA	TWXA_OPEN_OPT.IF_ANY	
C#	TWXA.OPEN_OPT.IF_ANY	
C/C++	TWXA_TYPE_I2219	I2219 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_I2219	
VB/VBA	TWXA_OPEN_OPT.TYPE_I2219	
C#	TWXA.OPEN_OPT.TYPE_I2219	
C/C++	TWXA_TYPE_IOx0x	IOx0x タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_IOx0x	
VB/VBA	TWXA_OPEN_OPT.TYPE_IOx0x	
C#	TWXA.OPEN_OPT.TYPE_IOx0x	
C/C++	TWXA_TYPE_AOx0x	AO800 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_AOx0x	
VB/VBA	TWXA_OPEN_OPT.TYPE_AOx0x	
C#	TWXA.OPEN_OPT.TYPE_AOx0x	
C/C++	TWXA_TYPE_I2424	I2424 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_I2424	
VB/VBA	TWXA_OPEN_OPT.TYPE_I2424	
C#	TWXA.OPEN_OPT.TYPE_I2424	

言語	値	説明
C/C++	TWXA_TYPE_ANY	製品タイプを問わずに接続します。I2219 タイプ、I0x0x タイプ、A0800 タイプ、I2424 タイプの順に接続可能なデバイスを検索します。
C++	TWXA::OPEN_OPT::TYPE_ANY	
VB/VBA	TWXA_OPEN_OPT.TYPE_ANY	
C#	TWXA.OPEN_OPT.TYPE_ANY	
C/C++	TWXA_LIST_UPDATE	LAN 内のデバイスを検索し、結果をライブラリ内部のテーブルに登録します。LAN インタフェースデバイス専用です。
C++	TWXA::OPEN_OPT::LIST_UPDATE	
VB/VBA	TWXA_OPEN_OPT.LIST_UPDATE	
C#	TWXA.OPEN_OPT.LIST_UPDATE	

装置番号を指定してデバイスに接続します。成功すると phDev にデバイスへのハンドルが格納されます。装置番号の設定方法は製品のユーザーズマニュアルを参照してください。Number を 0 とすると番号を指定せず、最初に見つかったデバイスに接続を行います。

Opt 引数により接続先製品のインタフェースやタイプを指定することができます。インタフェースやタイプを限定する場合には、インタフェースのオプションとタイプのオプションから 1 つずつ選択し、OR で結合してください。例えば C 言語のプログラムから、USB インタフェースのデバイスでタイプを指定せずに接続する場合、Opt 引数には"TWXA_IF_USB | TWXA_TYPE_ANY"を指定します。
デバイスを限定しない場合は TWXA_ANY_DEVICE(相当の)オプションを 1 つ指定します。

プログラムが起動後初めて LAN インタフェースのデバイスに接続する場合は、TWXA_LIST_UPDATE(相当の)オプションを指定する必要があります。このオプションにより、LAN 内のデバイスが検索され接続可能なデバイスを列挙した内部テーブルが更新されます。

TWXA_Close() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_Close(TW_HANDLE hDev)
VB	Function TWXA_Close(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_Close(ByVal hDev As Long) As Long
C#	STATUS Close(System.IntPtr hDev)

hDev ： デバイスのハンドル

ハンドルをクローズし、デバイスへのアクセスを終了します。

TWXA_CloseAll() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_CloseAll()
VB	Function TWXA_CloseAll() As Integer
VBA	Function TWXA_CloseAll() As Long
C#	STATUS CloseAll()

プロセスが接続しているデバイス全てをクローズします。

TWXA_OpenByAddress() LAN I2219 I0800 I0404 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_OpenByAddress(TW_HANDLE *phDev, LPCTSTR Address, long Opt)
VB	Function TWXA_OpenByAddress(ByRef phDev As System.IntPtr, ByVal Address As String, ByVal Opt As TWXA_OPEN_OPT) As Integer
VBA	Function TWXA_OpenByAddress(ByRef phDev As Long, ByVal Address As String, ByVal Opt As TWXA_OPEN_OPT) As Long
C#	STATUS OpenByAddress(out System.IntPtr phDev, string Address, OPEN_OPT Opt)

phDev　：[出力]取得したハンドルの格納先
Address：接続する製品の IP アドレスまたはドメイン名
Opt　　：接続オプション。以下を OR で結合

言語	値	説明
C/C++	TWXA_ANY_DEVICE	制御できるデバイスであればインタフェースや製品タイプを問わずに接続します。
C++	TWXA::OPEN_OPT::ANY_DEVICE	
VB/VBA	TWXA_OPEN_OPT.ANY_DEVICE	
C#	TWXA.OPEN_OPT.ANY_DEVICE	
C/C++	TWXA_IF_LAN	ホストインタフェースが LAN のデバイスに接続します。
C++	TWXA::OPEN_OPT::IF_LAN	
VB/VBA	TWXA_OPEN_OPT.IF_LAN	
C#	TWXA.OPEN_OPT.IF_LAN	
C/C++	TWXA_TYPE_I2219	I2219 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_I2219	
VB/VBA	TWXA_OPEN_OPT.TYPE_I2219	
C#	TWXA.OPEN_OPT.TYPE_I2219	
C/C++	TWXA_TYPE_I0x0x	I0800、I0404 いずれかのタイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_I0x0x	
VB/VBA	TWXA_OPEN_OPT.TYPE_I0x0x	
C#	TWXA.OPEN_OPT.TYPE_I0x0x	
C/C++	TWXA_TYPE_A0x0x	A0800 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_A0x0x	
VB/VBA	TWXA_OPEN_OPT.TYPE_A0x0x	
C#	TWXA.OPEN_OPT.TYPE_A0x0x	
C/C++	TWXA_TYPE_I2424	I2424 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_I2424	
VB/VBA	TWXA_OPEN_OPT.TYPE_I2424	
C#	TWXA.OPEN_OPT.TYPE_I2424	
C/C++	TWXA_TYPE_ANY	製品タイプを問わずに接続します。I2219 タイプ、I0x0x タイプ、A0800 タイプ、I2424 タイプの順に接続可能なデバイスを検索します。
C++	TWXA::OPEN_OPT::TYPE_ANY	
VB/VBA	TWXA_OPEN_OPT.TYPE_ANY	
C#	TWXA.OPEN_OPT.TYPE_ANY	

IP アドレスやドメインを指定してデバイスに接続します。成功すると phDev にデバイスへのハンドルが格納されます。
通常、デバイスのポート番号(デフォルトで 49,152)を指定する必要はありませんが、設定ツールでポート番号を変更した場合や、ルータの設定でポート番号の指定が必要な場合には、"ドメイン名:ポート番号"のようにアドレス指定に続けて":"(コロン)とポート番号を指定します。

Opt 引数により接続先製品のタイプを指定することができます。例えば C 言語のプログラムから、I2219 タイプのデバイスに接続する場合、Opt 引数には"TWXA_IF_LAN | TWXA_TYPE_I2219"を指定します。
デバイスを限定しない場合は TWXA_ANY_DEVICE(相当の)オプションを 1 つ指定します。

TWXA_Listen() LAN I2219 I0800 I0404 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_Listen(UINT_PTR *pListenSocket, LPCTSTR pLocalIP, DWORD PortNumber)
VB	Function TWXA_Listen(ByRef pListenSocket As System.IntPtr, ByVal pLocalIP As String, ByVal PortNumber As Integer) As Integer
VBA	Function TWXA_Listen(ByRef pListenSocket As Long, ByVal pLocalIP As String, ByVal PortNumber As Long) As Long
C#	STATUS Listen(out System.IntPtr pListenSocket, string pLocalIP, int PortNumber)

pListenSocket ： [出力]接続待ちソケットの格納先
pLocalIP　　： [入力]接続待ちを行うローカル(パソコン側)の IP アドレス
PortNumber　： 接続待ちを行うポート番号

指定のポート番号をオープンし、クライアントモードのデバイスの接続を待ちます。
pListenSocket に返された値を TWXA_Accept()、および、TWXA_AcceptSelect() に渡すことで、ここで設定したポートへの接続を受け入れることができます。
pLocalIP はパソコンのネットワークカードが複数ある場合に、接続待ちを行う IP アドレスを指定します。特に指定が無い場合は NULL または""とすることができます。

TWXA_Accept() LAN I2219 I0800 I0404 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_Accept(UINT_PTR ListenSocket, TW_HANDLE *phDev)
VB	Function TWXA_Accept (ByVal ListenSocket As System.IntPtr, ByRef phDev As System.IntPtr) As Integer
VBA	Function TWXA_Accept(ByVal ListenSocket As Long, ByRef phDev As Long) As Long
C#	STATUS Accept(System.IntPtr ListenSocket, out System.IntPtr phDev)

ListenSocket ： 接続待ちソケット
phDev　　： [出力]デバイスへのハンドルの格納先

TWXA_Listen()でオープンした接続待ちソケットに対して、接続要求があれば受け入れてデバイスへのハンドルを返します。
接続要求が無い場合は TW_DEVICE_NOT_FOUND を返します。接続待ちを行う間はこの関数を繰り返し呼び出して、接続要求の有無を確認してください。

TWXA_AcceptSelect() LAN I2219 I0800 I0404 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_AcceptSelect(UINT_PTR ListenSocket, TW_HANDLE *phDev, Long Opt)
VB	Function TWXA_AcceptSelect(ByVal ListenSocket As System.IntPtr, ByRef phDev As System.IntPtr, ByVal Opt As TWXA_TYPE_OPT) As Integer
VBA	Function TWXA_AcceptSelect(ByVal ListenSocket As Long, ByRef phDev As Long, ByVal Opt As TWXA_TYPE_OPT) As Long
C#	STATUS AcceptSelect(System.IntPtr ListenSocket, out System.IntPtr phDev, TYPE_OPT Opt) STATUS AcceptSelect(System.IntPtr ListenSocket, out System.IntPtr phDev)

ListenSocket ： 接続待ちソケット

phDev　　：[出力]デバイスへのハンドルの格納先
Opt　　：接続を受け入れる製品のタイプ。以下を OR で結合

言語	値	説明
C/C++	TWXA_TYPE_I2219	I2219 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_I2219	
VB/VBA	TWXA_OPEN_OPT.TYPE_I2219	
C#	TWXA.OPEN_OPT.TYPE_I2219	
C/C++	TWXA_TYPE_IOx0x	I0800、I0404 いずれかのタイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_IOx0x	
VB/VBA	TWXA_OPEN_OPT.TYPE_IOx0x	
C#	TWXA.OPEN_OPT.TYPE_IOx0x	
C/C++	TWXA_TYPE_AOx0x	A0800 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_AOx0x	
VB/VBA	TWXA_OPEN_OPT.TYPE_AOx0x	
C#	TWXA.OPEN_OPT.TYPE_AOx0x	
C/C++	TWXA_TYPE_I2424	I2424 タイプに接続します。
C++	TWXA::OPEN_OPT::TYPE_I2424	
VB/VBA	TWXA_OPEN_OPT.TYPE_I2424	
C#	TWXA.OPEN_OPT.TYPE_I2424	
C/C++	TWXA_TYPE_ANY	デバイスのタイプを問わずに接続します。
C++	TWXA::OPEN_OPT::TYPE_ANY	
VB/VBA	TWXA_OPEN_OPT.TYPE_ANY	
C#	TWXA.OPEN_OPT.TYPE_ANY	

TWXA_Listen()でオープンした接続待ちソケットに対して、接続要求があるデバイスのうち、指定タイプのデバイスを受け入れてデバイスへのハンドルを返します。
接続要求が無い場合は TW_DEVICE_NOT_FOUND を返します。接続待ちを行う間はこの関数を繰り返し呼び出して、接続要求の有無を確認してください。

TWXA_CloseListenSocket() LAN I2219 I0800 I0404 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_CloseListenSocket(UINT_PTR ListenSocket)
VB	Function TWXA_CloseListenSocket(ByVal ListenSocket As System.IntPtr) As Integer
VBA	Function TWXA_CloseListenSocket(ByVal ListenSocket As Long) As Long
C#	STATUS CloseListenSocket(System.IntPtr ListenSocket)

ListenSocket ： 接続待ちソケット

TWXA_Listen()でオープンした接続待ちのソケットをクローズします。

□ ポート操作関数

TWXA_PortWrite() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PortWrite(TW_HANDLE hDev, DWORD Port, BYTE Data, BYTE Mask)
VB	Function TWXA_PortWrite(ByVal hDev As System.IntPtr, ByVal Port As TWXA_WPORT, ByVal Data As Byte, ByVal Mask As Byte) As Integer
VBA	Function TWXA_PortWrite(ByVal hDev As Long, ByVal Port As TWXA_WPORT, ByVal Data As Byte, ByVal Mask As Byte) As Long
C#	STATUS PortWrite(System.IntPtr hDev, WPORT Port, byte Data) STATUS PortWrite(System.IntPtr hDev, WPORT Port, byte Data, byte Mask)

hDev ： デバイスのハンドル
Port ： 書込み先の指定
I2219 タイプの場合：以下の値またはアドレスを指定します

言語	値	説明
C/C++	TWXA_POd	0d0-0d7 出力を変更します。
C++	TWXA::WPORT::POd	
VB/VBA	TWXA_WPORT.POd	
C#	TWXA.WPORT.POd	
C/C++	TWXA_POe	0e0-0e7 出力を変更します。
C++	TWXA::WPORT::POe	
VB/VBA	TWXA_WPORT.POe	
C#	TWXA.WPORT.POe	
C/C++	TWXA_POf	0f0-0f2 出力を変更します。
C++	TWXA::WPORT::POf	
VB/VBA	TWXA_WPORT.POf	
C#	TWXA.WPORT.POf	
C/C++	TWXA_DA0	DA0 出力を変更します。
C++	TWXA::WPORT::DA0	
VB/VBA	TWXA_WPORT.DA0	
C#	TWXA.WPORT.DA0	
C/C++	TWXA_DA1	DA1 出力を変更します。
C++	TWXA::WPORT::DA1	
VB/VBA	TWXA_WPORT.DA1	
C#	TWXA.WPORT.DA1	
C/C++	TWXA_USER_STATUS	ユーザー用ステータスレジスタを変更します。
C++	TWXA::WPORT::USER_STATUS	
VB/VBA	TWXA_WPORT.USER_STATUS	
C#	TWXA.WPORT.USER_STATUS	

IOxOx タイプの場合：以下の値またはアドレスを指定します

言語	値	説明
C/C++	TWXA_POd	0d0-0d5 出力を変更します。
C++	TWXA::WPORT::POd	
VB/VBA	TWXA_WPORT.POd	
C#	TWXA.WPORT.POd	

言語	値	説明
C/C++	TWXA_POf	0f0-0f1 出力を変更します。
C++	TWXA::WPORT::POf	
VB/VBA	TWXA_WPORT.POf	
C#	TWXA.WPORT.POf	
C/C++	TWXA_USER_STATUS	ユーザー用ステータスレジスタを変更します。
C++	TWXA::WPORT::USER_STATUS	
VB/VBA	TWXA_WPORT.USER_STATUS	
C#	TWXA.WPORT.USER_STATUS	

A0800 タイプの場合：以下の値またはアドレスを指定します

言語	値	説明
C/C++	TWXA_USER_STATUS	ユーザー用ステータスレジスタを変更します。
C++	TWXA::WPORT::USER_STATUS	
VB/VBA	TWXA_WPORT.USER_STATUS	
C#	TWXA.WPORT.USER_STATUS	

I2424 タイプの場合：以下の値のいずれか

言語	値	説明
C/C++	TWXA_POUT0	OUT00-OUT07 出力を変更します。
C++	TWXA::WPORT::POUT0	
VB/VBA	TWXA_WPORT.POUT0	
C#	TWXA.WPORT.POUT0	
C/C++	TWXA_POUT1	OUT10-OUT17 出力を変更します。
C++	TWXA::WPORT::POUT1	
VB/VBA	TWXA_WPORT.POUT1	
C#	TWXA.WPORT.POUT1	
C/C++	TWXA_POUT2	OUT20-OUT27 出力を変更します。
C++	TWXA::WPORT::POUT2	
VB/VBA	TWXA_WPORT.POUT2	
C#	TWXA.WPORT.POUT2	
C/C++	TWXA_USER_STATUS_Rx	ユーザー用ステータスレジスタ Rx を変更します。 (x = 0～9, A～F)
C++	TWXA::WPORT::USER_STATUS_Rx	
VB/VBA	TWXA_WPORT.USER_STATUS_Rx	
C#	TWXA.WPORT.USER_STATUS_Rx	

Data ： 書き込むデータ
Mask ： 操作するビットを指定するマスク(1 と対応するビットのみ影響を受けます)

デジタル出力、アナログ出力の変更に使用します。また、ユーザー用ステータスレジスタやユーザーメモリなどの領域への書込みにも使用します。
Mask 引数は特定のビットだけを操作する場合に使用します。Mask 引数が 1 となっているビットのみ書込みの影響を受けます。例えば 0d7 だけを操作する場合、Mask = 0x80 とします。全てのビットを操作する場合 Mask = 0xff です。

TWXA_PortRead() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PortRead(TW_HANDLE hDev, DWORD Port, BYTE *pData)
VB	Function TWXA_PortRead(ByVal hDev As System.IntPtr, ByVal Port As TWXA_RPORT, ByRef pData As Byte) As Integer
VBA	Function TWXA_PortRead(ByVal hDev As Long, ByVal Port As TWXA_RPORT, ByRef pData As Byte) As Long
C#	STATUS PortRead(System.IntPtr hDev, RPORT Port, out byte pData)

hDev ： デバイスのハンドル
Port ： 読み出す対象を指定
I2219 タイプの場合：以下の値またはアドレスを指定します

言語	値	説明
C/C++	TWXA_PIa	Ia0-Ia7 入力を読み取ります。
C++	TWXA::RPORT::PIa	
VB/VBA	TWXA_RPORT.PIa	
C#	TWXA.RPORT.PIa	
C/C++	TWXA_PIb	Ib0-Ib7 入力を読み取ります。
C++	TWXA::RPORT::PIb	
VB/VBA	TWXA_RPORT.PIb	
C#	TWXA.RPORT.PIb	
C/C++	TWXA_PIc	Ic0-Ic5 入力を読み取ります。
C++	TWXA::RPORT::PIc	
VB/VBA	TWXA_RPORT.PIc	
C#	TWXA.RPORT.PIc	
C/C++	TWXA_POd	Od0-Od7 の出力値を読み取ります。
C++	TWXA::RPORT::POd	
VB/VBA	TWXA_RPORT.POd	
C#	TWXA.RPORT.POd	
C/C++	TWXA_POe	Oe0-Oe7 の出力値を読み取ります。
C++	TWXA::RPORT::POe	
VB/VBA	TWXA_RPORT.POe	
C#	TWXA.RPORT.POe	
C/C++	TWXA_POf	Of0-Of2 の出力値を読み取ります。
C++	TWXA::RPORT::POf	
VB/VBA	TWXA_RPORT.POf	
C#	TWXA.RPORT.POf	
C/C++	TWXA_DA0	DA0 の出力値を読み取ります。
C++	TWXA::RPORT::DA0	
VB/VBA	TWXA_RPORT.DA0	
C#	TWXA.RPORT.DA0	
C/C++	TWXA_DA1	DA1 の出力値を読み取ります。
C++	TWXA::RPORT::DA1	
VB/VBA	TWXA_RPORT.DA1	
C#	TWXA.RPORT.DA1	
C/C++	TWXA_USER_STATUS	ユーザー用ステータスレジスタの値を読み取ります。
C++	TWXA::RPORT::USER_STATUS	
VB/VBA	TWXA_RPORT.USER_STATUS	
C#	TWXA.RPORT.USER_STATUS	

I0x0x タイプの場合：以下の値またはアドレスを指定します

言語	値	説明
C/C++	TWXA_PIa	Ia0-Ia3 入力を読み取ります。
C++	TWXA::RPORT::PIa	
VB/VBA	TWXA_RPORT.PIa	
C#	TWXA.RPORT.PIa	
C/C++	TWXA_PIc	Ic0-Ic3 入力を読み取ります。
C++	TWXA::RPORT::PIc	
VB/VBA	TWXA_RPORT.PIc	
C#	TWXA.RPORT.PIc	
C/C++	TWXA_POd	Od0-Od5 の出力値を読み取ります。
C++	TWXA::RPORT::POd	
VB/VBA	TWXA_RPORT.POd	
C#	TWXA.RPORT.POd	
C/C++	TWXA_POd	Od0-Od5 出力を変更します。
C++	TWXA::WPORT::POd	
VB/VBA	TWXA_WPORT.POd	
C#	TWXA.WPORT.POd	
C/C++	TWXA_POf	Of0-Of1 出力を変更します。
C++	TWXA::WPORT::POf	
VB/VBA	TWXA_WPORT.POf	
C#	TWXA.WPORT.POf	
C/C++	TWXA_USER_STATUS	ユーザー用ステータスレジスタを変更します。
C++	TWXA::WPORT::USER_STATUS	
VB/VBA	TWXA_WPORT.USER_STATUS	
C#	TWXA.WPORT.USER_STATUS	

