REX-USB220

USB2.0 to GPIB Converter

ユーザーズマニュアル

2021 年 11 月

第 9.0 版

RATOC Systems, Inc.

ラトックシステム株式会社

目　　次

I&L サポートセンターへのお問い合わせ

技術的なご質問やご相談の窓口を用意していますのでご利用ください。

ラトックシステム株式会社

I&L サポートセンター

〒550-0015　大阪市西区南堀江 1-18-4 Osaka Metro 南堀江ビル　8F

TEL 06-7670-5064

FAX 06-7670-5066

https://www.ratocsystems.com/mail/support.html

〈サポート対応時間〉

月曜ー金曜（祝祭日は除く）　10:00 - 13:00

14:00 - 17:00

【ご注意】

☑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。

☑本書の内容につきましては万全を期して作成しましたが、万一ご不審な点や誤りなどお気づきになられましたら I&L サポートセンターまでご連絡願います。

☑本製品および本製品添付のマニュアルに記載されている会社名および製品名は、各社の商品または登録商標です。

☑本製品の運用を理由とする損失、免失利益などの請求につきましては、いかなる責任も負いかねますので予めご了承願います。

第１章 ご使用になる前に

この章では、本製品の特徴並びに製品仕様について説明します。

1-1. はじめに

このたびは、REX-USB220 USB2.0 to GPIB Converter をご購入いただきましてありがとうございます。REX-USB220 USB2.0 to GPIB Converter（以下、REX-USB220 コンバータと呼ぶ）は USB Specification2.0 の 480Mbps ハイスピード仕様準拠の USB-GPIB コンバータです。専用のアプリケーションを構築することにより、GPIB インターフェイスを持たないパソコンで、GPIB インターフェイスの各社計測器を制御することが可能です。

（特徴）

■ DMA 転送サポート

DMA 転送をサポートすることにより、USB と GPIB の間で、高速一括転送が可能となりました。（最大 900KByte/Sec）

■ アプリケーション開発支援

Microsoft VC/VB対応のライブラリとサンプルプログラムを添付。

Windows 11/10/8.1/7/Vista/XP/2000/Me/98SE 環境でのプログラム開発を容易に進めることができます。

1-2. 梱包内容のご確認

内 容	個 数	備 考
REX-USB220 コンバータ本体 インストールガイド 保証書	1 1 1	

万一、不足品等ございましたら I&L サポートセンターまでご連絡願います。

1-3. 製品仕様

1.3.1 ハードウェア仕様

項　目	仕　様	備　考
GPIB コントローラ	National Instruments 製 NAT9914 互換	
接続コネクタ	IEEE488 インターフェースコネクタ	
電源供給方式	バスパワード方式	
消費電流	MAX 350mA(動作時)	
USB 規格	USB Specification Revision2.0 準拠	
USB 接続コネクタ	TypeA	
外形寸法	W61×L62×H22、ケーブル長 1500mm	
重量	約 105g	
動作温湿度	0～55℃	但し、結露しないこと

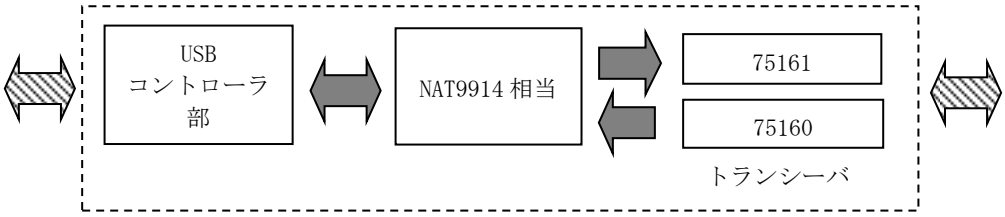
1.3.2 GPIB コネクタピンアサイン（アンフェノール 24 ピンオス）

端子番号	信号名	説明	端子番号	信号名	説明
1	DIO1	Data Line	13	DIO5	Data Line
2	DIO2	Data Line	14	DIO6	Data Line
3	DIO3	Data Line	15	DIO7	Data Line
4	DIO4	Data Line	16	DIO8	Data Line
5	EOI	End-or-Identify	17	REN	Remote Enable
6	DAV	Data Valid	18	GND	Ground
7	NRFD	Not Ready For Data	19	GND	Ground
8	NDAC	Not Data Accepted	20	GND	Ground
9	IFC	Interface Clear	21	GND	Ground
10	SRQ	Service Request	22	GND	Ground
11	ATN	Attention	23	GND	Ground
12	NC	No Connect	24	GND	Ground

1.3.3 ソフトウェア環境

項　目	仕　様	備　考
対応 OS	Windows 11/10/8.1/7/Vista/XP/2000/Me/98SE	64-bit 版 OS にも対応
製品付属ドライバ	OHCI/UHCI/EHCI 準拠 WDM ドライバ	
製品付属ライブラリ	VC 用 32/64 ビット DLL ライブラリ VB 用 32 ビット ActiveX	
対応開発言語	Microsoft Visual C/C++ Microsoft Visual BASIC	

1.3.4 ブロック図



1-4. USB220 の GPIB インターフェイス機能

GPIB には、下記の 10 種類のインターフェイス機能が定められています。そして、実際には、これらの機能のうち必要なものを選択して組合せて使用します。GPIB 機器やコントローラ(パソコン)を選択する場合には、この機能コードをあらかじめ調べておく必要があります。その機能を持っているかどうかということ、どのレベルまでの機能を持っているかということは、SR0, C4 のような機能シンボルコードと 0～9 の数字の組み合わせで示され、0 はその機能を持たないことを示します。

機能ｼﾝﾎﾞﾙ	インターフェイス	機　　能
ｺｰﾄﾞ	機能	
SH	ｿｰｽﾊﾝﾄﾞｼｪｲｸ	バス上のデータを送信する
AH	ｱｸｾﾌﾟﾀﾊﾝﾄﾞｼｪｲｸ	バス上のデータを受信する
T	ﾄｰｶ	SH機能を使って、他の装置にデータを送る
L	ﾘｽﾅ	AH機能を使って、他の装置からデータを受け取る
C	ｺﾝﾄﾛｰﾗ	バス上にコマンドを送り出して、GPIB システムをコントロールする
DT	ﾃﾞﾊﾞｲｽﾄﾘｶﾞ	トリガコマンドを受信し、装置をトリガする
DC	ﾃﾞﾊﾞｲｽｸﾘｱ	クリアコマンドを受信し、装置をリセットする
PP	ﾊﾟﾗﾚﾙﾎﾟｰﾙ	コントローラのパラレルポールに応答する
SR	ｻｰﾋﾞｽﾘｸｴｽﾄ	コントローラに対し SRQ を送り出す
RL	ﾘﾓｰﾄ･ﾛｰｶﾙ	コントローラからの指令により装置のリモートとローカル状態とを切りかえる

GPIB では、すべての機器がバスに対して、並列に接続されています。したがってバス上のデータは、L(リスナ)機能をもつ装置であれば同時に受信することができます。しかし送信(バス上へのデータの送り出し)は、必ずどれか一台のみしか行えません。

バス上でデータの衝突(同時に２台以上がトーカとなる)が発生したり、受信データの指定などを行うために GPIB システムでは、コントローラ(C)機能が用意され各装置にはアドレスが割付けられます。通常のシステムでは、コントローラはバス上に１台のみ存在します。

[REX-USB220 の機能表]　パソコンをコントローラとしてのみ使用します。

機　能	サブセット	内　　容
SH	SH1	ソースハンドシェイク機能を持つ
AH	AH1	アクセプタハンドシェイク機能を持つ
C	C1	コントローラ機能を持つ
	C2	コントローラインチャージ機能を持つ
	C3	リモートイネーブル機能を持つ
	C4	SRQ に対する応答機能を持つ
	C28	インターフェイスメッセージ送信機能を持つ
T	T8	基本的なトーカ機能を持つ MLA によってトーカ機能が解除される
L	L4	基本的なリスナ機能を持つ MTA によりリスナ機能が解除される
SR	SR0	システムコントローラとしてのみ動作しますので これらの機能はありません。
RL	RL0	
PP	PP0	
DC	DC0	
DT	DT0	

REX-USB220 コンバータは、パソコンをコントローラとして機能させるためのインターフェイスセットで、GPIB バス上において他のコントローラとの同居はできません。従って REX-USB220 と同時に GPIB 上で使用できる機器は、下記の機能を持つ装置に限られます。

- アドレス可能な装置であること。
- コントローラ機能を持たない（Ｃ０）こと。

(ATN, IFC, REN ラインの管理機能を持たないこと)

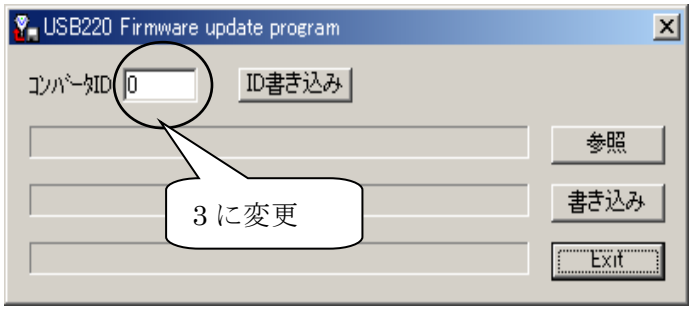
但し、REX-USB220 を複数台パソコンに接続して、それぞれの GPIB バスを管理するコントローラとして使用することは可能です。

1-5. 2 台以上の USB220 をご使用になる場合

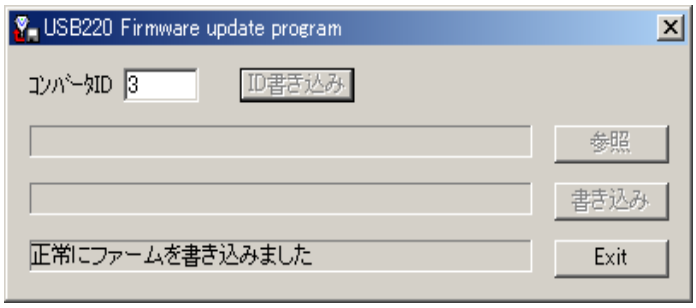
本コンバータをパソコンに複数台接続されてご使用になる場合には、個々のコンバータに対してユニークな ID を 0～255 の範囲で設定しておく必要があります。
ダウンロード提供のファームウェアアップデートプログラムにより、本体のコンバータ ID を変更することが可能です。なお、コンバータ ID は製品出荷時「0」にセットされています。

（注意！） REX-USB220 コンバータはコントローラとして機能するため、GPIB バス上で他のコントローラとの同居はできませんので、ご注意ください。

（コンバータ ID の変更方法）



パソコンの USB ポートには本コンバーター台のみ接続してください。HUB 等を使用して複数の USB 機器を接続されている場合は、全て取り外しておいてください。
ファームウェアアップデートプログラム Firmup.exe 起動後、左図が表示されます。「コンバータ ID」には、現在接続されているコンバータの ID が表示されますので、変更したい番号を記述します。本例では 0 から 3 に ID を変更します。
値を変更後、「ID 書き込み」ボタンを押します。



左図が表示されれば、コンバータ ID の変更は完了です。「Exit」ボタンを押して、プログラムを終了してください。
プログラム終了後、USB のケーブルを一旦取り外し、再度、接続してください。

1-6. 使用上の注意

1.6.1 USB の制約について

USB2.0 は 480Mbps のバンド幅を持つトークンパケット方式の高速シリアル通信インターフェイスです。しかしながら、実際の運用環境では本製品以外にアイソクロナス転送を使って膨大な量のデータを通信するデバイスの他に USB キーボード、USB マウス等が接続されていることが考えられます。このような複合環境では、アイソクロナス転送を行うデバイスのバンド幅は保証されますが、バルク転送を使ってデータ転送を行う REX-USB220 のバンド幅は保証されていません。これにより、作成した REX-USB220 のアプリケーションのレスポンスが思うように得られなかったり、一時的に転送レートが落ちる場合があります。本問題の有無については、運用される環境により異なりますのでお客様の実環境にて事前に評価する必要があります。

1.6.2 取り扱い上の注意

本製品を取り扱う場合は以下の注意事項を守ってください

1. 本製品を下記のような環境で使用もしくは保管しないでください。
   直射日光のあたる場所や、高温になる場所
   内部に異物（水・ごみ）の入りやすい場所
   湿気の多い場所や結露しやすい場所
   強い磁界・電波を発生する機器の近くや、静電気の影響の多い場所
   振動・衝撃の加わる場所
2. 外部電源電圧やドライブ電流等の定格を超えないように注意してください。
3. 本製品を改造しないでください。改造を行ったものについては一切の責任を負いません。

この装置は、情報処理装置等電波障害自主規制協議会（ＶＣＣＩ）の基準に基づくクラスＡ情報技術装置です。この装置を家庭環境で使用すると電波妨害を引き起こすことがあります。この場合には使用者が適切な対策を講ずるよう要求されることがあります。

Warning

This is a class A product. In a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.

-FCC Statement-

This device complies with Part 15 of the FCC Rules. Operation is

Subject to the following two conditions: (1) this device may not cause

Harmful interference, and (2) this device must accept any interference

Received , including interference that may cause undesired operation.

