REX-PE20/20L

GPIB PCI Express Board

ユーザーズ マニュアル

2021 年 11 月

第 2.0 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

<サポート受付時間>

月曜－金曜（祝祭日は除く） 10:00 - 13:00

14:00 - 17:00

また、インターネットのホームページでも受け付けています。

http://www.ratocsystems.com/

【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。

FAX：06-7670-5066

REX-PE20/20L 質問用紙

●下記ユーザ情報をご記入願います。

法人登録の方のみ	会社名・学校名			
	所属部署			
ご担当者名				
E-Mail				
住所	〒			
TEL		FAX		
製品型番		シリアルNo.		
ご購入情報	販売店名		ご購入日	

●下記運用環境情報とお問い合わせ内容をご記入願います。

【パソコン/マザーボードのメーカ名と機種名】
【ご利用の OS】
【接続機器】
【お問合せ内容】
【添付資料】

⚠ 個人情報取り扱いについて

ご連絡いただいた氏名、住所、電話番号、メールアドレス、その他の個人情報は、お客様への回答など本件に関わる業務のみに利用し、他の目的では利用致しません。

安全にご使用いただくために

本製品は安全に充分配慮して設計を行っていますが、誤った使い方をすると火災や感電などの事故につながり大変危険です。ご使用の際は、警告/注意事項を必ず守ってください。

表示について

この取扱説明書は、次のような表示をしています。表示の内容をよく理解してから本文をお読みください。

⚠警告　この表示を無視して誤った取扱いをすると、火災や感電などにより、人が死亡または重傷を負う可能性がある内容を示しています。

⚠注意　この表示を無視して誤った取扱いをすると、感電やその他の事故により、人が負傷または物的損害が発生する可能性がある内容を示しています。

⚠警告

- 製品の分解や改造などは、絶対に行わないでください。
- 無理に曲げる、落とす、傷つける、上に重い物を載せることは行わないでください。
- 製品が水・薬品・油などの液体によって濡れた場合、ショートによる火災や感電の恐れがあるため使用しないでください。

⚠注意

- 本製品は電子機器ですので、静電気を与えないでください。
- ラジオやテレビ、オーディオ機器の近く、モーターなどのノイズが発生する機器の近くでは誤動作することがあります。必ず離してご使用ください。
- 高温多湿の場所、温度差の激しい場所、チリやほこりの多い場所、振動や衝撃の加わる場所、スピーカなどの磁気を帯びた物の近くで保管しないでください。
- 煙が出たり異臭がする場合は、直ちにパソコンや周辺機器の電源を切り、電源ケーブルもコンセントから抜いてください。
- 本製品は、医療機器、原子力機器、航空宇宙機器、輸送機器など人命に関わる設備や機器、及び高度な信頼性を必要とする設備や機器での使用は意図されておりません。これらの設備、機器制御システムに本製品を使用し、本製品の故障により人身事故/火災事故/その他の障害が発生した場合、いかなる責任も負いかねます。
- 取り付け時、鋭い部分で手を切らないように、十分注意して作業を行ってください。
- 配線を誤ったことによる損失、逸失利益などが発生した場合でも、いかなる責任も負いかねます。

その他のご注意

- 本書の内容に関して、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ておりますが、万一不審な点や誤りなどお気づきになりましたらご連絡お願い申し上げます。
- 本製品の運用を理由とする損失、逸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、予めご了承ください。
- 製品改良のため、将来予告なく外観または仕様の一部を変更する場合があります。
- 本製品は日本国内仕様となっており、海外での保守及びサポートは行っておりません。
- 本製品を廃棄するときは地方自治体の条例に従ってください。条例の内容については各地方自治体にお問い合わせください。
- 本製品の保証や修理に関しましては、添付の保証書に内容を明記しております。必ず内容をご確認の上、大切に保管してください。
- “REX”は株式会社リコーが商標権を所有しておりますが、弊社はその使用許諾契約により本商標の使用が認められています。
- Windowsは米国マイクロソフト社の米国およびその他の国における登録商標です。その他本書に記載されている商品名/社名などは、各社の商標または登録商標です。 なお本書では、™、Ⓡ マークは明記しておりません。

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-PE20/20L GPIB PCI Express Board をご購入いただきましてありがとうございます。REX-PE20/20L は PCI Express バス用 GPIB インターフェイスで専用アプリケーションを構築することにより GPIB インターフェイスの各社計測器を制御することが可能です。

(特徴)

●IEEE488.2 準拠の GPIB インターフェイス

●PCI Express Rev.1.0A

●ラインナップとして標準 PCI Express スロットのみに装着できる REX-PE20 と Low Profile PCI Express スロットにも装着可能な REX-PE20L の 2 製品を用意。(両製品の基板部分とソフトウェアは共通です。) REX-PE20L には標準 PCI Express 用と Low Profile PCI Express 用のブラケットを添付。さらに PC ケースと GPIB ケーブルの干渉を回避する GPIB コネクタアダプタを添付しています。

●VisualC++、VisualBasic 用のライブラリを豊富に提供。
アプリケーション開発を強力に支援します。

●弊社製GPIB PCカード REX-5052、USB2.0 to GPIBコンバータ REX-USB220、およびGPIB PCI Board REX-PCI20/20L 用に作成したプログラムは、簡単な変更を加えることにより本製品でも動作します。(REX-PCI20/20L 用はそのままご使用いただけます。)
Visual C++ではプロジェクトに組み込むヘッダファイルとライブラリファイルを変更、
Visual Basic では使用している gp_xxx()関数を対応するメソッドに変更、
することで使用可能となります。

●RoHS 指令対応。

1-2. 梱包内容のご確認

ご使用前に添付品のご確認をお願いします。

内　容	個　数	備　考
REX-PE20/20L GPIB PCI Express ボード本体	1	
サポートソフトウェア CD-ROM	1	
インストールガイド	1	
保証書	1	
標準 PCI Express ブラケット	1	※1 参照
GPIB コネクタアダプタ	1	※2 参照

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）標準 PCI Express ブラケットについて（REX-PE20L のみ付属）

REX-PE20L は製品出荷時に Low Profile PCI Express スロットのブラケットが取り付けてありますが、付属の標準 PCI Express ブラケットに取り替えて頂くことで、標準 PCI Express スロットでもご使用いただけます。

※2）GPIB コネクタアダプタについて（REX-PE20L のみ付属）

REX-PE20L を Low Profile PCI Express スロットのパソコンに装着すると、パソコン筐体と GPIB コネクタが干渉しケーブルが装着できないことがあります。そのような場合は付属の GPIB コネクタアダプタをボード側コネクタと GPIB ケーブルの間に取り付けてください。

1-3. 製品仕様

1.3.1 ハードウェア仕様

項　目	仕　様	備　考
入出力方式	GPIB（IEEE488.2 準拠）	
GPIB コントローラ	Texas Instruments 製 TMS99C14 互換	
入出力コネクタ	IEEE488 インターフェイスコネクタ (アンフェノール 24 ピン　メス)	
接続バス	PCI Express Rev.1.0 A	
外形寸法	約 64mm(W) x 約 120mm(L)（突起部含まず）	
重量	約 80g（REX-PE20 の場合）	※1 参照
動作温度	0～55℃	但し、結露しないこと

※1）REX-PE20 は約 80g（標準 PCI Express ブラケット含む）、
REX-PE20L は約 70g（Low Profile PCI Express ブラケット含む）になります。

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

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	SHIELD	Shield Ground	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	
製品付属ドライバ	WDM ドライバ	
製品付属ライブラリ	32 ビット DLL ライブラリ ActiveX コントロール	
対応開発言語	Microsoft Visual C/C++ Microsoft Visual BASIC	

1.4 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 ｺｰﾄﾞ	インターフェイス 機能	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-PE20/20L 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。

(※ 以降「REX-PE20/20L」は「REX-PE20」と表現してあります。)

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-PE20 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共存はできません。従って REX-PE20 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こと(C0)。

(ATN，IFC，REN ラインの管理機能を持たないこと)

また REX-PE20 を実装し REX-PE20 ライブラリが動作中のパソコンはすべてコントローラインチャージ(コントローラとしてバスの制御権を獲得している状態)ですので、GPIB 関係のコマンドを実行していなくとも、他のコントローラとバス上での共存はできません。