A0800 タイプの場合：以下の値またはアドレスを指定します

言語	値	説明
C/C++	TWXA_USER_STATUS	ユーザー用ステータスレジスタを変更します。
C++	TWXA::WPORT::USER_STATUS	
VB/VBA	TWXA_WPORT.USER_STATUS	
C#	TWXA.WPORT.USER_STATUS	

I2424 タイプの場合：以下の値のいずれか

言語	値	説明
C/C++	TWXA_PIN0	IN00-IN07 入力を読み取ります。
C++	TWXA::RPORT::PIN0	
VB/VBA	TWXA_RPORT.PIN0	
C#	TWXA.RPORT.PIN0	
C/C++	TWXA_PIN1	IN10-IN17 入力を読み取ります。
C++	TWXA::RPORT::PIN1	
VB/VBA	TWXA_RPORT.PIN1	
C#	TWXA.RPORT.PIN1	
C/C++	TWXA_PIN2	IN20-IN27 入力を読み取ります。
C++	TWXA::RPORT::PIN2	
VB/VBA	TWXA_RPORT.PIN2	
C#	TWXA.RPORT.PIN2	

言語	値	説明
C/C++	TWXA_POUT0	OUT00-OUT07 の出力値を読み取ります。
C++	TWXA::RPORT::POUT0	
VB/VBA	TWXA_RPORT.POUT0	
C#	TWXA.RPORT.POUT0	
C/C++	TWXA_POUT1	OUT10-OUT17 の出力値を読み取ります。
C++	TWXA::RPORT::POUT1	
VB/VBA	TWXA_RPORT.POUT1	
C#	TWXA.RPORT.POUT1	
C/C++	TWXA_POUT2	OUT20-OUT27 の出力値を読み取ります。
C++	TWXA::RPORT::POUT2	
VB/VBA	TWXA_RPORT.POUT2	
C#	TWXA.RPORT.POUT2	
C/C++	TWXA_USER_STATUS_Rx	ユーザー用ステータスレジスタ Rx の値を読み取ります。 (x = 0～9, A～F)
C++	TWXA::RPORT::USER_STATUS_Rx	
VB/VBA	TWXA_RPORT.USER_STATUS_Rx	
C#	TWXA.RPORT.USER_STATUS_Rx	

pData : [出力]読み出した値の格納先

デジタル入力値の読出しに使用します。また、デジタル出力、アナログ出力の出力値、ユーザー用ステータスレジスタやユーザーメモリなどの領域の読出しにも使用します。

□ アナログ入力／アナログ出力／アナログ値変換関数

TWXA_ADRead() USB LAN I2219 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADRead(TW_HANDLE hDev, long Ch, long *pData)
VB	Function TWXA_ADRead(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByRef pData As Integer) As Integer Function TWXA_ADRead(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal pData() As Integer) As Integer
VBA	Function TWXA_ADRead(ByVal hDev As Long, ByVal Ch As Long, ByRef pData As Long) As Long
C#	STATUS ADRead(System.IntPtr hDev, int Ch, out int pData) STATUS ADRead(System.IntPtr hDev, int Ch, int []pData)

hDev ： デバイスのハンドル
Ch ： 入力するチャンネル
I2219 タイプの場合 ： 0～3
A0800 タイプの場合 ： 0～7 or TWXA_AD_ALL
I2424 タイプの場合 ： 0～3 or TWXA_AD_ALL

pData ： [出力]AD 変換結果の格納先

- I2219 タイプの場合

指定チャンネル(0～3)を AD 変換した結果を読み出します。変換結果の上位 16 ビットは常に 0 で、下位 16 ビットの MSB から 10 ビットに格納されます。値の範囲は 16 進数で 0x0000-0xffc0、10 進数で 0-65,472 の範囲となります。

ビット	31-16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
値	常に 0	AD 変換結果										常に 0					

- A0800 タイプの場合

指定チャンネル(0～7)または全チャンネル(TWXA_AD_ALL)を AD 変換した結果を読み出します。全チャンネルを変換する場合、変換結果は全てのチャンネルを同時に変換した結果が返されます。値の範囲は 10 進数で -32,768～32,767 の範囲となります。
TWXA_AD_ALL を指定した場合 pData に AD0-AD7 までの値が連続して格納されますので pData には 8 チャンネル分のデータを格納できる領域(4 x 8 = 32 バイト)が必要です。

TWXA_ADSetMode()関数によりオーバーサンプリングレートを設定した場合、設定値に比例した回数のサンプリングが行われ、その平均値が変換結果として pData 引数に格納されます。

- I2424 タイプの場合

指定チャンネル(0～3)または全チャンネル(TWXA_AD_ALL)を AD 変換した結果を読み出します。全チャンネルを変換する場合、変換結果は全てのチャンネルを同時に変換した結果が返されます。値の範囲は 10 進数で -32,768～32,767 の範囲となります。
TWXA_AD_ALL を指定した場合 pData に AD0-AD3 までの値が連続して格納されますので pData には 4 チャンネル分のデータを格納できる領域(4 x 4 = 16 バイト)が必要です。

TWXA_ADSetMode()関数によりオーバーサンプリングレートを設定した場合、設定値に比例した回数のサンプリングが行われ、その平均値が変換結果として pData 引数に格納されます。

TWXA_DAWrite() USB LAN I2219 I2424

言語	関数宣言
C/C++	long TWXA_DAWrite(TW_HANDLE hDev, long Ch, long Data)
VB	Function TWXA_DAWrite(ByVal hDev As System.IntPtr, ByVal Ch As TWXA_DA_CH, ByVal Data As Integer) As Integer
VBA	Function TWXA_DAWrite(ByVal hDev As Long, ByVal Ch As TWXA_DA_CH, ByVal Data As Long) As Long
C#	STATUS DAWrite(System.IntPtr hDev, DA_CH Ch, int Data)

hDev ： デバイスのハンドル
Ch　 ： 出力値を変更するチャンネル。以下の値のいずれか

言語	値	説明
C/C++	TWXA_DA0	DA0 出力を変更します。
C++	TWXA::RPORT::DA0	
VB/VBA	TWXA_RPORT.DA0	
C#	TWXA.RPORT.DA0	
C/C++	TWXA_DA1	DA1 出力を変更します。
C++	TWXA::RPORT::DA1	
VB/VBA	TWXA_RPORT.DA1	
C#	TWXA.RPORT.DA1	

Data ： 出力する値
I2219 タイプの場合 ： 0～255
I2424 タイプの場合 ： 0～1,023

指定チャンネルのアナログ出力を変更します。

TWXA_DARead() USB LAN I2219 I2424

言語	関数宣言
C/C++	long TWXA_DARead(TW_HANDLE hDev, long Ch, long *pData)
VB	Function TWXA_DARead(ByVal hDev As System.IntPtr, ByVal Ch As TWXA_DA_CH, ByRef pData As Integer) As Integer
VBA	Function TWXA_DARead(ByVal hDev As Long, ByVal Ch As TWXA_DA_CH, ByRef pData As Long) As Long
C#	STATUS DARead(System.IntPtr hDev, DA_CH Ch, out int pData)

hDev ： デバイスのハンドル
Ch　 ： 現在の出力値を読み出すチャンネル。以下の値のいずれか

言語	値	説明
C/C++	TWXA_DA0	DA0 の出力値を読み出します。
C++	TWXA::RPORT::DA0	
VB/VBA	TWXA_RPORT.DA0	
C#	TWXA.RPORT.DA0	
C/C++	TWXA_DA1	DA1 の出力値を読み出します。
C++	TWXA::RPORT::DA1	
VB/VBA	TWXA_RPORT.DA1	
C#	TWXA.RPORT.DA1	

pData ： [出力]読み出した値の格納先

アナログ出力の現在の出力値の読出しに使用します。

TWXA_DASetConvertMode() USB LAN I2424

言語	関数宣言
C/C++	long TWXA_DASetConvertMode(TW_HANDLE hDev, long Ch, double *pFreq, DWORD Addr, DWORD Cnt, BOOL Flg)
VB	Function TWXA_DASetConvertMode(ByVal hDev As System.IntPtr, ByVal Ch As TWXA_DA_CH, ByRef pFreq As Double, ByVal Addr As Integer, ByVal Cnt As Integer, ByVal Flg As Integer) As Integer
VBA	Function TWXA_DASetConvertMode(ByVal hDev As Long, ByVal Ch As TWXA_DA_CH, ByRef pFreq As Double, ByVal Addr As Long, ByVal Cnt As Long, ByVal Flg As Long) As Long
C#	STATUS DASetConvertMode(System.IntPtr hDev, DA_CH Ch, ref double pFreq, Uint Addr, uint Cnt, int Flg)

hDev ： デバイスのハンドル
Ch ： 設定を行うチャンネル。以下の値のいずれか

言語	値	説明
C/C++	TWXA_DA0	DA0 出力の設定を行います。
C++	TWXA::RPORT::DA0	
VB/VBA	TWXA_RPORT.DA0	
C#	TWXA.RPORT.DA0	
C/C++	TWXA_DA1	DA1 出力の設定を行います。
C++	TWXA::RPORT::DA1	
VB/VBA	TWXA_RPORT.DA1	
C#	TWXA.RPORT.DA1	

pFreq ： [入力]希望の周波数(Hz 単位) (0～20,000[Hz])
[出力]実際に設定できた周波数(Hz 単位)

Addr ： 出力データが格納されたメモリの先頭アドレス
Cnt ： 出力するデータ数
Flg ： 繰り返し実行するかどうかのフラグ

アナログ出力の周期出力設定を行います。Addr で指定されたアドレスから、Cnt で指定されたデータ数のデータ出力を行います。アナログ出力は pFreq で指定されたレートで更新されます。この変換レートはデバイスの内部クロックを分周して作られるため値は離散的となります。そのため、希望値と設定可能な値が異なる場合があります。関数は pFreq に実際に設定できた値をセットして戻ります。Flg に 0 以外を指定すると、指定データ数のデータ出力を繰り返し実行します。

周期出力で出力するデータは TWXA_PortBWrite()でユーザーメモリに書き込みます。書込みを行う際は、1 データを 2 バイトで構成し、データはビッグエンディアンとしてください。

TWXA_DAStart() USB LAN I2424

言語	関数宣言
C/C++	long TWXA_DAStart(TW_HANDLE hDev, long Ch)
VB	Function TWXA_DAStart(ByVal hDev As System.IntPtr, ByVal Ch As TWXA_DA_CH) As Integer
VBA	Function TWXA_DAStart(ByVal hDev As Long, ByVal Ch As TWXA_DA_CH) As Long
C#	STATUS DAStart(System.IntPtr hDev, DA_CH Ch)

hDev ： デバイスのハンドル

Ch　：設定を行うチャンネル。以下の値のいずれか

言語	値	説明
C/C++	TWXA_DA0	DA0 の周期出力を開始します。
C++	TWXA::RPORT::DA0	
VB/VBA	TWXA_RPORT.DA0	
C#	TWXA.RPORT.DA0	
C/C++	TWXA_DA1	DA1 の周期出力を開始します。
C++	TWXA::RPORT::DA1	
VB/VBA	TWXA_RPORT.DA1	
C#	TWXA.RPORT.DA1	
C/C++	TWXA_DA_ALL	全てのチャンネルの周期出力を開始します。
C++	TWXA::RPORT::DA_ALL	
VB/VBA	TWXA_RPORT.DA_ALL	
C#	TWXA.RPORT.DA_ALL	

アナログ出力の周期出力を開始します。

TWXA_DAStop() USB LAN I2424

言語	関数宣言
C/C++	long TWXA_DAStop(TW_HANDLE hDev, long Ch)
VB	Function TWXA_DAStop(ByVal hDev As System.IntPtr, ByVal Ch As TWXA_DA_CH) As Integer
VBA	Function TWXA_DAStop(ByVal hDev As Long, ByVal Ch As TWXA_DA_CH) As Long
C#	STATUS DAStop(System.IntPtr hDev, DA_CH Ch)

hDev　：デバイスのハンドル
Ch　：設定を行うチャンネル。以下の値のいずれか

言語	値	説明
C/C++	TWXA_DA0	DA0 の周期出力を停止します。
C++	TWXA::RPORT::DA0	
VB/VBA	TWXA_RPORT.DA0	
C#	TWXA.RPORT.DA0	
C/C++	TWXA_DA1	DA1 の周期出力を停止します。
C++	TWXA::RPORT::DA1	
VB/VBA	TWXA_RPORT.DA1	
C#	TWXA.RPORT.DA1	
C/C++	TWXA_DA_ALL	全てのチャンネルの周期出力を停止します。
C++	TWXA::RPORT::DA_ALL	
VB/VBA	TWXA_RPORT.DA_ALL	
C#	TWXA.RPORT.DA_ALL	

アナログ出力の周期出力を停止します。

TWXA_An16ToVolt() USB LAN I2219 A0800 I2424

言語	関数宣言
C/C++	double TWXA_An16ToVolt(long Data, long Opt)
VB	Function TWXA_An16ToVolt(ByVal Data As Integer, ByVal Opt As Integer) As Double Function TWXA_An16ToVolt(ByVal Data As Integer, ByVal Opt As TWXA_AN_OPTION) As Double
VBA	Function TWXA_An16ToVolt(ByVal Data As Long, ByVal Opt As Long) As Double
C#	double An16ToVolt(int Data) double An16ToVolt(int Data, uint Opt) double An16ToVolt(int Data, AN_OPTION Opt)

Data ： AD 変換で得られた値
Opt ： I2219 タイプの場合 ： 0 としてください
A0800 タイプ、I2424 タイプの場合 ： 設定した入力レンジ。以下の値のいずれか

言語	値	説明
C/C++	TWXA_AN_10VPP	入力レンジが 10Vpp(-5～+5V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_10VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_10VPP	
C#	TWXA.AN_OPTION.RANGE_10VPP	
C/C++	TWXA_AN_20VPP	入力レンジが 20Vpp(-10～+10V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_20VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_20VPP	
C#	TWXA.AN_OPTION.RANGE_20VPP	

戻り値 ： Data を電圧に換算した値(ボルト単位)

- I2219 タイプの場合

TWXA_ADRead()で得られた AD 変換の結果を、ボルト単位に変換する際に使用します。なお、Opt 引数は 0 としてください。

- A0800 タイプ、I2424 タイプの場合

TWXA_ADRead()、および、TWXA_ADReadBuffer()で得られた AD 変換の結果を、ボルト単位に変換する際に使用します。Opt 引数には TWXA_ADSetRange()で設定した入力レンジを指定します。

TWXA_AnToVolt() USB LAN I2219 A0800 I2424

言語	関数宣言
C/C++	Double TWXA_AnToVolt(long Data, long Bit, long Opt)
VB	Function TWXA_AnToVolt(ByVal Data As Integer, ByVal Bit As Integer, ByVal Opt As Integer) As Double Function TWXA_AnToVolt(ByVal Data As Integer, ByVal Bit As Integer, ByVal Opt As TWXA_AN_OPTION) As Double
VBA	Function TWXA_AnToVolt(ByVal Data As Long, ByVal Bit As Long, ByVal Opt As Long) As Double
C#	double AnToVolt(int Data, int Bit, int Opt) double AnToVolt(int Data, int Bit, AN_OPTION Opt)

Data ： AD 変換で得られた値、または、アナログ出力の出力値

Bit ： コンバータの分解能
I2219 タイプの場合 ： 8[bit](DA コンバータ)、または、10[bit](AD コンバータ)
A0800 タイプの場合 ： 16[bit]
I2424 タイプの場合 ： 10[bit](DA コンバータ)、または、16[bit](AD コンバータ)

Opt ： I2219 タイプの場合 ： 0 としてください
A0800 タイプの場合 ： 設定した入力レンジ。以下の値のいずれか

言語	値	説明
C/C++	TWXA_AN_10VPP	入力レンジが 10Vpp(-5～+5V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_10VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_10VPP	
C#	TWXA.AN_OPTION.RANGE_10VPP	
C/C++	TWXA_AN_20VPP	入力レンジが 20Vpp(-10～+10V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_20VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_20VPP	
C#	TWXA.AN_OPTION.RANGE_20VPP	

I2424 タイプの場合 ： 設定した入力レンジ、または、アナログ出力の電圧範囲。
以下の値のいずれか

言語	値	説明
C/C++	TWXA_AN_5VPP	アナログ出力の出力値を電圧値に変換する場合に指定します。
C++	TWXA::AN_OPTION::RANGE_5VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_5VPP	
C#	TWXA.AN_OPTION.RANGE_5VPP	
C/C++	TWXA_AN_10VPP	アナログ入力レンジが 10Vpp(-5～+5V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_10VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_10VPP	
C#	TWXA.AN_OPTION.RANGE_10VPP	
C/C++	TWXA_AN_20VPP	アナログ入力レンジが 20Vpp(-10～+10V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_20VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_20VPP	
C#	TWXA.AN_OPTION.RANGE_20VPP	

戻り値 ： Data を電圧に換算した値(ボルト単位)

- I2219 タイプの場合

TWXA_ADRead()で得られた AD 変換の結果をボルト単位に変換する場合、Bit 引数は 10、Opt 引数は 0 としてください。
アナログ出力の出力値をボルト単位に変換する場合、Bit 引数は 8、Opt 引数は 0 としてください。

- A0800 タイプの場合

TWXA_ADRead()、および、TWXA_ADReadBuffer()で得られた AD 変換の結果をボルト単位に変換する場合、Bit 引数は 16、Opt 引数には TWXA_ADSetRange()で設定した入力レンジを指定します。

- I2424 タイプの場合

TWXA_ADRead()、および、TWXA_ADReadBuffer()で得られた AD 変換の結果をボルト単位に変換する場合、Bit 引数は 16、Opt 引数には TWXA_ADSetRange()で設定した入力レンジを指定します。
アナログ出力の出力値をボルト単位に変換する場合、Bit 引数は 10、Opt 引数は TWXA_AN_5VPP(相当の値)としてください。

TWXA_An8FromVolt() USB LAN I2219

言語	関数宣言
C/C++	BYTE TWXA_An8FromVolt(double *pData, long Opt)
VB	Function TWXA_An8FromVolt(ByRef pData As Double, ByVal Opt As Integer) As Byte
VBA	Function TWXA_An8FromVolt(ByRef pData As Double, ByVal Opt As Long) As Byte
C#	byte An8FromVolt(ref double pData) byte An8FromVolt(ref double pData, uint Opt)

pData ： [入力]DA コンバータの設定値に変換したい電圧値(ボルト単位)
[出力]DA コンバータの精度で表現可能な電圧の理論値(ボルト単位)

Opt ： 予約。0 としてください

戻り値 ： pData を DA への書込み値に変換した値

ボルト単位の電圧値から DA コンバータに書き込む値を計算して返します。DA コンバータの精度で表現可能な理論値を pData に返します。

TWXA_AnFromVolt() USB LAN I2219 I2424

言語	関数宣言
C/C++	long TWXA_AnFromVolt(double *pData, long Bit, long Opt)
VB	Function TWXA_AnFromVolt(ByRef pData As Double, ByVal Bit As Integer, ByVal Opt As Integer) As Integer Function TWXA_AnFromVolt(ByRef pData As Double, ByVal Bit As Integer, ByVal Opt As TWXA_AN_OPTION) As Integer
VBA	Function TWXA_AnFromVolt(ByRef pData As Double, ByVal Bit As Long, ByVal Opt As Long) As Long
C#	int AnFromVolt(ref double pData, int Bit, int Opt) int AnFromVolt(ref double pData, int Bit, AN_OPTION Opt)

pData ： [入力]DA コンバータの設定値に変換したい電圧値、または、ハードウェアイベントで使用する AD コンバータの比較値(ボルト単位)
[出力]DA コンバータ、または、AD コンバータの精度で表現可能な電圧の理論値(ボルト単位)