第 2 章 インストレーション

この章では、Windows でのインストール方法について説明します。

● ドライバーのダウンロード

ホームページ右上の検索欄に「USB220 ダウンロード」と入力し検索します。

https://www.ratocsystems.com/

下記ダウンロードページへのリンクをクリックし、表示されたページのインストーラーをダウンロードします。

2-1. インストール

2.1.1 ドライバーのセットアップ

まず、ダウンロードしたインストーラーを実行し、ドライバーのセットアップを行います。

ユーザーアカウント制御の画面が表示される場合は、「はい(Y)」をクリックします。

「次へ(N)」をクリックします。

「このデバイスソフトウェアをインストールしますか？」の発行元確認画面が表示される場合は
「インストール(I)」をクリックします。

以上でドライバーのセットアップは完了です。

「完了」をクリックして Windows を再起動し、「2.1.2 デバイスのインストール」へ進んでください。

2.1.2 デバイスのインストール

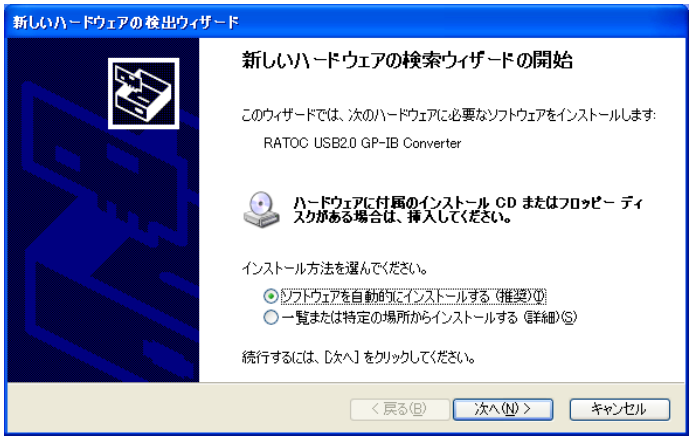
(本作業を行う前に、2.1.1 項 「ドライバーのインストール」を行ってください。)

1) Windows 11/10/8.1/7/Vista/2000/Me/98SE の場合

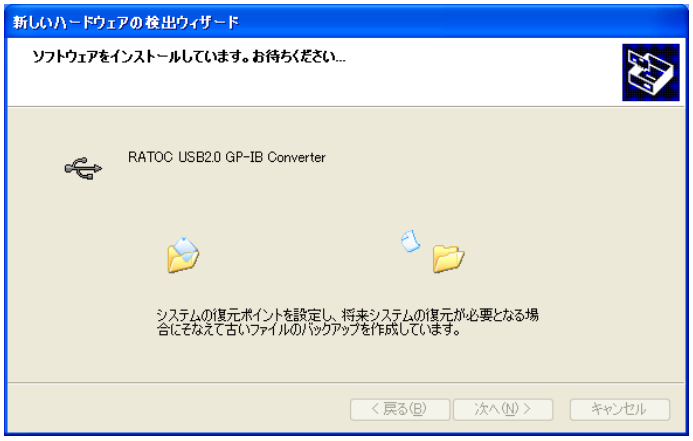
本製品を USB ポートに接続すると、自動的にインストールが行われます。引き続き、インストールの確認を行ってください。

2) WindowsXP x32 の場合

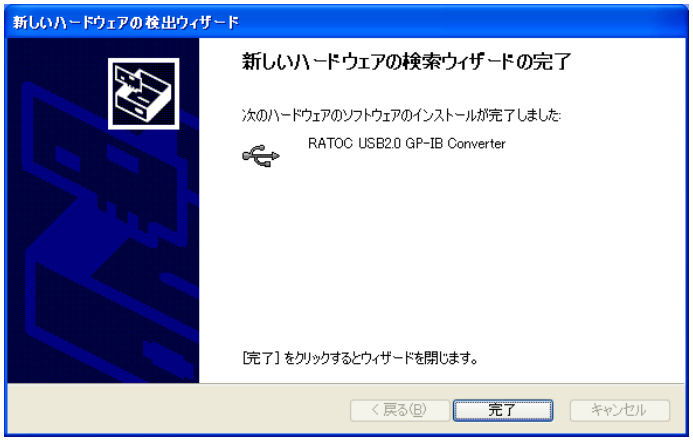
本製品をはじめて USB ポートに接続すると、左下の画面が表示されます。下記に従ってデバイスのインストールを行ってください。



「ソフトウェアを自動的にインストールする(推奨)」を選んで、「次へ(N)」をクリックします。



ドライバーを自動検索し、インストールが行われます。



「完了」をクリックします。

以上でデバイスのインストールは完了です。

3) WindowsXP x64 の場合

本製品をはじめて USB ポートに接続すると、左下の画面が表示されます。

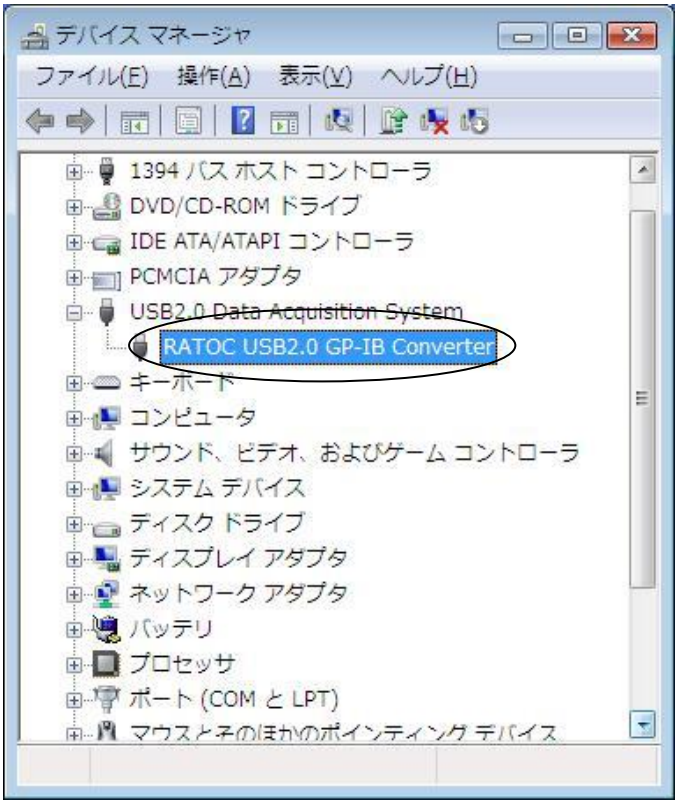
下記に従ってデバイスのインストールを行ってください。

「はい(Y)」をクリックします。

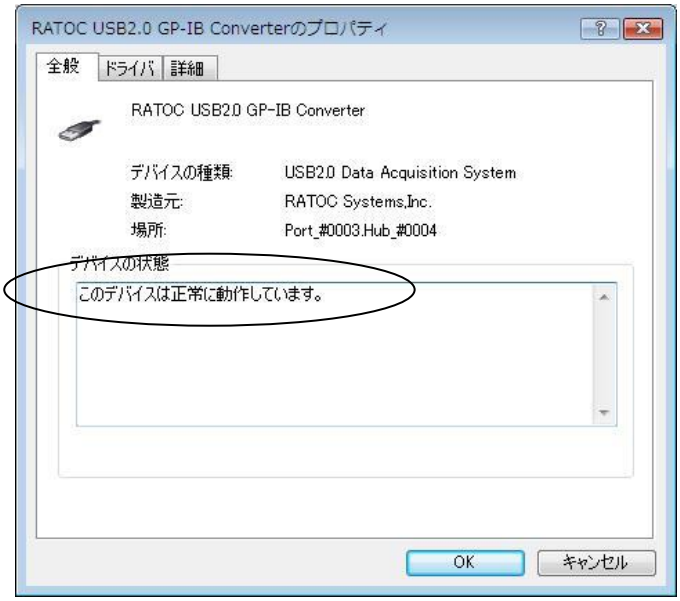
以上でインストールは完了となります。

2.1.3 インストールの確認

コントロールパネルのデバイスマネージャーを起動します。
（Windows XP/2000 では、コントロールパネルのシステムを起動し「ハードウェア」のタブから「デバイスマネージャ」のボタンをクリックします。
Windows Me/98SE では、コントロールパネルのシステムを起動し「デバイスマネージャ」のタブをクリックします。）



左の画面のようにデバイス一覧から USB2.0 Data Acquisition System クラスに「RATOC USB2.0 GP-IB Converter」が登録されているか確認します。



上記より、「RATOC USB2.0 GP-IB Converter」をダブルクリックしてプロパティを開き、左図のようにデバイスの状態に「このデバイスは正常に動作しています」と表示されていることを確認します。

以上で、インストールの確認は終了です。

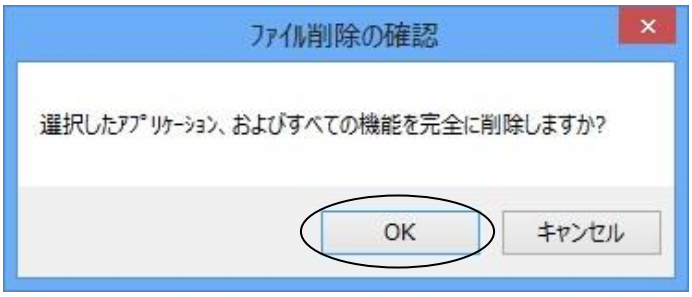
2-2. アンインストールについて

インストールした内容を削除する方法について説明します。

コントロールパネルの「プログラムと機能」を起動します。

(Windows XP では「プログラムの追加と削除」、Windows 2000/Me/98SE では「アプリケーションの追加と削除」を起動します。)

「RATOC I&L USB2.0 Series」を選択し「アンインストール」(または「削除」をクリックします。



「OK」をクリックします。

「完了」をクリックします。

以上でアンインストールは完了です。

第 3 章 Windows アプリケーション作成

この章では、ライブラリ関数仕様とサンプルプログラムについて説明します。

● サンプルプログラム/ライブラリのダウンロード

ホームページ右上の検索欄に「USB220 ダウンロード」と入力し検索します。

https://www.ratocsystems.com/

下記ダウンロードページへのリンクをクリックし、表示されたページの
「REX-USB220 用サンプルプログラム(VB/VC)」「REX-USB220 用ライブラリ」
をダウンロードします。

3-1. ライブラリの呼び出し方法

3.1.1 VC からの呼び出し方法

Visual C/C++のアプリケーションから DLL ライブラリ「U2GPLIB.DLL」の API を呼び出すには以下の二つの作業が必要です。

(1) DLL ヘッダーファイルのインクルード

ダウンロードした VC 用サンプルプログラム内の「U2GPLIB.H」を作成されたプロジェクトにコピーし、アプリケーションプログラムにインクルードします。

(2) DLL ライブラリファイルのプロジェクト追加

ダウンロードしたライブラリファイル(U2GPLIB.LIB)を作成されたプロジェクトにコピーし、プロジェクトメニューの「プロジェクトへ追加」->「ファイル」を選択し、ファイルの種類「ライブラリファイル(*.lib)」指定後、プロジェクトファイルに追加します。

以上で、DLL の API 呼び出しが可能になります。

3.1.2 VB からの呼び出し方法

Visual BASIC のアプリケーションから ActiveX コンポーネントを利用するためには、以下の方法により ActiveX の登録が必要です。(32bit 版アプリケーションのみ対応)

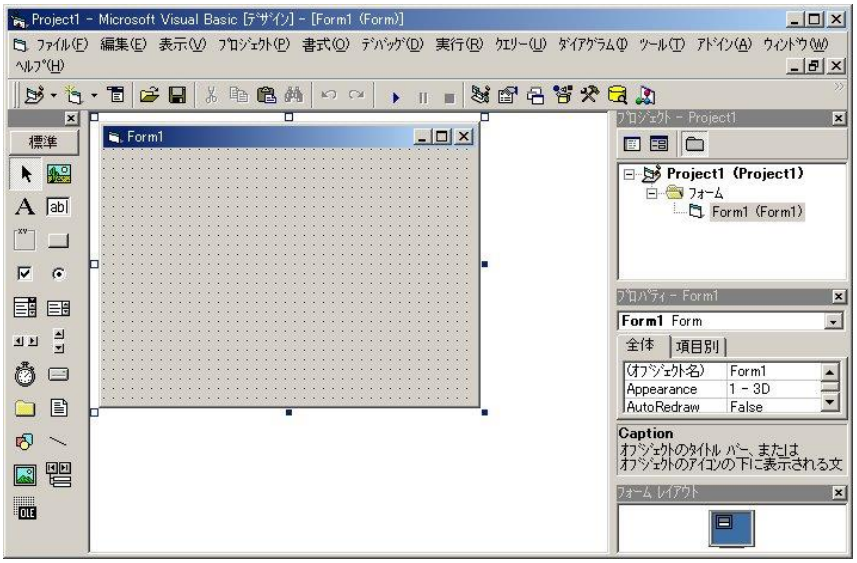
(1) DLL のコピー,ActiveX の登録

2.1.1 項 「ドライバーのインストール」を行ってください。自動的に DLL,ActiveX のコピー, ActiveX のレジストリ登録が行われます。すでに終了している場合は次に進んでください。

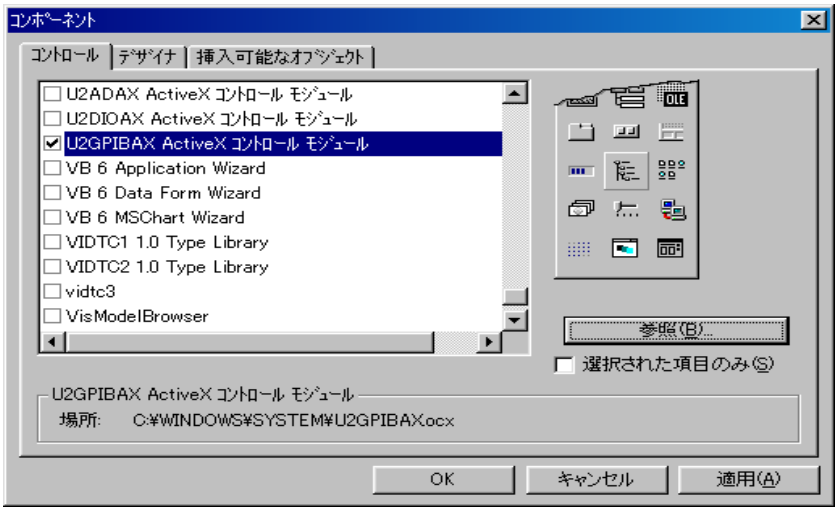
※Visual BASIC で 64bit 版のアプリケーションを作成される場合は、
ダウンロードした VB 用サンプル内の Declare.vb をご参照ください。
(ActiveX を利用せず DLL から直接 API を呼び出します)