第 2 章 インストレーション

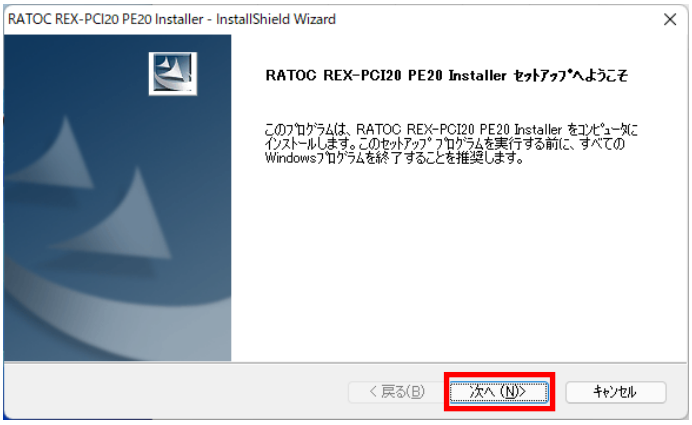
この章では、Windows でのインストール方法について説明します。

2-1. Windows 11 / 10 / 8.1 / 7 セットアップ

ホームページよりダウンロードしたドライバーをセットアップし、PC をシャットダウンしてから本製品を PC に接続します。

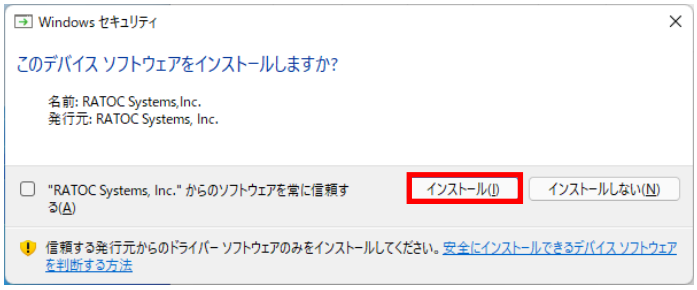
ユーザーアカウント制御の画面で「はい」をクリックします。

「次へ」をクリックします。

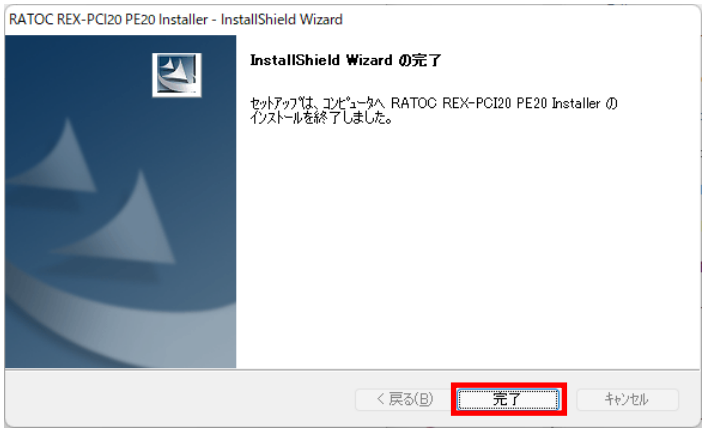


インストール準備の加完了画面で
「インストール」をクリックします。

Windows セキュリティ画面が表示される場合は
「インストール」をクリックします。



以上でREX-PCI20 のインストールは
完了です。

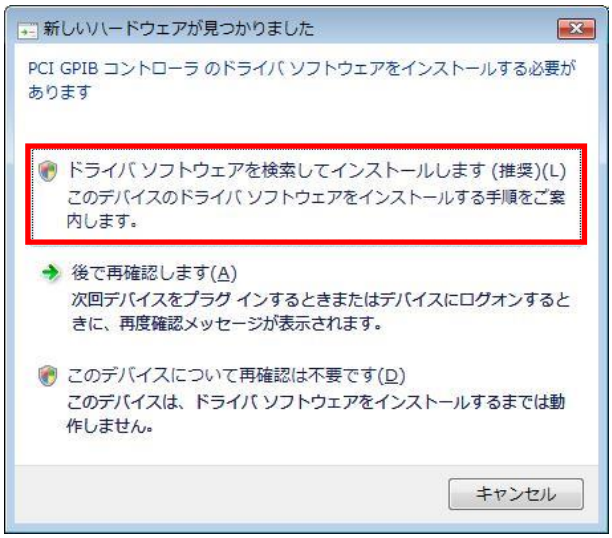


「2-5 REX-PCI20 インストールの確認」へ進み、正常にインストールされたのか確認を行ってください。

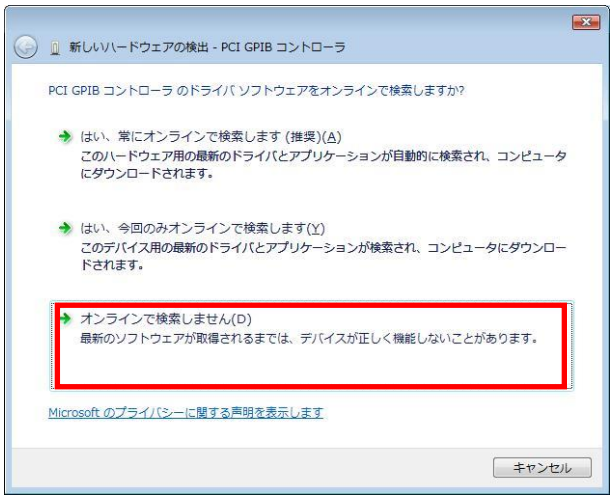
2-2. Windows Vista / Vista x64 セットアップ

本製品を PCI Express スロットへ装着し、PC の電源を ON にしてください。
新しいハードウェアの検出ウィザードが起動しますので、以下の手順でインストールを行ってください。

「PCI GPIB コントローラのドライバソフトウエアをインストールする必要があります」と表示されていることを確認し、「ドライバソフトウェアを検索してインストールします(推奨)(L)」をクリックします。



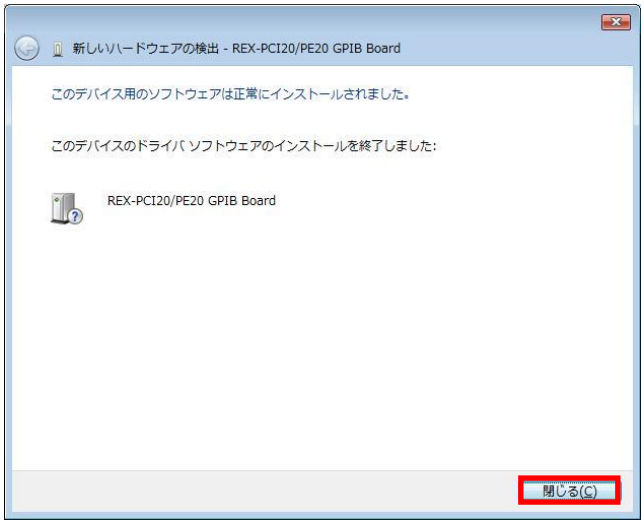
「PCI GPIB コントローラのドライバソフトウェアをオンラインで検索しますか？」のダイアログが表示されますので、「オンラインで検索しません(D)」をクリックします。



製品添付 CD-ROM を挿入し、
「次へ(N)」ボタンをクリックします。

「このデバイスのソフトウェアをインストールしますか？」のダイアログで、「"RATOC Systems,Inc.”からのソフトウェアを常に信頼する(A)」にチェックを入れ、「インストール(I)」ボタンをクリックします。

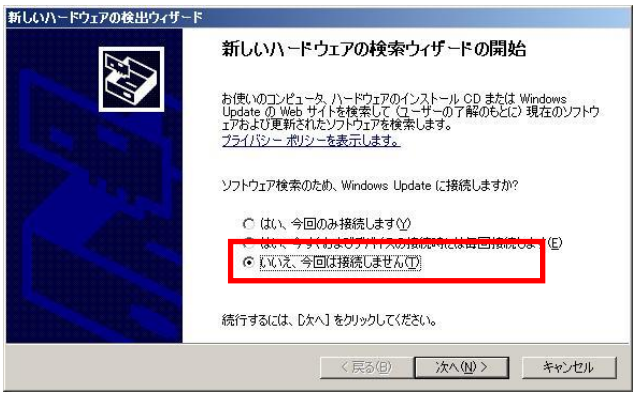
以上で REX-PE20 のインストールは完了です。



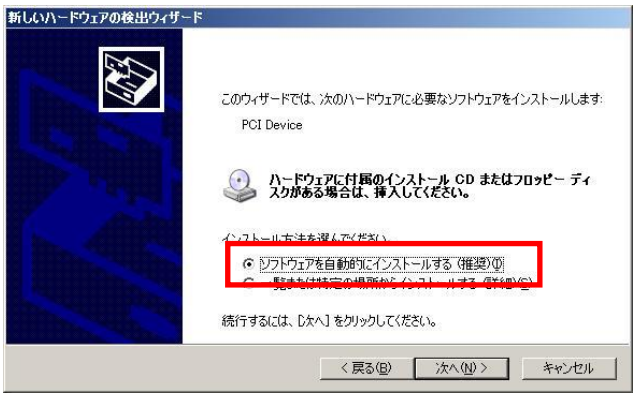
「2-5 REX-PE20 インストールの確認」へ進み、正常にインストールされたのか確認を行ってください。

2-3.　Windows XP / XPx64 セットアップ

「新しいハードウェアの検索ウィザードの開始」のダイアログが表示されますので、「いいえ、今回は接続しません(T)」を選択し「次へ(N)」をクリックします。



製品添付 CD-ROM を挿入し、「ソフトウェアを自動的にインストールする(推奨)(I)」を選択し、「次へ(N)」ボタンをクリックします。

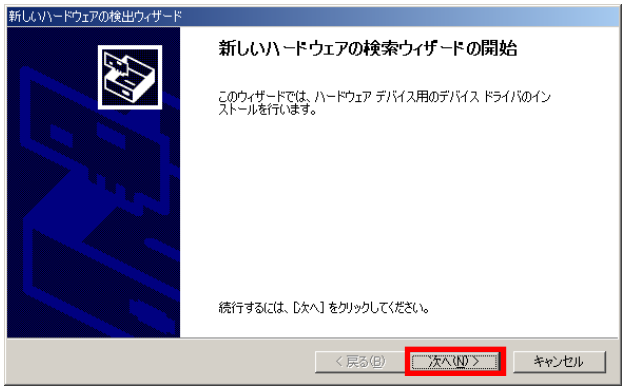


以上で REX-PE20 のインストールは完了です。

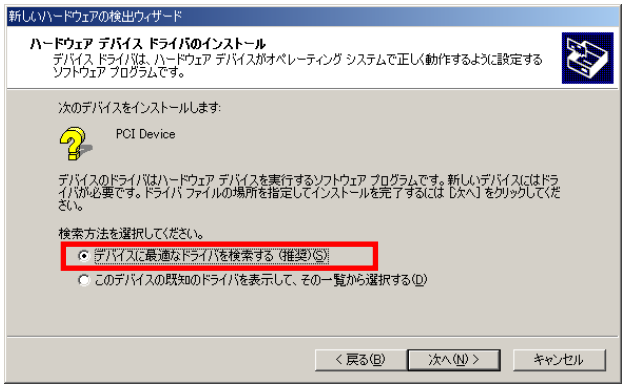
「2-5 REX-PE20 インストールの確認」へ進み、正常にインストールされたのか確認を行ってください。