Bit ： コンバータの分解能
I2219 タイプの場合 ： 8[bit](DA コンバータ)、または、10[bit](AD コンバータ)
I2424 タイプの場合 ： 10[bit](DA コンバータ)、または、16[bit](AD コンバータ)

Opt ： I2219 タイプの場合 ： 0 としてください
I2424 タイプの場合 ： 設定した入力レンジ、または、アナログ出力の電圧範囲。
以下の値のいずれか

言語	値	説明
C/C++	TWXA_AN_5VPP	アナログ出力の電圧値を設定値に変換する場合に指定します。
C++	TWXA::AN_OPTION::RANGE_5VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_5VPP	
C#	TWXA.AN_OPTION.RANGE_5VPP	
C/C++	TWXA_AN_10VPP	アナログ入力レンジが 10Vpp(-5～+5V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_10VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_10VPP	
C#	TWXA.AN_OPTION.RANGE_10VPP	

言語	値	説明
C/C++	TWXA_AN_20VPP	アナログ入力レンジが 20Vpp(-10～+10V)の場合に指定します。
C++	TWXA::AN_OPTION::RANGE_20VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_20VPP	
C#	TWXA.AN_OPTION.RANGE_20VPP	

戻り値 : pData を DA コンバータへの設定値、または、AD コンバータの比較値に変換した値

ボルト単位の電圧値から DA コンバータへの設定値、または、ハードウェアイベントで使用する AD コンバータの比較値を計算して返します。また、コンバータの精度で表現可能な理論値を pData に返します。

□ パルスカウンタ(ソフトウェアカウンタ)操作関数

TWXA_PCSetMode() USB LAN I2219 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PCSetMode(TW_HANDLE hDev, long ChBits, long Mode)
VB	Function TWXA_PCSetMode(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_PC_BITS, ByVal Mode As TWXA_PC_MODE) As Integer
VBA	Function TWXA_PCSetMode(ByVal hDev As Long, ByVal ChBits As TWXA_PC_BITS, ByVal Mode As TWXA_PC_MODE) As Long
C#	STATUS PCSetMode(System.IntPtr hDev, PC_BITS ChBits, PC_MODE Mode)

hDev : デバイスのハンドル
ChBits : 設定チャンネル
I2219 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 の動作を設定します。Mode 指定が 2 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 の動作を設定します。Mode 指定が 2 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 の動作を設定します。Mode 指定が 2 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 の動作を設定します。Mode 指定が 2 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1の動作を設定します。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3の動作を設定します。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC_ALL	全てのチャンネルの動作を設定します。Mode 指定が 2 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC4	チャンネル 4 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC4	
VB/VBA	TWXA_PC_BITS.PC4	
C#	TWXA.PC_BITS.PC4	
C/C++	TWXA_PC5	チャンネル 5 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC5	
VB/VBA	TWXA_PC_BITS.PC5	
C#	TWXA.PC_BITS.PC5	
C/C++	TWXA_PC6	チャンネル 6 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC6	
VB/VBA	TWXA_PC_BITS.PC6	
C#	TWXA.PC_BITS.PC6	
C/C++	TWXA_PC7	チャンネル 7 の動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC7	
VB/VBA	TWXA_PC_BITS.PC7	
C#	TWXA.PC_BITS.PC7	
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1の動作を設定します。Mode 指定が 3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3の動作を設定します。Mode 指定が 3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC4_PC5	チャンネル4とチャンネル5の動作を設定します。Mode 指定が 3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC4_PC5	
VB/VBA	TWXA_PC_BITS.PC4_PC5	
C#	TWXA.PC_BITS.PC4_PC5	
C/C++	TWXA_PC6_PC7	チャンネル6とチャンネル7の動作を設定します。Mode 指定が 3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC6_PC7	
C#	TWXA.PC_BITS.PC6_PC7	

言語	値	説明
C/C++	TWXA_PC0_PC2_PC3	チャンネル 0、チャンネル 2、チャンネル 3 の動作を設定します。Mode 指定が 3 相カウントの場合指定可能です。
C++	TWXA::PC_BITS::PC0_PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC0_PC2_PC3	
C#	TWXA.PC_BITS.PC0_PC2_PC3	
C/C++	TWXA_PC4_PC6_PC7	チャンネル 4、チャンネル 6、チャンネル 7 の動作を設定します。Mode 指定が 3 相カウントの場合指定可能です。
C++	TWXA::PC_BITS::PC4_PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC4_PC6_PC7	
C#	TWXA.PC_BITS.PC4_PC6_PC7	
C/C++	TWXA_PC_ALL	全てのチャンネルの動作を設定します。Mode 指定が 2 相カウント、および、3 相カウント以外の場合指定可能です。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

Mode　：カウントモード

I2219 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC_OFF_TO_ON	入力が OFF から ON に変化したときにカウントします。
C++	TWXA::PC_MODE::COUNT_OFF_TO_ON	
VB/VBA	TWXA_PC_MODE.COUNT_OFF_TO_ON	
C#	TWXA.PC_MODE.COUNT_OFF_TO_ON	
C/C++	TWXA_PC_ON_TO_OFF	入力が ON から OFF に変化したときにカウントします。
C++	TWXA::PC_MODE::COUNT_ON_TO_OFF	
VB/VBA	TWXA_PC_MODE.COUNT_ON_TO_OFF	
C#	TWXA.PC_MODE.COUNT_ON_TO_OFF	
C/C++	TWXA_PC_2PHASE	90° 位相差の A 相､B 相の 2 相信号をカウントするモードです。
C++	TWXA::PC_MODE::COUNT_2PHASE	
VB/VBA	TWXA_PC_MODE.COUNT_2PHASE	
C#	TWXA.PC_MODE.COUNT_2PHASE	
C/C++	TWXA_PC_3PHASE	チャンネル 2 とチャンネル 3 で 2 相信号のカウントを行い、チャンネル 0 でカウンタをクリアします。ChBits の値は無視されます。
C++	TWXA::PC_MODE::COUNT_3PHASE	
VB/VBA	TWXA_PC_MODE.COUNT_3PHASE	
C#	TWXA.PC_MODE.COUNT_3PHASE	

I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC_OFF_TO_ON	入力が OFF から ON に変化したときにカウントします。
C++	TWXA::PC_MODE::COUNT_OFF_TO_ON	
VB/VBA	TWXA_PC_MODE.COUNT_OFF_TO_ON	
C#	TWXA.PC_MODE.COUNT_OFF_TO_ON	
C/C++	TWXA_PC_ON_TO_OFF	入力が ON から OFF に変化したときにカウントします。
C++	TWXA::PC_MODE::COUNT_ON_TO_OFF	
VB/VBA	TWXA_PC_MODE.COUNT_ON_TO_OFF	
C#	TWXA.PC_MODE.COUNT_ON_TO_OFF	
C/C++	TWXA_PC_BOTH	入力が変化したときにカウントします。
C++	TWXA::PC_MODE::COUNT_BOTH	
VB/VBA	TWXA_PC_MODE.COUNT_BOTH	
C#	TWXA.PC_MODE.COUNT_BOTH	

言語	値	説明
C/C++	TWXA_PC_2PHASE	90° 位相差の A 相､B 相の 2 相信号をカウントするモードです。
C++	TWXA::PC_MODE::COUNT_2PHASE	
VB/VBA	TWXA_PC_MODE.COUNT_2PHASE	
C#	TWXA.PC_MODE.COUNT_2PHASE	
C/C++	TWXA_PC_3PHASE	90° 位相差の A 相、B 相の 2 相信号に、Z 相によるカウンタクリアを追加した 3 相信号をカウントするモードです。
C++	TWXA::PC_MODE::COUNT_3PHASE	
VB/VBA	TWXA_PC_MODE.COUNT_3PHASE	
C#	TWXA.PC_MODE.COUNT_3PHASE	

パルスカウンタの指定チャンネルのカウントモードを設定します。

- I2219 タイプの場合

PC0 と PC1 で 2 相カウントする場合、PC0、PC1 両方に A 相信号、Ib6 に B 相信号を入力します。
PC2 と PC3 で 2 相カウントする場合、PC2、PC3 両方に A 相信号、Ib7 に B 相信号を入力します。
Mode に TWXA_PC_3PHASE を指定すると、PC0 がカウンタクリア、PC2 と PC3 で 2 相カウントする設定になります。PC2、PC3 に A 相、Ib7 に B 相信号を入力し、PC0 に Z 相信号を入力すると 1 周毎に PC2 と PC3 のカウンタがクリアされ、PC0 のカウンタがカウントアップします。

- I2424 タイプの場合

チャンネル 0 とチャンネル 1 で 2 相カウントする場合、チャンネル 0(SC0)に A 相信号、チャンネル 1(SC1)に B 相信号を入力します。
チャンネル 2 とチャンネル 3 で 2 相カウントする場合、チャンネル 2(SC2)に A 相信号、チャンネル 3(SC3)に B 相信号を入力します。
チャンネル 4 とチャンネル 5 で 2 相カウントする場合、チャンネル 4(SC4)に A 相信号、チャンネル 5(SC5)に B 相信号を入力します。
チャンネル 6 とチャンネル 7 で 2 相カウントする場合、チャンネル 6(SC6)に A 相信号、チャンネル 7(SC7)に B 相信号を入力します。
チャンネル 0、チャンネル 2 とチャンネル 3 で 3 相信号をカウントする場合、チャンネル 2(SC2)に A 相信号、チャンネル 3(SC3)に B 相信号、チャンネル 0(SC0)に Z 相信号を入力します。チャンネル 2 とチャンネル 3 で 2 相信号をカウントし、チャンネル 0 で 2 相信号のカウントをクリアします。また、チャンネル 0 のカウンタは 2 相信号のカウントクリア時にカウントアップします。
チャンネル 4、チャンネル 6 とチャンネル 7 で 3 相信号をカウントする場合、チャンネル 6(SC6)に A 相信号、チャンネル 7(SC7)に B 相信号、チャンネル 4(SC4)に Z 相信号を入力します。チャンネル 6 とチャンネル 7 で 2 相信号をカウントし、チャンネル 4 で 2 相信号のカウントをクリアします。また、チャンネル 4 のカウンタは 2 相信号のカウントクリア時にカウントアップします。

TWXA_PCStart() USB LAN I2219 I0800 I0404 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PCStart(TW_HANDLE hDev, long ChBits)
VB	Function TWXA_PCStart(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_PC_BITS) As Integer
VBA	Function TWXA_PCStart(ByVal hDev As Long, ByVal ChBits As TWXA_PC_BITS) As Long
C#	STATUS PCStart(System.IntPtr hDev, PC_BITS ChBits)

hDev　: デバイスのハンドル
ChBits : パルスカウンタのチャンネル

I2219 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウントを開始します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウントを開始します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウントを開始します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウントを開始します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1のカウントを開始します。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3のカウントを開始します。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウントを開始します。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

I0800 タイプ、I0404 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウントを開始します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウントを開始します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウントを開始します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウントを開始します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	

言語	値	説明
C/C++	TWXA_PC_ALL	全てのチャンネルのカウントを開始します。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

I2424 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウントを開始します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウントを開始します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウントを開始します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウントを開始します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC4	チャンネル 4 のカウントを開始します。
C++	TWXA::PC_BITS::PC4	
VB/VBA	TWXA_PC_BITS.PC4	
C#	TWXA.PC_BITS.PC4	
C/C++	TWXA_PC5	チャンネル 5 のカウントを開始します。
C++	TWXA::PC_BITS::PC5	
VB/VBA	TWXA_PC_BITS.PC5	
C#	TWXA.PC_BITS.PC5	
C/C++	TWXA_PC6	チャンネル 6 のカウントを開始します。
C++	TWXA::PC_BITS::PC6	
VB/VBA	TWXA_PC_BITS.PC6	
C#	TWXA.PC_BITS.PC6	
C/C++	TWXA_PC7	チャンネル 7 のカウントを開始します。
C++	TWXA::PC_BITS::PC7	
VB/VBA	TWXA_PC_BITS.PC7	
C#	TWXA.PC_BITS.PC7	
C/C++	TWXA_PC0_PC1	チャンネル 0 とチャンネル 1 のカウントを開始します。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル 2 とチャンネル 3 のカウントを開始します。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC4_PC5	チャンネル 4 とチャンネル 5 のカウントを開始します。
C++	TWXA::PC_BITS::PC4_PC5	
VB/VBA	TWXA_PC_BITS.PC4_PC5	
C#	TWXA.PC_BITS.PC4_PC5	

言語	値	説明
C/C++	TWXA_PC6_PC7	チャンネル 6 とチャンネル 7 のカウントを開始します。
C++	TWXA::PC_BITS::PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC6_PC7	
C#	TWXA.PC_BITS.PC6_PC7	
C/C++	TWXA_PC0_PC2_PC3	チャンネル 0,、チャンネル 2、チャンネル 3 のカウントを開始します。
C++	TWXA::PC_BITS::PC0_PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC0_PC2_PC3	
C#	TWXA.PC_BITS.PC0_PC2_PC3	
C/C++	TWXA_PC4_PC6_PC7	チャンネル 4、チャンネル 6、チャンネル 7 のカウントを開始します。
C++	TWXA::PC_BITS::PC4_PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC4_PC6_PC7	
C#	TWXA.PC_BITS.PC4_PC6_PC7	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウントを開始します。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

指定チャンネルのパルスカウンタのカウントをスタートさせます。

TWXA_PCStop() USB LAN I2219 I0800 I0404 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PCStop(TW_HANDLE hDev, long ChBits)
VB	Function TWXA_PCStop(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_PC_BITS) As Integer
VBA	Function TWXA_PCStop(ByVal hDev As Long, ByVal ChBits As TWXA_PC_BITS) As Long
C#	STATUS PCStop(System.IntPtr hDev, PC_BITS ChBits)

hDev　：デバイスのハンドル
ChBits：パルスカウンタのチャンネル
　　I2219 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウントを停止します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウントを停止します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウントを停止します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウントを停止します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	

言語	値	説明
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1のカウントを停止します。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3のカウントを停止します。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウントを停止します。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

I0800 タイプ、I0404 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウントを停止します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウントを停止します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウントを停止します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウントを停止します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウントを停止します。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

I2424 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウントを停止します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウントを停止します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウントを停止します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	

言語	値	説明
C/C++	TWXA_PC3	チャンネル 3 のカウントを停止します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC4	チャンネル 4 のカウントを停止します。
C++	TWXA::PC_BITS::PC4	
VB/VBA	TWXA_PC_BITS.PC4	
C#	TWXA.PC_BITS.PC4	
C/C++	TWXA_PC5	チャンネル 5 のカウントを停止します。
C++	TWXA::PC_BITS::PC5	
VB/VBA	TWXA_PC_BITS.PC5	
C#	TWXA.PC_BITS.PC5	
C/C++	TWXA_PC6	チャンネル 6 のカウントを停止します。
C++	TWXA::PC_BITS::PC6	
VB/VBA	TWXA_PC_BITS.PC6	
C#	TWXA.PC_BITS.PC6	
C/C++	TWXA_PC7	チャンネル 7 のカウントを停止します。
C++	TWXA::PC_BITS::PC7	
VB/VBA	TWXA_PC_BITS.PC7	
C#	TWXA.PC_BITS.PC7	
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1のカウントを停止します。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3のカウントを停止します。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC4_PC5	チャンネル4とチャンネル5のカウントを停止します。
C++	TWXA::PC_BITS::PC4_PC5	
VB/VBA	TWXA_PC_BITS.PC4_PC5	
C#	TWXA.PC_BITS.PC4_PC5	
C/C++	TWXA_PC6_PC7	チャンネル6とチャンネル7のカウントを停止します。
C++	TWXA::PC_BITS::PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC6_PC7	
C#	TWXA.PC_BITS.PC6_PC7	
C/C++	TWXA_PC0_PC2_PC3	チャンネル 0,、チャンネル 2、チャンネル 3 のカウントを停止します。
C++	TWXA::PC_BITS::PC0_PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC0_PC2_PC3	
C#	TWXA.PC_BITS.PC0_PC2_PC3	
C/C++	TWXA_PC4_PC6_PC7	チャンネル 4、チャンネル 6、チャンネル 7 のカウントを停止します。
C++	TWXA::PC_BITS::PC4_PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC4_PC6_PC7	
C#	TWXA.PC_BITS.PC4_PC6_PC7	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウントを停止します。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

指定チャンネルのパルスカウンタのカウントを停止します。

TWXA_PCReadCnt() USB LAN I2219 I0800 I0404 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PCReadCnt(TW_HANDLE hDev, long ChBits, long *pCnt)
VB	Function TWXA_PCReadCnt(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_PC_BITS, ByRef pCnt As Integer) As Integer Function TWXA_PCReadCnt(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_PC_BITS, ByVal pCnt() As Integer) As Integer
VBA	Function TWXA_PCReadCnt(ByVal hDev As Long, ByVal ChBits As TWXA_PC_BITS, ByRef pCnt As Long) As Long
C#	STATUS PCReadCnt(System.IntPtr hDev, PC_BITS ChBits, out int pCnt) STATUS PCReadCnt(System.IntPtr hDev, PC_BITS ChBits, out uint pCnt) STATUS PCReadCnt(System.IntPtr hDev, PC_BITS ChBits, int []pCnt) STATUS PCReadCnt(System.IntPtr hDev, PC_BITS ChBits, uint []pCnt)

hDev　：デバイスのハンドル
ChBits：パルスカウンタのチャンネル
　I2219 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1のカウント値の合計を読み出します。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3のカウント値の合計を読み出します。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウント値を同時に読み出します。pCnt には 4 チャンネル分の領域が必要です。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

I0800 タイプ、I0404 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウント値を同時に読み出します。pCnt には 4 チャンネル分の領域が必要です。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC4	チャンネル 4 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC4	
VB/VBA	TWXA_PC_BITS.PC4	
C#	TWXA.PC_BITS.PC4	
C/C++	TWXA_PC5	チャンネル 5 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC5	
VB/VBA	TWXA_PC_BITS.PC5	
C#	TWXA.PC_BITS.PC5	

言語	値	説明
C/C++	TWXA_PC6	チャンネル 6 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC6	
VB/VBA	TWXA_PC_BITS.PC6	
C#	TWXA.PC_BITS.PC6	
C/C++	TWXA_PC7	チャンネル 7 のカウント値を読み出します。
C++	TWXA::PC_BITS::PC7	
VB/VBA	TWXA_PC_BITS.PC7	
C#	TWXA.PC_BITS.PC7	
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1のカウント値の合計を読み出します。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3のカウント値の合計を読み出します。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	
C/C++	TWXA_PC4_PC5	チャンネル4とチャンネル5のカウント値の合計を読み出します。
C++	TWXA::PC_BITS::PC4_PC5	
VB/VBA	TWXA_PC_BITS.PC4_PC5	
C#	TWXA.PC_BITS.PC4_PC5	
C/C++	TWXA_PC6_PC7	チャンネル6とチャンネル7のカウント値の合計を読み出します。
C++	TWXA::PC_BITS::PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC6_PC7	
C#	TWXA.PC_BITS.PC6_PC7	
C/C++	TWXA_PC_ALL	全てのチャンネルのカウント値を同時に読み出します。pCnt には 4、または、8 チャンネル分の領域が必要です。
C++	TWXA::PC_BITS::PC_ALL	
VB/VBA	TWXA_PC_BITS.PC_ALL	
C#	TWXA.PC_BITS.PC_ALL	

pCnt　：[出力]カウント値の格納先

指定チャンネルのパルスカウンタの値を読み出します。

- I2219 タイプの場合

TWXA_PC0_PC1、TWXA_PC2_PC3 を指定した場合、それぞれ、PC0 と PC1 のカウンタ値の合計、PC2 と PC3 のカウンタ値の合計を返します。
TWXA_PC_ALL を指定した場合 pCnt に全てのチャンネルの値が連続して格納されますので pCnt には 4 チャンネル分のデータを格納できる領域(4 x 4 = 16 バイト)が必要です。

- I0800 タイプ、I0404 タイプの場合

TWXA_PC_ALL を指定した場合 pCnt に全てのチャンネルの値が連続して格納されますので pCnt には 4 チャンネル分のデータを格納できる領域(4 x 4 = 16 バイト)が必要です。