(2) VB からの ActiveX 参照方法

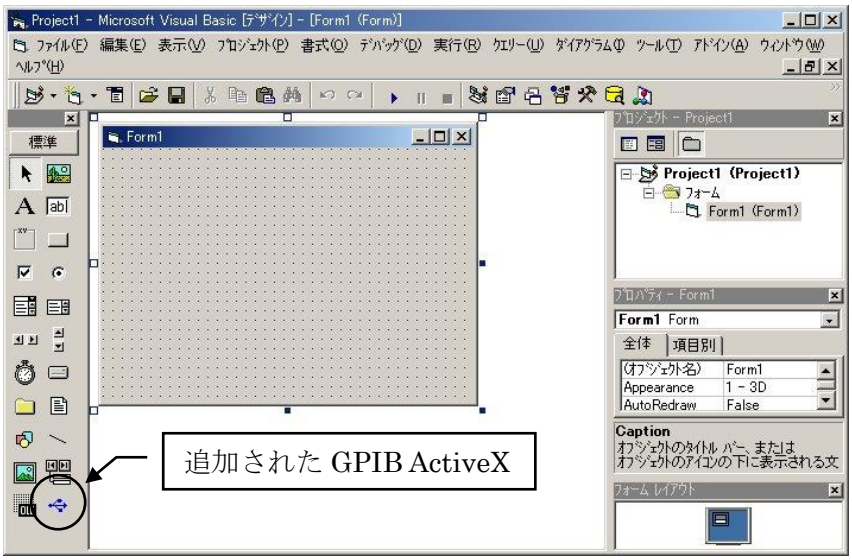
新しいプロジェクトを作成します。



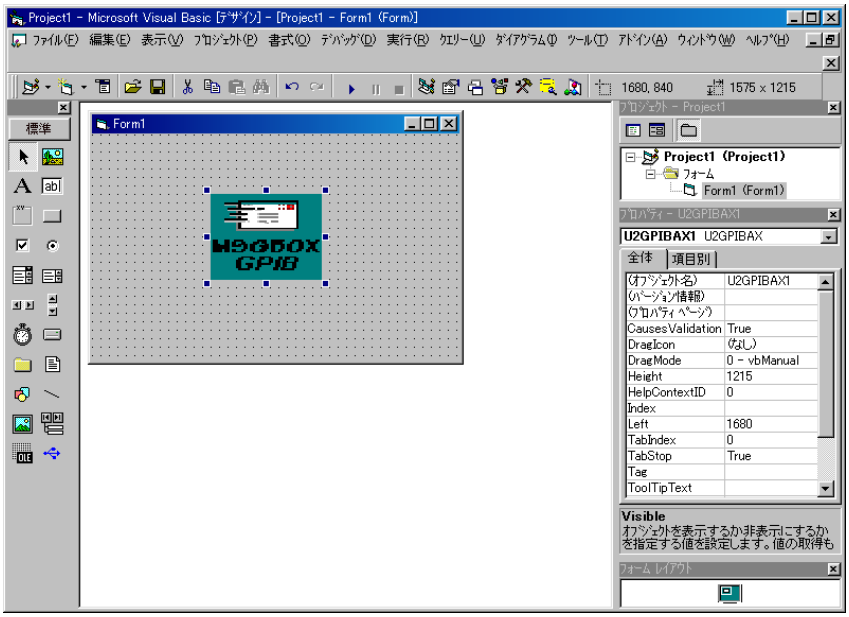
プロジェクトメニューのコンポーネントを選択します。コントロール一覧の、「U2GPIBAX ActiveX ｺﾝﾄﾛｰﾙﾓｼﾞｭｰﾙ」にチェックを入れて OK ボタンをクリックします。



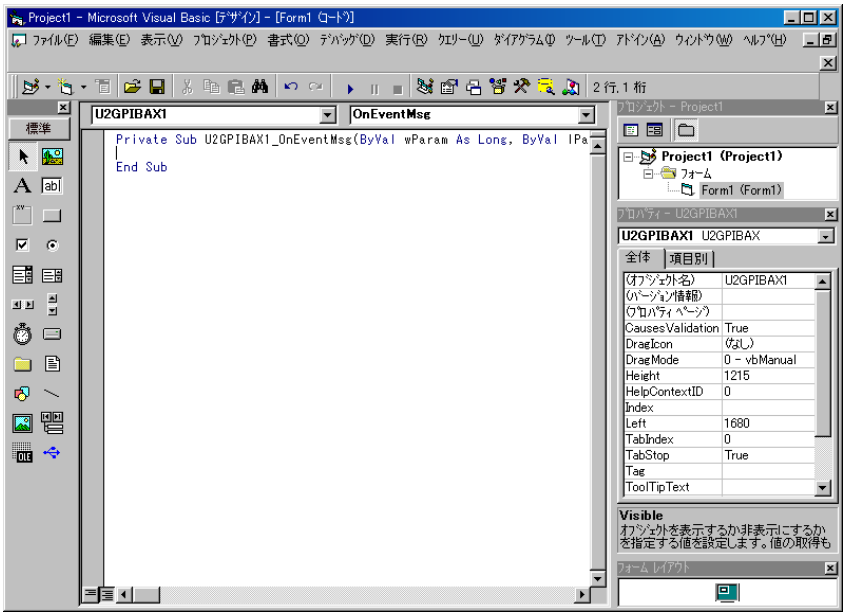
GPIB ActiveX コンポーネントが追加されます。



追加された GPIB ActiveX コンポーネントを選択し、フォームにオブジェクトを貼り付けます。
右例のように「MSGBOX GPIB」と表示されたオブジェクトが貼り付けられます。
オブジェクトのプロパティ内の「Visible」を False にして、実行時表示されないようにしておきます。



オブジェクトをダブルクリックすると、イベント発生時の呼び出されるサブルーチン
Sub U2GPIBAX1_OnEventMsg（…）
が表示されます。
SRQ 割り込みモードサンプルプログラムの説明を参照願います。



3-2. API 関数・ActiveX コントロール仕様

API 関数は、デバイスオープン・クローズに関する関数、9914 レジスタ制御関数、GPIB 機器制御関数に分類されます。以下に、関数の動作概要を示します。ActiveX コントロールについても同等の機能を持ったメソッドを用意しております。

■ デバイスオープン・クローズに関する関数として下記の関数を用意しています。
※USB220 を複数台接続する際に使用してください。

関数名	動作概要
U2GpOpenUnit	指定 ID のデバイスをオープン
U2GpCloseUnit	指定 ID のデバイスをクローズ
U2GpEnumUnit	接続されている全デバイスのコンバータ ID 情報を取得

■ USB220 のレジスタを直接制御する関数として下記の関数を用意しています。
※USB220 を複数台接続する際に頭に”U2”が付く関数を使用してください

関数名	動作概要
OutPort, U2OutPort	USB220 のレジスタに書き込み
InPort, U2InPort	USB220 のレジスタから読み込み

■ GPIB 機器の制御に関する関数として下記の関数を用意しています。
※USB220 を複数台接続する際に頭に”U2”が付く関数を使用してください

関数名	動作概要
gp_init, U2Gpinit	USB220 の初期化
gp_cli, U2Gpcli	IFC ラインを TRUE にします
gp_ren, U2Gpren	REN ラインを TRUE にします
gp_clr, U2Gpclr	CLR 又は SDC コマンド送信
gp_wrt, U2Gpwrt	GPIB 機器にデータ送信
gp_red, U2Gpred	GPIB 機器からデータ受信
gp_trg, U2Gptrg	GET コマンド送信
gp_wsrq, U2Gpwsrq	指定時間 SRQ を待つ(ステータスレジスタ 1)
gp_wsrqb, U2Gpwsrqb	指定時間 SRQ を待つ(バスステータスレジスタ)
gp_rds, U2Gprds	シリアルポールを実行
gp_rds1, U2Gprds1	シリアルポールを実行
gp_srq, U2Gpsrq	SRQ 割り込み
gp_srqCallBack, U2GpsrqCallBack	SRQ 割り込みの実行と解除を行います 割り込み発生時の処理は登録した CallBack 関数で行います
gp_lcl, U2Gplcl	GPIB 機器をローカル状態に設定

gp_llo, U2Gpllo	LLO コマンド送信
gp_tmout, U2Gptmout	バスタイムアウト時間設定
gp_setdelay, U2Gpsetdelay	外部変数のディレイ時間設定
gp_count, U2Gpcount	受信データ数の取得
gp_delm, U2Gpdelm	デリミタの設定
gp_tfrout, U2Gptfrout	GPIB 機器にバイナリデータ送信
gp_tfrin, U2Gptfrin	GPIB 機器からバイナリデータ受信
gp_tfrinit, U2Gptfrinit	GPIB 機器からバイナリデータ受信するためのトーカ指定
gp_tfrins, U2Gptfrins	GPIB 機器からバイナリデータ受信
gp_tfrend, U2Gptfrend	GPIB 機器からバイナリデータ受信するためのトーカ解除
gp_wtb, U2Gpwtb	コマンド文字列を送信
gp_myadr, U2Gpmyadr	USB220 の機器アドレスを取得

■ その他の関数として下記の関数を用意しています。

※USB220 を複数台接続する際もそのまま使用できます

関数名	動作概要
gp_wait	指定時間待つ
gp_strtoflt	4byte のデータを Single 型の実数に変換(VB 専用)
gp_strtodbl	8byte のデータを Double 型の実数に変換(VB 専用)

■ 補助関数として下記の関数を用意しています。

関数名	動作概要
gp_srqCheck, U2GpsrqCheck	SRQ ラインの現在の状態を取得
gp_wrtd, U2Gpwrtd	GPIB バス上にデータ送信
gp_tfroutd, U2Gptfroutd	GPIB バス上にバイナリデータ送信
gp_redd, U2Gpredd	GPIB 上からのデータ受信
gp_redah, U2Gpredah	GPIB 機器からデータ受信
gp_redrst, U2Gpredrst	リスナ解除、RFD ホールドオフの解除
gp_findlstn, U2Gpfindlstn	リスナ機器の検出

○REX-5052 からの移行について

弊社製品 GPIB PC カード REX-5052 用に作成したプログラムを REX-USB220 でご使用いただく場合、以下の作業を行って、アプリケーションを再構築ください。なお、以下 2 つの関数はサポートしておりません。

gp_cardinfo()

GetMyCardResource()

(VC) プロジェクトに組み込むヘッダファイル、ライブラリファイルを変更します。

(VB) Visual BASIC で DLL ライブラリ関数を使用するための ActiveX を提供していますので、Declare 宣言の必要はありません。使用している関数 gp_xxx()を U2GPIBAX オブジェクトの対応するメソッドに変更してください。但し、gp_rds(), gp_rds1(), gp_tfrout(), gp_tfrin(), gp_tfrins(), gp_strtoflt()の引数の型が一部異なっておりますので、ご注意ください。

○API 関数使用上の注意

(1) 1 台のパソコンで USB220 コンバータを複数台使用する場合には、先頭が U2xxxx となる関数(U2Gpinit など)を使用します。プログラムの始めに、指定コンバータ ID の USB220 のハンドルを関数 U2GpOpenUnit()で取得し、U2xxxx の第一引数に指定します。

(2) 1 台の USB220 コンバータで複数台の GPIB 機器(計測器)の制御を行うには、機器アドレス間にカンマ","を指定します。U2xxxx 関数を使用する必要はありません。

機器アドレス指定を行う関数 gp_clr(), gp_wrt(), gp_red(), gp_trg(), gp_rds(), gp_rds1(), gp_lcl(), gp_tfrout(), gp_tfrin(), gp_tfrinit()で使用します。

たとえば、以下のように使用してください。

```
gp_clr("3,5"); // リスナ 3 と 5 にコマンド SDC を送信します。
gp_wrt("6,20,30", "*CLS"); // リスナ 6 と 20 と 30 にデータ "*CLS"を送信します。
gp_red("3,20", buf, bufLen); /* アドレス 3 をトーカに、アドレス 20 をリスナに指
                                定してトーカ 3 からのデータを受信します。*/
gp_rds("3,20", status_byte); /* シリアルポールを実行し、アドレス 3 と 20 にステ
                                ータスバイトを問い合わせます。*/
```

(3) ニ次アドレスをもつ GPIB 機器の制御を行うには、一次アドレスに続いて、ニ次コマンド(96(0x60h)+ニ次アドレス) を指定します。たとえば、以下のように使用してください。

```
gp_clr("3,111"); /* 一次アドレス 3, 二次アドレス 15 のリスナにコマンド SDC を送信
                    します。*/
```

【API 関数・ActiveX コントロール仕様】

オープンクローズ関数

書 式

VC ➢ HANDLE **U2GpOpenUnit**(USHORT UnitId)

VB ➢ Function U2GPIBAX.**U2GpOpenUnit**(ByVal UnitId As Integer) As Long

機 能 指定の ID のコンバータをオープンします。正常にオープンされた時に返されるコンバータのハンドルは他の関数呼び出し時の第一引数として必要となります。
1 台のコンバータのみ接続の場合は、本関数を呼び出す必要はありません。

引 数 UnitId (IN) オープンするコンバータ ID（工場出荷時の設定値はゼロ）

戻 値 指定のコンバータを正常にオープンした場合はコンバータのハンドルを返します。
指定された ID のコンバータを見つけられなかった場合、またはオープン時エラーが発生した場合は NULL が返されます。

補 足 プログラム終了時、U2GpCloseUnit()によりコンバータをクローズするようにしてください。

書 式

VC ➢ VOID **U2GpCloseUnit**(HANDLE hUnit)

VB ➢ Sub U2GPIBAX.**U2GpCloseUnit**(ByVal hUnit As Long)

機 能 コンバータをクローズします。

引 数 hUnit (IN) クローズするコンバータのハンドル

戻 値 ありません。

書 式

VC ➢ INT **U2GpEnumUnit**(PUSHORT pUnitId, USHORT MaxUnit)

VB ➢ Function U2GPIBAX.**U2GpEnumUnit**(pUnitId As Integer, ByVal MaxUnit As Integer) As Long

機 能 接続されているコンバータを列挙し、コンバータ ID 情報をユーザに返します。

本関数は、複数台の USB2.0 GPIB コンバータを同一 USB バスに接続して利用する場合に必要になります。
複数台同時使用する場合は、各コンバータにユニークなコンバータ ID を設定します。1 台のパソコンに接続可能な最大コンバータ台数は 256 台で、コンバータ ID は 0-255 までの範囲で設定できます。
本関数により個々のコンバータに設定されたコンバータ ID 情報を得ることができます。その情報をもとに U2GpOpenUnit()を呼び出すことにより、コンバータ ID で指定されたコンバータのハンドルを取得することができます。
1 台のコンバータしか接続されない場合は、本関数を呼び出す必要はありません。

引 数

pUnitId	(OUT)	列挙したコンバータ ID の格納先アドレス。MaxUnit で指定した個数分の配列を確保してその先頭アドレスを指定します。
MaxUnit	(IN)	列挙する最大コンバータ台数。

戻 値

N	正常終了時、接続されているコンバータの台数が返されます。
-1	ライブラリ DeviceIoControl()リクエストエラー
-2	コンバータ ID 重複エラー この時呼び出し元の pUnitId[0]には重複コンバータ ID 番号がセットされます。

補 足 すべてのコンバータがクローズされた状態で呼び出してください。

9914 レジスタ制御関数

書 式　VC ➢ USHORT **OutPort**(USHORT Reg, USHORT OutVal)

VB ➢ Function U2GPIBAX.**OutPort**(ByVal Reg As Integer, ByVal OutVal As Integer) As Integer