2-4. Windows 2000 セットアップ

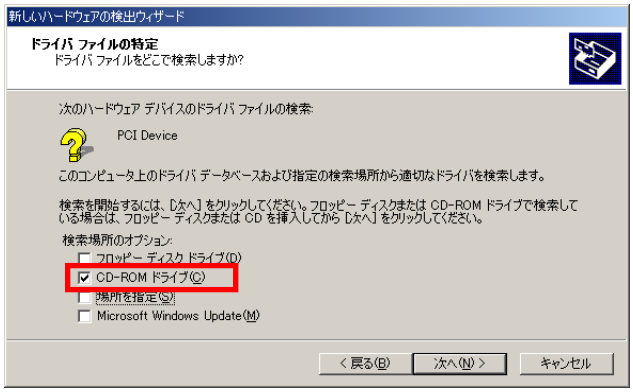
「新しいハードウェアの検索ウィザードの開始」のダイアログが表示されますので、「次へ(N)」をクリックします。



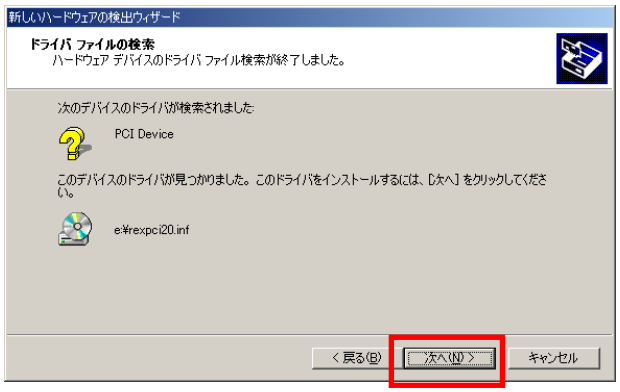
検索方法の選択画面で、「デバイスに最適なドライバを検索する(推奨) (S)」を選択し、「次へ(N)」ボタンをクリックします。



製品添付 CD-ROM を挿入し、「CD-ROM ドライブ(C)」を選択し、「次へ(N)」ボタンをクリックします。



右図のようにドライバファイルが見つかりましたら、「次へ(N)」ボタンをクリックします。



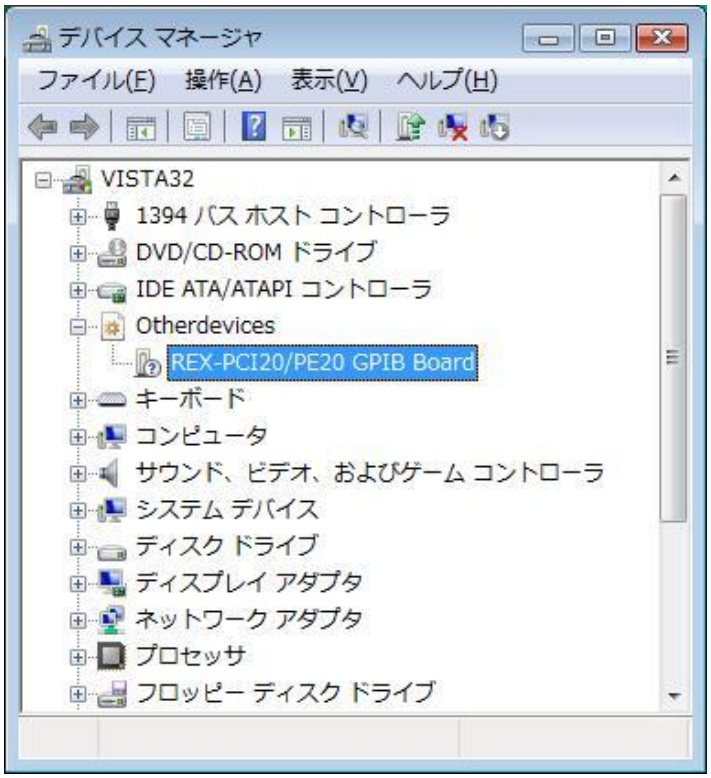
以上で REX-PE20 のインストールは完了です。

「2-5 REX-PE20 インストールの確認」へ進み、正常にインストールされたのか確認を行ってください。

2-4.　REX-PE20 インストールの確認

コントロールパネルの表示をクラシック表示に切り替え、「デバイスマネージャ」を起動します。(※ Windows XP/XPx64/2000 では、コントロールパネルのシステムを起動し「システムのプロパティ」の「ハードウェア」タブから「デバイスマネージャ」ボタンをクリックします。)

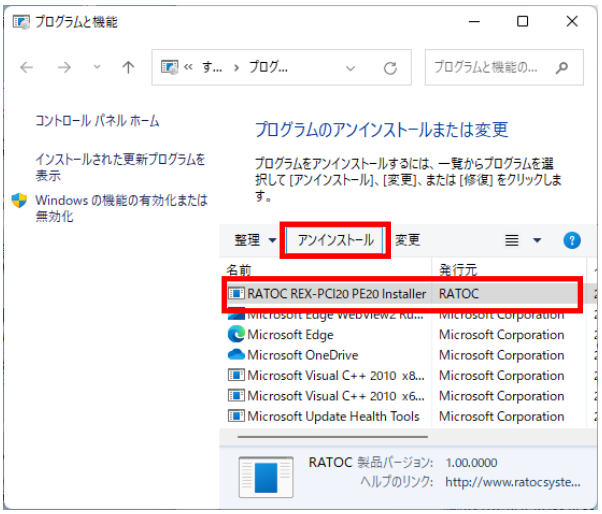
「Otherdevices」クラスの下に「REX-PCI20/PE20 GPIB Board」が正常に認識されていることを確認してください。



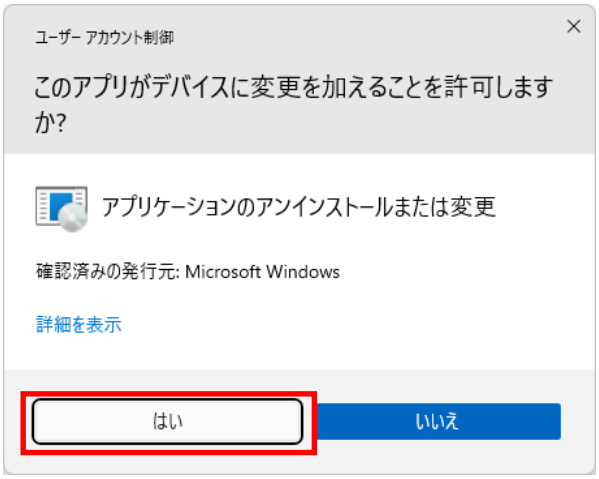
2-6. アンインストール方法

● Windows 11/10/8.1/7 でのアンインストール

コントロールパネルの[プログラムと機能]を起動し、
「RATOC REX-PCI20PE20 Installer」選択してから
「アンインストール」をクリックします。

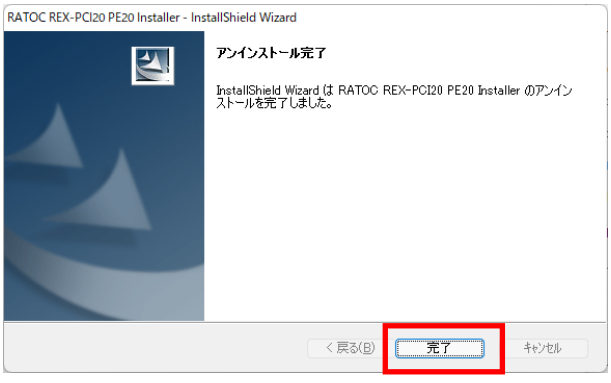


ユーザーアカウント制御画面で
「はい」をクリックします。



「はい」をクリックします。

以上でアンインストールは完了です。

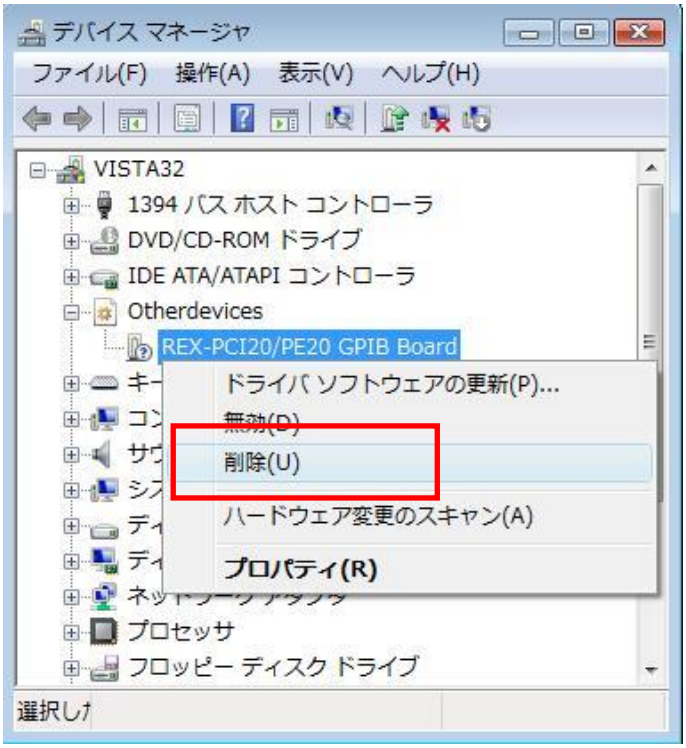


● Windows Vista/XP/2000 でのアンインストール

以下の「ドライバの削除」と「INF ファイルの削除」を行います。
(Windows Vista では「ドライバの削除」のみでアンインストールは完了です。)

・ドライバの削除

コントロールパネルの表示をクラシック表示に切り替え、「デバイスマネージャ」を起動します。
(※ Windows XP/XPx64/2000 では、コントロールパネルのシステムを起動し「システムのプロパティ」の「ハードウェア」タブから「デバイスマネージャ」ボタンをクリックします。)