- I2424 タイプの場合

TWXA_PC0_PC1、TWXA_PC2_PC3、TWXA_PC4_PC5、TWXA_PC6_PC7 を指定した場合、それぞれ、チャンネル 0 とチャンネル 1 のカウンタ値の合計、チャンネル 2 とチャンネル 3 のカウンタ値の合計、チャンネル 4 とチャンネル 5 のカウンタ値の合計、チャンネル 6 とチャンネル 7 のカウンタ値の合計を返します。
TWXA_PC_ALL を指定した場合 pCnt に全てのチャンネルの値が連続して格納されますので pCnt には 8 チャンネル分のデータを格納できる領域(4 x 8 = 32 バイト)が必要です。

TWXA_PCSetCnt() USB LAN I2219 I0800 I0404 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PCSetCnt(TW_HANDLE hDev, long ChBits, long Cnt)
VB	Function TWXA_PCSetCnt(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_PC_BITS, ByVal Cnt As Integer) As Integer
VBA	Function TWXA_PCSetCnt(ByVal hDev As Long, ByVal ChBits As TWXA_PC_BITS, ByVal Cnt As Long) As Long
C#	STATUS PCSetCnt(System.IntPtr hDev, PC_BITS ChBits, int Cnt) STATUS PCSetCnt(System.IntPtr hDev, PC_BITS ChBits, uint Cnt)

hDev　：デバイスのハンドル
ChBits：パルスカウンタのチャンネル
　I2219 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC0_PC1	チャンネル 0 とチャンネル 1 のカウンタに合計が Cnt となるように値をセットします。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル 2 とチャンネル 3 のカウンタに合計が Cnt となるように値をセットします。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	

　I0800 タイプ、I0404 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	

言語	値	説明
C/C++	TWXA_PC2	チャンネル 2 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	

I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_PC0	チャンネル 0 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC0	
VB/VBA	TWXA_PC_BITS.PC0	
C#	TWXA.PC_BITS.PC0	
C/C++	TWXA_PC1	チャンネル 1 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC1	
VB/VBA	TWXA_PC_BITS.PC1	
C#	TWXA.PC_BITS.PC1	
C/C++	TWXA_PC2	チャンネル 2 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC2	
VB/VBA	TWXA_PC_BITS.PC2	
C#	TWXA.PC_BITS.PC2	
C/C++	TWXA_PC3	チャンネル 3 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC3	
VB/VBA	TWXA_PC_BITS.PC3	
C#	TWXA.PC_BITS.PC3	
C/C++	TWXA_PC4	チャンネル 4 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC4	
VB/VBA	TWXA_PC_BITS.PC4	
C#	TWXA.PC_BITS.PC4	
C/C++	TWXA_PC5	チャンネル 5 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC5	
VB/VBA	TWXA_PC_BITS.PC5	
C#	TWXA.PC_BITS.PC5	
C/C++	TWXA_PC6	チャンネル 6 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC6	
VB/VBA	TWXA_PC_BITS.PC6	
C#	TWXA.PC_BITS.PC6	
C/C++	TWXA_PC7	チャンネル 7 のカウンタに値をセットします。
C++	TWXA::PC_BITS::PC7	
VB/VBA	TWXA_PC_BITS.PC7	
C#	TWXA.PC_BITS.PC7	
C/C++	TWXA_PC0_PC1	チャンネル0とチャンネル1のカウンタに合計が Cnt となるように値をセットします。
C++	TWXA::PC_BITS::PC0_PC1	
VB/VBA	TWXA_PC_BITS.PC0_PC1	
C#	TWXA.PC_BITS.PC0_PC1	
C/C++	TWXA_PC2_PC3	チャンネル2とチャンネル3のカウンタに合計が Cnt となるように値をセットします。
C++	TWXA::PC_BITS::PC2_PC3	
VB/VBA	TWXA_PC_BITS.PC2_PC3	
C#	TWXA.PC_BITS.PC2_PC3	

言語	値	説明
C/C++	TWXA_PC4_PC5	チャンネル4とチャンネル5のカウンタに合計が Cnt となるように値をセットします。
C++	TWXA::PC_BITS::PC4_PC5	
VB/VBA	TWXA_PC_BITS.PC4_PC5	
C#	TWXA.PC_BITS.PC4_PC5	
C/C++	TWXA_PC6_PC7	チャンネル6とチャンネル7のカウンタに合計が Cnt となるように値をセットします。
C++	TWXA::PC_BITS::PC6_PC7	
VB/VBA	TWXA_PC_BITS.PC6_PC7	
C#	TWXA.PC_BITS.PC6_PC7	

Cnt　：セットする値

指定チャンネルのパルスカウンタに値をセットします。カウンタをクリアする場合は Cnt を 0 として呼び出します。

- I2219 タイプ、I0800 タイプ、I0404 タイプの場合

ChBits が TWXA_PC0_PC1、TWXA_PC2_PC3(相当の値)の場合、2 つのカウンタの合計値が Cnt となるように調整され 2 つのカウンタに値がセットされます。

- I2424 タイプの場合

ChBits が TWXA_PC0_PC1、TWXA_PC2_PC3、TWXA_PC4_PC5、または、TWXA_PC6_PC7(相当の値)の場合、2 つのカウンタの合計値が Cnt となるように調整され 2 つのカウンタに値がセットされます。

□ 16 ビットタイマ（ハードウェアカウンタ、PWM）操作関数

TWXA_TimerSetMode() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerSetMode(TW_HANDLE hDev, long Ch, long Mode)
VB	Function TWXA_TimerSetMode(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal Mode As TWXA_TIMER_MODE) As Integer
VBA	Function TWXA_TimerSetMode(ByVal hDev As Long, ByVal Ch As Long, ByVal Mode As TWXA_TIMER_MODE) As Long
C#	STATUS TimerSetMode(System.IntPtr hDev, int Ch, TIMER_MODE Mode)

hDev ： デバイスのハンドル
Ch　： タイマチャンネル
I2219 タイプの場合 ： 0～2
I0404 タイプ、I0008 タイプの場合 ： 0～1
I2424 タイプの場合、以下を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 の動作を設定します。Mode 指定が PWM モード、および、2 相カウントモードの場合は指定できません。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 の動作を設定します。Mode 指定が PWM モード、および、2 相カウントモードの場合は指定できません。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の動作を設定します。Mode 指定が 2 相カウントモードの場合は指定できません。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BIT3	チャンネル 3 の動作を設定します。Mode 指定が 2 相カウントモードの場合は指定できません。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 の動作を設定します。全てのモードで指定可能です。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	
C/C++	TWXA_TIMER_BIT5	チャンネル 5 の動作を設定します。Mode 指定が PWM モード、および、2 相カウントモードの場合は指定できません。
C++	TWXA::TIMER_BITS::TIMER5	
VB/VBA	TWXA_TIMER_BITS.TIMER5	
C#	TWXA.TIMER_BITS.TIMER5	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 の動作を設定します。全てのモードで指定可能です。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 の動作を設定します。Mode 指定が 2 相カウントモードの場合は指定できません。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	

言語	値	説明
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルの動作を設定します。Mode 指定が PWM モード、および、2 相カウントモードの場合は指定できません。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

Mode : 動作モード

I2219 タイプ、I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_DISABLE	PWM モード、および、カウンタ機能を解除する場合に指定します。PWM 端子はデジタル出力端子として、カウンタ端子はデジタル入力端子として使用可能になります。
C++	TWXA::TIMER_MODE::DISABLE	
VB/VBA	TWXA_TIMER_MODE.DISABLE	
C#	TWXA.TIMER_MODE.DISABLE	
C/C++	TWXA_TIMER_PWM	指定チャンネルを PWM モードに設定します。対応する端子は PWM 出力用となります。
C++	TWXA::TIMER_MODE::PWM	
VB/VBA	TWXA_TIMER_MODE.PWM	
C#	TWXA.TIMER_MODE.PWM	
C/C++	TWXA_TIMER_OFF_TO_ON	指定チャンネルを単相パルスカウントモードとし、対応する入力が OFF から ON に変化したときカウントします。
C++	TWXA::TIMER_MODE::COUNT_OFF_TO_ON	
VB/VBA	TWXA_TIMER_MODE.COUNT_OFF_TO_ON	
C#	TWXA.TIMER_MODE.COUNT_OFF_TO_ON	
C/C++	TWXA_TIMER_ON_TO_OFF	指定チャンネルを単相パルスカウントモードとし、対応する入力が ON から OFF に変化したときカウントします。
C++	TWXA::TIMER_MODE::COUNT_ON_TO_OFF	
VB/VBA	TWXA_TIMER_MODE.COUNT_ON_TO_OFF	
C#	TWXA.TIMER_MODE.COUNT_ON_TO_OFF	
C/C++	TWXA_TIMER_BOTH	指定チャンネルを単相パルスカウントモードとし、極性によらず対応する入力が変化したときにカウントします。
C++	TWXA::TIMER_MODE::COUNT_BOTH	
VB/VBA	TWXA_TIMER_MODE.COUNT_BOTH	
C#	TWXA.TIMER_MODE.COUNT_BOTH	
C/C++	TWXA_TIMER_2PHASE	90° 位相差の A 相、B 相の 2 相信号をカウントします。
C++	TWXA::TIMER_MODE::COUNT_2PHASE	
VB/VBA	TWXA_TIMER_MODE.COUNT_2PHASE	
C#	TWXA.TIMER_MODE.COUNT_2PHASE	

I0404 タイプ、I0008 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_DISABLE	PWM モードを解除する場合に指定します。PWM 端子がデジタル出力端子として使用可能になります。
C++	TWXA::TIMER_MODE::DISABLE	
VB/VBA	TWXA_TIMER_MODE.DISABLE	
C#	TWXA.TIMER_MODE.DISABLE	
C/C++	TWXA_TIMER_PWM	指定チャンネルを PWM モードに設定します。対応する端子は PWM 出力用となります。
C++	TWXA::TIMER_MODE::PWM	
VB/VBA	TWXA_TIMER_MODE.PWM	
C#	TWXA.TIMER_MODE.PWM	

16 ビットタイマの動作モードを変更します。

- I2219 タイプの場合

PWM モードに設定すると、チャンネルに対応する端子が PWM 出力用端子となりデジタル出力端子としての制御ができなくなります。デジタル出力端子に戻す場合は Mode 引数に TWXA_TIMER_DISABLE(または相当する定数)を指定して呼び出してください。

単相のパルスカウントモードはチャンネル 1 と 2 のみ指定可能です。それぞれ CLK1、CLK2 の入力パルスを

カウントします。2 相のパルスカウントモードはチャンネル 2 のみ指定可能です。CLK1 に B 相、CLK2 に A 相を接続して使用します。

- I0404 タイプ、I0008 タイプの場合

PWM モードに設定すると、チャンネルに対応する端子が PWM 出力用端子となりデジタル出力端子としての制御ができなくなります。デジタル出力端子に戻す場合は Mode 引数に TWXA_TIMER_DISABLE(または相当する定数)を指定して呼び出してください。

- I2424 タイプの場合

PWM モードに設定すると、チャンネルに対応する端子が PWM 出力用端子となりデジタル出力端子としての制御ができなくなります。デジタル出力端子に戻す場合は Mode 引数に TWXA_TIMER_DISABLE(または相当する定数)を指定して呼び出してください。PWM モードはチャンネル 2、3、4、6、7 が指定可能です。
単相のパルスカウントモードは全てのチャンネルで指定可能です。それぞれ HC0～HC7 の入力パルスをカウントします。2 相のパルスカウントモードはチャンネル 4 と 6 のみ指定可能です。チャンネル 4 を 2 相パルスカウントモードで使用する場合、HC4 に A 相、HC5 に B 相を接続します。この場合、チャンネル 5 は単相パルスカウントモードで使用できなくなります。チャンネル 6 を 2 相パルスカウントモードで使用する場合、HC6 に A 相、HC7 に B 相を接続します。この場合、チャンネル 7 は単相パルスカウントモード、および、PWM モードで使用できなくなります。

TWXA_TimerSetPwm() USB LAN I2219 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerSetPwm(TW_HANDLE hDev, long Ch, double *pFrequency, double *pDuty, double *pPhase)
VB	Function TWXA_TimerSetPwm(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByRef pFrequency As Double, ByRef pDuty As Double, ByRef pPhase As Double) As Integer
VBA	Function TWXA_TimerSetPwm(ByVal hDev As Long, ByVal Ch As Long, ByRef pFrequency As Double, ByRef pDuty As Double, ByRef pPhase As Double) As Long
C#	STATUS TimerSetPwm(System.IntPtr hDev, int Ch, ref double pFrequency, ref double pDuty, ref double pPhase)

hDev　　: デバイスのハンドル
Ch　　: 設定する 16 ビットタイマのチャンネル
I2219 タイプの場合 : 0～2
I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BIT3	チャンネル 3 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	

言語	値	説明
C/C++	TWXA_TIMER_BIT6	チャンネル 6 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	

pFrequency ： [入力]希望の周波数(Hz 単位)
I2219 タイプの場合 ： 約 48(=3,125,000 / 65,535)～1,000,000[Hz]
I2424 タイプの場合 ： 1～1,000,000[Hz](チャンネル 6 は 3～1,000,000[Hz])
[出力]実際に設定できた周波数(Hz 単位)

pDuty ： [入力]希望のデューティ [出力]実際に設定できたデューティ(0-1.0)
pPhase ： [入力]希望の初期位相 [出力]実際に設定できた初期位相(0-1.0)

PWM 出力の設定を行います。
デューティ(pDuty)は 0-1.0 の範囲で ON デューティを指定します。例えば 0.3 を指定した場合、周期の約 30%が ON 出力になります。
初期位相(pPhase)は該当のタイマチャンネルが停止中のみ変更可能です。0-1.0 の範囲で指定します。1.0 は 360°、0.5 は 180°に相当します。
波形は製品の内部クロックを分周して作られるため、周波数、デューティ、初期位相の各パラメータは設定できる値が離散的になります。そのため、引数に与えた希望値と設定可能な値が異なる場合があります。関数は各引数に実際に設定できた値をセットして戻ります。

TWXA_TimerSetPwmExt() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerSetPwmExt(TW_HANDLE hDev, long Ch, double dClkFreq, double *pFrequency, double *pDuty, double *pPhase)
VB	Function TWXA_TimerSetPwmExt(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal dClkFreq As Double, ByRef pFrequency As Double, ByRef pDuty As Double, ByRef pPhase As Double) As Integer
VBA	Function TWXA_TimerSetPwmExt(ByVal hDev As Long, ByVal Ch As Long, ByVal dClkFreq As Double, ByRef pFrequency As Double, ByRef pDuty As Double, ByRef pPhase As Double) As Long
C#	STATUS TimerSetPwmExt(System.IntPtr hDev, int Ch, double dClkFreq, ref double pFrequency, ref double pDuty, ref double pPhase)

hDev ： デバイスのハンドル
Ch ： 設定する 16 ビットタイマのチャンネル
I2219 タイプの場合 ： 0～2
I0404 タイプ、I0008 タイプの場合 ： 0～1
I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	

言語	値	説明
C/C++	TWXA_TIMER_BIT3	チャンネル 3 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 の動作を設定します。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	

dClkFreq ： 外部クロックの周波数(Hz 単位)
I2219 タイプ、I2424 タイプの場合 ： 最大 5,000,000[Hz]
I0404 タイプ、I0008 タイプの場合 ： 1,000[Hz]としてください

pFrequency ： [入力]希望の周波数(最大 dClkFreq / 2[Hz])
I2219 タイプ、I2424 タイプの場合 ： dClkFreq / 65,535～dClkFreq / 2[Hz]
I0404 タイプ、I0008 タイプの場合 ： 約 0.015(=1,000 / 65,535)～50[Hz]
[出力]実際に設定できた周波数(Hz 単位)

pDuty ： [入力]希望のデューティ [出力]実際に設定できたデューティ(0-1.0)

pPhase ： [入力]希望の初期位相 [出力]実際に設定できた初期位相(0-1.0)

- I2219 タイプの場合

基準となるクロックとして外部入力を使用する点を除いて TWXA_TimerSetPwm()関数と同様です。
内部クロックを用いた場合、出力できる周波数の下限が約 50Hz となりますので、それより低い周波数を出力する場合に使用してください。
外部クロックは CLK1 に入力します。別の機器からの出力信号を用いることもできますが、他のチャンネルの PWM 出力を入力することも可能です。

- I0404 タイプ、I0008 タイプの場合

PWM 出力の設定を行います。
dClkFreq は常に 1,000 にしてください。
デューティ(pDuty)は 0-1.0 の範囲で ON デューティを指定します。例えば 0.3 を指定した場合、周期の約 30%が ON 出力になります。
初期位相(pPhase)は該当のタイマチャンネルが停止中のみ変更可能です。0-1.0 の範囲で指定します。1.0 は 360°、0.5 は 180°に相当します。
波形は製品の内部クロック(1,000[Hz])を分周して作られるため、周波数、デューティ、初期位相の各パラメータは設定できる値が離散的になります。そのため、引数に与えた希望値と設定可能な値が異なる場合があります。関数は各引数に実際に設定できた値をセットして戻ります。

- I2424 タイプの場合

基準となるクロックとして外部入力を使用する点を除いて TWXA_TimerSetPwm()関数と同様です。
内部クロックを用いた場合、出力できる周波数の下限が 1Hz、または、3Hz となりますので、それよりも低い周波数を出力する場合に使用してください。
外部クロックは、チャンネル 2 の場合 HC2 へ、チャンネル 3 の場合 HC3 へ、チャンネル 4 の場合 HC4 へ、チャンネル 6 の場合 HC6 へ、チャンネル 7 の場合 HC7 へ入力します。別の機器からの出力信号を用いることもできますが、他のチャンネルの PWM 出力を入力することも可能です。

TWXA_TimerStart() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerStart(TW_HANDLE hDev, long ChBits)
VB	Function TWXA_TimerStart(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_TIMER_BITS) As Integer
VBA	Function TWXA_TimerStart(ByVal hDev As Long, ByVal ChBits As TWXA_TIMER_BITS) As Long
C#	STATUS TimerStart(System.IntPtr hDev, TIMER_BITS ChBits)

hDev　：デバイスのハンドル
ChBits：スタートする 16 ビットタイマのチャンネル。以下の値を OR で結合
I2219 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルの動作を開始します。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

I0404 タイプ、I0008 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルの動作を開始します。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

I2424 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BIT3	チャンネル 3 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	
C/C++	TWXA_TIMER_BIT5	チャンネル 5 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER5	
VB/VBA	TWXA_TIMER_BITS.TIMER5	
C#	TWXA.TIMER_BITS.TIMER5	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 の動作を開始します。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルの動作を開始します。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

指定のタイマチャンネルの動作を開始します。
PWM 出力に設定したチャンネルはパルス出力を開始し、パルスカウントモードに設定したチャンネルはパルスのカウントを開始します。

TWXA_TimerStop() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerStop(TW_HANDLE hDev, long ChBits)
VB	Function TWXA_TimerStop(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_TIMER_BITS) As Integer
VBA	Function TWXA_TimerStop(ByVal hDev As Long, ByVal ChBits As TWXA_TIMER_BITS) As Long
C#	STATUS TimerStop(System.IntPtr hDev, TIMER_BITS ChBits)

hDev　：デバイスのハンドル
ChBits：停止する 16 ビットタイマのチャンネル
I2219 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルの動作を停止します。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

I0404 タイプ、I0008 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルの動作を停止します。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

I2424 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BIT3	チャンネル 3 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	
C/C++	TWXA_TIMER_BIT5	チャンネル 5 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER5	
VB/VBA	TWXA_TIMER_BITS.TIMER5	
C#	TWXA.TIMER_BITS.TIMER5	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 の動作を停止します。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルの動作を停止します。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

指定のタイマチャンネルの動作を停止します。

TWXA_TimerSetLevel() USB LAN I2219 I0404 I0008

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerSetLevel(TW_HANDLE hDev, long ChBits)
VB	Function TWXA_TimerSetLevel(ByVal hDev As System.IntPtr, ByVal ChBits As TWXA_TIMER_BITS) As Integer
VBA	Function TWXA_TimerSetLevel(ByVal hDev As Long, ByVal ChBits As TWXA_TIMER_BITS) As Long
C#	STATUS TimerSetLevel(System.IntPtr hDev, TIMER_BITS ChBits)

hDev　：デバイスのハンドル

ChBits ： 出力を"ON"とするビットを指定します。以下を OR で結合
I2219 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	PWM0 を ON 出力にします。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	PWM1 を ON 出力にします。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	PWM2 を ON 出力にします。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BITS_ALL	PWM0、PWM1、PWM2 を ON 出力にします。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