VC ➢ USHORT **U2OutPort**(HANDLE hUnit, USHORT Reg, USHORT OutVal)

VB ➢ Function U2GPIBAX.**U2OutPort**(ByVal hUnit As Long, ByVal Reg As Integer, ByVal OutVal As Integer) As Integer

機 能　GPIB コントローラのレジスタに直接、値を書き込みます。(4-2 項 レジスタセット参照)

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
Reg	(IN)	レジスタオフセット値(0～7 を指定)
OutVal	(IN)	レジスタに書き込む値

戻 値　N　正常終了時、書き込んだ値が返されます。
エラー時には、0xFFFFh が返されます。

書 式　VC ➢ USHORT **InPort**(USHORT Reg)

VB ➢ Function U2GPIBAX.**InPort**(ByVal Reg As Integer) As Integer

VC ➢ USHORT **U2InPort**(HANDLE hUnit, USHORT Reg)

VB ➢ Function U2GPIBAX.**U2InPort**(ByVal hUnit As Long, ByVal Reg As Integer) As Integer

機 能　GPIB コントローラのレジスタから直接､値を読み込みます｡(4-2 項 レジスタセット参照)

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
Reg	(IN)	レジスタオフセット値(0～7 を指定)

戻 値　N　正常終了時、読み込んだ値が返されます。
エラー時には、0xFFFFh が返されます。

GPIB 機器制御関数

書 式

VC ➢ INT **gp_init**(USHORT GpAdrs, USHORT IOBase, USHORT IrqNo)

VB ➢ Function U2GPIBAX.**gpinit**(ByVal GpAdrs As Integer, ByVal IOBase As Integer, ByVal IrqNo As Integer) As Long

VC ➢ INT **U2Gpinit**(HANDLE hUnit, USHORT GpAdrs)

VB ➢ Function U2GPIBAX.**U2Gpinit**(ByVal hUnit As Long, ByVal GpAdrs As Integer) As Long

機 能 USB220 の GPIB 機器アドレスをセットし、GPIB コントローラの初期化を行います。また、各パラメータ(バスタイムアウト時間,ディレイ時間, デリミタ)の初期値を設定します。GPIB 制御を行う前に必ず呼び出してください。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
GpAdrs	(IN)	USB220 の GPIB 機器アドレス
IOBase	(IN)	REX-5052 互換のための引数です。0 を指定してください
IrqNo	(IN)	REX-5052 互換のための引数です。0 を指定してください

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
60	USB220 の GPIB 機器アドレス設定エラー

書 式　VC ➢　INT **gp_cli**(void)

VB ➢　Function U2GPIBAX.**gpcli**() As Long

VC ➢　INT **U2Gpcli**(HANDLE hUnit)

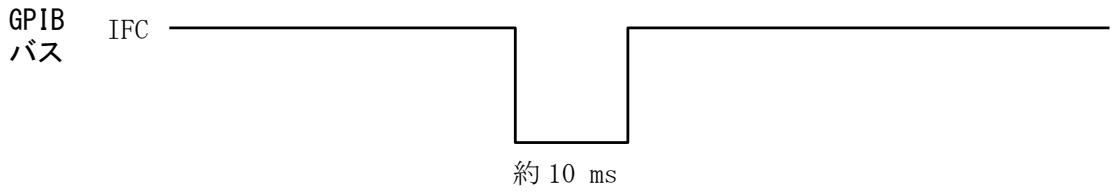
VB ➢　Function U2GPIBAX.**U2Gpcli**(ByVal hUnit As Long) As Long

機 能　IFC ラインを TRUE にします。

引 数　hUnit　　(IN)　　コンバータのハンドル(複数台接続時使用)

戻 値　0　　正常終了

-1　　ライブラリ DeviceIoControl()リクエストエラー



書 式　VC ➢　INT **gp_ren**(void)

VB ➢　Function U2GPIBAX.**gpren**() As Long

VC ➢　INT **U2Gpren**(HANDLE hUnit)

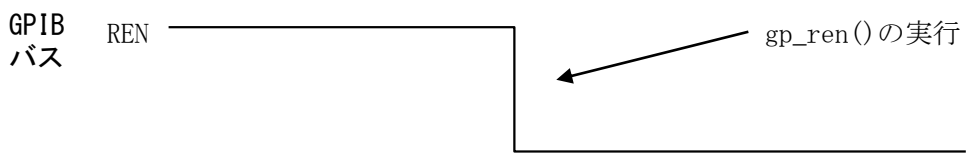
VB ➢　Function U2GPIBAX.**U2Gpren**(ByVal hUnit As Long) As Long

機 能　REN ラインを TRUE にします。

引 数　hUnit　　(IN)　　コンバータのハンドル(複数台接続時使用)

戻 値　0　　正常終了

-1　　ライブラリ DeviceIoControl()リクエストエラー



書 式　VC ➢　INT **gp_clr**(PCHAR adrs)

VB ➢　Function U2GPIBAX.**gpclr**(ByVal adrs As String) As Long

VC ➢　INT **U2Gpclr**(HANDLE hUnit, PCHAR adrs)

VB ➢　Function U2GPIBAX.**U2Gpclr**(ByVal hUnit As Long, ByVal adrs As String) As Long

機 能　クリアコマンド（DCL 又は SDC ）を送信します。引数 adrs に機器アドレスを指定しない場合は DCL コマンドを、指定する場合は SDC コマンドを送信します。

引 数

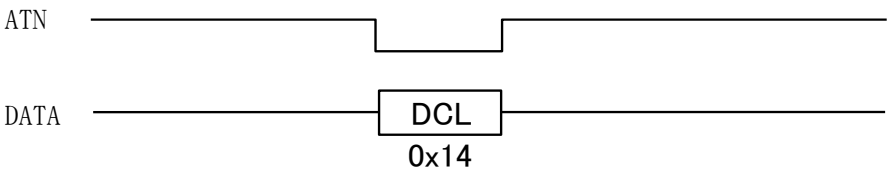
hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
-5,-6,-7	USB 転送時エラー

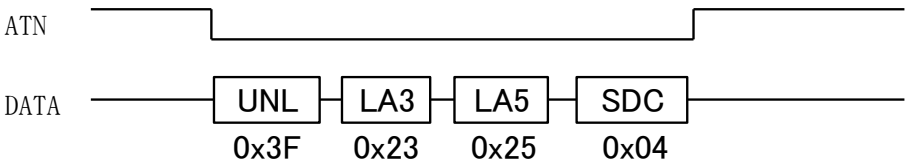
補 足　機器アドレスの指定が無い場合は、GPIB 上の全機器に対して DCL(Device Clear)コマンドを送信します。

(使用例)VC: gp_clr("");
VB: Call U2GPIBAX.gpclr("")



機器アドレスの指定がある場合は、指定の機器に対して SDC(Selected Device Clear)コマンドを送信します。

(使用例)VC: gp_clr("3,5");
VB: Call U2GPIBAX.gpclr("3,5")



書 式

VC ➢ INT **gp_wrt**(PCHAR adrs, PCHAR buf)

VB ➢ Function U2GPIBAX.**gpwrt**(ByVal adrs As String, ByVal buf As String) As Long

VC ➢ INT **U2Gpwrt**(HANDLE hUnit, PCHAR adrs, PCHAR buf)

VB ➢ Function U2GPIBAX.**U2Gpwrt**(ByVal hUnit As Long, ByVal adrs As String, ByVal buf As String) As Long

機 能 引数 adrs で指定した GPIB 機器に対してデータ送信します。デリミタ指定関数 gp_delm および gpdelm で指定されたデリミタを送信データに自動的に付加して送信を行います。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス
buf	(IN)	送信文字列を格納するバッファアドレス

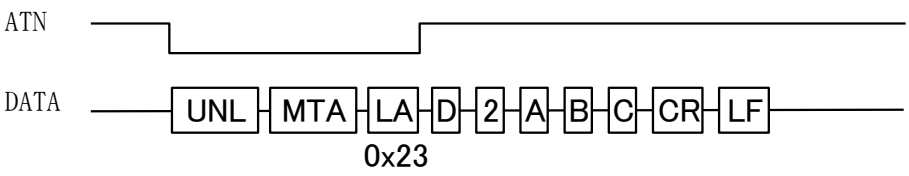
戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
64	送信データ設定エラー
-5,-6,-7	USB 転送時エラー

補 足 機器アドレス 3 にアスキーデータ"D2ABC"を送信する場合の例

(使用例)VC: gp_wrt("3","D2ABC");

VB: Call U2GPIBAX.gpwrt("3","D2ABC")



書 式

VC ➢ INT **gp_red**(PCHAR adrs, PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**gpred**(ByVal adrs As String, buf As String, ByVal bufLen As Long) As Long

VC ➢ INT **U2Gpred**(HANDLE hUnit, PCHAR adrs, PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**U2Gpred**(ByVal hUnit As Long, ByVal adrs As String, buf As String, ByVal bufLen As Long) As Long

機 能

引数 adrs で指定した GPIB 機器をトーカに指定し、データの受信を行います。デリミタ指定関数 gp_delm および gpdelm で指定されたデリミタ(もしくは EOI)を受信するかバスタイムアウトになるまで制御を返しません。

注)アプリケーションにはデリミタコードを返しません。

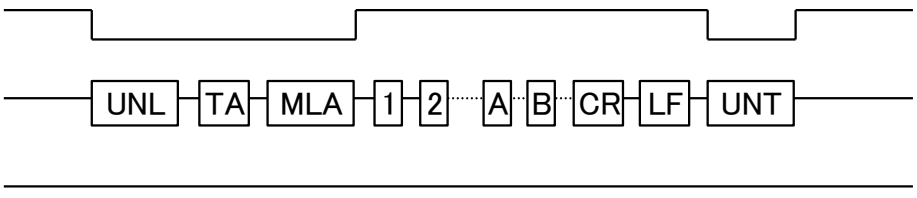
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス
buf	(OUT)	受信文字列を格納するバッファアドレス
bufLen	(IN)	受信バッファのサイズ

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
61	バッファオーバーフロー(デリミタ受信しないまま、サイズ分を受信)
63	GPIB 機器アドレス設定エラー
64	受信バッファサイズ設定エラー
-5,-6,-7	USB 転送時エラー

GPIB バス



書 式　VC ➢　INT **gp_trg**(PCHAR adrs)

VB ➢　Function U2GPIBAX.**gptrg**(ByVal adrs As String) As Long

VC ➢　INT **U2Gptrg**(HANDLE hUnit, PCHAR adrs)

VB ➢　Function U2GPIBAX.**U2Gptrg**(ByVal hUnit As Long, ByVal adrs As String) As Long

機 能　トリガコマンド（GET）を送信します。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス

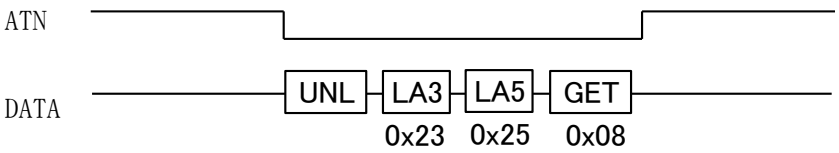
戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
-5,-6,-7	USB 転送時エラー

補 足　機器アドレス 3 と 5 に GET(Group Execute Trigger)コマンドを送信する場合の例

(使用例)VC: gp_trg("3,5");

VB: Call U2GPIBAX.gptrg("3,5")



書 式　VC ➢　INT **gp_wsrq**(INT WaitSecTime)

VB ➢　Function U2GPIBAX.**gpwsrq**(ByVal WaitSecTime As Long) As Long

VC ➢　INT **U2Gpwsrq**(HANDLE hUnit, INT WaitSecTime)

VB ➢　Function U2GPIBAX.**U2Gpwsrq**(ByVal hUnit As Long, ByVal WaitSecTime As Long) As Long

機 能　指定された時間、SRQ が発行されるのを待ちます。(インタラプトステータスレジスタ 1 をリード) SRQ を受信した場合、直ちに制御を返します。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
WaitSecTime	(IN)	SRQ を待つ時間(秒単位で指定)

戻 値

0	SRQ 受信
-1	タイムアウト(SRQ 未受信)

書 式　VC ➢　INT **gp_wsrqb**(INT WaitSecTime)

VB ➢　Function U2GPIBAX.**gpwsrqb**(ByVal WaitSecTime As Long) As Long

VC ➢　INT **U2Gpwsrqb**(HANDLE hUnit, INT WaitSecTime)

VB ➢　Function U2GPIBAX.**U2Gpwsrqb**(ByVal hUnit As Long, ByVal WaitSecTime As Long) As Long

機 能　指定された時間、SRQ が発行されるのを待ちます。(バスステータスレジスタをリード) SRQ を受信した場合、直ちに制御を返します。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
WaitSecTime	(IN)	SRQ を待つ時間(秒単位で指定)

戻 値

0	SRQ 受信
-1	タイムアウト(SRQ 未受信)

書 式

VC ➢ INT **gp_rds**(PCHAR adrs, PUCHAR status_byte)

VB ➢ Function U2GPIBAX.**gprds**(ByVal adrs As String, status_byte As Integer) As Long

VC ➢ INT **U2Gprds**(HANDLE hUnit, PCHAR adrs, PUCHAR status_byte)

VB ➢ Function U2GPIBAX.**U2Gprds**(ByVal hUnit As Long, ByVal adrs As String, status_byte As Integer) As Long

機 能 シリアルポールを実行し、ステータスバイトを取得します。

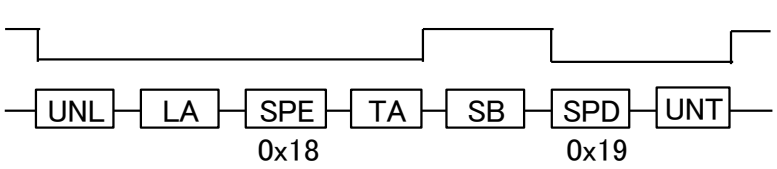
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス
status_byte	(OUT)	ステータスバイトを受け取るための配列。接続機器台数分以上の配列を確保してその先頭アドレスを指定します。

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
-5,-6,-7	USB 転送時エラー

GPIB バス



SB ： ステータスバイト
SPE ： シリアルポールイネーブル
SPD ： シリアルポールディセーブル

書 式

VC ➢ INT **gp_rds1**(PCHAR adrs, PUCHAR status_byte)

VB ➢ Function U2GPIBAX.**gprds1**(ByVal adrs As String, status_byte As Integer) As Long