「REX-PCI20/PE20 GPIB Board」上で右クリックをし「削除」をクリックしてください。

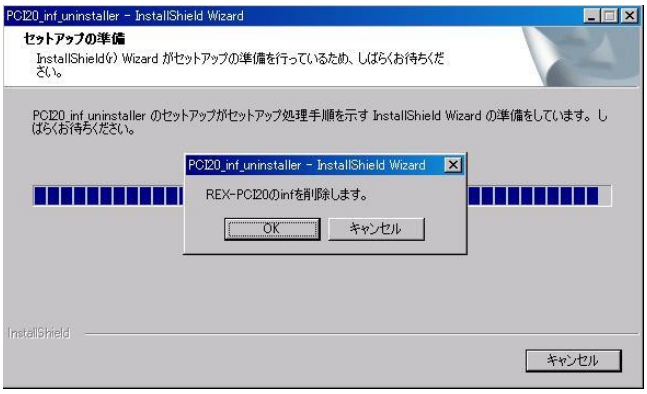
Windows Vista では「このデバイスのドライバソフトウェアを削除する」にチェックを入れ「OK」ボタンをクリックします。

・INF ファイルの削除

(Windows XP/XPx64/2000 のみ必要)

※「PCI20」と表示されておりますが、REX-PCI20/REX-PE20 共通のツールとなります。

製品添付の CD-ROM 内にある「PCI20_uninst.exe」を実行します。
(D:¥PCI20_uninst.exe)



右図画面が表示されましたら「OK」ボタンをクリックします。

完了の画面が表示されましたら「OK」ボタンをクリックしてください。

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。

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製品に添付された DLL ライブラリ「RexPCI20.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

製品付属 CD-ROM の VC フォルダから「RexPCI20.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

製品付属 CD-ROM の VC6 または VC2005 フォルダから「RexPCI20.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製品に添付された 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。

(1) ActiveX の登録

第 2 章インストレーションを参照しドライバのインストールを行ってください。自動的に DLL,ActiveX のコピーが行われます。

PCIGPIBAX.ocx を VB で使用するためには、Visual BASIC に添付されているツール”Regsvr32.exe”を使って登録を行います。”Regsvr32.exe”は 32 ビットコンソールアプリケーションですのでコマンドプロンプトから実行します。(Windows Vista では管理者権限で実行する必要があります。)

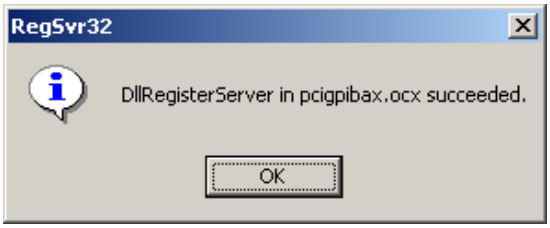
登録の際にはコマンドプロンプトから

```
>regsvr32 pcigpibax.ocx
```

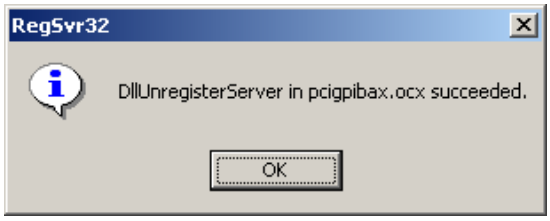
登録から削除する際には

```
>regsvr32 /u pcigpibax.ocx
```

と実行します。



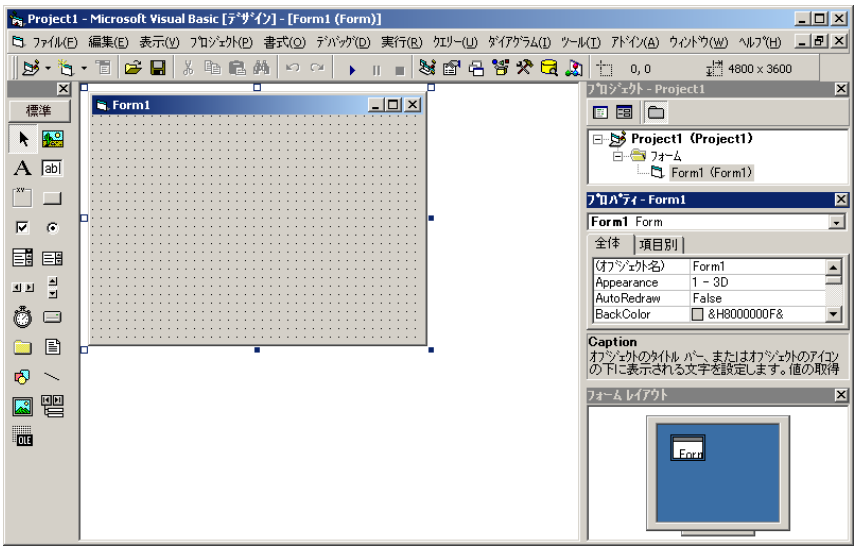
登録成功メッセージ



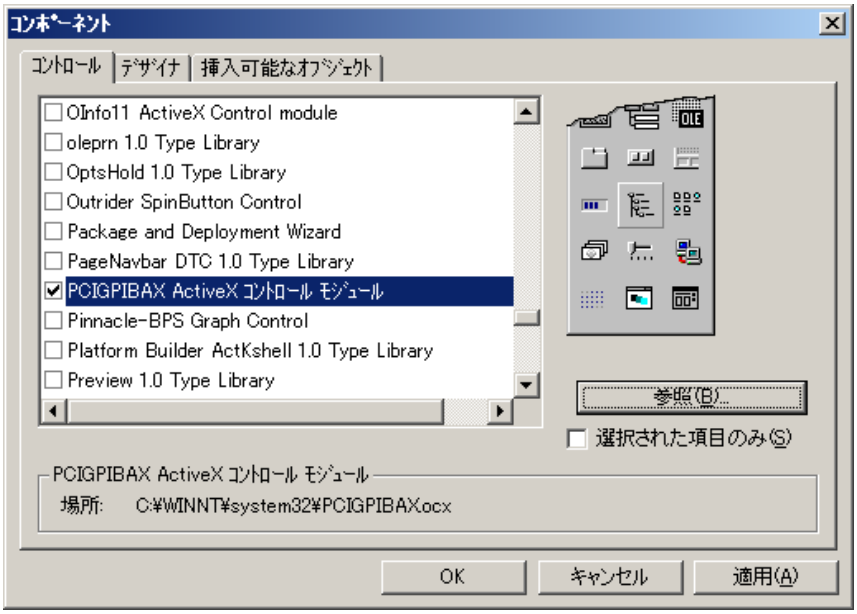
登録削除成功メッセージ

(2) VB6 からの ActiveX 参照方法

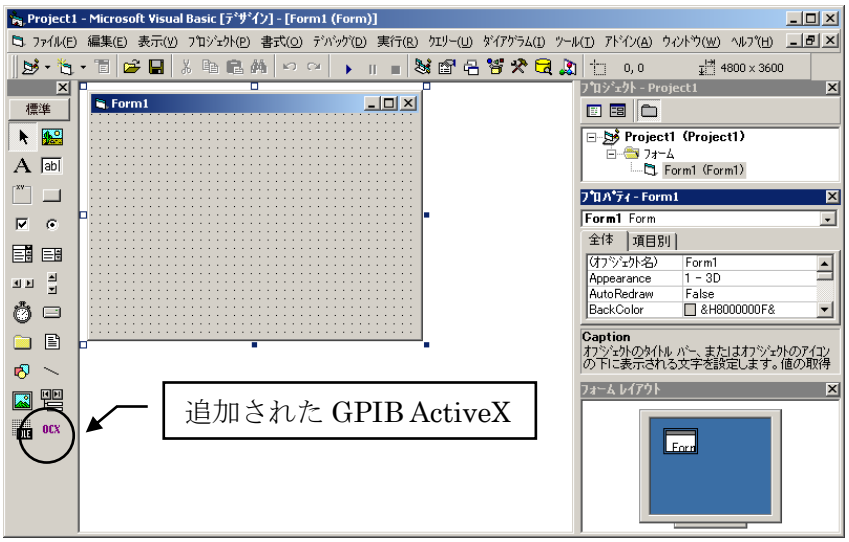
新しいプロジェクトを作成します。



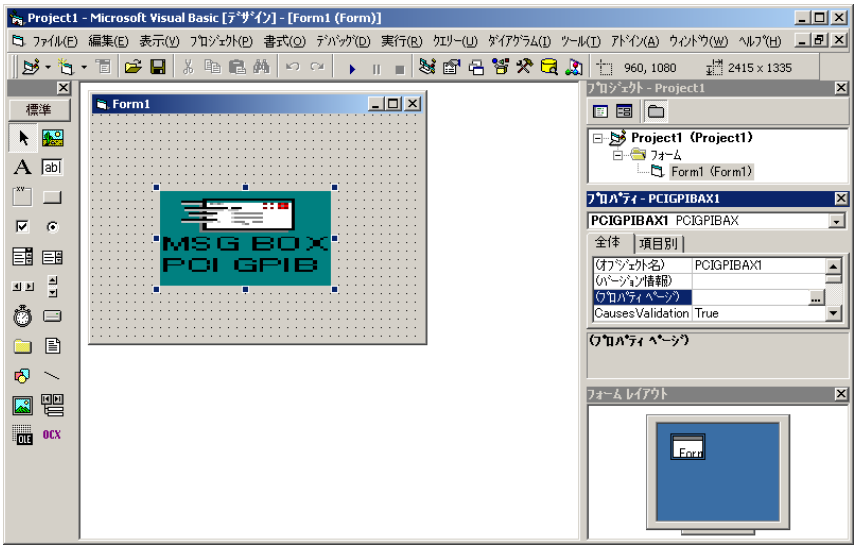
プロジェクトメニューのコンポーネントを選択します。コントロール一覧の、「PCIGPIBAX ActiveX ｺﾝﾄﾛｰﾙﾓｼﾞｭｰﾙ」にチェックを入れて OK ボタンをクリックします。