I2219 タイプの場合、以下の値を OR で結合

言語	値	説明
C/C++	TWXA_TIMER_BIT0	PWM0 を ON 出力にします。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	PWM1 を ON 出力にします。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BITS_ALL	PWM0、PWM1 を ON 出力にします。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

製品タイプによって指定できる値が異なります。

PWM 出力となっている端子状態を変更します。ChBits で指定した端子は ON、その他の端子は OFF となります。
呼び出しはタイマの停止中に行ってください。

TWXA_TimerReadStatus() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerReadStatus(TW_HANDLE hDev, long *pStatus)
VB	Function TWXA_TimerReadStatus(ByVal hDev As System.IntPtr, ByRef pStatus As Integer) As Integer
VBA	Function TWXA_TimerReadStatus(ByVal hDev As Long, ByRef pStatus As Long) As Long
C#	STATUS TimerReadStatus(System.IntPtr hDev, out int pStatus) STATUS TimerReadStatus(System.IntPtr hDev, out TIMER_BITS pStatus)

hDev　： デバイスのハンドル

pStatus ： [出力]タイマの状態
ビット 0 ： CH0 が動作状態のとき 1 となります
ビット 1 ： CH1 が動作状態のとき 1 となります
ビット 2 ： CH2 が動作状態のとき 1 となります
ビット 3 ： CH3 が動作状態のとき 1 となります
ビット 4 ： CH4 が動作状態のとき 1 となります
ビット 5 ： CH5 が動作状態のとき 1 となります
ビット 6 ： CH6 が動作状態のとき 1 となります
ビット 7 ： CH7 が動作状態のとき 1 となります
ビット 8-31(MSB) ： 常に 0 です

タイマの動作状態を返します。動作中のチャンネルは pStatus の対応するビットが 1 になります。

TWXA_TimerReadCnt() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerReadCnt(TW_HANDLE hDev, long Ch, short *pCnt)
VB	Function TWXA_TimerReadCnt(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByRef pCnt As Short) As Integer Function TWXA_TimerReadCnt(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByRef pCnt As System.UInt16) As Integer
VBA	Function TWXA_TimerReadCnt(ByVal hDev As Long, ByVal Ch As Long, ByRef pCnt As Integer) As Long
C#	STATUS TimerReadCnt(System.IntPtr hDev, int Ch, out short pCnt) STATUS TimerReadCnt(System.IntPtr hDev, int Ch, out ushort pCnt)

hDev ： デバイスのハンドル
Ch ： チャンネル
I2219 タイプの場合 ： 0～2
I0404 タイプ、I0008 タイプの場合 ： 0～1
I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	
C/C++	TWXA_TIMER_BIT1	チャンネル 1 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	チャンネル 2 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BIT3	チャンネル 3 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	

言語	値	説明
C/C++	TWXA_TIMER_BIT5	チャンネル 5 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER5	
VB/VBA	TWXA_TIMER_BITS.TIMER5	
C#	TWXA.TIMER_BITS.TIMER5	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 のカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	
C/C++	TWXA_TIMER_BITS_ALL	全てのチャンネルのカウンタの値を読み出します。
C++	TWXA::TIMER_BITS::TIMER_ALL	
VB/VBA	TWXA_TIMER_BITS.TIMER_ALL	
C#	TWXA.TIMER_BITS.TIMER_ALL	

pCnt　：[出力]カウント数の格納先

- I2219 タイプ、I0404 タイプ、I0008 タイプの場合

指定されたチャンネルのカウンタの値を読み出します。

- I2424 タイプの場合

指定されたチャンネルのカウンタの値を読み出します。
TWXA_TIMER_BITS_ALL(相当の値)を指定した場合、pCnt に全てのチャンネルの値が連続して格納されますので pCnt には 8 チャンネル分のデータを格納できる領域(2 x 8 = 16 バイト)が必要です。

TWXA_TimerSetCnt() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerSetCnt(TW_HANDLE hDev, long Ch, short Cnt)
VB	Function TWXA_TimerSetCnt(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal Cnt As Short) As Integer Function TWXA_TimerSetCnt(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal Cnt As System.UInt16) As Integer
VBA	Function TWXA_TimerSetCnt(ByVal hDev As Long, ByVal Ch As Long, ByVal Cnt As Integer) As Long
C#	STATUS TimerSetCnt(System.IntPtr hDev, int Ch, short Cnt) STATUS TimerSetCnt(System.IntPtr hDev, int Ch, ushort Cnt)

hDev　：デバイスのハンドル
Ch　　：チャンネル
　　I2219 タイプの場合 ： 0～2
　　I0404 タイプ、I0008 タイプの場合 ： 0～1
　　I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_BIT0	チャンネル 0 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER0	
VB/VBA	TWXA_TIMER_BITS.TIMER0	
C#	TWXA.TIMER_BITS.TIMER0	

言語	値	説明
C/C++	TWXA_TIMER_BIT1	チャンネル 1 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER1	
VB/VBA	TWXA_TIMER_BITS.TIMER1	
C#	TWXA.TIMER_BITS.TIMER1	
C/C++	TWXA_TIMER_BIT2	チャンネル 2 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BIT3	チャンネル 3 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	
C/C++	TWXA_TIMER_BIT5	チャンネル 5 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER5	
VB/VBA	TWXA_TIMER_BITS.TIMER5	
C#	TWXA.TIMER_BITS.TIMER5	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 のカウンタに値をセットします。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	

Cnt　：カウント数

指定されたチャンネルのカウンタに値をセットします。

TWXA_TimerSetNumOfPulse() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerSetNumOfPulse(TW_HANDLE hDev, long Ch, DWORD nPulse)
VB	Function TWXA_TimerSetNumOfPulse(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal nPulse As Integer) As Integer
VBA	Function TWXA_TimerSetNumOfPulse(ByVal hDev As Long, ByVal Ch As Long, ByVal nPulse As Long) As Long
C#	STATUS TimerSetNumOfPulse(System.IntPtr hDev, int Ch, uint nPulse)

hDev　：デバイスのハンドル
Ch　：チャンネル
I2219 タイプの場合 ： 0～2
I0404 タイプ、I0008 タイプの場合 ： 0～1

I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の出力するパルス数を設定します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	
C/C++	TWXA_TIMER_BIT3	チャンネル 3 の出力するパルス数を設定します。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 の出力するパルス数を設定します。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 の出力するパルス数を設定します。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 の出力するパルス数を設定します。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	

nPulse ： 出力するパルス数
0x00000000 ： 設定できません。TW_INVALID_ARGS が返ります
0xffffffff ： 停止するまでパルス出力を続けます(デフォルト動作)
上記以外 ： 指定した数のパルスを出力して停止します

PWM 出力で出力するパルス数を指定します。

TWXA_TimerReadNumOfPulse() USB LAN I2219 I0404 I0008 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_TimerReadNumOfPulse(TW_HANDLE hDev, long Ch, DWORD *pnPulse)
VB	Function TWXA_TimerReadNumOfPulse(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByRef pnPulse As Integer) As Integer
VBA	Function TWXA_TimerReadNumOfPulse(ByVal hDev As Long, ByVal Ch As Long, ByRef pnPulse As Long) As Long
C#	STATUS TimerReadNumOfPulse(System.IntPtr hDev, int Ch, out uint pnPulse)

hDev ： デバイスのハンドル
Ch ： チャンネル
I2219 タイプの場合 ： 0～2
I0404 タイプ、I0008 タイプの場合 ： 0～1
I2424 タイプの場合、以下の値のいずれか

言語	値	説明
C/C++	TWXA_TIMER_BIT2	チャンネル 2 の残りのパルス数を取得します。
C++	TWXA::TIMER_BITS::TIMER2	
VB/VBA	TWXA_TIMER_BITS.TIMER2	
C#	TWXA.TIMER_BITS.TIMER2	

言語	値	説明
C/C++	TWXA_TIMER_BIT3	チャンネル 3 の残りのパルス数を取得します。
C++	TWXA::TIMER_BITS::TIMER3	
VB/VBA	TWXA_TIMER_BITS.TIMER3	
C#	TWXA.TIMER_BITS.TIMER3	
C/C++	TWXA_TIMER_BIT4	チャンネル 4 の残りのパルス数を取得します。
C++	TWXA::TIMER_BITS::TIMER4	
VB/VBA	TWXA_TIMER_BITS.TIMER4	
C#	TWXA.TIMER_BITS.TIMER4	
C/C++	TWXA_TIMER_BIT6	チャンネル 6 の残りのパルス数を取得します。
C++	TWXA::TIMER_BITS::TIMER6	
VB/VBA	TWXA_TIMER_BITS.TIMER6	
C#	TWXA.TIMER_BITS.TIMER6	
C/C++	TWXA_TIMER_BIT7	チャンネル 7 の残りのパルス数を取得します。
C++	TWXA::TIMER_BITS::TIMER7	
VB/VBA	TWXA_TIMER_BITS.TIMER7	
C#	TWXA.TIMER_BITS.TIMER7	

pnPulse : [出力]残りパルス数の格納先

PWM 出力の残りのパルス数を返します。出力パルス数を設定していない場合、pnPulse には 0 が返ります。

□ シリアルポート操作関数

TWXA_SCISetMode() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SCISetMode(TW_HANDLE hDev, long Ch, long Mode, long Baud)
VB	Function TWXA_SCISetMode(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal Mode As TWXA_SCI_MODE, ByVal Baud As TWXA_SCI_BAUD) As Integer
VBA	Function TWXA_SCISetMode(ByVal hDev As Long, ByVal Ch As Long, ByVal Mode As TWXA_SCI_MODE, ByVal Baud As TWXA_SCI_BAUD) As Long
C#	STATUS SCISetMode(System.IntPtr hDev, int Ch, SCI_MODE Mode, SCI_BAUD Baud)

hDev ： デバイスのハンドル
Ch ： チャンネル(0,1)
Mode ： シリアルポートの動作設定。次の値からデータ長、パリティ、ストップビットに関する設定を 1 つずつ選択して OR で結合します。指定しない項目はデフォルトと書かれた設定になります

言語	値	説明
C/C++	TWXA_SCI_DATA8	データ長を 8 ビットにします(デフォルト)。
C++	TWXA::SCI_MODE::DATA8	
VB/VBA	TWXA_SCI_MODE.DATA8	
C#	TWXA.SCI_MODE.DATA8	
C/C++	TWXA_SCI_DATA7	データ長を 7 ビットにします。
C++	TWXA::SCI_MODE::DATA7	
VB/VBA	TWXA_SCI_MODE.DATA7	
C#	TWXA.SCI_MODE.DATA7	
C/C++	TWXA_SCI_NOPARITY	パリティビットを使用しません(デフォルト)。
C++	TWXA::SCI_MODE::NO_PARITY	
VB/VBA	TWXA_SCI_MODE.NO_PARITY	
C#	TWXA.SCI_MODE.NO_PARITY	
C/C++	TWXA_SCI_EVEN	偶数パリティを使用します。
C++	TWXA::SCI_MODE::EVEN	
VB/VBA	TWXA_SCI_MODE.EVEN	
C#	TWXA.SCI_MODE.EVEN	
C/C++	TWXA_SCI_ODD	奇数パリティを使用します。
C++	TWXA::SCI_MODE::ODD	
VB/VBA	TWXA_SCI_MODE.ODD	
C#	TWXA.SCI_MODE.ODD	
C/C++	TWXA_SCI_STOP1	ストップビットを 1 ビットとします(デフォルト)。
C++	TWXA::SCI_MODE::STOP1	
VB/VBA	TWXA_SCI_MODE.STOP1	
C#	TWXA.SCI_MODE.STOP1	
C/C++	TWXA_SCI_STOP2	ストップビットを 2 ビットとします。
C++	TWXA::SCI_MODE::STOP2	
VB/VBA	TWXA_SCI_MODE.STOP2	
C#	TWXA.SCI_MODE.STOP2	

Baud ： ボーレート

I2219 タイプ、I0x0x タイプ、A0800 タイプの場合、次の値のいずれか

言語	値	説明
C/C++	TWXA_SCI_BAUD300	ボーレートを 300bps にします。
C++	TWXA::SCI_BAUD::BAUD300	
VB/VBA	TWXA_SCI_BAUD.BAUD300	
C#	TWXA.SCI_BAUD.BAUD300	
C/C++	TWXA_SCI_BAUD600	ボーレートを 600bps にします。
C++	TWXA::SCI_BAUD::BAUD600	
VB/VBA	TWXA_SCI_BAUD.BAUD600	
C#	TWXA.SCI_BAUD.BAUD600	
C/C++	TWXA_SCI_BAUD1200	ボーレートを 1200bps にします。
C++	TWXA::SCI_BAUD::BAUD1200	
VB/VBA	TWXA_SCI_BAUD.BAUD1200	
C#	TWXA.SCI_BAUD.BAUD1200	
C/C++	TWXA_SCI_BAUD2400	ボーレートを 2400bps にします。
C++	TWXA::SCI_BAUD::BAUD2400	
VB/VBA	TWXA_SCI_BAUD.BAUD2400	
C#	TWXA.SCI_BAUD.BAUD2400	
C/C++	TWXA_SCI_BAUD4800	ボーレートを 4800bps にします。
C++	TWXA::SCI_BAUD::BAUD4800	
VB/VBA	TWXA_SCI_BAUD.BAUD4800	
C#	TWXA.SCI_BAUD.BAUD4800	
C/C++	TWXA_SCI_BAUD9600	ボーレートを 9600bps にします。
C++	TWXA::SCI_BAUD::BAUD9600	
VB/VBA	TWXA_SCI_BAUD.BAUD9600	
C#	TWXA.SCI_BAUD.BAUD9600	
C/C++	TWXA_SCI_BAUD14400	ボーレートを 14400bps にします。
C++	TWXA::SCI_BAUD::BAUD14400	
VB/VBA	TWXA_SCI_BAUD.BAUD14400	
C#	TWXA.SCI_BAUD.BAUD14400	
C/C++	TWXA_SCI_BAUD19200	ボーレートを 19200bps にします。
C++	TWXA::SCI_BAUD::BAUD19200	
VB/VBA	TWXA_SCI_BAUD.BAUD19200	
C#	TWXA.SCI_BAUD.BAUD19200	
C/C++	TWXA_SCI_BAUD38400	ボーレートを 38400bps にします。
C++	TWXA::SCI_BAUD::BAUD38400	
VB/VBA	TWXA_SCI_BAUD.BAUD38400	
C#	TWXA.SCI_BAUD.BAUD38400	

I2424 タイプの場合、次の値のいずれか

言語	値	説明
C/C++	TWXA_SCI_BAUD300	ボーレートを 300bps にします。
C++	TWXA::SCI_BAUD::BAUD300	
VB/VBA	TWXA_SCI_BAUD.BAUD300	
C#	TWXA.SCI_BAUD.BAUD300	
C/C++	TWXA_SCI_BAUD600	ボーレートを 600bps にします。
C++	TWXA::SCI_BAUD::BAUD600	
VB/VBA	TWXA_SCI_BAUD.BAUD600	
C#	TWXA.SCI_BAUD.BAUD600	

言語	値	説明
C/C++	TWXA_SCI_BAUD1200	ボーレートを 1200bps にします。
C++	TWXA::SCI_BAUD::BAUD1200	
VB/VBA	TWXA_SCI_BAUD.BAUD1200	
C#	TWXA.SCI_BAUD.BAUD1200	
C/C++	TWXA_SCI_BAUD2400	ボーレートを 2400bps にします。
C++	TWXA::SCI_BAUD::BAUD2400	
VB/VBA	TWXA_SCI_BAUD.BAUD2400	
C#	TWXA.SCI_BAUD.BAUD2400	
C/C++	TWXA_SCI_BAUD4800	ボーレートを 4800bps にします。
C++	TWXA::SCI_BAUD::BAUD4800	
VB/VBA	TWXA_SCI_BAUD.BAUD4800	
C#	TWXA.SCI_BAUD.BAUD4800	
C/C++	TWXA_SCI_BAUD9600	ボーレートを 9600bps にします。
C++	TWXA::SCI_BAUD::BAUD9600	
VB/VBA	TWXA_SCI_BAUD.BAUD9600	
C#	TWXA.SCI_BAUD.BAUD9600	
C/C++	TWXA_SCI_BAUD14400	ボーレートを 14400bps にします。
C++	TWXA::SCI_BAUD::BAUD14400	
VB/VBA	TWXA_SCI_BAUD.BAUD14400	
C#	TWXA.SCI_BAUD.BAUD14400	
C/C++	TWXA_SCI_BAUD19200	ボーレートを 19200bps にします。
C++	TWXA::SCI_BAUD::BAUD19200	
VB/VBA	TWXA_SCI_BAUD.BAUD19200	
C#	TWXA.SCI_BAUD.BAUD19200	
C/C++	TWXA_SCI_BAUD38400	ボーレートを 38400bps にします。
C++	TWXA::SCI_BAUD::BAUD38400	
VB/VBA	TWXA_SCI_BAUD.BAUD38400	
C#	TWXA.SCI_BAUD.BAUD38400	
C/C++	TWXA_SCI_BAUD56000	ボーレートを 56000bps にします。
C++	TWXA::SCI_BAUD::BAUD56000	
VB/VBA	TWXA_SCI_BAUD.BAUD56000	
C#	TWXA.SCI_BAUD.BAUD56000	
C/C++	TWXA_SCI_BAUD57600	ボーレートを 57600bps にします。
C++	TWXA::SCI_BAUD::BAUD57600	
VB/VBA	TWXA_SCI_BAUD.BAUD57600	
C#	TWXA.SCI_BAUD.BAUD57600	
C/C++	TWXA_SCI_BAUD115200	ボーレートを 115200bps にします。
C++	TWXA::SCI_BAUD::BAUD115200	
VB/VBA	TWXA_SCI_BAUD.BAUD115200	
C#	TWXA.SCI_BAUD.BAUD115200	
C/C++	TWXA_SCI_BAUD128000	ボーレートを 128000bps にします。
C++	TWXA::SCI_BAUD::BAUD128000	
VB/VBA	TWXA_SCI_BAUD.BAUD128000	
C#	TWXA.SCI_BAUD.BAUD128000	
C/C++	TWXA_SCI_BAUD256000	ボーレートを 256000bps にします。
C++	TWXA::SCI_BAUD::BAUD256000	
VB/VBA	TWXA_SCI_BAUD.BAUD256000	
C#	TWXA.SCI_BAUD.BAUD256000	

- I2219 タイプ、I0x0x タイプ、A0800 タイプの場合

シリアルポートの動作設定と速度設定を行います。Baud に指定可能な値は 300bps から 38400bps までです。

チャンネル 1 に対してこの関数を呼び出すと、再起動するまでユーザーファームのデバッグ用ポートとして使用できなくなります。

- I2424 タイプの場合

シリアルポートの動作設定と速度設定を行います。Baud に指定可能な値は 300bps から 256000bps までです。

TWXA_SCIReadStatus() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SCIReadStatus(TW_HANDLE hDev, long Ch, BYTE *pStatus, long *pnReceive)
VB	Function TWXA_SCIReadStatus(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByRef pStatus As Byte, ByRef pnReceive As Integer) As Integer
VBA	Function TWXA_SCIReadStatus(ByVal hDev As Long, ByVal Ch As Long, ByRef pStatus As Byte, ByRef pnReceive As Long) As Long
C#	STATUS SCIReadStatus(System.IntPtr hDev, int Ch, out byte pStatus, out int pnReceive)

hDev : デバイスのハンドル
Ch : チャンネル(0,1)
pStatus : [出力]ステータスの格納先。各ビットの意味は以下です
0(LSB)ビット～2 ビット : 常に 0 です
3 ビット : パリティエラーが起こった場合に 1 になります
4 ビット : フレーミングエラーが起こった場合に 1 になります
5 ビット : オーバーランエラーが起こった場合に 1 になります
6 ビット～7(MSB)ビット : 常に 0 です
pnReceive : [出力]受信データバイト数の格納先

- I2219 タイプ、I0x0x タイプ、A0800 タイプの場合

シリアルポートのステータスと、デバイス内部のシリアル用受信バッファ中のデータバイト数を取得します。
シリアル用受信バッファは、チャンネル毎に 127 バイトまでのデータを格納できます。

- I2424 タイプの場合

シリアルポートのステータスと、デバイス内部のシリアル用受信バッファ中のデータバイト数を取得します。
シリアル用受信バッファは、チャンネル毎に 511 バイトまでのデータを格納できます。