VC ➢ INT **U2Gprds1**(HANDLE hUnit, PCHAR adrs, PUCHAR status_byte)

VB ➢ Function U2GPIBAX.**U2Gprds1**(ByVal hUnit As Long, ByVal adrs As String, status_byte As Integer) As Long

機 能

シリアルポールを実行し、ステータスバイトを取得します。

gp_rds(gprds)との違いは最後に UNT(Untalk)コマンドを送信しない点です。

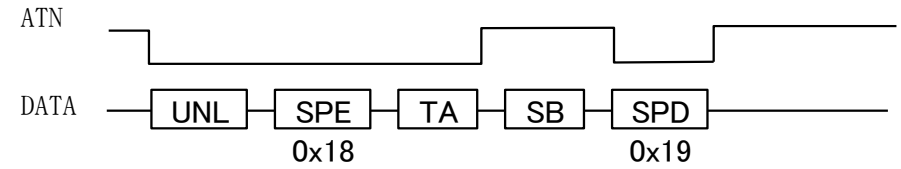
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス
status_byte	(OUT)	ステータスバイトを受け取るための配列。接続機器台数分以上の配列を確保してその先頭アドレスを指定します。

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
-5,-6,-7	USB 転送時エラー

GPIB バス



SB ： ステータスバイト
SPE ： シリアルポールイネーブル
SPD ： シリアルポールディセーブル

書 式

VC ➢ INT **gp_srq**(HWND hwnd, INT SrqMode)

VB ➢ Function U2GPIBAX.**gpsrq**(ByVal hwnd AsLong, ByVal SrqMode As Long) As Long

VC ➢ INT **U2Gpsrq**(HANDLE hUnit, HWMD hwnd, INT SrqMode)

VB ➢ Function U2GPIBAX.**U2Gpsrq**(ByVal hUnit As Long, ByVal hwnd AsLong, ByVal SrqMode As Long) As Long

機 能 SRQ 割り込みの実行および解除を行います。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
hwnd	(IN)	ウィンドウハンドル(VB の場合は 0 を指定してください)
SrqMode	(IN)	モードフラグ(0:解除フラグ, 1:実行フラグ)

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
-4	スレッドクリエートエラー
71	SRQ 割り込みは実行されている(実行時のエラー)

書 式　VC ➢　INT **gp_srqCallBack**(INT SrqMode, SRQCallback CallBackSrqFunc)

VB ➢　Function **gp_srqCallBack**(ByVal SrqMode As Integer,
ByVal CallBackSrqFunc As SRQCallback) As Integer

VC ➢　INT **U2GpsrqCallBack**(HANDLE hUnit, INT SrqMode,
SRQCallback CallBackSrqFunc)

VB ➢　Function **U2GpsrqCallBack**(ByVal hUnit As Integer, ByVal SrqMode As Integer,
ByVal CallBackSrqFunc As SRQCallback) As Integer

機 能　SRQ 割り込みの実行および解除を行います。
割り込みが発生すると、登録した CallBack 関数が処理されます。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
SrqMode	(IN)	モードフラグ(0:解除フラグ, 1:実行フラグ)
CallBackSrqFunc	(IN)	SRQ 割り込み発生時に呼び出される関数

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
-4	スレッドクリエートエラー
71	SRQ 割り込みは実行されている(実行時のエラー)

補 足　本関数に登録した CallBack 関数は、SRQ 割り込み時に呼び出されます。
CallBack 関数の引数で指定する SRQ_CALLBACK 構造体は、次ページを参照してください。

以下はサンプルプログラムでの定義例になります。

```
------- VC -------
typedef VOID (CALLBACK* SRQCallback)(PSRQ_CALLBACK);

------- VB -------
Delegate Sub SRQCallback(ByRef Srq_CallBack As SRQ_CALLBACK)

------- C# -------
unsafe public delegate void SRQCallback(SRQ_CALLBACK* pSrq_CallBack);
```

書 式　SRQ_CALLBACK（構造体）

機 能　gp_srqCallBack 関数で登録した CallBack 関数の構造体

引 数

wParam	(IN)	コンバータのハンドル(複数台接続時使用)
lParam	(IN)	(上位 2Byte) ： メッセージの種類
		0：SRQ 割り込み待ち状態開始
		1：SRQ 割り込み
		2：SRQ 割り込み解除
		3：エラー発生
		(下位 2Byte) ： コンバータ ID

戻 値　なし

補 足

------- VC -------

```
typedef struct _SRQ_CALLBACK
{
    UINT32      wParam;
    UINT32      lParam;
} SRQ_CALLBACK, *PSRQ_CALLBACK;
```

------- VB -------

```
<StructLayout(LayoutKind.Explicit, Size:=8, CharSet:=CharSet.Ansi)> _
Public Structure SRQ_CALLBACK
    <FieldOffset(0)> Dim wParam As UInt32
    <FieldOffset(4)> Dim lParam As UInt32
End Structure
```

------- C# -------

```
[StructLayout(LayoutKind.Explicit, Size = 8)]
unsafe public struct SRQ_CALLBACK
{
    [FieldOffset(0)] public UInt32 wParam;
    [FieldOffset(4)] public UInt32 lParam;
}
```

書 式　VC ➢　INT **gp_lcl**(PCHAR adrs)

VB ➢　Function U2GPIBAX.**gplcl**(ByVal adrs As String) As Long

VC ➢　INT **U2Gplcl**(HANDLE hUnit, PCHAR adrs)

VB ➢　Function U2GPIBAX.**U2Gplcl**(ByVal hUnit As Long, ByVal adrs As String) As Long

機 能　GPIB 機器をローカル状態に設定します。

引 数

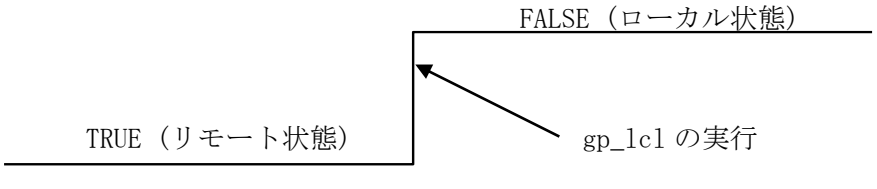
hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
-5,-6,-7	USB 転送時エラー

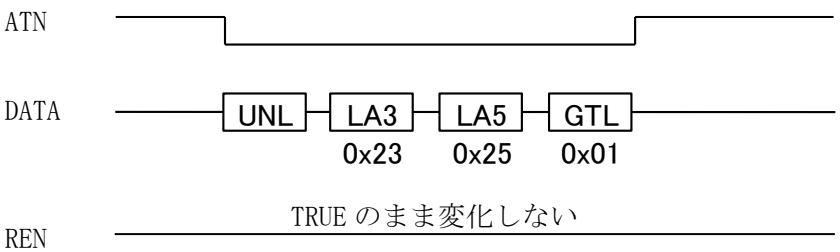
補 足　機器アドレスの指定が無い場合は、REN ラインを High(FALSE)にします。

(使用例)VC: gp_lcl("");
VB: Call U2GPIBAX.gplcl("")



機器アドレスの指定がある場合は、指定の機器に対して GTL(Go To Local)コマンドを送信します。

(使用例)VC: gp_lcl("3,5");
VB: Call U2GPIBAX.gplcl("3,5")



書 式　VC ➢　INT **gp_llo**(void)

VB ➢　Function U2GPIBAX.**gpllo**() As Long

VC ➢　INT **U2Gpllo**(HANDLE hUnit)

VB ➢　Function U2GPIBAX.**U2Gpllo**(ByVal hUnit As Long) As Long

機 能　GPIB 上の全機器に対して LLO(Local Lock Out)コマンド送信します。

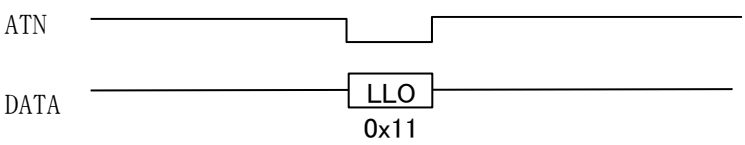
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
-5,-6,-7	USB 転送時エラー

GPIB バス



書 式　VC ➢　INT **gp_tmout**(INT SecTime)

VB ➢　Function U2GPIBAX.**gptmout**(ByVal SecTime As Long) As Long

VC ➢　INT **U2Gptmout**(HANDLE hUnit, INT SecTime)

VB ➢　Function U2GPIBAX.**U2Gptmout**(ByVal hUnit As Long, ByVal SecTime As Long) As Long

機 能　バスタイムアウト時間の設定を変更します。初期値は 10 秒です。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
SecTime	(IN)	タイムアウト時間(秒単位で指定)

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
72	パラメータ設定エラー

補 足　初期設定(10 秒)は gp_init()で行いますので、本関数呼び出しは、gp_init()の後に行ってください。設定可能な最長タイムアウト時間は 655 秒です。

書 式　VC ➢　INT **gp_setdelay**(INT DelayTime)

VB ➢　Function U2GPIBAX.**gpsetdelay**(ByVal DelayTime As Long) As Long

VC ➢　INT **U2Gpsetdelay**(HANDLE hUnit, INT DelayTime)

VB ➢　Function U2GPIBAX.**U2Gpsetdelay**(ByVal hUnit As Long, ByVal DelayTime As Long) As Long

機 能　ATN ラインを TRUE 又は FALSE にする際のディレイ時間を設定します。コマンド送信時に GPIB タイムアウトとなる場合に調整します。初期値は 0usec です。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
DelayTime	(IN)	ディレイ時間(マイクロ秒単位で指定)

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー

補 足　初期設定(0 マイクロ秒)は gp_init()で行いますので、本関数呼び出しは、gp_init()の後に行ってください。設定可能な最長ディレイ時間は 65500 マイクロ秒です。

書 式　VC ➢　INT **gp_count**(void)

VB ➢　Function U2GPIBAX.**gpcount**() As Long

VC ➢　INT **U2Gpcount**(HANDLE hUnit)

VB ➢　Function U2GPIBAX.**U2Gpcount**(ByVal hUnit As Long) As Long

機 能　GPIB 機器からの受信データ数または GPIB 機器へ送信完了したデータ数を取得します。関数 gp_red(gpred), gp_tfrin(gptfrin), gp_tfrins(gptfrins) gp_wrt(gpwrt), gp_tfrout(gptfrout)の後に呼び出すことで、実際にハンドシェイクが完了したデータ数を知ることができます。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)

戻 値

N	受信データ数または送信データ数が返されます。

補 足　デリミタコードのカウントは行いません。

書 式

VC ➢ INT **gp_delm**(PCHAR mode, UINT dlm)

VB ➢ Function U2GPIBAX.**gpdelm**(ByVal mode As String, ByVal dlm As Long) As Long

VC ➢ INT **U2Gpdelm**(HANDLE hUnit, PCHAR mode, UINT dlm)

VB ➢ Function U2GPIBAX.**U2Gpdelm**(ByVal hUnit As Long, ByVal mode As String, ByVal dlm As Long) As Long

機 能

送信時(gp_wrt および gpwrt)、受信時(gp_red および gpred)のデリミタの設定を行ないます。初期設定では送信時デリミタは CR+LF、受信時デリミタは LF(0x0A)となっています。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
mode	(IN)	“l”で受信時、“t”で送信時の設定を行います。 “b”で受信時・送信時の設定を行います。
dlm	(IN)	デリミタコードを指定します。

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー

補 足

・初期設定は gp_init()で行いますので、本関数呼び出しは、gp_init()の後に行ってください。

・デリミタコード dlm については以下のような設定を行います。

(送信時) ：mode = “t”での設定

Bit6～Bit0 の 7bit でデリミタコードを設定します。Bit7 を 1 に設定すると EOI を出力し、全ての bit を 0(dlm=0)にすると、CR+LF(0x0D＋0x0A)が設定されます。

(受信時) ：mode = “l”での設定

Bit7～Bit0 の 8bit でデリミタコードを設定します。EOI 検出時は常にデリミタとして扱い、データ受信を終了します。

(送信・受信時) ：mode = “b”での設定

dlm の値を以下のように設定することで、送信・受信時の設定を同時に行います。“t”“l”で設定されたデリミタは無効となります。

dlm = 0x0400	デリミタなし
dlm = 0x000D	CR
dlm = 0x000A	LF
dlm = 0x0200	CR+LF
dlm = 0x0C00	EOI のみ
dlm = 0x080D	CR+EOI(受信時は CR もしくは EOI で受信終了)
dlm = 0x080A	LF+EOI(受信時は CR もしくは EOI で受信終了)
dlm = 0x0A00	CR+LF+EOI(受信時は CR+LF もしくは EOI で受信終了)

※mode=“b”で送信時・受信時に異なる設定を行いたい場合は gp_wrt(),gp_red()関数呼び出し直前に、本関数で再設定を行ってください。

書 式

VC ➢ INT **gp_tfrout**(PCHAR adrs, INT bufLen, PCHAR buf)

VB ➢ Function U2GPIBAX.**gptfrout**(ByVal adrs As String, ByVal bufLen As Long, ByVal buf As Integer) As Long

VC ➢ INT **U2Gptfrout**(HANDLE hUnit, PCHAR adrs, INT bufLen, PCHAR buf)

VB ➢ Function U2GPIBAX.**U2Gptfrout**(ByVal hUnit As Long, ByVal adrs As String, ByVal bufLen As Long, ByVal buf As Integer) As Long

機 能 引数 adrs で指定した GPIB 機器に対してバイナリデータを送信します。デリミタは EOI のみです。

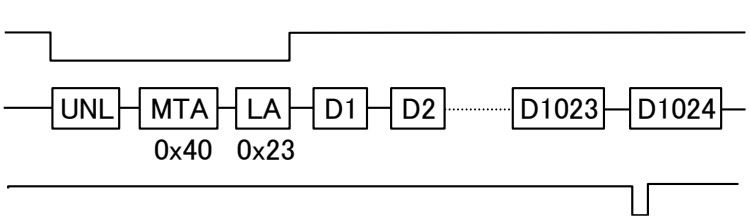
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス
bufLen	(IN)	送信するデータの長さ
buf	(IN)	送信データを格納する配列の先頭アドレス。

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
64	送信データ長設定エラー
-5,-6,-7	USB 転送時エラー

GPIB バス



書 式

VC ➢ INT **gp_tfrin**(PCHAR adrs, INT bufLen, PCHAR buf)

VB ➢ Function U2GPIBAX.**gptfrin**(ByVal adrs As String, ByVal bufLen As Long, buf As Integer) As Long