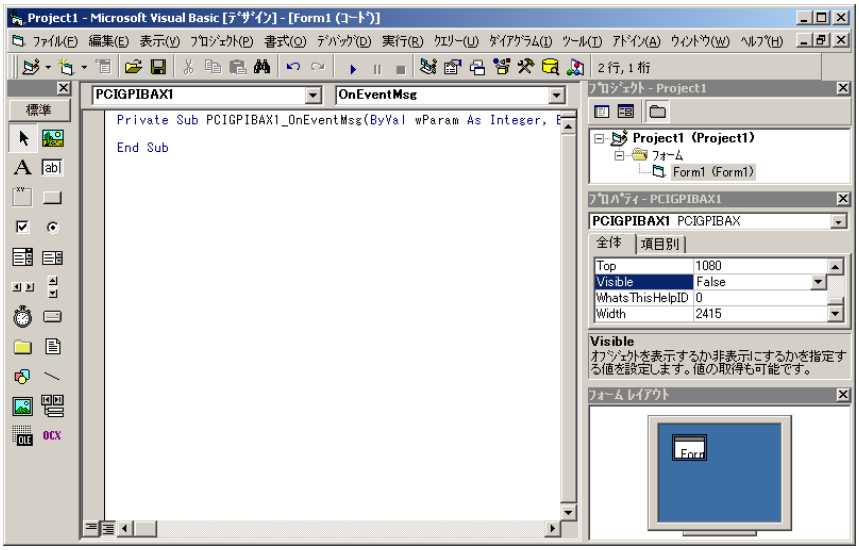
GPIB ActiveX コンポーネントが追加されます。



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 PCI 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。

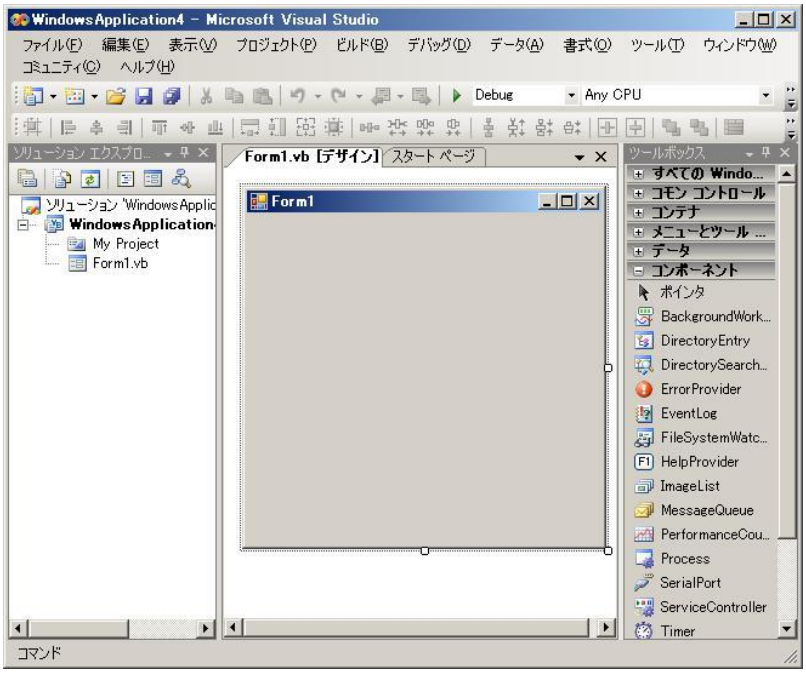


オブジェクトをダブルクリックすると、イベント発生時に呼び出されるサブルーチン Sub PCIGPIBAX1_OnEventMsg（）が表示されます。
SRQ 割り込みモードサンプルプログラムの説明を参照願います。

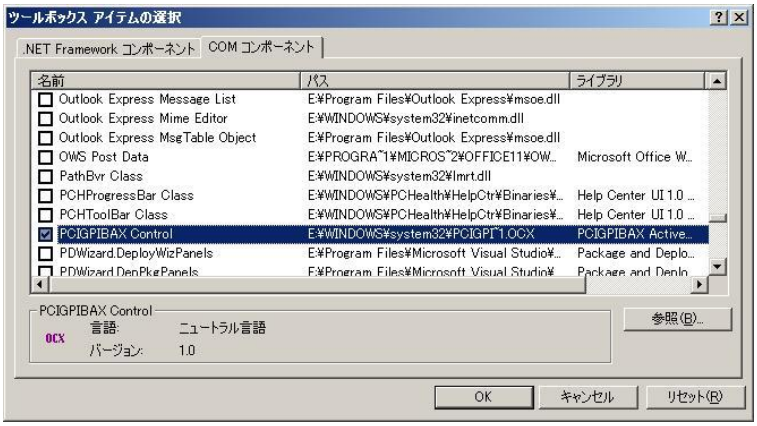


(2) VB.NET からの ActiveX 参照方法

新しいプロジェクトを作成します。

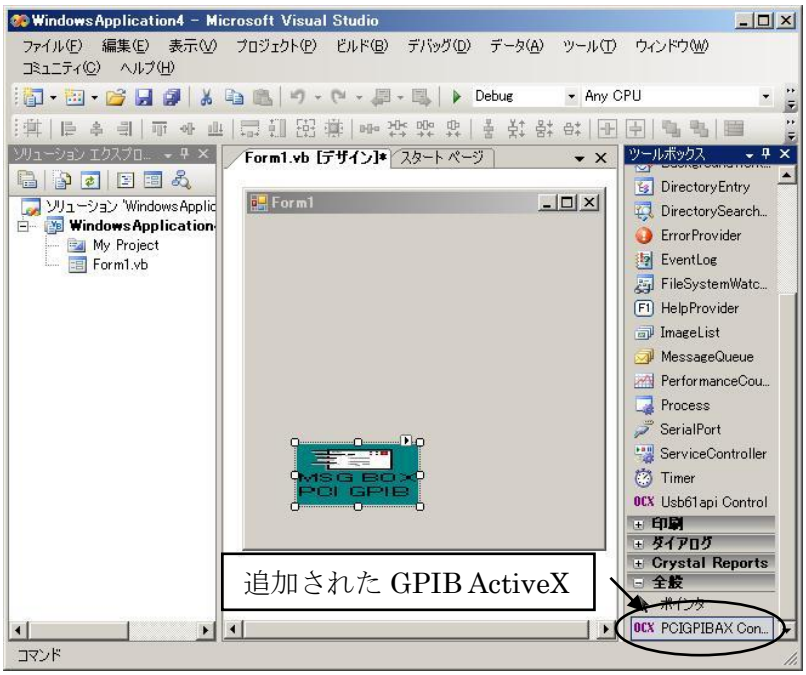


ツールメニューの[ツールボックス アイテムの選択]-[COM コンポーネント]で「PCIGPIBAX Control」にチェックを入れて OK ボタンをクリックします。

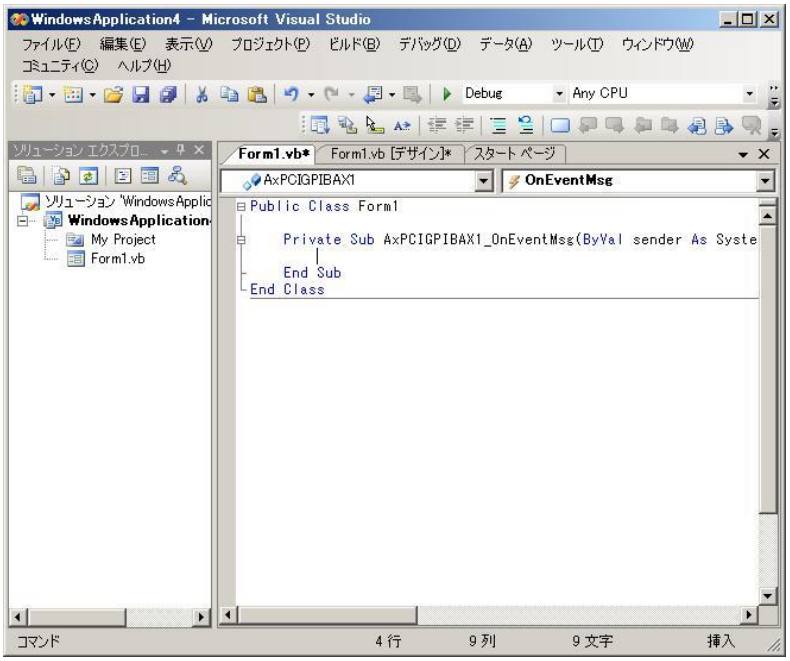


PCIGPIBAX Control コンポーネントが追加されます。

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 PCI 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、イベント発生時に呼び出されるサブルーチン Sub AxPCIGPIBAX1_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

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。

■ 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。

関数名	動作概要
gp_cardinfo	REX-PE20 のリソース情報取得
gp_init	REX-PE20 の初期化
gp_cli	IFC ラインを TRUE にします
gp_ren	REN ラインを TRUE にします
gp_clr	DCL 又は SDC コマンド送信
gp_wrt	GPIB 機器にデータ送信
gp_red	GPIB 機器からデータ受信
gp_trg	GET コマンド送信
gp_wsrq	指定時間 SRQ を待つ(ステータスレジスタ 1)
gp_wsrqb	指定時間 SRQ を待つ(バスステータスレジスタ)
gp_rds	シリアルポールを実行
gp_rds1	シリアルポールを実行
gp_srq	SRQ 割り込み
gp_lcl	GPIB 機器をローカル状態に設定
gp_llo	LLO コマンド送信
gp_tmout	バスタイムアウト時間設定
gp_setdelay	外部変数のディレイ時間設定
gp_count	受信データ数の取得
gp_delm	デリミタの設定
gp_tfrout	GPIB 機器にバイナリデータ送信
gp_tfrin	GPIB 機器からバイナリデータ受信
gp_tfrinit	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	GPIB 機器からバイナリデータ受信
gp_tfrend	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	コマンド文字列を送信
gp_myadr	REX-PE20 の GPIB 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。