TWXA_SCIRead() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SCIRead(TW_HANDLE hDev, long Ch, void *pData, long nData, long *pnRead)
VB	Function TWXA_SCIRead(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal pData As Object, ByVal nData As Integer, ByRef pnRead As Integer) As Integer
VBA	Function TWXA_SCIRead(ByVal hDev As Long, ByVal Ch As Long, ByRef pData As Any, ByVal nData As Long, ByRef pnRead As Long) As Long
C#	STATUS SCIRead(System.IntPtr hDev, int Ch, object pData, int nData, out int pnRead)

hDev : デバイスのハンドル
Ch : チャンネル(0,1)
pData : [出力]読み出したデータの格納先
nData : 受信するバイト数
I2219 タイプ、I0x0x タイプ、A0800 タイプの場合 : 0～255
I2424 タイプの場合 : 0～511
pnRead : [出力]実際に受信したバイト数の格納先

デバイスのシリアルポートからデータを読み出します。
デバイスの受信バッファ中のデータ数よりも大きなデータを要求すると、関数と同時にデバイスもデータ待ち状態(ブロッキング)となります。
ブロッキングを起こさないようにするには、この関数を呼び出す前に TWXA_SCIReadStatus()を使用し、受信バッファ中のデータバイト数を確認した上で、受信バイト数以下のデータを要求するようにしてください。

TWXA_SCIWrite() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SCIWrite(TW_HANDLE hDev, long Ch, void *pData, long nData)
VB	Function TWXA_SCIWrite(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_SCIWrite(ByVal hDev As Long, ByVal Ch As Long, ByRef pData As Any, ByVal nData As Long) As Long
C#	STATUS SCIWrite(System.IntPtr hDev, int Ch, object pData, int nData)

hDev ： デバイスのハンドル
Ch ： チャンネル(0,1)
pData ： [入力]送信データの格納先
nData ： 送信するバイト数
I2219 タイプ、I0x0x タイプ、A0800 タイプの場合 ： 0～255
I2424 タイプの場合 ： 0～511

デバイスのシリアルポートからデータを送信します。

TWXA_SCISetDelimiter() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SCISetDelimiter(TW_HANDLE hDev, long Ch, void *pDelimiter, long nDelimiter)
VB	Function TWXA_SCISetDelimiter(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal pDelimiter() As Byte, ByVal nDelimiter As Integer) As Integer Function TWXA_SCISetDelimiter(ByVal hDev As System.IntPtr, ByVal Ch As Integer, ByVal pDelimiter As String, ByVal nDelimiter As Integer) As Integer
VBA	Function TWXA_SCISetDelimiter(ByVal hDev As Long, ByVal Ch As Long, ByVal pDelimiter As Byte, ByVal nDelimiter As Long) As Long
C#	STATUS SCISetDelimiter(System.IntPtr hDev, int Ch, byte []pDelimiter, int nDelimiter) STATUS SCISetDelimiter(System.IntPtr hDev, int Ch, string pDelimiter, int nDelimiter)

hDev ： デバイスのハンドル
Ch ： チャンネル(0,1)
pDelimiter ： [入力]デリミタコードの格納先
nDelimiter ： デリミタコードのバイト数(0～2)

シリアルポートにデリミタコードを設定します。デリミタの設定は TWXA_SCIRead()の動作に影響します。
TWXA_SCIRead()関数はデリミタコード(1 バイトまたは 2 バイト)が現れると、シリアルポートからの読み取りを一旦中止し、デリミタコードより後には指定バイトまで 0 をコピーしてデータを返します。

□ ユーザーファームサポート関数

TWXA_ATF_INFO 構造体

言語	宣言

C/C++	typedef struct {
	  DWORD FormatVer;
	  DWORD FirmwareVer;
	  char Description[32];
	  char Manufacture[32];
	  DWORD ProgramVer;
	  DWORD AddressTop;
	  DWORD AddressBottom;
	  DWORD CommandAddress;
	  WORD EncryptMode;
	  WORD wRsv;
	  DWORD MainAddress;
	} TWXA_ATF_INFO;

VB	Public Structure TWXA_ATF_INFO
	  Public FormatVer As Integer
	  Public FirmwareVer As Integer
	  <MarshalAs(UnmanagedType.ByValTStr, SizeConst:=32)> _
	  Public Description As String
	  <MarshalAs(UnmanagedType.ByValTStr, SizeConst:=32)> _
	  Public Manufacture As String
	  Public ProgramVer As Integer
	  Public AddressTop As Integer
	  Public AddressBottom As Integer
	  Public CommandAddress As Integer
	  Public EncryptMode As Short
	  Public wRsv As Short
	  Public MainAddress As Integer
	End Structure

VBA	Public Type TWXA_ATF_INFO FormatVer As Long FirmwareVer As Long Description As String * 32 Manufacture As String * 32 ProgramVer As Long AddressTop As Long AddressBottom As Long CommandAddress As Long EncryptMode As Integer wRsv As Integer MainAddress As Long End Type

言語	宣言

C#	public struct ATF_INFO
	{
	  public uint FormatVer;
	  public uint FirmwareVer;
	  [MarshalAs(UnmanagedType.ByValTStr, SizeConst=32)]
	  public string Description;
	  [MarshalAs(UnmanagedType.ByValTStr, SizeConst = 32)]
	  public string Manufacture;
	  public uint ProgramVer;
	  public uint AddressTop;
	  public uint AddressBottom;
	  public uint CommandAddress;
	  public short EncryptMode;
	  public short wRsv;
	  public uint MainAddress;
	}

FormatVer : ATF ファイルのフォーマットに関する情報です
FirmwareVer : 「ATF Maker」の「要求するファームウェアバージョン」に記載した内容です
Description : 「ATF Maker」の「プログラムの説明」に記載した内容です
Manufacture : 「ATF Maker」の「作成者」に記載した内容です
ProgramVer : 「ATF Maker」の「プログラムのバージョン」に記載した内容です
AddressTop : ATF で使用する RAM 領域の先頭アドレスです
AddressBottom : ATF で使用する RAM 領域の最終アドレス+1 です
CommandAddress : コマンドハンドラ(ATF_Command)のアドレスです
EncryptMode : 予約。使用されません
wRsv : 予約。使用されません
MainAddress : メイン関数(ATF_Main)のアドレスです

ATF ファイルの情報を取得するための構造体です。TWXA_ATFGetInfo()関数で使用します。

TWXA_ATFGetInfo() USB LAN I2219 I0800 I0404 I0008 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_ATFGetInfo(LPCTSTR FileName, TWXA_ATF_INFO *pInfo)
VB	Function TWXA_ATFGetInfo(ByVal FileName As String, ByRef pInfo As TWXA_ATF_INFO) As Integer
VBA	Function TWXA_ATFGetInfo(ByVal FileName As String, ByRef pInfo As TWXA_ATF_INFO) As Long
C#	STATUS ATFGetInfo(string FileName, out ATF_INFO pInfo)

FileName : ATF ファイルのパス
pInfo : [出力]ATF ファイルの情報

ATF ファイルの情報を返します。内容は TWXA_ATF_INFO 構造体の説明を参照してください。

TWXA_ATFUserCommand() USB LAN I2219 I0800 I0404 I0008 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_ATFUserCommand(TW_HANDLE hDev, WORD Command, DWORD Param1, DWORD Param2, void *pData, long nData)
VB	Function TWXA_ATFUserCommand(ByVal hDev As System.IntPtr, ByVal Command As Short, ByVal Param1 As Integer, ByVal Param2 As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_ATFUserCommand(ByVal hDev As Long, ByVal Command As Integer, ByVal Param1 As Long, ByVal Param2 As Long, ByRef pData As Variant, ByVal nData As Long) As Long
C#	STATUS ATFUserCommand(System.IntPtr hDev, ushort Command, uint Param1, uint Param2) STATUS ATFUserCommand(System.IntPtr hDev, ushort Command, uint Param1, uint Param2, object pData, int nData)

hDev : デバイスのハンドル
Command : コマンドを識別する番号
Param1 : コマンドパラメータ 1
Param2 : コマンドパラメータ 2
pData : [入力]付加するデータの格納先
nData : 付加するデータのバイト数

ユーザーファームにコマンドを送信します。Command, Param1, Param2 の各パラメータを引数としてユーザーファームの ATF_Command()関数が呼び出されます。
追加の送信データがある場合 pData に指定すると送信することができます。追加したデータはユーザーファーム側で TWFA_Receive()関数を呼び出して全て読み取る必要があります。

TWXA_ATFDownload() USB LAN I2219 I0800 I0404 I0008 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_ATFDownload(TW_HANDLE hDev, LPCTSTR FileName, LPCTSTR Reserve)
VB	Function TWXA_ATFDownload(ByVal hDev As System.IntPtr, ByVal FileName As String, ByVal Reserve As String) As Integer
VBA	Function TWXA_ATFDownload(ByVal hDev As Long, ByVal FileName As String, ByVal Reserve As String) As Long
C#	STATUS ATFDownload(System.IntPtr hDev, string FileName)

hDev : デバイスのハンドル
FileName : ATF ファイルのパス
Reserve : 予約。C 言語では NULL とし、その他ではデフォルト値のままとしてください。

ATF ファイルをデバイスにダウンロードし使用可能にします。

TWXA_PortBWrite() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PortBWrite(TW_HANDLE hDev, DWORD Port, void *pData, long nData)
VB	Function TWXA_PortBWrite(ByVal hDev As System.IntPtr, ByVal Port As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_PortBWrite(ByVal hDev As Long, ByVal Port As Long, ByRef pData As Any, ByVal nData As Long) As Long
C#	STATUS PortBWrite(System.IntPtr hDev, uint Port, object pData, int nData)

hDev ： デバイスのハンドル
Port ： 書き込むメモリのアドレス
pData ： [入力]書き込むデータの格納先
nData ： 書き込むバイト数

デバイスの内部メモリにデータを書き込みます。

TWXA_PortBRead() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_PortBRead(TW_HANDLE hDev, DWORD Port, void *pData, long nData)
VB	Function TWXA_PortBRead(ByVal hDev As System.IntPtr, ByVal Port As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_PortBRead(ByVal hDev As Long, ByVal Port As Long, ByRef pData As Any, ByVal nData As Long) As Long
C#	STATUS PortBRead(System.IntPtr hDev, uint Port, object pData, int nData)

hDev ： デバイスのハンドル
Port ： 読み出すメモリのアドレス
pData ： [出力]読み出したデータの格納先
nData ： 読み出すバイト数

デバイスの内部メモリからデータを読み出します。

TWXA_Write() USB LAN I2219 I0800 I0404 I0008 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_Write(TW_HANDLE hDev, void *pData, DWORD nData, DWORD *pWritten)
VB	Function TWXA_Write(ByVal hDev As System.IntPtr, ByVal pData As Object, ByVal nData As Integer, ByRef pWritten As Integer) As Integer
VBA	Function TWXA_Write(ByVal hDev As Long, ByRef pData As Any, ByVal nData As Long, ByRef pWritten As Long) As Long
C#	STATUS Write(System.IntPtr hDev, object pData, int nData) STATUS Write(System.IntPtr hDev, object pData, int nData, out int pWritten)

hDev ： デバイスのハンドル
pData ： [入力]送信するデータの格納先
nData ： 送信するバイト数
pWritten ： [出力]実際に送信したバイト数の格納先

ユーザーファームにデータを送信する場合に使用します。通常は TWXA_ATFUserCommand()で送信しきれない

データがある場合に補助的に呼び出します。
この関数によるデータはコマンドとしてではなく、そのままデバイスに送信されます。システムファームのコマンドループは受信したデータを全てコマンドとして処理しようとしますので、この関数の呼び出しタイミングには注意が必要です。この関数で送信したデータがデバイスの受信バッファに残ったままシステムファームに処理が移ると、システムファームはデータをコマンドとして解釈し誤った動作を行います。
送信したデータはユーザーファーム内の ATF_Command()や ATF_Main()関数内で確実に読み出してください。

TWXA_Read() USB LAN I2219 I0800 I0404 I0008 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_Read(TW_HANDLE hDev, void *pData, DWORD nData, DWORD *pRead)
VB	Function TWXA_Read(ByVal hDev As System.IntPtr, ByVal pData As Object, ByVal nData As Integer, ByRef pRead As Integer) As Integer
VBA	Function TWXA_Read(ByVal hDev As Long, ByRef pData As Any, ByVal nData As Long, ByRef pRead As Long) As Long
C#	STATUS Read(System.IntPtr hDev, object pData, int nData) STATUS Read(System.IntPtr hDev, object pData, int nData, out int pRead)

hDev ： デバイスのハンドル
pData ： [出力]受信したデータの格納先
nData ： 受信するバイト数
pRead ： [出力]実際に受信したバイト数の格納先

デバイスから送信されたデータを読み出します。通常はユーザーファームから送信したデータを受信するために使用します。

TWXA_GetQueueStatus() USB LAN I2219 I0800 I0404 I0008 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_GetQueueStatus(TW_HANDLE hDev, DWORD *pnQueue)
VB	Function TWXA_GetQueueStatus(ByVal hDev As System.IntPtr, ByRef pnQueue As Integer) As Integer
VBA	Function TWXA_GetQueueStatus(ByVal hDev As Long, ByRef pnQueue As Long) As Long
C#	STATUS GetQueueStatus(System.IntPtr hDev, out int pnQueue)

hDev ： デバイスのハンドル
pnQueue ： [出力]受信バッファ中のバイト数の格納先

デバイスから送信されたデータを蓄積するパソコン内のバッファに何バイト受信しているかを調べます。

TWXA_Purge() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_Purge(TW_HANDLE hDev)
VB	Function TWXA_Purge(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_Purge(ByVal hDev As Long) As Long
C#	STATUS Purge(System.IntPtr hDev)

hDev ： デバイスのハンドル

デバイスから送信されたデータを蓄積するパソコン内の受信バッファをクリアします。

□ フラッシュメモリ操作関数

TWXA_FlashAttachWriter() USB LAN I2219 I0800 I0404 I0008 A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_FlashAttachWriter(TW_HANDLE hDev)
VB	Function TWXA_FlashAttachWriter(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_FlashAttachWriter(ByVal hDev As Long) As Long
C#	STATUS FlashAttachWriter(System.IntPtr hDev)

hDev ： デバイスのハンドル

フラッシュメモリ制御用のファームウェア(M3069FlashWriter.atf)をデバイスにダウンロードし、フラッシュメモリ操作関数を使用できるようにします。ファイルが見つからない場合 TW_FILE_ERROR を返します。
フラッシュメモリ制御用のファームウェアはユーザーメモリにダウンロードされますので、この関数の呼び出しによりユーザーメモリの内容は破壊されます。また、フラッシュメモリの操作が必要な間はユーザーメモリを利用することができません。

TWXA_FlashEraseBlk() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_FlashEraseBlk(TW_HANDLE hDev, long Blk)
VB	Function TWXA_FlashEraseBlk(ByVal hDev As System.IntPtr, ByVal Blk As Integer) As Integer
VBA	Function TWXA_FlashEraseBlk(ByVal hDev As Long, ByVal Blk As Long) As Long
C#	STATUS FlashEraseBlk(System.IntPtr hDev, int Blk)

hDev ： デバイスのハンドル
Blk ： ブロック番号(1～3)

フラッシュメモリの指定ブロック内のデータを全て消去します。
I2219 タイプ、I0x0x タイプ、A0800 タイプの場合、この関数を使用するには予め TWXA_FlashAttachWriter() が正常に終了している必要があります。

TWXA_FlashWrite() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_FlashWrite(TW_HANDLE hDev, DWORD Address, void *pData, long nData)
VB	Function TWXA_FlashWrite(ByVal hDev As System.IntPtr, ByVal Address As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_FlashWrite(ByVal hDev As Long, ByVal Address As Long, ByRef pData As Any, ByVal nData As Long) As Long
C#	STATUS FlashWrite(System.IntPtr hDev, uint Address, object pData, int nData)

hDev ： デバイスのハンドル
Address ： 書き込むアドレス(0x1000-0x3f80,128 バイト境界を指定)
pData ： [入力]書き込むデータの格納先
nData ： データのバイト数(0-4096, 128 バイト単位)

フラッシュメモリにデータを書き込みます。書き込むアドレスの指定は 128 バイト境界(下位 7 ビットが 0)を指定する必要があります。また書き込むデータのバイト数は 128 の倍数を指定してください。
I2219 タイプ、I0x0x タイプ、A0800 タイプの場合、この関数を使用するには予め TWXA_FlashAttachWriter() が正常に終了している必要があります。

TWXA_FlashRead() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_FlashRead(TW_HANDLE hDev, DWORD Address, void *pData, long nData)
VB	Function TWXA_FlashRead(ByVal hDev As System.IntPtr, ByVal Address As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_FlashRead(ByVal hDev As Long, ByVal Address As Long, ByRef pData As Any, ByVal nData As Long) As Long
C#	STATUS FlashRead(System.IntPtr hDev, uint Address, object pData, int nData)

hDev : デバイスのハンドル
Address : 読み出すアドレス
I2219 タイプ、I0x0x タイプ、A0800 タイプの場合 : 0x1000-0x3ffff
I2424 タイプの場合 : 0x1000-0x3f80, 128 バイト境界を指定

pData : [出力]読み出したデータの格納先
nData : データのバイト数(0-4096, 128 バイト単位)
I2219 タイプ、I0x0x タイプ、A0800 タイプの場合 : 0-4096, 1 バイト単位
I2424 タイプの場合 : 0-4096,128 バイト単位

フラッシュメモリからデータを読み出します。
I2424 タイプの場合、読み出すアドレスの指定は 128 バイト境界(下位 7 ビットが 0)を指定する必要があります。また読み出すデータのバイト数は 128 の倍数を指定してください。

□ ハードウェアイベント操作関数

TWXA_HW_EVENT 構造体

言語	宣言

C/C++	typedef struct tagTwxaHwEvent
	{
	  HWND hRecvWindow;
	  DWORD idRecvThread;
	  LPVOID lpRsv;
	  UINT Message;
	  DWORD EventBits;
	  long PCCnt[4];
	  long PCCmp[4];
	  long ADVal[4];
	  short ADCmp[4];
	} TWXA_HW_EVENT;

VB	Public Structure TWXA_HW_EVENT
	  Public hRecvWindow As System.IntPtr
	  Public idRecvThread As Integer
	  Public lpRsv As System.IntPtr
	  Public Message As Integer
	  Public EventBits As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
	  Public PCCnt() As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
	  Public PCCmp() As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
	  Public ADVal() As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
	  Public ADCmp() As Short
	
	  Public Sub Initialize()
	  ReDim PCCnt(3)
	  ReDim PCCmp(3)
	  ReDim ADVal(3)
	  ReDim ADCmp(3)
	  End Sub
	End Structure

言語	宣言

C#	public struct HW_EVENT
	{
	  public System.IntPtr hRecvWindow;
	  public uint idRecvThread;
	  public System.IntPtr lpRsv;
	  public int Message;
	  public uint EventBits;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)]
	  public int[] PCCnt;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)]
	  public int[] PCCmp;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)]
	  public int[] ADVal;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)]
	  public short[] ADCmp;
	
	  public void Initialize()
	  {
	  PCCnt = new int[4];
	  PCCmp = new int[4];
	  ADVal = new int[4];
	  ADCmp = new short[4];
	  }
	}

hRecvWindow ： イベント発生時にメッセージを受け取るウィンドウのハンドルを指定
idRecvThread ： イベント発生時にメッセージを受け取るスレッドの ID を指定
lpRsv ： 予約
Message ： メッセージ番号。アプリケーション独自のメッセージでは通常 WM_APP(0x8000)以上の値
EventBits ： 監視するイベントに対応するビットを 1 とします
I2219 タイプの場合 ：

ビット0 ： PC0を監視する
ビット1 ： PC1を監視する
ビット2 ： PC2を監視する
ビット3 ： PC3を監視する
ビット4 ： AD0を監視する
ビット5 ： AD1を監視する
ビット6 ： AD2を監視する
ビット7 ： AD3を監視する
ビット31 ： ユーザーファームから独自のイベントを通知する
上記以外 ： 予約。0としてください

I0x0x タイプの場合 ：

ビット0 ： PC0を監視する
ビット1 ： PC1を監視する
ビット2 ： PC2を監視する
ビット3 ： PC3を監視する
ビット31 ： ユーザーファームから独自のイベントを通知する
上記以外 ： 予約。0としてください