VC ➢ INT **U2Gptfrin**(HANDLE hUnit, PCHAR adrs, INT bufLen, PCHAR buf)

VB ➢ Function U2GPIBAX.**U2Gptfrin**(ByVal hUnit As Long, ByVal adrs As String, ByVal bufLen As Long, buf As Integer) As Long

機 能

引数 adrs で指定した GPIB 機器をトーカに指定し、バイナリデータを受信します。デリミタは EOI のみです。EOI を受信するかバスタイムアウトになるまで制御を返しません。

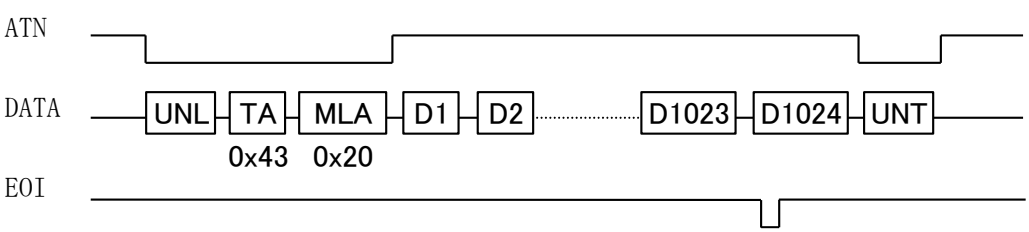
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス
bufLen	(IN)	用意する配列数
buf	(OUT)	受信データを格納する配列の先頭アドレス

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
61	受信バッファオーバーフロー(EOI 受信しないまま、サイズ分を受信)
63	GPIB 機器アドレス設定エラー
64	受信用配列設定エラー
-5,-6,-7	USB 転送時エラー

GPIB バス



書 式　VC ➢　INT **gp_tfrinit**(PCHAR adrs)

VB ➢　Function U2GPIBAX.**gptfrinit**(ByVal adrs As String) As Long

VC ➢　INT **U2Gptfrinit**(HANDLE hUnit, PCHAR adrs)

VB ➢　Function U2GPIBAX.**U2Gptfrinit**(ByVal hUnit As Long, ByVal adrs As String) As Long

機 能　GPIB 機器からバイナリデータを受信するためにトーカアドレスを指定します。
(gp_tfrins または gptfrins, gp_tfrend または gptfrend と共に使用します)

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
63	GPIB 機器アドレス設定エラー
-5,-6,-7	USB 転送時エラー

補 足　受信すべきデータ数が不明な場合、関数 gp_tfrin(gptfrin)の代わりに 3 つの関数 gp_tfrinit(gptfrinit), gp_tfrins(gptfrins), gp_tfrend(gptfrend)を組み合わせて使用し、データを受信することが可能です。gp_tfrins(gptfrins)を繰りかえし呼び出すことで、連続してデータを受信することができます。

(使用例) 機器アドレス 3 からデータを受信する場合。通常は gp_tfrins(gptfrins)の戻り値より EOI 受信の有無を調べ、EOI 未受信であれば、再度を呼び出します。

```
VC: BYTE RxBuf[256];
    gp_tfrinit("3"); // トーカ指定
    gp_tfrins(256, RxBuf); // 256Byte データ受信
    gp_tfrins(256, RxBuf);
    gp_tfrins(256, RxBuf);
    ・・・(EOI 受信するまで繰返し呼び出す)
    gp_tfrend(); // トーカ指定解除
VB: Dim RxBuf(256) As Integer
    Call U2GPIBAX.gptfrinit("3") ' トーカ指定
    Call U2GPIBAX.gptfrins(256, RxBuf(0)) '256Byte データ受信
    Call U2GPIBAX.gptfrins(256, RxBuf(0))
    Call U2GPIBAX.gptfrins(256, RxBuf(0))
    ・・・(EOI 受信するまで繰返し呼び出す)
    Call U2GPIBAX.gptfrend() ' トーカ指定解除
```

書 式　VC ➢　INT **gp_tfrins**(INT bufLen, PCHAR buf)

VB ➢　Function U2GPIBAX.**gptfrins**(ByVal bufLen As Long, buf As Integer) As Long

VC ➢　INT **U2Gptfrins**(HANDLE hUnit, INT bufLen, PCHAR buf)

VB ➢　Function U2GPIBAX.**U2Gptfrins**(ByVal hUnit As Long, ByVal bufLen As Long, buf As Integer) As Long

機 能　GPIB 機器からバイナリデータを受信します。デリミタは EOI のみです。
(gp_tfrinit または gptfrinit, gp_tfrend または gptfrend と共に使用します)

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
bufLen	(IN)	受信するデータの長さ
buf	(OUT)	受信データを格納する配列の先頭アドレス

戻 値

0	指定サイズ分のデータを受信して正常終了
24	EOI を受信して正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
64	受信用配列設定エラー
-5,-6,-7	USB 転送時エラー

書 式　VC ➢　VOID **gp_tfrend**(void)

VB ➢　Sub U2GPIBAX.**gptfrend**()

VC ➢　VOID **U2Gptfrend**(HANDLE hUnit)

VB ➢　Sub U2GPIBAX.**U2Gptfrend**(ByVal hUnit As Long)

機 能　GPIB 機器からバイナリデータを受信するために指定したトーカアドレスを解除します。
(gp_tfrinit または gptfrinit, gp_tfrins または gptfrins と共に使用します)

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)

戻 値　ありません。

書 式　VC ➢　INT **gp_wtb**(PCHAR buf)

VB ➢　Function U2GPIBAX.**gpwtb**(ByVal buf As String) As Long

VC ➢　INT **U2Gpwtb**(HANDLE hUnit, PCHAR buf)

VB ➢　Function U2GPIBAX.**U2Gpwtb**(ByVal hUnit As Long, ByVal buf As String) As Long

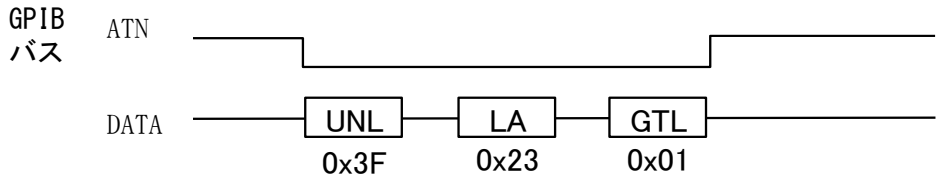
機 能　ATN ラインを TRUE にしてコマンド文字列を送信します。コマンド文字列の最後に、データ終了を示す NULL コード(0x00)を指定してください。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
buf	(IN)	送信文字列を格納するバッファアドレス

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
64	送信データ設定エラー
-5,-6,-7	USB 転送時エラー



書 式　VC ➢　INT **gp_myadr**(void)

VB ➢　Function U2GPIBAX.**gpmyadr**() As Long

VC ➢　INT **U2Gpmyadr**(HANDLE hUnit)

VB ➢　Function U2GPIBAX.**U2Gpmyadr**(ByVal hUnit As Long) As Long

機 能　関数 gp_init(gpinit)で設定された USB220 の GPIB 機器アドレスを取得します。
プログラムで新たに USB220 の GPIB アドレスを知る必要が無い場合は、本関数を呼び出す必要はありません。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)

戻 値

N	正常終了時、USB220 の GPIB 機器アドレスが返されます。
-1	ライブラリ DeviceIoControl()リクエストエラー

その他の関数

書 式　VC ➢　VOID **gp_wait**(int WaitSecTime)

VB ➢　Sub U2GPIBAX.**gpwait**(ByVal WaitSecTime As Long)

機 能　指定時間プログラムを停止させます。

引 数　WaitSecTime　(IN)　プログラムを停止する時間(秒単位で指定)

戻 値　ありません。

書 式　VC ➢　VOID **gp_strtoflt**(BYTE *bPoint, float *data)

VB ➢　Sub U2GPIBAX.**gpstrtoflt**(ByVal bPoint As Integer, data As Single)

機 能　4 バイトのデータの格納するメモリへの BYTE 型ポインタを Single 型ポインタにキャストします。(VC では、直接キャスト可能であるため、使用する必要はありません。)

引 数　bPoint　(IN)　4 バイトデータを格納するポインタ

data　(OUT)　キャストした Single 型アドレス

戻 値　ありません。

補 足　(使用例)

```
VB: Dim Buf(4) As Integer
    Dim Data As Single

    Buf(0)=&H52
    Buf(1)=&H6
    Buf(2)=&H9E
    Buf(3)=&H3F

    Call U2GPIBAX.gpstrtoflt(Buf(0), Data)
    ‘ 結果は Data =1.234568 となります。
```

書 式　VC ➢　VOID **gp_strtodbl**(BYTE *bPoint, double *data)

VB ➢　Sub U2GPIBAX.**gpstrtodbl**(ByVal bPoint As Integer, data As Double)

機 能　8バイトのデータの格納するメモリへのBYTE型ポインタをDouble型ポインタにキャストします。(VCでは、直接キャスト可能であるため、使用する必要はありません。)

引 数

bPoint	(IN)	8バイトデータを格納するポインタ
data	(OUT)	キャストしたDouble型アドレス

戻 値　ありません。

補 足　(使用例)

```
VB: Dim Buf(8) As Integer
    Dim Data As Double

    Buf(0)=&H1B
    Buf(1)=&HDE
    Buf(2)=&H83
    Buf(3)=&H42
    Buf(4)=&HCA
    Buf(5)=&HC0
    Buf(6)=&HF3
    Buf(7)=&H3F

    Call U2GPIBAX.gpstrtodbl(Buf(0), Data)
    ‘ 結果はData =1.23456789となります。
```

補助関数

書 式　VC ➢　INT **gp_srqCheck**(void)

VB ➢　Function U2GPIBAX.**gpsrqCheck**() As Long

VC ➢　INT **U2GpsrqCheck**(HANDLE hUnit)

VB ➢　Function U2GPIBAX.**U2GpsrqCheck**(ByVal hUnit As Long) As Long

機 能　SRQ ラインの現在の状態を返します。

引 数　hUnit　　(IN)　　コンバータのハンドル(複数台接続時使用)

戻 値　1　　SRQ ラインが TRUE

0　　SRQ ラインが FALSE

-1　　ライブラリ DeviceIoControl()リクエストエラー

書 式

VC ➢ INT **gp_wrtd**(PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**gpwrtd**(ByVal buf As String, ByVal bufLen As Long) As Long

VC ➢ INT **U2Gpwrtd**(HANDLE hUnit, PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**U2Gpwrtd**(ByVal hUnit As Long, ByVal buf As String, ByVal bufLen As Long) As Long

機 能 GPIB バス上にデータ送信します。デリミタ指定関数 gp_delm および gpdelm で指定されたデリミタを送信データに自動的に付加して送信を行います。

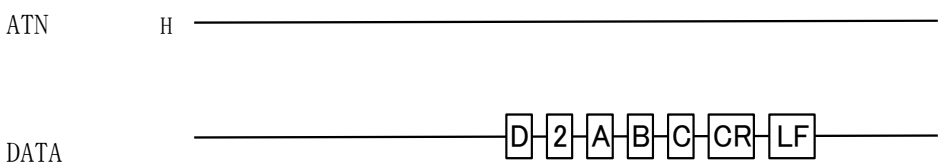
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
buf	(IN)	送信文字列を格納するバッファアドレス
bufLen	(IN)	送信するデータサイズ

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
64	送信データ設定エラー
-5,-6,-7	USB 転送時エラー

補 足 gp_wrt(gpwrt)と異なる点は、データ送信前にコマンド送信をしない点です。
通常、gp_wtb(gpwtb)と組み合わせて使用します。



書 式

VC ➢ INT **gp_tfroutd**(INT bufLen, PCHAR buf)

VB ➢ Function U2GPIBAX.**gptfroutd**(ByVal bufLen As Long, ByVal buf As Integer) As Long

VC ➢ INT **U2Gptfroutd**(HANDLE hUnit, INT bufLen, PCHAR buf)

VB ➢ Function U2GPIBAX.**U2Gptfroutd**(ByVal hUnit As Long, ByVal bufLen As Long, ByVal buf As Integer) As Long

機 能 GPIB バス上にバイナリデータ送信します。

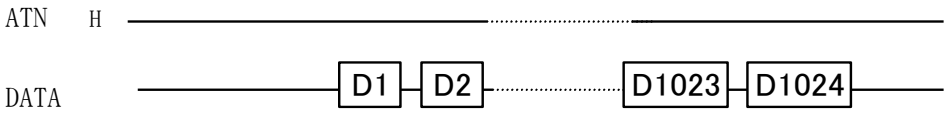
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
bufLen	(IN)	送信するデータサイズ
buf	(IN)	送信データを格納する配列の先頭アドレス。

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
64	送信データ長設定エラー
-5,-6,-7	USB 転送時エラー

補 足 gp_tfrout(gptfrout)と異なる点は、データ送信前にコマンド送信をしない点です。
また、送信デリミタ EOI の有無は gp_delm(gpdelm)の設定に従います。デリミタコードは付加しません。通常、gp_wtb(gpwtb)と組み合わせて使用します。



書 式

VC ➢ INT **gp_redd**(PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**gpredd**(buf As String, ByVal bufLen As Long) As Long

VC ➢ INT **U2Gpredd**(HANDLE hUnit, PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**U2Gpredd**(ByVal hUnit As Long, buf As String, ByVal bufLen As Long) As Long

機 能 GPIB 上からのデータ受信を行います。デリミタ指定関数 gp_delm(gpdelm)で指定されたデリミタを受信するかバスタイムアウトになるまで制御を返しません。
注)アプリケーションにはデリミタコードを返しません。

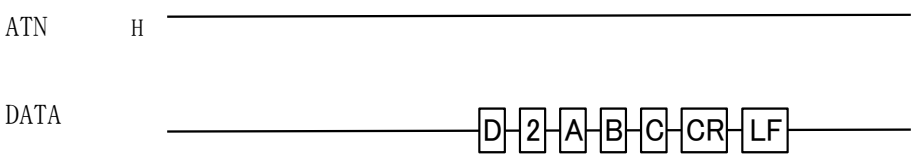
引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
buf	(OUT)	受信文字列を格納するバッファアドレス
bufLen	(IN)	受信バッファのサイズ

戻 値

0	指定サイズ分データ受信して正常終了
24	指定のデリミタを受信して正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
64	受信バッファサイズ設定エラー
-5,-6,-7	USB 転送時エラー

補 足 gp_red(gpred)と異なる点は、データ受信前にコマンド送信をしない点と最後に UNT コマンドを送信しない点です。また、本関数から制御を戻したとき、RFD ホールドオフとなります。通常、gp_wtb(gpwtb)と組み合わせて使用します。