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	SRQ ラインの現在の状態を取得
gp_findlstn	リスナ機器の検出

○REX-PCI20/20L からの移行について

弊社製品 GPIB PCI Board REX-PCI20/20L で作成したプログラムはそのままご使用いただけます。

○REX-5052/REX-USB220 からの移行について

弊社製品 GPIB PC カード REX-5052 又は USB to GPIB REX-USB220 用に作成したプログラムを REX-PE20 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。

(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 PCIGPIBAX 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。

関数の戻り値について正常終了の場合は同じですが、エラーの場合一部値の異なるものがありますのでご注意ください。

○API 関数使用上の注意

(1) 1 つの REX-PE20 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。

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。*/
```

(2) 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。*/
```

GPIB 機器制御関数

書 式

VC ➢ INT **gp_cardinfo**(USHORT* pSlotNo, USHORT* pIOBase, USHORT* pIrqNo)

VB ➢ Function PCIGPIBAX.**gpcardinfo**(pSlotNo As Integer, pIOBase As Integer, pIrqNo As Integer) As Long

VB.NET➢ Function PCIGPIBAX.**gpcardinfo**(ByRef pSlotNo As Short, ByRef pIOBase As Short, ByRef pIrqNo As Short) As Integer

機 能 REX-PE20 のリソース情報を取得します。

引 数

pSlotNo		REX-5052 互換のための引数です。NULL を指定してください。
pIOBase	(OUT)	REX-PE20 の I/O ポート
pIrqNo	(OUT)	REX-PE20 の IRQ 番号

戻 値

0	正常終了
-9	REX-PE20 認識エラー

書 式

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

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

VB.NET ➢ Function PCIGPIBAX.**gpinit**(ByVal GpAdrs As Short, ByVal IOBase As Short, ByVal IrqNo As Short) As Integer

機 能 REX-PE20 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

GpAdrs	(IN)	REX-PE20 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	コンフィグレーションエラー

書 式　VC ➢　INT **gp_cli**(void)

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

VB.NET ➢　Function PCIGPIBAX.**gpcli**() As Integer

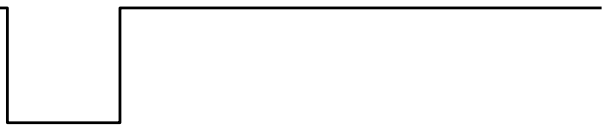
機 能　IFC ラインを TRUE にします。

引 数　なし

戻 値　0　正常終了

-9　REX-PE20 認識エラー

GPIB バス　IFC



約 10 ms

書 式　VC ➢　INT **gp_ren**(void)

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

VB.NET ➢　Function PCIGPIBAX.**gpren**() As Integer

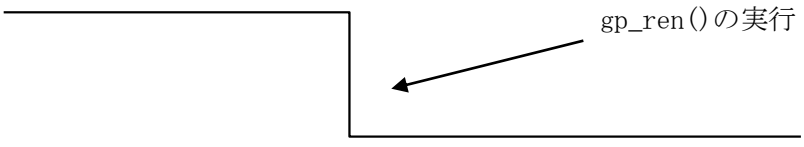
機 能　REN ラインを TRUE にします。

引 数　なし

戻 値　0　正常終了

-9　REX-PE20 認識エラー

GPIB バス　REN



書 式

VC ➢ INT **gp_clr**(PCHAR adrs)

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

VB.NET ➢ Function PCIGPIBAX.**gpclr**(ByVal adrs As String) As Integer

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

adrs		(IN)	GPIB 機器アドレス

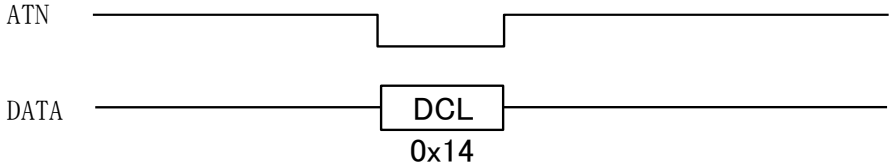
戻 値

0	正常終了
-1	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト

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 PCIGPIBAX.gpclr("")

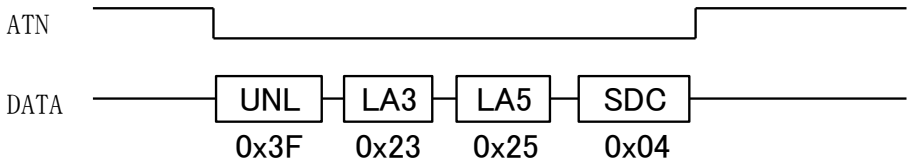
GPIB バス



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 PCIGPIBAX.gpclr("3,5")

GPIB バス



書 式

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

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

VB.NET ➢ Function PCIGPIBAX.**gpwrt**(ByVal adrs As String, ByVal buf As String) As Integer

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

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

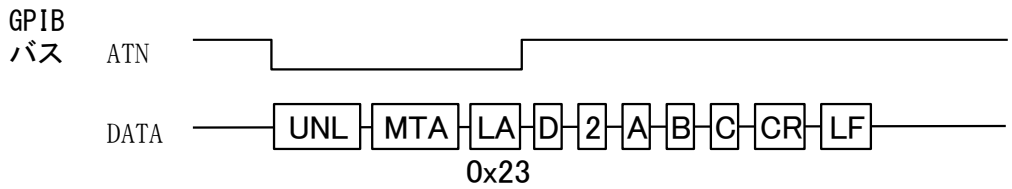
戻 値

0	正常終了
-1	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト

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 PCIGPIBAX.gpwrt("3","D2ABC")



書 式

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

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

VB.NET ➢ Function PCIGPIBAX.**gpred**(ByVal adrs As String, ByRef buf As String, ByVal bufLen As Integer) As Integer

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。

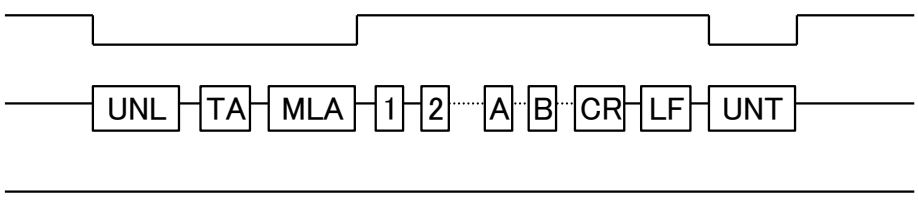
引 数

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	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
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)

GPIB バス



書 式　VC ➢　INT **gp_trg**(PCHAR adrs)

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

VB.NET ➢　Function PCIGPIBAX.**gptrg**(ByVal adrs As String) As Integer

機 能　トリガコマンド（GET）を送信します。

引 数　adrs　(IN)　GPIB 機器アドレス

戻 値　0　正常終了

-1　GPIB 機器アドレス設定エラー

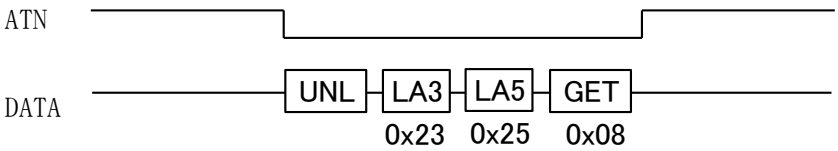
-9　REX-PE20 認識エラー

53　GPIB バスタイムアウト

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 PCIGPIBAX.gptrg("3,5")



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

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

VB.NET ➢　Function PCIGPIBAX.**gpwsrq**(ByVal WaitSecTime As Integer) As Integer

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　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 PCIGPIBAX.**gpwsrqb**(ByVal WaitSecTime As Long) As Long

VB.NET ➢ Function PCIGPIBAX.**gpwsrqb**(ByVal WaitSecTime As Integer) As Integer

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

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 PCIGPIBAX.**gprds**(ByVal adrs As String, status_byte As Integer) As Long

VB.NET ➢ Function PCIGPIBAX.**gprds**(ByVal adrs As String, ByRef status_byte As Short) As Integer

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。

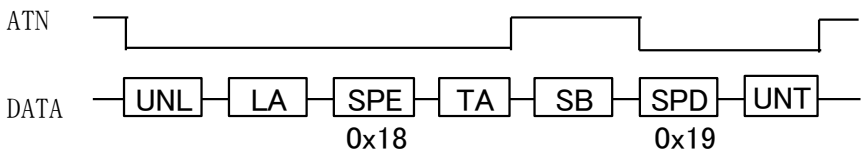
引 数

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	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト

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 PCIGPIBAX.**gprds1**(ByVal adrs As String, status_byte As Integer) As Long

VB.NET ➢ Function PCIGPIBAX.**gprds1**(ByVal adrs As String, ByRef status_byte As Short) As Integer

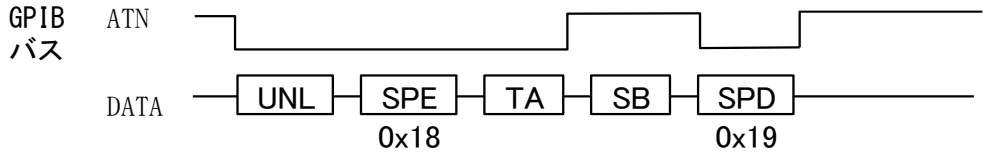
機 能 シリアルポールを実行し、ステータスバイトを取得します。
gp_rds(gprds)との違いは最後に UNT(Untalk)コマンドを送信しない点です。

引 数

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	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト



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 PCIGPIBAX.**gpsrq**(ByVal hwnd As Long, ByVal SrqMode As Long) As Long

VB.NET ➢ Function PCIGPIBAX.**gpsrq**(ByVal hwnd As Integer, ByVal SrqMode As Integer) As Integer