PCCnt[4] ： パルスカウンタ各チャンネルの閾値

PCCmp[4]　：パルスカウンタによるイベント発生条件
I2219 タイプの場合、以下の値のいずれか

TWXA_CMP_NO ： カウンタに変化があれば毎回イベントを発生
TWXA_CMP_GE ： カウンタ値が PCCnt 以上になった場合イベントを発生
TWXA_CMP_LE ： カウンタ値が PCCnt 以下になった場合イベントを発生

IOxOx タイプの場合、以下の値のいずれか

TWXA_CMP_NO ： カウンタに変化があれば毎回イベントを発生
TWXA_CMP_GE ： カウンタ値が PCCnt 以上になった場合イベントを発生

ADVal[4]　：AD 各チャンネルの閾値
I2219 タイプの場合、上位 16 ビットは 0、下位 16 ビットの MSB から 10 ビット使用

ADCmp[4]　：AD 入力によるイベント発生条件。大きさはヒステリシス
I2219 タイプの場合 ：

0以上の場合 ： アナログ入力値がADVal以上でイベント発生
負の場合　　： アナログ入力値がADVal以下でイベント発生

ハードウェアイベントを監視するためのパラメータを設定します。TWXA_SetHwEvent() 関数の呼び出しに使用します。
Visual Basic と C# では構造体を使用する前に Initialize()メソッドを呼び出して初期化を行ってください。

TWXA_SetHwEvent() USB LAN I2219 IO800 IO404 IO008

言語	関数宣言
C/C++	TW_STATUS TWXA_SetHwEvent(TW_HANDLE hDev, TWXA_HW_EVENT *pHwEvent)
VB	Function TWXA_SetHwEvent(ByVal hDev As System.IntPtr, ByRef pHwEvent As TWXA_HW_EVENT) As Integer
C#	STATUS SetHwEvent(System.IntPtr hDev, ref HW_EVENT pHwEvent)

hDev　　：デバイスのハンドル
pHwEvent ：[入力]ハードウェアイベントの通知設定の格納先

デバイスでパルスカウンタとアナログ入力の状態を監視し、予め設定した状態になったとき Windows のアプリケーションに対してメッセージによる通知を行います。メッセージの送信先としてウィンドウかスレッドを指定できます。どちらかを必ず指定する必要がありますが、両方とも指定することもできます(TWXA_HW_EVENT 構造体参照)。
監視を終了するには pHwEvent の EventBits を 0 として呼び出します。
ユーザーファームで SRV_TransmitEvent()関数を利用することで、独自のメッセージをアプリケーションに通知することが可能です。

TWXA_HW_EVENT_EXA 構造体

言語	宣言

C/C++	typedef struct tagTwxaHwEventExA
	{
	  HWND hRecvWindow;
	  DWORD idRecvThread;
	  LPVOID lpRsv;
	  UINT Message;
	  DWORD EventBits;
	  long PCCnt[8];
	  long PCCmp[8];
	  long ADVal[4];
	  long ADCmp[4];
	} TWXA_HW_EVENT_EXA;

VB	Public Structure TWXA_HW_EVENT_EXA
	  Public hRecvWindow As System.IntPtr
	  Public idRecvThread As Integer
	  Public lpRsv As System.IntPtr
	  Public Message As Integer
	  Public EventBits As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=8)> _
	  Public PCCnt() As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=8)> _
	  Public PCCmp() As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
	  Public ADVal() As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=4)> _
	  Public ADCmp() As Integer
	
	  Public Sub Initialize()
	  ReDim PCCnt(7)
	  ReDim PCCmp(7)
	  ReDim ADVal(3)
	  ReDim ADCmp(3)
	  End Sub
	End Structure

言語	宣言

C#	public struct HW_EVENT_EXA
	{
	  public System.IntPtr hRecvWindow;
	  public uint idRecvThread;
	  public System.IntPtr lpRsv;
	  public int Message;
	  public uint EventBits;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 8)]
	  public int[] PCCnt;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 8)]
	  public int[] PCCmp;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)]
	  public int[] ADVal;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 4)]
	  public int[] ADCmp;
	
	  public void Initialize()
	  {
	  PCCnt = new int[8];
	  PCCmp = new int[8];
	  ADVal = new int[4];
	  ADCmp = new int[4];
	  }
	}

hRecvWindow ： イベント発生時にメッセージを受け取るウィンドウのハンドルを指定
idRecvThread ： イベント発生時にメッセージを受け取るスレッドの ID を指定
lpRsv ： 予約
Message ： メッセージ番号。アプリケーション独自のメッセージでは通常 WM_APP(0x8000)以上の値
EventBits ： 監視するイベントに対応するビットを１とします
ビット0 ： PC0を監視する
ビット1 ： PC1を監視する
ビット2 ： PC2を監視する
ビット3 ： PC3を監視する
ビット4 ： AD0を監視する
ビット5 ： AD1を監視する
ビット6 ： AD2を監視する
ビット7 ： AD3を監視する
ビット8 ： PC4を監視する
ビット9 ： PC5を監視する
ビット10 ： PC6を監視する
ビット11 ： PC7を監視する
上記以外 ： 予約。0としてください

PCCnt[8] ： パルスカウンタ各チャンネルの閾値
PCCmp[8] ： パルスカウンタによるイベント発生条件。下記のいずれか
TWXA_CMP_NO ： カウンタに変化があれば毎回イベントを発生
TWXA_CMP_GE ： カウンタ値が PCCnt 以上になった場合に１度だけイベントを発生
TWXA_CMP_LE ： カウンタ値が PCCnt 以下になった場合に１度だけイベントを発生
上記以外の正の場合 ： カウンタ値が PCCnt 以上になった場合にイベントを発生し、
閾値を PCCnt+PCCmp に更新。そのあと閾値以上になれば
イベントを発生して閾値を更新。これを繰り返す
上記以外の負の場合 ： カウンタ値が PCCnt 以下になった場合にイベントを発生し、
閾値を PCCnt+PCCmp に更新。そのあと閾値以下になれば
イベントを発生して閾値を更新。これを繰り返す

ADVal[4]　　: AD 各チャンネルの閾値。
ADCmp[4]　　: AD 入力によるイベント発生条件。大きさはヒステリシス
　　正の場合 : アナログ入力値がADVal以上でイベント発生
　　負の場合 : アナログ入力値がADVal以下でイベント発生

ハードウェアイベントを監視するためのパラメータを設定します。TWXA_SetHwEventEx() 関数の呼び出しに使用します。
Visual Basic と C# では構造体を使用する前に Initialize()メソッドを呼び出して初期化を行ってください。

TWXA_SetHwEventEx() USB LAN I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SetHwEventEx(TW_HANDLE hDev, void *pHwEventEx)
VB	Function TWXA_SetHwEventEx(ByVal hDev As System.IntPtr, ByVal pHwEventEx As Object) As Integer
C#	STATUS SetHwEventEx(System.IntPtr hDev, object pHwEventEx)

hDev　　: デバイスのハンドル
pHwEventEx : [入力]ハードウェアイベントの通知設定の格納先

デバイスでパルスカウンタとアナログ入力の状態を監視し、あらかじめ設定した状態になったとき Windows のアプリケーションに対してメッセージによる通知を行います。メッセージの送信先としてウィンドウかスレッドを指定できます。どちらかを必ず指定する必要がありますが、両方とも指定することもできます(TWXA_HW_EVENT_EXA 構造体参照)。
監視を終了するには pHwEventEx の EventBits を 0 として呼び出します。

□ EEPROM/FRAM 制御関数

TWXA_EEWrite() USB LAN I2219 I0800 I0404 I0008

言語	関数宣言
C/C++	TW_STATUS TWXA_EEWrite(TW_HANDLE hDev, DWORD Address, void *pData, DWORD nData)
VB	Function TWXA_EEWrite(ByVal hDev As System.IntPtr, ByVal Address As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_EEWrite(ByVal hDev As Long, ByVal Address As Long, ByRef pData As Any, ByVal nData As Long) As Long
C#	STATUS EEWrite(System.IntPtr hDev, uint Address, object pData, int nData)

hDev : デバイスのハンドル
Address : EEPROM/FRAM のアドレス(0～8191)
pData : [入力]書き込むデータの格納先
nData : 書き込むバイト数(0～8192)

デバイスに搭載された EEPROM/FRAM に任意のデータを書き込みます。EEPROM/FRAM は RAM やフラッシュメモリとは異なるメモリ空間に配置されていますので、TWXA_EEWrite()、TWXA_EERead()以外の関数ではアクセスできません。
※「USBX-I2219」には対応していません。

TWXA_EERead() USB LAN I2219 I0800 I0404 I0008

言語	関数宣言
C/C++	TW_STATUS TWXA_EERead(TW_HANDLE hDev, DWORD Address, void *pData, DWORD nData)
VB	Function TWXA_EERead(ByVal hDev As System.IntPtr, ByVal Address As Integer, ByVal pData As Object, ByVal nData As Integer) As Integer
VBA	Function TWXA_EERead(ByVal hDev As Long, ByVal Address As Long, ByRef pData As Any, ByVal nData As Long) As Long
C#	STATUS EERead(System.IntPtr hDev, uint Address, object pData, int nData)

hDev : デバイスのハンドル
Address : EEPROM/FRAM のアドレス(0～8191)
pData : [出力]読み出したデータの格納先
nData : 読み出すバイト数(0～8192)

デバイスに搭載された EEPROM/FRAM からデータを読み出します。EEPROM/FRAM は RAM やフラッシュメモリとは異なるメモリ空間に配置されていますので、TWXA_EEWrite()、TWXA_EERead()以外の関数ではアクセスできません。
※「USBX-I2219」には対応していません。

□ 16 ビット AD コンバータ制御関数

TWXA_ADSetRange() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADSetRange(TW_HANDLE hDev, long Range)
VB	Function TWXA_ADSetRange(ByVal hDev As System.IntPtr, ByVal Range As TWXA_AN_OPTION) As Integer
VBA	Function TWXA_ADSetRange(ByVal hDev As Long, ByVal Range As TWXA_AN_OPTION) As Long
C#	STATUS ADSetRange(System.IntPtr hDev, AN_OPTION Range)

hDev　：デバイスのハンドル
Range　：アナログ入力端子の入力レンジを指定。以下の値のいずれか

言語	値	説明
C/C++	TWXA_AN_10VPP	入力レンジを 10Vpp(-5～+5V)に設定します。
C++	TWXA::AN_OPTION::RANGE_10VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_10VPP	
C#	TWXA.AN_OPTION.RANGE_10VPP	
C/C++	TWXA_AN_20VPP	入力レンジを 20Vpp(-10～+10V)に設定します。
C++	TWXA::AN_OPTION::RANGE_20VPP	
VB/VBA	TWXA_AN_OPTION.RANGE_20VPP	
C#	TWXA.AN_OPTION.RANGE_20VPP	

アナログ入力端子の入力レンジを設定します。本関数の呼び出しは連続サンプリングが停止した状態で行ってください。

TWXA_ADGetRange() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADGetRange(TW_HANDLE hDev, long *pRange)
VB	Function TWXA_ADGetRange(ByVal hDev As System.IntPtr, ByRef pRange As TWXA_AN_OPTION) As Integer
VBA	Function TWXA_ADGetRange(ByVal hDev As Long, ByRef pRange As TWXA_AN_OPTION) As Long
C#	STATUS ADGetRange(System.IntPtr hDev, out AN_OPTION pRange) STATUS ADGetRange(System.IntPtr hDev, out int pRange)

hDev　：デバイスのハンドル
pRange　：[出力]取得した現在の入力レンジ設定値の格納先

16 ビット AD コンバータの現在設定されている入力レンジを読み出します。本関数の呼び出しは連続サンプリングが停止した状態で行ってください。

TWXA_ADSetMode() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADSetMode(TW_HANDLE hDev, long Mode)
VB	Function TWXA_ADSetMode(ByVal hDev As System.IntPtr, ByVal Mode As TWXA_AN_MODE) As Integer
VBA	Function TWXA_ADSetMode(ByVal hDev As Long, ByVal Mode As TWXA_AN_MODE) As Long
C#	STATUS ADSetMode(System.IntPtr hDev, AN_MODE Mode)

hDev　：デバイスのハンドル
Mode　：16 ビット AD コンバータの動作モードを指定。以下の値のいずれか

言語	値	説明
C/C++	TWXA_AN_OSR_NON	オーバーサンプリング機能を使用しません（デフォルト）。
C++	TWXA::AN_MODE::OSR_NON	
VB/VBA	TWXA_AN_MODE.OSR_NON	
C#	TWXA.AN_MODE.OSR_NON	
C/C++	TWXA_AN_OSR2	オーバーサンプリングレートを 2 に設定します。
C++	TWXA::AN_MODE::OSR2	
VB/VBA	TWXA_AN_MODE.OSR2	
C#	TWXA.AN_MODE.OSR2	
C/C++	TWXA_AN_OSR4	オーバーサンプリングレートを 4 に設定します。
C++	TWXA::AN_MODE::OSR4	
VB/VBA	TWXA_AN_MODE.OSR4	
C#	TWXA.AN_MODE.OSR4	
C/C++	TWXA_AN_OSR8	オーバーサンプリングレートを 8 に設定します。
C++	TWXA::AN_MODE::OSR8	
VB/VBA	TWXA_AN_MODE.OSR8	
C#	TWXA.AN_MODE.OSR8	
C/C++	TWXA_AN_OSR16	オーバーサンプリングレートを 16 に設定します。
C++	TWXA::AN_MODE::OSR16	
VB/VBA	TWXA_AN_MODE.OSR16	
C#	TWXA.AN_MODE.OSR16	
C/C++	TWXA_AN_OSR32	オーバーサンプリングレートを 32 に設定します。
C++	TWXA::AN_MODE::OSR32	
VB/VBA	TWXA_AN_MODE.OSR32	
C#	TWXA.AN_MODE.OSR32	
C/C++	TWXA_AN_OSR64	オーバーサンプリングレートを 64 に設定します。
C++	TWXA::AN_MODE::OSR64	
VB/VBA	TWXA_AN_MODE.OSR64	
C#	TWXA.AN_MODE.OSR64	

16 ビット AD コンバータの動作モードを設定します。本関数の呼び出しは連続サンプリングが停止した状態で行ってください。
※A0800 タイプの場合、本関数は Ver.1.2.1 以降のユーザーファームが必要です。
※I2424 タイプの場合、本関数は Ver.1.4.1 以降のシステムファームが必要です。

TWXA_ADGetMode() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADGetMode(TW_HANDLE hDev, long *pMode)
VB	Function TWXA_ADGetMode(ByVal hDev As System.IntPtr, ByRef pMode As TWXA_AN_MODE) As Integer
VBA	Function TWXA_ADGetMode(ByVal hDev As Long, ByRef pMode As TWXA_AN_MODE) As Long
C#	STATUS ADGetMode(System.IntPtr hDev, out AN_MODE pMode) STATUS ADGetMode(System.IntPtr hDev, out int pMode)

hDev　：デバイスのハンドル
pMode　：[出力]取得した現在の動作モード設定値の格納先

16 ビット AD コンバータの現在設定されている動作モードを読み出します。本関数の呼び出しは連続サンプリングが停止した状態で行ってください。
※A0800 タイプの場合、本関数は Ver.1.2.1 以降のユーザーファームが必要です。
※I2424 タイプの場合、本関数は Ver.1.4.1 以降のシステムファームが必要です。

TWXA_ADStartFastSampling() USB LAN A0800

言語	関数宣言
C/C++	TW_STATUS TWXA_ADStartFastSampling(TW_HANDLE hDev, double *pRate)
VB	Function TWXA_ADStartFastSampling(ByVal hDev As System.IntPtr, ByRef pRate As Double) As Integer
VBA	Function TWXA_ADStartFastSampling(ByVal hDev As Long, ByRef pRate As Double) As Long
C#	STATUS ADStartFastSampling(System.IntPtr hDev, ref double pRate)

hDev　：デバイスのハンドル
pRate　：[入力]希望の周波数 [出力]実際に設定出来た周波数(Hz 単位)

オーバーサンプリングレート	設定可能周波数範囲[Hz]	
	USBX-A0800	LANX-A0800
未使用	1,000-200,000	1,000-50,000
2	1,000-100,000	
4	1,000-50,000	
8	1,000-25,000	
16	1,000-12,500	
32	1,000-6,250	
64	1,000-3,125	

pRate で設定した周波数でアナログ入力の連続サンプリングを開始します。ホストインタフェース、および、オーバーサンプリングレートの設定により、pRate へ設定可能な最大周波数が異なりますの注意してください。
オーバーサンプリングレートの設定は、本関数を呼び出す前に TWXA_ADSetMode()関数にて行います。
※高速でのサンプリングが可能ですが、サンプリング中のデバイスアクセスができません。

TWXA_ADStartAutoSampling() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADStartAutoSampling(TW_HANDLE hDev, double *pRate, DWORD nSampling)
VB	Function TWXA_ADStartAutoSampling(ByVal hDev As System.IntPtr, ByRef pRate As Double, ByVal nSampling As Integer) As Integer
VBA	Function TWXA_ADStartAutoSampling(ByVal hDev As Long, ByRef pRate As Double, ByVal nSampling As Long) As Long
C#	STATUS ADStartAutoSampling(System.IntPtr hDev, ref double pRate, uint nSampling) STATUS ADStartAutoSampling(System.IntPtr hDev, ref double pRate)

hDev　　：デバイスのハンドル
pRate　　：[入力]希望の周波数 [出力]実際に設定出来た周波数(Hz 単位)

オーバーサンプリングレート	設定可能周波数範囲[Hz]		
	USBX-A0800	LANX-A0800	I2424 タイプ
未使用	0.02-40,000	0.02-30,000	1-50,000
2			
4			
8	0.02-25,000		1-25,000
16	0.02-12,500		1-12,500
32	0.02-6,250		1-6,250
64	0.02-3,125		1-3,125

nSampling ： AD 変換を行う回数
0x00000000 ： 設定できません。TW_INVALID_ARGS が返ります
0xffffffff ： 停止するまで AD 変換を行います（デフォルト動作）
上記以外　 ： 指定した回数の AD 変換後に停止します

pRate で設定した周波数でアナログ入力の連続サンプリングを開始します。なお、製品タイプ、ホストインタフェース、オーバーサンプリングレートの設定により、pRate へ設定可能な最大周波数が異なりますの注意してください。
オーバーサンプリングレートの設定は、本関数を呼び出す前に TWXA_ADSetMode()関数にて行います。

nSampling で変換回数を指定することが可能です。

TWXA_ADStartAutoSamplingEx() USB LAN I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADStartAutoSamplingEx(TW_HANDLE hDev, double *pRate, DWORD nSampling, long Opt)
VB	Function TWXA_ADStartAutoSamplingEx(ByVal hDev As System.IntPtr, ByRef pRate As Double, ByVal nSampling As Integer, ByVal Opt As TWXA_AN_MODE) As Integer
VBA	Function TWXA_ADStartAutoSamplingEx(ByVal hDev As Long, ByRef pRate As Double, ByVal nSampling As Long, ByVal Opt As TWXA_AN_MODE) As Long
C#	STATUS ADStartAutoSamplingEx(System.IntPtr hDev, ref double pRate, uint nSampling, AN_MODE Opt) STATUS ADStartAutoSamplingEx(System.IntPtr hDev, ref double pRate, uint nSampling) STATUS ADStartAutoSamplingEx(System.IntPtr hDev, ref double pRate)

hDev　　：デバイスのハンドル
pRate　　：[入力]希望の周波数 [出力]実際に設定出来た周波数(Hz 単位)

オーバーサンプリングレート	設定可能周波数範囲[Hz]
未使用	1-50,000
2	
4	
8	1-25,000
16	1-12,500
32	1-6,250
64	1-3,125

nSampling ：AD 変換を行う回数
0x00000000 ： 設定できません。TW_INVALID_ARGS が返ります
0xffffffff ： 停止するまで AD 変換を行います（デフォルト動作）
上記以外　 ： 指定した回数の AD 変換後に停止します