書 式

VC ➢ INT **gp_redah**(PCHAR adrs, PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**gpredah**(ByVal adrs As String, buf As String, ByVal bufLen As Long) As Long

VC ➢ INT **U2Gpredah**(HANDLE hUnit, PCHAR adrs, PCHAR buf, INT bufLen)

VB ➢ Function U2GPIBAX.**U2Gpredah**(ByVal hUnit As Long, ByVal adrs As String, buf As String, ByVal bufLen As Long) As Long

機 能

引数 adrs で指定した GPIB 機器をトーカに指定し、データの受信を行います。デリミタ指定関数 gp_delm(gpdelm)で指定されたデリミタを受信するかバスタイムアウトになるまで制御を返しません。

注)アプリケーションにはデリミタコードを返しません。

引 数

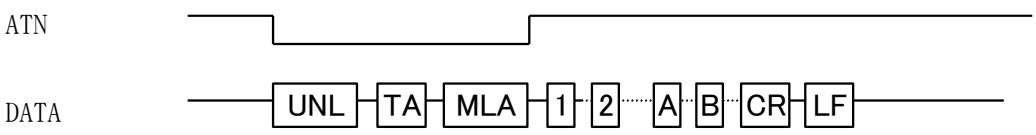
hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(IN)	GPIB 機器アドレス
buf	(OUT)	受信文字列を格納するバッファアドレス
bufLen	(IN)	受信バッファのサイズ

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
61	バッファオーバーフロー(デリミタ受信しないまま、サイズ分を受信)
63	GPIB 機器アドレス設定エラー
64	受信用配列数設定エラー
-5,-6,-7	USB 転送時エラー

補 足

gp_red(gpred)と異なる点は、最後に UNT コマンドを送信しない点です。また、本関数から制御を戻したとき、RFD ホールドオフとなります。



書 式　VC ➢　INT **gp_redrst**(void)

VB ➢　Function U2GPIBAX.**gpredrst**() As Long

VC ➢　INT **U2Gpredrst**(HANDLE hUnit)

VB ➢　Function U2GPIBAX.**U2Gpredrst**(ByVal hUnit As Long) As Long

機 能　本関数は、リスナ解除、RFD ホールドオフの解除を行います。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー

書 式　VC ➢　INT **gp_findlstn**(PCHAR adrs, INT adrsLen)

VB ➢　Function U2GPIBAX.**gpfindlstn**(adrs As String, ByVal adrsLen As Long) As Long

VC ➢　INT **U2Gpfindlstn**(HANDLE hUnit, PCHAR adrs, INT adrsLen)

VB ➢　Function U2GPIBAX.**U2Gpfindlstn**(ByVal hUnit As Long, adrs As String, ByVal adrsLen As Long) As Long

機 能　GPIB バスに接続されているリスナ機器を検出し、GPIB アドレスを取得します。

引 数

hUnit	(IN)	コンバータのハンドル(複数台接続時使用)
adrs	(OUT)	GPIB アドレスを格納するバッファアドレス
adrsLen	(IN)	バッファのサイズ

戻 値

0	リスナ未検出
-1	ライブラリ DeviceIoControl()リクエストエラー
53	GPIB バスタイムアウト
64	アドレス格納用バッファサイズが正しくない
N	リスナ検出台数

補 足　本関数では、取得した GPIB アドレスを ASCII データの形で adrs に格納します。取得した adrs を gp_wrt(), gp_red()等の第一引数でそのまま使用できます。
戻り値に 64 が返る場合は、確保するバッファ adrs を大きめに確保してください。

3-3. サンプルプログラム解説

本製品には、GPIB インターフェースを持つ各種測定器の制御を行うアプリケーション作成のためのサンプルプログラム(VisualC/C++, VisualBASIC)として、下記に掲げる測定器用のサンプルプログラムを提供しています。
※サンプルプログラムについてのご質問につきましては、弊社サポートセンターまでお問い合わせください。
なお、各機器の操作に関してのサポートは行うことはできませんので予めご了承ください。
・デジタルオシロスコープ DS-8812(岩通計測株式会社製)
・デジタルオシロスコープ TDS3054B(日本テクトロニクス株式会社製)
・デジタルマルチメータ HP3478A(Hewlett-Packard 製)
・デジタルマルチメータ R6552(ADVANTEST 製)
・ファンクションジェネレータ 33120A(Hewlett-Packard 製)
・デジタルオシロスコープ Infiniium 54846A(Agilent Technologies 製)
・デジタルパワーメータ WT110E(横河電機製)
・プログラマブル直流電圧・電流源 7651(横河電機製)
・直流標準電圧・電流発生器 YEW2553(横河電機製)
・ユニバーサルカウンタ SC-7201(岩通計測株式会社製)
・直流電圧源 E3631A(Hewlett-Packard 製)

ここでは、代表例として HP3478A のサンプルプログラムについて説明します。その他のサンプルプログラムの詳細については、Readme もしくはプログラムソースコードを参照ください。

HP3478A 制御プログラム

HP3478A 制御プログラムには、1) ポーリングモード（割り込みを使用せず、SRQ が来るのをポーリングしデータを取得するプログラム）と 2) 割り込みモード（SRQ の検知に割り込みを使用し、データを取得するプログラム）の 2 つを用意しています。

1) ポーリングモード
① 最初に機器側で設定されている GPIB 機器アドレスをエディットボックスに入力します。(初期値は 3)
② 初期化ボタンを押して DLL ライブラリの初期化を行います。
③ 計測開始ボタンを押して 10 秒間 SRQ を監視し、SRQ 信号がきたときの計測データを表示します。

2) 割り込みモード
① 最初に機器側で設定されている GPIB 機器アドレスをエディットボックスに入力します。(初期値は 3)
② 初期化ボタンを押して DLL ライブラリの初期化を行います。
③ 計測開始ボタンを押して SRQ 待ちになり、SRQ 信号がきたときに計測データを表示します。計測停止ボタンを押すと、SRQ 待ちを止めます。

VisualC サンプルプログラム抜粋(ポーリング/割り込みモード共通)

初期化ボタンを押したときの処理・・・GPIB コントローラの初期化を行った後、続けて HP3478A に対しクリアコマンド、測定用コマンドを送信します。

```
void Cmd_OnCmdGpInit ( HWND hwnd )
{
    INT   GpStatus;
    CHAR   szCommand[] = "H0KM01";

    // GPIB コントローラ初期化
    GpStatus = gp_init( MyGPIBAdrs, 0, 0 );
    if( GpStatus != 0 )
    {
        sprintf( szBuf, "gp_init()初期化ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    // IFC ﾗｲﾝを TRUE にする
    gp_cli();

    // REN ﾗｲﾝを TRUE にする
    gp_ren();

    // HP3478A で設定されている GPIB 機器ｱﾄﾞﾚｽ取得
    GetDlgItemText( hwnd, IDE_3478GPIBADRS, szHP3478A, sizeof(szHP3478A) );

    // GPIB バスタイムアウト時間を 3 秒に設定
    gp_tmout(3);

    // SDC コマンド送出
    GpStatus = gp_clr( szHP3478A );
    if ( GpStatus != 0 )
    {
        sprintf( szBuf, "gp_clr()ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    // LLO コマンド送出
    gp_llo();

    // HP3478A GPIB ｺﾏﾝﾄﾞ送信
    GpStatus = gp_wrt( szHP3478A, szCommand );
    if ( GpStatus != 0 )
    {
        sprintf( szBuf, "gp_wrt()ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    SetDlgItemText( hwnd, IDS_STATUS, "初期化正常終了" );
}
```

VisualC サンプルプログラム抜粋(ポーリングモード)

計測開始ボタンを押したときの処理・・・HP3478A に対してトリガコマンドを送ることで測定が開始され、SRQ を指定時間待った後、シリアルポールを行い、測定データを読み込みます。

```
void Cmd_OnCmdStart ( HWND hwnd )
{
    INT   GpStatus;
    char   RcvData[256]; // 受信バッファ
    BYTE   StatusByte[16]; // ステータスバイト格納用バッファ

    // ﾄﾘｶﾞｰｺﾏﾝﾄﾞ実行
    GpStatus = gp_trg( szHP3478A );
    if ( GpStatus != 0 )
    {
        sprintf( szBuf, "gp_trg()ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    // 指定時間 SRQ を待つ
    GpStatus = gp_wsrq( 10 );
    if ( GpStatus != 0 )
    {
        sprintf( szBuf, "gp_wsrq()ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    // ｼﾘｱﾙﾎﾟｰﾙを実行しｽﾃｰﾀｽﾊﾞｲﾄを受信
    GpStatus = gp_rds( szHP3478A, StatusByte );
    if( GpStatus != 0 )
    {
        sprintf( szBuf,"ｽﾃｰﾀｽﾊﾞｲﾄﾘｰﾄﾞ gp_rds()ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    sprintf( szBuf,"%x", StatusByte[0] );
    SetDlgItemText( hwnd, IDS_SBYTE, szBuf );

    // GPIB ﾊﾞｽからﾃﾞｰﾀをﾘｰﾄﾞ
    memset( RcvData, 0x00, sizeof(RcvData) );
    GpStatus = gp_red( szHP3478A, RcvData, sizeof(RcvData) );
    if( GpStatus != 0 )
    {
        sprintf( szBuf,"gp_red()ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    // 測定値を表示
    SetDlgItemText( hwnd, IDS_READVAL, RcvData );
}
```

VisualC サンプルプログラム抜粋(割り込みモード)

計測開始ボタンを押したときの処理・・・HP3478A からの SRQ 検知に割り込みを使用します。SRQ を検知した場合、ユーザ定義メッセージ(次頁)によってアプリケーションに知らされます。

```
void Cmd_OnCmdStart ( HWND hwnd )
{
    INT   GpStatus;

    // ｼﾘｱﾙﾎﾟｰﾙ割り込み実行
    GpStatus = gp_srq( hwnd, ENABLE_SRQ_INTERRUPT );
    if ( GpStatus != 0 )
    {
        sprintf( szBuf,"gp_srq()...ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }

    // ﾄﾘｶﾞｰｺﾏﾝﾄﾞ実行
    GpStatus = gp_trg( szHP3478A );
    if ( GpStatus != 0 )
    {
        sprintf( szBuf, "gp_trg()ｴﾗｰ [ERROR:%d]", GpStatus );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        return;
    }
}
```

VisualC サンプルプログラム抜粋(割り込みモード)

ユーザ定義メッセージの処理・・・SRQ を検知した場合、LPARAM に GPIB_EVENTOCCUR がセットされます。シリアルポール実行後、測定データを読み込みます。

```
void Dlg_OnUserDefineMessage (HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    INT   GpStatus;
    char   RcvData[256]; // 受信バッファ
    BYTE   StatusByte[16]; // ステータスバイト格納用バッファ

    switch(HIWORD(lParam))
    {
    case GPIB_EVENTSTART: // gp_srq(hwnd, ENABLE_SRQ_INTERRUPT)が呼ばれた場合
        break;
    case GPIB_EVENTOCCUR: // SRQ 割り込みが発生した場合
        sprintf( szBuf,"SRQ イベント発生" );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        // ｼﾘｱﾙﾎﾟｰﾙを実行しｽﾃｰﾀｽﾊﾞｲﾄを受信
        GpStatus = gp_rds( szHP3478A, StatusByte );
        if( GpStatus != 0 )
        {
            sprintf( szBuf,"ｽﾃｰﾀｽﾊﾞｲﾄﾘｰﾄﾞ gp_rds()ｴﾗｰ [ERROR:%d]", GpStatus );
            SetDlgItemText( hwnd, IDS_STATUS, szBuf );
            return;
        }
        sprintf( szBuf,"%x", StatusByte[0] );
        SetDlgItemText( hwnd, IDS_SBYTE, szBuf );

        // GPIB ﾊﾞｽからﾃﾞｰﾀをﾘｰﾄﾞ
        memset( RcvData, 0x00, sizeof(RcvData) );
        GpStatus = gp_red( szHP3478A, RcvData, sizeof(RcvData) );
        if( GpStatus != 0 )
        {
            sprintf( szBuf,"gp_red()ｴﾗｰ [ERROR:%d]", GpStatus );
            SetDlgItemText( hwnd, IDS_STATUS, szBuf );
            return;
        }

        // 測定値を表示
        SetDlgItemText( hwnd, IDS_READVAL, RcvData );

        gp_srq( hwnd, DISABLE_SRQ_INTERRUPT );
        break;
    case GPIB_EVENTSTOP: // SRQ 待ち状態を終了する場合
        sprintf( szBuf,"SRQ 待ち終了" );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        break;
    case GPIB_EVENTERROR: // 予期せぬエラー
        sprintf( szBuf,"予期せぬエラー" );
        SetDlgItemText( hwnd, IDS_STATUS, szBuf );
        break;
    } // End of switch(HIWORD(lParam))

}
```

VisualBASIC サンプルプログラム抜粋(ポーリング/割り込みモード共通)

初期化ボタンを押したときの処理・・・GPIB コントローラの初期化を行った後、続けて HP3478A に対しクリアコマンド、測定用コマンドを送信します。

```
Private Sub INIT_Click()

    GpAdrs = GpibAdrs.Text
    ' GPIB コントローラ初期化
    Status = U2GPIBAX.gpinit(MyGpibAdrs, 0, 0)
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gpinit()ｴﾗｰ :" & Status
        Exit Sub
    End If

    '   IFC ﾗｲﾝを TRUE にする
    U2GPIBAX.gpcli

    '   REN ﾗｲﾝを TRUE にする
    U2GPIBAX.gpren

    ' ｾﾚｸﾃｯﾄﾞﾃﾞﾊﾞｲｽｸﾘｱｺﾏﾝﾄﾞ送出
    Status = U2GPIBAX.gpclr(GpAdrs)
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gpclr()ｴﾗｰ :" & Status
        Exit Sub
    End If

    Status = U2GPIBAX.gpllo()
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gpllo()ｴﾗｰ :" & Status
        Exit Sub
    End If

    '   HP3478A GPIB ｺﾏﾝﾄﾞ送信
    Status = U2GPIBAX.gpwrt(GpAdrs, "H0KM01")
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gpwrt()ｴﾗｰ :" & Status
        Exit Sub
    End If
    ERROR.Text = "初期化正常終了"

End Sub
```

VisualBASIC サンプルプログラム抜粋(ポーリングモード)