機 能 SRQ 割り込みの実行および解除を行います。

引 数

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	モード設定エラー
-2	開始エラー
-4	開始エラー
-9	REX-PE20 認識エラー

書 式

VC ➢ INT **gp_lcl**(PCHAR adrs)

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

VB.NET ➢ Function PCIGPIBAX.**gplcl**(ByVal adrs As String) As Integer

機 能 GPIB 機器をローカル状態に設定します。

引 数

adrs		(IN)	GPIB 機器アドレス

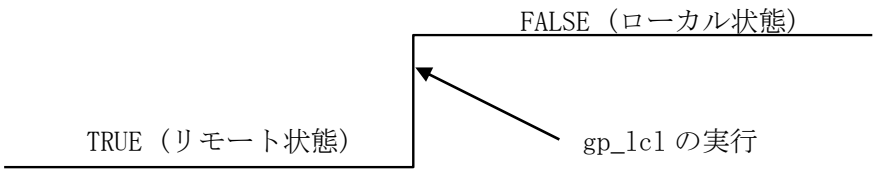
戻 値

0	正常終了
-1	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト

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 PCIGPIBAX.gplcl("")

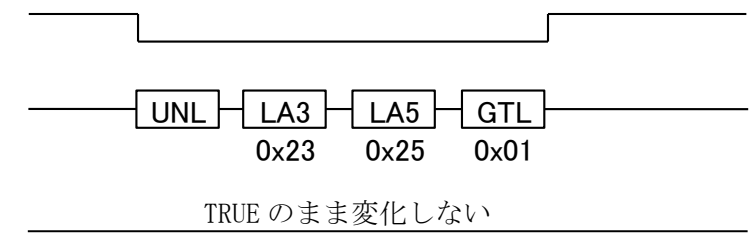
GPIB バス



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 PCIGPIBAX.gplcl("3,5")

GPIB バス



書 式　VC ➢　INT **gp_llo**(void)

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

VB.NET ➢　Function PCIGPIBAX.**gpllo**() As Integer

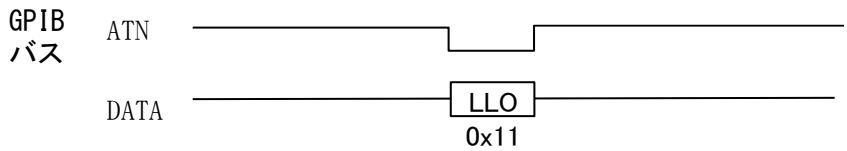
機 能　GPIB 上の全機器に対して LLO(Local Lock Out)コマンド送信します。

引 数　なし

戻 値　0　正常終了

-1　REX-PE20 認識エラー

53　GPIB バスタイムアウト



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

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

VB.NET ➢　Function PCIGPIBAX.**gptmout**(ByVal SecTime As Integer) As Integer

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　SecTime　(IN)　タイムアウト時間(秒単位で指定)

戻 値　0　正常終了

-9　REX-PE20 認識エラー

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 PCIGPIBAX.**gpsetdelay**(ByVal DelayTime As Long) As Long

VB.NET ➢ Function PCIGPIBAX.**gpsetdelay**(ByVal DelayTime As Integer) As Integer

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 DelayTime (IN) ディレイ時間(マイクロ秒単位で指定)

戻 値

N	正常終了時、引数をそのまま返します。
0	失敗

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 PCIGPIBAX.**gpcount**() As Long

VB.NET ➢ Function PCIGPIBAX.**gpcount**() As Integer

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 なし

戻 値

N	受信データ数または送信データ数が返されます。
-9	REX-PE20 認識エラー

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 PCIGPIBAX.**gpdelm**(ByVal mode As String, ByVal dlm As Long) As Long

VB.NET ➢ Function PCIGPIBAX.**gpdelm**(ByVal mode As String, ByVal dlm As Integer) As Integer

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

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	モード設定エラー
-9	REX-PE20 認識エラー

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 PCIGPIBAX.**gptfrout**(ByVal adrs As String, ByVal bufLen As Long, ByVal buf As Integer) As Long

VB.NET ➢ Function PCIGPIBAX.**gptfrout**(ByVal adrs As String, ByVal bufLen As Integer, ByRef buf As Short) As Integer

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。

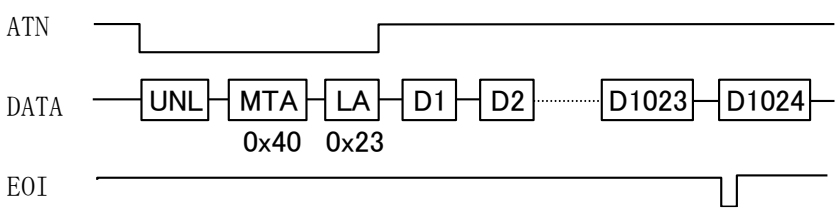
引 数

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	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト

GPIB バス



書 式

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

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

VB.NET ➢ Function PCIGPIBAX.**gptfrin**(ByVal adrs As String, ByVal bufLen As Integer, ByRef buf As Short) As Integer

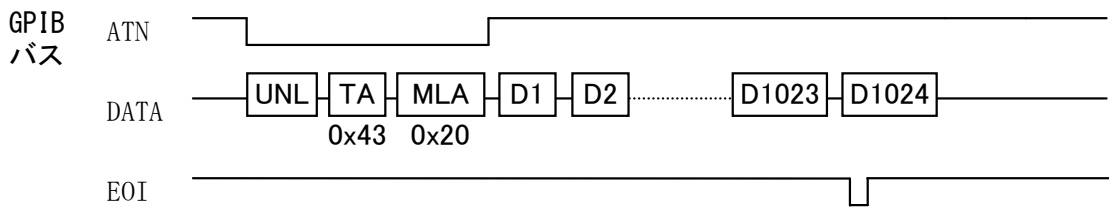
機 能 引数 adrs で指定した GPIB 機器をトーカに指定し、バイナリデータを受信します。デリミタは EOI のみです。EOI を受信するかバスタイムアウトになるまで制御を返しません。

引 数

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	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
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)



書 式

VC ➢ INT **gp_tfrinit**(PCHAR adrs)

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

VB.NET ➢ Function PCIGPIBAX.**gptfrinit**(ByVal adrs As String) As Integer

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

adrs	(IN)	GPIB 機器アドレス

戻 値

0	正常終了
-1	GPIB 機器アドレス設定エラー
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト

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 PCIGPIBAX.gptfrinit("3") ' トーカ指定
    Call PCIGPIBAX.gptfrins(256, RxBuf(0)) '256Byte データ受信
    Call PCIGPIBAX.gptfrins(256, RxBuf(0))
    Call PCIGPIBAX.gptfrins(256, RxBuf(0))
    ・・・(EOI 受信するまで繰返し呼び出す)
    Call PCIGPIBAX.gptfrend() ' トーカ指定解除
```

書 式

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

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

VB.NET ➢ Function PCIGPIBAX.**gptfrins**(ByVal bufLen As Integer, ByRef buf As Short) As Integer

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

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
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト

書 式

VC ➢ VOID **gp_tfrend**(void)

VB ➢ Sub PCIGPIBAX.**gptfrend**()

VB.NET ➢ Sub PCIGPIBAX.**gptfrend**()

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 なし

戻 値 なし

書 式

VC ➢ INT **gp_wtb**(PCHAR buf)

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

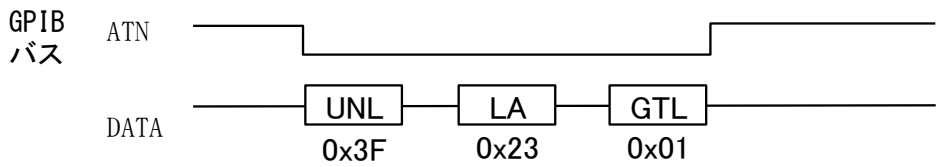
VB.NET ➢ Function PCIGPIBAX.**gpwtb**(ByVal buf As String) As Integer

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 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
-9	REX-PE20 認識エラー
53	GPIB バスタイムアウト



書 式

VC ➢ INT **gp_myadr**(void)

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

VB.NET ➢ Function PCIGPIBAX.**gpmyadr**() As Integer

機 能 関数 gp_init(gpinit)で設定された REX-PE20 の GPIB アドレスを取得します。
プログラムで新たに REX-PE20 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 なし

戻 値

N	正常終了時、REX-PE20 の GPIB アドレスが返されます。
-9	REX-PE20 認識エラー

その他の関数

書 式
VC ➢ VOID **gp_wait**(int WaitSecTime)
VB ➢ Sub PCIGPIBAX.**gpwait**(ByVal WaitSecTime As Long)
VB.NET ➢ Sub PCIGPIBAX.**gpwait**(ByVal WaitSecTime As Integer)

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 なし

書 式
VC ➢ VOID **gp_strtoflt**(BYTE *bPoint, float *data)
VB ➢ Sub PCIGPIBAX.**gpstrtoflt**(ByVal bPoint As Integer, data As Single)
VB.NET ➢ Sub PCIGPIBAX.**gpstrtoflt**(ByRef bPoint As Short, ByRef 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 なし

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 PCIGPIBAX.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 PCIGPIBAX.**gpstrtodbl**(ByVal bPoint As Integer, data As Double)

VB.NET ➢ Sub PCIGPIBAX.**gpstrtodbl**(ByRef bPoint As Short, ByRef 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 なし

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 PCIGPIBAX.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 PCIGPIBAX.**gpsrqCheck**() As Long

VB.NET ➢ Function PCIGPIBAX.**gpsrqCheck**() As Integer

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
-9	REX-PE20 認識エラー

書 式

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

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

VB.NET ➢ Function PCIGPIBAX.**gpfindlstn**(ByRef adrs As String, ByVal adrsLen As Integer) As Integer

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

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
-9	REX-PE20 認識エラー
61	バッファオーバーフロー
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 61 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. 製品付属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が付属しています。その他については、随時、弊社ホームページにて公開予定です。

■付属 CD-ROM 収録サンプルプログラム	
メーカー	デバイス
岩通計測	デジタルオシロスコープ(DS-8812)、ユニバーサルカウンタ(SC-7201,SC-7205,SC-7206,SC-7207)、ファンクションジェネレータ(SG-4105,SG-4115) 、デジタルマルチメータ(VOAC-7520)
ヒューレットパッカード	デジタルマルチメータ(HP3478A)、直流電圧源(E3631A)、ファンクションジェネレータ(33120A)
横河電機	デジタルパワーメータ(WT110E)、プログラマブル直流電圧・電流源(7651)、直流標準電圧・電流発生器(YEW2553)
アドバンテスト	デジタルマルチメータ(R6552)
アジレント・テクノロジー	デジタルオシロスコープ(54846A)
テクトロニクス	デジタルオシロスコープ(TDS3054B)
菊水電子工業	電子負荷装置(PLZ164W)
鶴賀電機	抵抗計(3573/3574)
日置電機	パワーハイテスタ(3332)

※本製品に添付し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。

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

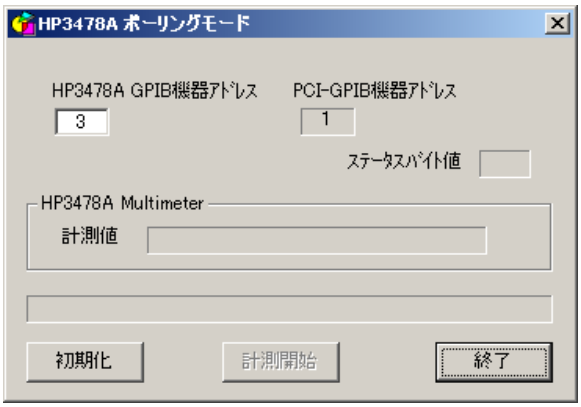
HP3478A 制御プログラムには、1) ポーリングモード（割り込みを使用せず、SRQ が来るのをポーリングしデータを取得するプログラム）と 2) 割り込みモード（SRQ の検知に割り込みを使用し、データを取得するプログラム）の 2 つを用意しています。

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。

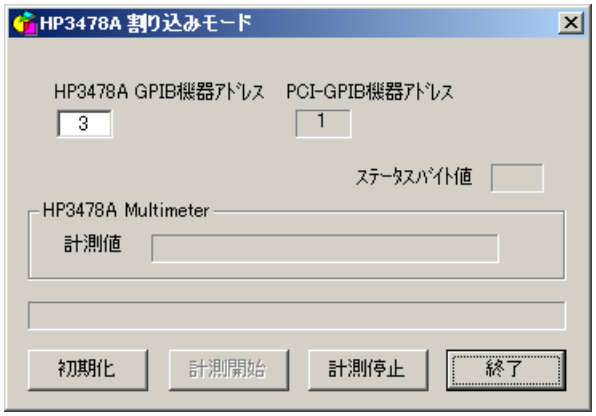


2) 割り込みモード

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、wParam に EVENT_INTERRUPT 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(wParam)
    {
    case EVENT_INTERRUPT: // 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 STOP_INTERRUPT: // 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 ERROR_INTERRUPT: // 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(wParam)

}
```