Opt　　：動作オプションを指定。以下の値のいずれか

言語	値	説明
C/C++	TWXA_AN_TRIG_NON	本関数の呼び出しで、直ちに連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_NON	
VB/VBA	TWXA_AN_MODE.TRIG_NON	
C#	TWXA.AN_MODE.TRIG_NON	
C/C++	TWXA_AN_TRIG_OFF_TO_ON	AD トリガ入力端子の状態が[OFF]から[ON]になると連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_OFF_TO_ON	
VB/VBA	TWXA_AN_MODE.TRIG_OFF_TO_ON	
C#	TWXA.AN_MODE.TRIG_OFF_TO_ON	
C/C++	TWXA_AN_TRIG_ON_TO_OFF	AD トリガ入力端子の状態が[ON]から[OFF]になると連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_ON_TO_OFF	
VB/VBA	TWXA_AN_MODE.TRIG_ON_TO_OFF	
C#	TWXA.AN_MODE.TRIG_ON_TO_OFF	
C/C++	TWXA_AN_TRIG_BOTH	AD トリガ入力端子の状態が変化すると連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_BOTH	
VB/VBA	TWXA_AN_MODE.TRIG_BOTH	
C#	TWXA.AN_MODE.TRIG_BOTH	

pRate で設定したサンプリングレートでアナログ入力の連続サンプリングを開始します。なお、オーバーサンプリングレートの設定により、pRate へ設定可能な最大周波数が異なりますの注意してください。
オーバーサンプリングレートの設定は、本関数を呼び出す前に TWXA_ADSetMode()関数にて行います。

nSampling で変換回数を指定することが可能です。
Opt 引数には連続サンプリングの開始タイミングを指定します。
※この関数は Ver.1.4.1 以降のシステムファームが必要です。

TWXA_ADStartExtSyncSampling() USB LAN I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADStartExtSyncSampling(TW_HANDLE hDev, DWORD nSampling, long Opt)
VB	Function TWXA_ADStartExtSyncSampling(ByVal hDev As System.IntPtr, ByVal nSampling As Integer, ByVal Opt As TWXA_AN_MODE) As Integer
VBA	Function TWXA_ADStartExtSyncSampling(ByVal hDev As Long, ByVal nSampling As Long, ByVal Opt As TWXA_AN_MODE) As Long
C#	STATUS ADStartExtSyncSampling(System.IntPtr hDev, uint nSampling, AN_MODE Opt) STATUS ADStartExtSyncSampling(System.IntPtr hDev, uint nSampling) STATUS ADStartExtSyncSampling(System.IntPtr hDev)

hDev　　：デバイスのハンドル
nSampling：AD 変換を行う回数
　0x00000000 ：設定できません。TW_INVALID_ARGS が返ります
　0xffffffff ：停止するまで AD 変換を行います（デフォルト動作）
　上記以外　：指定した回数の AD 変換後に停止します

Opt　　：動作オプションを指定。以下の値からサンプリング信号と開始トリガに関する設定を
　1 つずつ選択して OR で結合します。

言語	値	説明
C/C++	TWXA_AN_SYNC_OFF_TO_ON	AD クロック入力端子が[OFF]から[ON]になるとサンプリングを 1 回行います。
C++	TWXA::AN_MODE::SYNC_OFF_TO_ON	
VB/VBA	TWXA_AN_MODE.SYNC_OFF_TO_ON	
C#	TWXA.AN_MODE.SYNC_OFF_TO_ON	
C/C++	TWXA_AN_SYNC_ON_TO_OFF	AD クロック入力端子が[ON]から[OFF]になるとサンプリングを 1 回行います。
C++	TWXA::AN_MODE::SYNC_ON_TO_OFF	
VB/VBA	TWXA_AN_MODE.SYNC_ON_TO_OFF	
C#	TWXA.AN_MODE.SYNC_ON_TO_OFF	
C/C++	TWXA_AN_TRIG_NON	本関数の呼び出しで、直ちに連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_NON	
VB/VBA	TWXA_AN_MODE.TRIG_NON	
C#	TWXA.AN_MODE.TRIG_NON	
C/C++	TWXA_AN_TRIG_OFF_TO_ON	AD トリガ入力端子の状態が[OFF]から[ON]になると連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_OFF_TO_ON	
VB/VBA	TWXA_AN_MODE.TRIG_OFF_TO_ON	
C#	TWXA.AN_MODE.TRIG_OFF_TO_ON	
C/C++	TWXA_AN_TRIG_ON_TO_OFF	AD トリガ入力端子の状態が[ON]から[OFF]になると連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_ON_TO_OFF	
VB/VBA	TWXA_AN_MODE.TRIG_ON_TO_OFF	
C#	TWXA.AN_MODE.TRIG_ON_TO_OFF	
C/C++	TWXA_AN_TRIG_BOTH	AD トリガ入力端子の状態が変化すると連続サンプリングを開始します。
C++	TWXA::AN_MODE::TRIG_BOTH	
VB/VBA	TWXA_AN_MODE.TRIG_BOTH	
C#	TWXA.AN_MODE.TRIG_BOTH	

外部クロック入力に同期した連続サンプリングを開始します。なお、オーバーサンプリングレートの設定により、入力可能な最大周波数が異なりますので注意してください。
オーバーサンプリングレートの設定は、本関数を呼び出す前に TWXA_ADSetMode()関数にて行います。

オーバーサンプリングレート	入力可能最大周波数[Hz]
未使用	50,000
2	
4	
8	25,000
16	12,500
32	6,250
64	3,125

nSampling で変換回数を指定することが可能です。
Opt 引数にはサンプリングを行う外部信号の選択、および、連続サンプリングの開始タイミングを指定します。
※この関数は Ver.1.4.1 以降のシステムファームが必要です。

TWXA_ADStopSampling() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADStopSampling(TW_HANDLE hDev)
VB	Function TWXA_ADStopSampling(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_ADStopSampling(ByVal hDev As Long) As Long
C#	STATUS ADStopSampling(System.IntPtr hDev)

hDev　：デバイスのハンドル

連続サンプリングを停止します。

TWXA_ADGetQueueStatus() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADGetQueueStatus(TW_HANDLE hDev, int *pStatus, long *pnReceive)
VB	Function TWXA_ADGetQueueStatus(ByVal hDev As System.IntPtr, ByRef pStatus As Integer, ByRef pnReceive As Integer) As Integer
VBA	Function TWXA_ADGetQueueStatus(ByVal hDev As Long, ByRef pStatus As Long, ByRef pnReceive As Long) As Long
C#	STATUS ADGetQueueStatus(System.IntPtr hDev, out int pStatus, out int pnReceive)

hDev　：デバイスのハンドル
pStatus　：[出力]連続サンプリングの動作状態
　ビット０：TWXA_ADStartFastSampling()関数による連続サンプリングが動作状態のとき１となります
　ビット１：TWXA_ADStartAutoSampling()関数、TWXA_ADStartAutoSamplingEx()関数、TWXA_ADStartExtSyncSampling()関数のいずれかによる連続サンプリングが動作状態のとき１となります。
pnReceive：パソコン上の受信バッファに蓄えられたサンプリングデータのデータ数

連続サンプリングの動作状態と受信データ数を取得します。pStatus のビット 1 については、TWXA_ADStartAutoSampling()関数、TWXA_ADStartAutoSamplingEx()関数、TWXA_ADStartExtSyncSampling()関数のいずれかでサンプリング回数が指定されている場合、指定回数に到達すると０にクリアされます。

TWXA_AOxOx_DATA 構造体

言語	宣言

C/C++	typedef struct tagTwxaAOxOxData
	{
	  DWORD Index;
	  short Data[8];
	} TWXA_AOxOx_DATA;

VB	Public Structure TWXA_AOxOx_DATA
	  Public Index As Integer
	  <MarshalAs(UnmanagedType.ByValArray, SizeConst:=8)> _
	  Public Data() As Short
	
	  Public Sub Initialize()
	  ReDim Data(7)
	  End Sub
	End Structure

VBA	Public Type TWXA_AOxOx_DATA
	  Index As Long
	  Data(7) As Integer
	End Type

C#	public struct AOxOx_DATA
	{
	  public uint Index;
	  [MarshalAs(UnmanagedType.ByValArray, SizeConst = 8)]
	  public short[] Data;
	
	  public void Initialize()
	  {
	  Data = new short[8];
	  }
	}

Index ： データのインデックス
Data ： アナログ入力の各チャンネルの AD 変換値。配列のインデックスとチャンネルが一致
A0800 タイプの場合 ： インデックス 0～7 が有効
I2424 タイプの場合 ： インデックス 0～3 が有効、4～7 は常に 0

デバイスから受信したサンプリングデータを読み出すための構造体です。TWXA_ADReadBuffer()関数の呼び出しに使用します。
Visual Basic と C# では構造体を使用する前に Initialize()メソッドを呼び出して初期化を行ってください。

TWXA_ADReadBuffer() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADReadBuffer(TW_HANDLE hDev, void *pData, long nData, long *pnRead)
VB	Function TWXA_ADReadBuffer(ByVal hDev As System.IntPtr, ByVal pData() As TWXA_AOxOx_DATA, ByVal nData As Integer, ByRef pnRead As Integer) As Integer
VBA	Function TWXA_ADReadBuffer(ByVal hDev As Long, ByRef pData As Any, ByVal nData As Long, ByRef pnRead As Long) As Long
C#	STATUS ADReadBuffer(System.IntPtr hDev, AOxOx_DATA []pData, int nData, out int pnRead)

hDev　：デバイスのハンドル
pData　：[出力]読み出したデータの格納先
nData　：読み出すデータ数(0～65536)
pnRead　：[出力]実際に読み出したデータ数の格納先

パソコン内のバッファに蓄えられている、デバイスから受信したサンプリングデータを読み出します。

TWXA_ADPurgeBuffer() USB LAN A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ADPurgeBuffer(TW_HANDLE hDev)
VB	Function TWXA_ADPurgeBuffer(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_ADPurgeBuffer(ByVal hDev As Long) As Long
C#	STATUS ADPurgeBuffer(System.IntPtr hDev)

hDev　：デバイスのハンドル

デバイスから受信したサンプリングデータを蓄えるパソコン内のバッファをクリアします。本関数の呼び出しは連続サンプリングが停止した状態で行ってください。

□ その他の関数

TWXA_Initialize() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_Initialize(TW_HANDLE hDev, long Opt)
VB	Function TWXA_Initialize(ByVal hDev As System.IntPtr, ByVal Opt As TWXA_INIT_OPT) As Integer
VBA	Function TWXA_Initialize(ByVal hDev As Long, ByVal Opt As TWXA_INIT_OPT) As Long
C#	STATUS Initialize(System.IntPtr hDev, INIT_OPT Opt)

hDev ： デバイスのハンドル
Opt ： 初期化する機能を指定。以下の値を OR で結合

言語	値	説明
C/C++	TWXA_INIT_ALL	全機能を初期化します。
C++	TWXA::INIT_OPT::ALL	
VB/VBA	TWXA_INIT_OPT.ALL	
C#	TWXA.INIT_OPT.ALL	
C/C++	TWXA_INIT_PORT_DATA	デジタル出力端子を初期化します。
C++	TWXA::INIT_OPT::PORT_DATA	
VB/VBA	TWXA_INIT_OPT.PORT_DATA	
C#	TWXA.INIT_OPT.PORT_DATA	
C/C++	TWXA_INIT_TIMER	16 ビットタイマ(ハードウェアカウンタ)を初期化します。
C++	TWXA::INIT_OPT::TIMER	
VB/VBA	TWXA_INIT_OPT.TIMER	
C#	TWXA.INIT_OPT.TIMER	
C/C++	TWXA_INIT_SCI	シリアルポートを初期化します。
C++	TWXA::INIT_OPT::SCI	
VB/VBA	TWXA_INIT_OPT.SCI	
C#	TWXA.INIT_OPT.SCI	
C/C++	TWXA_INIT_PC	パルスカウンタ(ソフトウェアカウンタ)を初期化します。
C++	TWXA::INIT_OPT::PC	
VB/VBA	TWXA_INIT_OPT.PC	
C#	TWXA.INIT_OPT.PC	
C/C++	TWXA_INIT_DA	アナログ出力端子を初期化します。
C++	TWXA::INIT_OPT::DA	
VB/VBA	TWXA_INIT_OPT.DA	
C#	TWXA.INIT_OPT.DA	

デバイスの初期化を行います。デバイスは起動時に初期化されますので呼び出しは必須ではありません。
Opt 引数で初期化する機能を指定できます。
ユーザー用ステータスレジスタは Opt によらず 0 に初期化されます。

TWXA_ReadVersion() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_ReadVersion(TW_HANDLE hDev, DWORD *pVersion)
VB	Function TWXA_ReadVersion(ByVal hDev As System.IntPtr, ByRef pVersion As Integer) As Integer
VBA	Function TWXA_ReadVersion(ByVal hDev As Long, ByRef pVersion As Long) As Long
C#	STATUS ReadVersion(System.IntPtr hDev, out uint pVersion) STATUS ReadVersion(System.IntPtr hDev, out int pVersion)

hDev　　：デバイスのハンドル
pVersion：[出力]バージョン情報の格納先

ファームウェアのバージョン情報を読み出します。pVersion に読み出された値は、最上位バイトが予約、次いでメジャーバージョン、マイナーバージョン、最下位バイトがリビジョンを示します。

ビット	31-24	23-16	15-8	7-0
意味	予約(0)	メジャーバージョン	マイナーバージョン	リビジョン

ファームウェアのバージョンが 1.2.3 の場合、格納される値は 0x00010203 となります。

TWXA_SetTimeouts() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SetTimeouts(TW_HANDLE hDev, DWORD dwReadTimeout, DWORD dwWriteTimeout)
VB	Function TWXA_SetTimeouts(ByVal hDev As System.IntPtr, ByVal dwReadTimeout As Integer, ByVal dwWriteTimeout As Integer) As Integer
VBA	Function TWXA_SetTimeouts(ByVal hDev As Long, ByVal dwReadTimeout As Long, ByVal dwWriteTimeout As Long) As Long
C#	STATUS SetTimeouts(System.IntPtr hDev, uint dwReadTimeout, uint dwWriteTimeout) STATUS SetTimeouts(System.IntPtr hDev, int dwReadTimeout, int dwWriteTimeout)

hDev　　：デバイスのハンドル
dwReadTimeout　：読出しのタイムアウト時間(msec 単位)
dwWriteTimeout：書込みのタイムアウト時間(msec 単位)

デバイスからデータの読出し、デバイスへの書込みの際のタイムアウト時間を設定します。デフォルトではどちらも 5000msec に設定されています。

TWXA_GetNumber() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	long TWXA_GetNumber(TW_HANDLE hDev)
VB	Function TWXA_GetNumber(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_GetNumber(ByVal hDev As Long) As Long
C#	int GetNumber(System.IntPtr hDev)

hDev ：デバイスのハンドル

戻り値 ：接続中のデバイスの装置番号

接続中のデバイスの装置番号を返します。ハンドルが無効な場合には 0 を返します。

TWXA_SetPassword() LAN I2219 I0800 I0404 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SetPassword(LPCTSTR pPass)
VB	Function TWXA_SetPassword(ByVal pPass As String) As Integer
VBA	Function TWXA_SetPassword(ByVal pPass As String) As Long
C#	STATUS SetPassword(string pPass)

pPass ： パスワード文字列

LAN デバイスと接続する際のパスワードを入力します。パスワードは設定ツールでデバイスに書き込んだものと同じものを指定してください。pPass が NULL または ""の場合、デフォルトのパスワードが使用されます。

TWXA_SetNetworkPort() LAN I2219 I0800 I0404 A0800 I2424

言語	関数宣言
C/C++	TW_STATUS TWXA_SetNetworkPort(DWORD PortNumber)
VB	Function TWXA_SetNetworkPort(ByVal PortNumber As Integer) As Integer
VBA	Function TWXA_SetNetworkPort(ByVal PortNumber As Long) As Long
C#	STATUS SetNetworkPort(int PortNumber)

PortNumber ： デバイスとの接続に使用するポート番号

LAN デバイスと接続する際に使用するポート番号を変更します。この関数でポート番号を変更した場合、以後のそのプロセスからの接続は全て設定したポート番号に対して行われます。
ポート番号のデフォルト値は 49152 です。設定ツールでデバイスに設定したポート番号と同じ値を指定してください。

TWXA_GetSocket() LAN I2219 I0800 I0404 A0800 I2424

言語	関数宣言
C/C++	UINT_PTR TWXA_GetSocket(TW_HANDLE hDev)
VB	Function TWXA_GetSocket(ByVal hDev As System.IntPtr) As System.IntPtr
VBA	Function TWXA_GetSocket(ByVal hDev As Long) As Long
C#	System.IntPtr GetSocket(System.IntPtr hDev)

hDev ： デバイスのハンドル

戻り値としてハンドルに結びついた SOCKET を返します。LAN デバイスでは無い場合や無効なハンドルが渡された場合は TW_INVALID_SOCKET が返ります。

TWXA_GetDeviceType() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	long TWXA_GetDeviceType(TW_HANDLE hDev)
VB	Function TWXA_GetDeviceType(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_GetDeviceType(ByVal hDev As Long) As Long
C#	uint GetDeviceType(System.IntPtr hDev)

hDev ： デバイスのハンドル

戻り値 ： 接続中のデバイスの製品タイプ
製品タイプとして以下のビットのいずれかが 1 になります
TWXA_TYPE_I2219(0x00020000) ： I2219 タイプ
TWXA_TYPE_I0x0x(0x00040000) ： I0800、I0404、I0008 いずれかのタイプ
TWXA_TYPE_A0x0x(0x00080000) ： A0800 タイプ
TWXA_TYPE_I2424(0x00100000) ： I2424 タイプ

接続中のデバイスの製品タイプを返します。ハンドルが無効な場合には 0 を返します。

TWXA_GetIFType() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

言語	関数宣言
C/C++	long TWXA_GetIFType(TW_HANDLE hDev)
VB	Function TWXA_GetIFType(ByVal hDev As System.IntPtr) As Integer
VBA	Function TWXA_GetIFType(ByVal hDev As Long) As Long
C#	uint GetIFType(System.IntPtr hDev)

hDev ： デバイスのハンドル

戻り値 ： 接続中のデバイスのインタフェース種別
インタフェース種別として以下のビットのいずれかが 1 になります
TWXA_IF_USB(0x20000000) ： USB デバイス
TWXA_IF_LAN(0x40000000) ： LAN デバイス

接続中のデバイスのインタフェースタイプを返します。ハンドルが無効な場合には 0 を返します。

□ VBA 用ヘルパー関数

以下の関数は VBA のプリミティブな変数と、16 ビットの符号無し整数を変換するためのヘルパー関数です。16 ビットタイマのカウンタ値を符号無し整数として扱いたい場合などに使用します。

TWXA_ToUINT16() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

```
Function TWXA_ToUINT16(ByVal data As Long) As Integer
```

data ： 16 ビットコードに変換したい値

0～65535 の値を 16 ビットコード(0000h～FFFFh)に変換します。
data が負の場合は 0 として、また 65535 を超える場合は 65535 として変換します。

TWXA_ToINT32() USB LAN I2219 I0800 I0404 I0008 A0800 I2424

```
Function TWXA_ToINT32(ByVal data As Integer) As Long
```

data ： 32 ビット値に変換したい値

data を符号無し 16 ビットコード(0000h～FFFFh)とみなし、32 ビット値(0～65535)に変換します。

サポート情報

製品に関する情報、最新のファームウェア、ユーティリティなどは弊社ホームページにてご案内しております。また、お問い合わせ、ご質問などは下記までご連絡ください。

テクノウェーブ(株)
URL ： http://www.techw.co.jp
E-mail ： support@techw.co.jp

(1) 本書、および本製品のホームページに掲載されている応用回路、プログラム、使用方法などは、製品の代表的動作・応用例を説明するための参考資料です。これらに起因する第三者の権利(工業所有権を含む)侵害、損害に対し、弊社はいかなる責任も負いません。

(2) 本書の内容の一部または全部を無断転載することをお断りします。

(3) 本書の内容については、将来予告なしに変更することがあります。

(4) 本書の内容については、万全を期して作成いたしましたが、万一ご不審な点や誤り、記載もれなど、お気づきの点がございましたらご連絡ください。

改訂記録

年月	版	改訂内容
2011 年 12 月	初	
2012 年 4 月	2	・C# に対応
2013 年 6 月	3	・Ver.1.1.0.1 以降のライブラリバージョンに対応 ・誤記の修正
2014 年 10 月	4	・Ver.1.3.0.1 以降のライブラリバージョンに対応 ・FRAM の操作に関する記述を追加
2016 年 3 月	5	・Ver.1.4.0.1 以降のライブラリバージョンに対応 ・16 ビット AD コンバータの操作に関する記述を追加 ・誤記の修正
2018 年 10 月	6	・Ver.1.5.0.1 以降のライブラリバージョンに対応 ・対応製品を追加
2020 年 8 月	7	・対応 OS を変更 ・16 ビット AD コンバータの操作に関する追加機能に対応 ・誤記の修正