計測開始ボタンを押したときの処理・・・HP3478A に対してトリガコマンドを送ることで測定が開始され、SRQ を指定時間待った後、シリアルポールを行い、測定データを読み込みます。

```
Private Sub OK_Click()
    Dim Code(8) As Integer

    '  ﾄﾘｶﾞｰｺﾏﾝﾄﾞ実行
    Status = U2GPIBAX.gptrg(GpAdrs)
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gptrg()ｴﾗｰ :" & Status
        Exit Sub
    End If

    '  指定時間 SRQ を待つ
    Status = U2GPIBAX.gpwsrq(10)
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gpwsrq()ｴﾗｰ :" & Status
        Exit Sub
    End If
    ' ｼﾘｱﾙﾎﾟｰﾙを実行しｽﾃｰﾀｽﾊﾞｲﾄを受信
    Status = U2GPIBAX.gprds(GpAdrs, Code(0))
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gprds()ｴﾗｰ :" & Status
        Exit Sub
    End If
    SBYTE.Text = Hex(Code(0))

    ' GPIB ﾊﾞｽからﾃﾞｰﾀをﾘｰﾄﾞ
    szBuf = String(256, &H0)
    Status = U2GPIBAX.gpred(GpAdrs, szBuf, Len(szBuf))
    If Status <> 0 Then
        ERROR.Text = "U2GPIBAX.gpred()ｴﾗｰ :" & Status
        Exit Sub
    End If

    ' 測定値を表示
    READVAL.Text = szBuf
End Sub
```

VisualBASIC サンプルプログラム抜粋(割り込みモード)

計測開始ボタンを押したときの処理・・・HP3478A からの SRQ 検知に割り込みを使用します。SRQ を検知した場合、ユーザ定義メッセージ(次頁)によってアプリケーションに知らされます。

```
Private Sub OK_Click()

    ' ｼﾘｱﾙﾎﾟｰﾙ割り込み実行
    Status = U2GPIBAX.gpsrq(0, ENABLE_SRQ_INTERRUPT)
    If Status <> 0 Then
        ERROR.Text = "u2gpibax.gpsrq()ｴﾗｰ :" & Status
        Exit Sub
    End If

    '  ﾄﾘｶﾞｰｺﾏﾝﾄﾞ実行
    Status = U2GPIBAX.gptrg(GpAdrs)
    If Status <> 0 Then
        ERROR.Text = "u2gpibax.gptrg()ｴﾗｰ :" & Status
        Exit Sub
    End If

End Sub
```

VisualBASIC サンプルプログラム抜粋(割り込みモード)

ユーザ定義メッセージの処理・・・SRQ を検知した場合、LPARAM に GPIB_EVENTOCCUR がセットされます。シリアルポール実行後、測定データを読み込みます。

```
Private Sub U2GPIBAX_OnEventMsg(ByVal wParam As Long, ByVal lParam As Long)
    Dim Msg As Long
    Dim Unit As Long

    ' wParam：hUnit      lParam：MAKELPARAM(0, xxx)
    Msg = (lParam And &HFFFF0000) / (2 ^ 16)
    Unit = lParam And &HFFFF
    ' ﾒｯｾｰｼﾞの判定
    Select Case Msg
        Case GP_STARTSRQ 'U2GPIBAX.gpsrq(0, ENABLE_SRQ_INTERRUPT)が呼ばれた場合
        Case GP_SRQ_EVENT 'SRQ 割り込みが発生した場合
            ERROR.Text = "SRQ イベント発生"
            Data_Read ‘ データの読込み
            Call U2GPIBAX.gpsrq(0, DISABLE_SRQ_INTERRUPT)
        Case GP_STOPSRQ 'SRQ 待ち状態を終了する場合
            ERROR.Text = "SRQ 待ち終了"
        Case GP_ERROR '予期しないエラー
            ERROR.Text = "予期しないエラー"
    End Select

End Sub

Sub Data_Read()
    Dim Code(8) As Integer

    ' ｼﾘｱﾙﾎﾟｰﾙを実行しｽﾃｰﾀｽﾊﾞｲﾄを受信
    Status = U2GPIBAX.gprds(GpAdrs, Code(0))
    If Status <> 0 Then
        ERROR.Text = "u2gpibax.gprds()ｴﾗｰ :" & Status
        Exit Sub
    End If
    SBYTE.Text = Hex(Code(0))

    ' GPIB ﾊﾞｽからﾃﾞｰﾀをﾘｰﾄﾞ
    szBuf = String(256, &H0)
    Status = U2GPIBAX.gpred(GpAdrs, szBuf, Len(szBuf))
    If Status <> 0 Then
        ERROR.Text = "u2gpibax.gpred()ｴﾗｰ :" & Status
        Exit Sub
    End If

    ' 測定値を表示
    READVAL.Text = szBuf

End Sub
```

第 4 章 追加情報

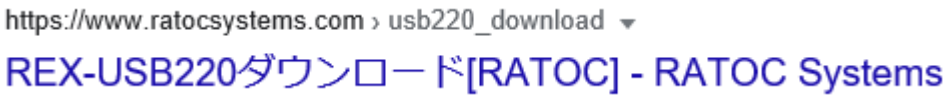
この章では、ファームウェアの更新方法について説明します。

● ファームウェアアップデートプログラムのダウンロード

ホームページ右上の検索欄に「USB220 ダウンロード」と入力し検索します。

https://www.ratocsystems.com/

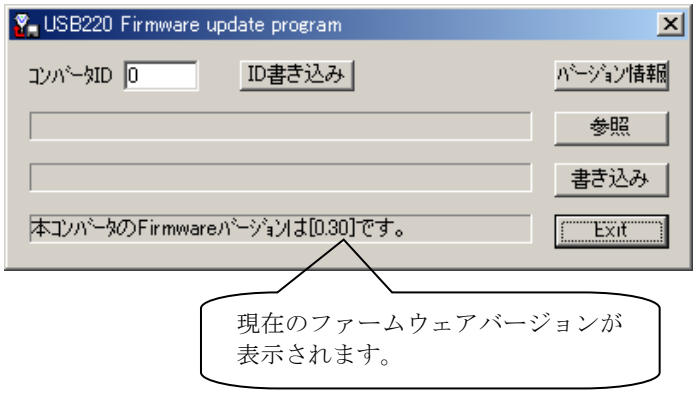
下記ダウンロードページへのリンクをクリックし、表示されたページの
「REX-USB220 用ファームウェア」をダウンロードします。
ダウンロードしたデータを解凍したフォルダー内に、ファームウェアアップデートプログラム(Firmup.exe)が収録されています。



4-1. ファームウェアアップデート

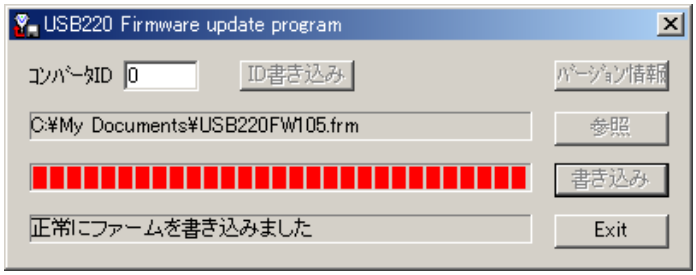
ファームウェアアップデートプログラム（Firmup.exe）により、コンバータ内のファームウェアを書き換えて本コンバータの機能を拡張したり、不具合動作を修正することができます。弊社ホームページで公開されたファームウェアのバージョンアップ情報に基づいて新しいファームウェア（拡張子 FRM ファイル）をダウンロードしてください。ファームウェアアップデートプログラム起動後、以下の手順で書き換えを行います。

注意
ファームウェアアップデート実行中に USB のケーブルを抜いたり、パソコンをリセットしたりしないでください。万一、再起動後に本コンバータの認識ができなくなった場合は、弊社に送付していただき書き換えを行うことになります。



パソコンの USB ポートには本コンバーター一台のみ接続してください。HUB 等を使用して複数の USB 機器を接続されている場合は、全てとりはずしておいてください。
Firmup.exe 起動後、左図が表示されます。
(コンバータ ID には、現在接続されているコンバータの ID が表示されます。)
「参照」ボタンを押して、ダウンロードした拡張子 FRM のファイルを指定してください。

指定したファイルのパス名を確認してから「書き込み」ボタンを押してください。
ファームウェアの書き込みには数十秒かかります。ファームウェアの書き込み中は他の作業を行わないでください。



書き込みが終了すると、左図のように表示されます。「Exit」ボタンを押して、プログラムを終了してください。
プログラム終了後、USB のケーブルを一旦取り外し、再度、接続してください。
以上で、ファームウェアの書き込みは終了です。

4-2. レジスタセット

GPIB コントローラのレジスタセットは以下のとおりです。各レジスタの詳細については、NAT9914 リファレンスマニュアルを参照してください。(NAT9914 リファレンスマニュアルは National Instruments 社ホームページよりダウンロード可能です。)

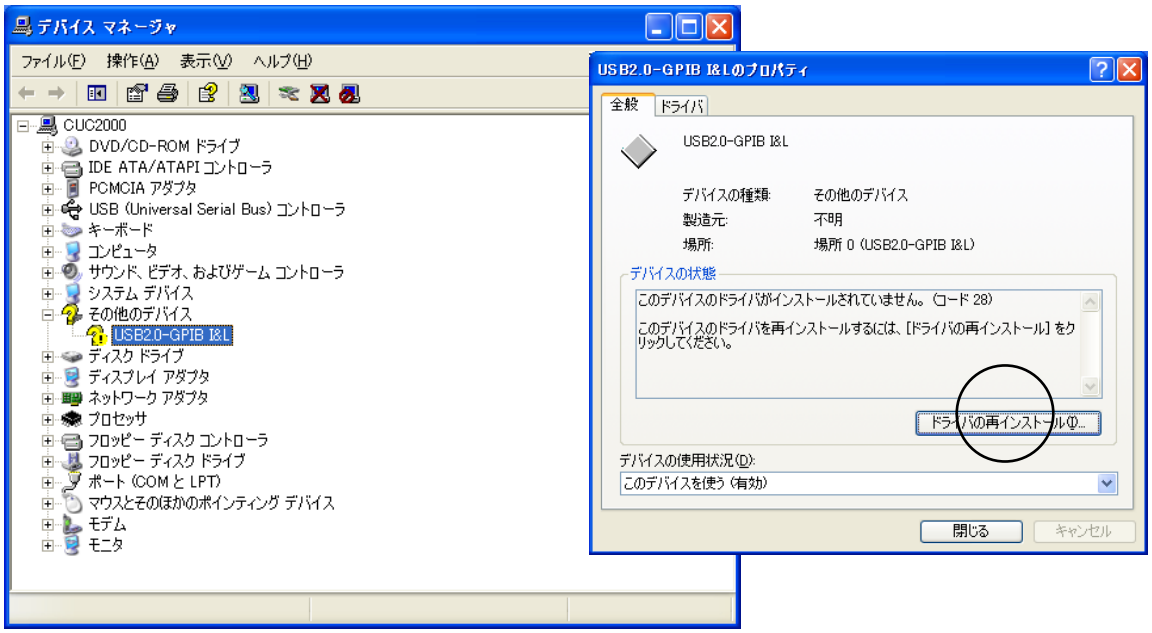
オフセット	R/W	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
+0	R	INT0	INT1	BI	BO	END	SPAS	RLC	MAC
+0	W	DMA0	DMA1	BIIE	BOIE	ENDIE	SPASIE	RLCIE	MACIE
+1	R	GET	ERR	UNC	APT	DCAS	MA	SRQ	IFC
+1	W	GETIE	ERRIE	UNCIE	APTIE	DCASIE	MAIE	SRQIE	IFCIE
+2	R	REM	LLO	ATN	LPAS	TPAS	LA	TA	ulpa
+2	W	GLINT	STBOIE	NLEN	0	LLOCIE	ATNIIE	0	CICIE
+2	W	EOS7	EOS6	EOS5	EOS4	EOS3	EOS2	EOS1	EOS0
+2	W	ATN	DAV	NDAC	NRFD	EOI	SRQ	IFC	REN
+2	W	ACC7	ACC6	ACC5	ACC4	ACC3	ACC2	ACC1	ACC0
+3	R	ATN	DAV	NDAC	NRFD	EOI	SRQ	IFC	REN
+3	W	C/S	0	0	F4	F3	F2	F1	F0
+4	R	nba	STB0	NL	EOS	LLOC	ATNI	X	CIC
+4	W	edpa	dal	dat	A5	A4	A3	A2	A1
+5	R	S8	PEND	S6	S5	S4	S3	S2	S1
+5	W	S8	rsv/RQS	S6	S5	S4	S3	S2	S1
+6	R	CPT7	CPT6	CPT5	CPT4	CPT3	CPT2	CPT1	CPT0
+6	W	PP8	PP7	PP6	PP5	PP4	PP3	PP2	PP1
+7	R	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1
+7	W	DIO8	DIO7	DIO6	DIO5	DIO4	DIO3	DIO2	DIO1

+0(R)　インタラプトステータスレジスタ 0
+1(R)　インタラプトステータスレジスタ 1
+2(R)　アドレスステータスレジスタ
+3(R)　バスステータスレジスタ
+4(R)　インタラプトステータスレジスタ 2
+5(R)　シリアルポールステータスレジスタ
+6(R)　コマンドパススルーレジスタ
+7(R)　データインレジスタ

+0(W)　インタラプトマスクレジスタ 0
+1(W)　インタラプトマスクレジスタ 1
+2(W)　インタラプトマスクレジスタ 2
+2(W)　エンドオブストリングレジスタ
+2(W)　バスコントロールレジスタ
+2(W)　アクセサリーリードレジスタ
+3(W)　補助コマンドレジスタ
+4(W)　アドレスレジスタ
+5(W)　シリアルポールモードレジスタ
+6(W)　パラレルポールレジスタ
+7(W)　コマンド/データアウトレジスタ

4.3.1 インストールに失敗した場合

2.1.1 項 「ドライバのセットアップ」を行わず、USB220 コンバータをインストールしようとした場合には、インストールに失敗します。デバイスマネージャ上で左下画面のようにその他のデバイス「USB2.0-GPIB I&L」と表示されている場合には、2.1.1 項 「ドライバのインストール」を行った後、右下画面の「プロパティ」からドライバの再インストールを行ってください。



4.3.2 サンプルプログラムの使用について

以前に CD-ROM で提供されたサンプルプログラムをハードディスクにコピーして使用する場合、ファイル属性が「読み取り専用」になっています。ファイルの編集を行う場合には、ファイルのプロパティで「読み取り専用」となっている属性を解除してください。