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 = PCIGPIBAX.gpinit(MyGpibAdrs, 0, 0)
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.gpinit()ｴﾗｰ :" & Status
        Exit Sub
    End If

    '   IFC ﾗｲﾝを TRUE にする
    PCIGPIBAX.gpcli

    '   REN ﾗｲﾝを TRUE にする
    PCIGPIBAX.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 = PCIGPIBAX.gpclr(GpAdrs)
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.gpclr()ｴﾗｰ :" & Status
        Exit Sub
    End If

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

    '   HP3478A GPIB ｺﾏﾝﾄﾞ送信
    Status = PCIGPIBAX.gpwrt(GpAdrs, "H0KM01")
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.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 = PCIGPIBAX.gptrg(GpAdrs)
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.gptrg()ｴﾗｰ :" & Status
        Exit Sub
    End If

    '  指定時間 SRQ を待つ
    Status = PCIGPIBAX.gpwsrq(10)
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.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 = PCIGPIBAX.gprds(GpAdrs, Code(0))
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.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 = PCIGPIBAX.gpred(GpAdrs, szBuf, Len(szBuf))
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.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 = PCIGPIBAX.gpsrq(0, ENABLE_SRQ_INTERRUPT)
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.gpsrq()ｴﾗｰ :" & Status
        Exit Sub
    End If

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

End Sub
```

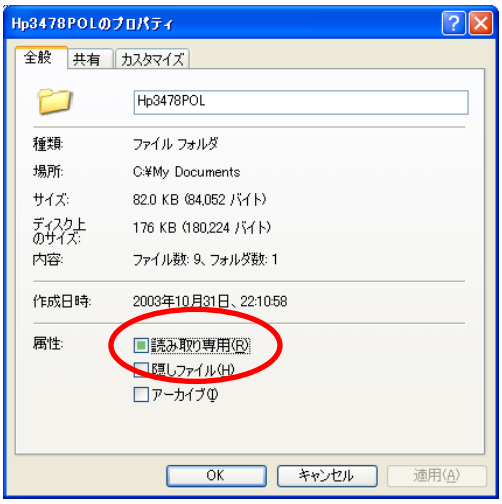
VisualBASIC サンプルプログラム抜粋(割り込みモード)

ユーザ定義メッセージの処理・・・SRQ を検知した場合、wParam に EVENT_INTERRUPT 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 PCIGPIBAX_OnEventMsg(ByVal wParam As Long, ByVal lParam As Long)
    ' ﾒｯｾｰｼﾞの判定
    Select Case wParam
        Case EVENT_INTERRUPT '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 PCIGPIBAX.gpsrq(0, DISABLE_SRQ_INTERRUPT)
        Case STOP_INTERRUPT 'SRQ 待ち状態を終了する場合
            ERROR.Text = "SRQ 待ち終了"
        Case ERROR_INTERRUPT 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
    Dim szBuf As String 'メッセージ格納用バッファ

    ' 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 = PCIGPIBAX.gprds(GpAdrs, Code(0))
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.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 = PCIGPIBAX.gpred(GpAdrs, szBuf, Len(szBuf))
    If Status <> 0 Then
        ERROR.Text = "PCIGPIBAX.gpred()ｴﾗｰ :" & Status
        Exit Sub
    End If

    ' 測定値を表示
    READVAL.Text = szBuf

End Sub
```

サンプルプログラムの使用について

製品付属 CD-ROM の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。

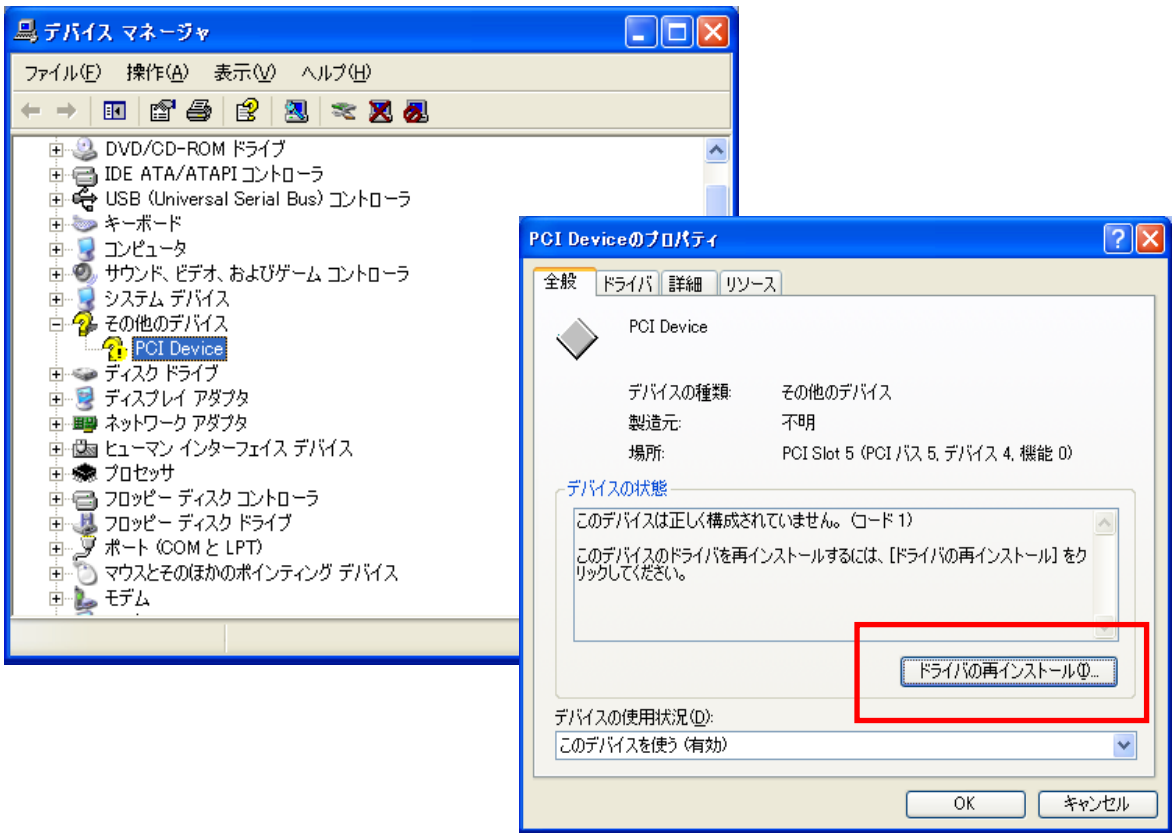


第 4 章　追加情報

4-1. トラブルシューティング

4.1.1 インストールに失敗した場合

「第 2 章インストレーション」を正常に行わなかった、またはインストールに失敗した場合など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－PCI Device」と表示されている場合には、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。



ⓒRATOC Systems, Inc. All rights reserved.