REX-PCI20

PCI GPIB Converter

Linux 用ライブラリ仕様書

2008 年 8 月 8 日

第1.2 版

ライブラリ仕様

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。

ファイル構成について

ライブラリ構成ファイル一覧

ファイル名	説明
rexpci20_gpib.c	REX-PCI20 ライブラリ ソースコード
rexpci20_gpib.h	REX-PCI20 ライブラリ ヘッダーファイル
def9914.h	9914 レジスタ定義ヘッダーファイル
sample.c	ライブラリ呼び出し用サンプルコード
Makefile	Makefile
install_lib	ライブラリ インストール用 スクリプトファイル
uninstall_lib	ライブラリ アンインストール用 スクリプトファイル

ライブラリ関数一覧

・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。

関数名	動作概要
gp_attach	デバイスをオープン
gp_detach	指定 ID のデバイスをクローズ
gp_cardinfo	接続されているデバイスのリソース情報を取得

・REX-PCI20 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。

関数名	動作概要
OutPort	REX-PCI20 のレジスタに書き込み
InPort	REX-PCI20 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。

関数名	動作概要
gp_init	REX-PCI20 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	CLR 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-PCI20 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。

関数名	動作概要
gp_wait	指定時間待つ

・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_wrtd	GPIB バス上にデータ送信
gp_tfroutd	GPIB バス上にバイナリデータ送信
gp_redd	GPIB 上からのデータ受信
gp_redah	GPIB 機器からデータ受信
gp_redrst	リスナ解除、RFD ホールドオフの解除
gp_findlstn	リスナ機器の検出

○API 関数使用上の注意

(1) １台の REX-PCI20 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。*/
```

ライブラリ関数詳細

オープンクローズ関数

書 式　int **gp_attach** ()

機 能　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（**ファイルディスクリプタ**）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。

引 数　なし　(IN)

戻 値　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。
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、gp_detach ()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　VOID **gp_detach** (int num)

機 能　コンバータをクローズします。

引 数　num　(IN)　クローズするコンバータのハンドル

戻 値　なし

書 式　　int **gp_cardinfo**(unsigned int *pIOBase, unsigned int *pIrqNo);

機 能　接続されている REX-PCI20 の IOBase、Irq 情報をユーザに返します。

引 数

*pIOBase	(OUT)	IOBase 情報格納用ポインタ
*pIrqNo	(IN)	Irq 情報格納用ポインタ

戻 値

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

補 足

9914 レジスタ制御関数

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

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　Reg　　(IN)　レジスタオフセット値
(0, 1, 2, 3, 4, 5, 6, 7 のいずれかを指定)
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　　USHORT **InPort**(USHORT Reg)

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　Reg　　(IN)　レジスタオフセット値
(0, 1, 2, 3, 4, 5, 6, 7 のいずれかを指定)

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

機 能　REX-PCI20 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-PCI20 の GPIB 機器アドレス
IOBase	(IN)	未使用
IrqNo	(IN)	未使用

戻 値

0	正常終了
-1	ライブラリ DeviceIoControl()リクエストエラー
60	REX-PCI20 の GPIB 機器アドレス設定エラー

書 式　　INT **gp_cli**(void)

機 能　IFC ラインを TRUE にします。

引 数　なし

戻 値

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

GPIB バス　IFC

約 10 ms

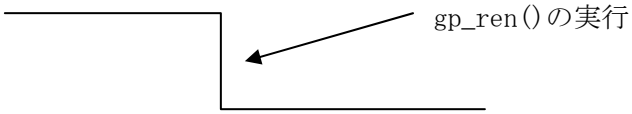
書 式　　INT gp_ren(void)

機 能　REN ラインを TRUE にします。

引 数　なし

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

GPIB バス　REN



書 式　　INT **gp_clr**(PCHAR adrs)

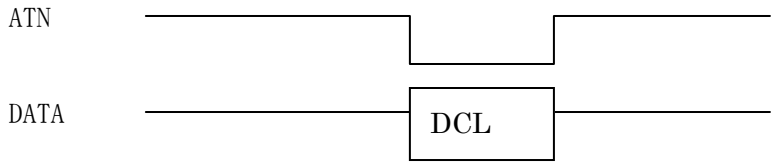
機 能　クリアコマンド（DCL 又は SDC ）を送信します。引数 adrs に機器アドレスを指定しない場合は DCL コマンドを、指定する場合は SDC コマンドを送信します。

引 数　adrs　　(IN)　　GPIB 機器アドレス

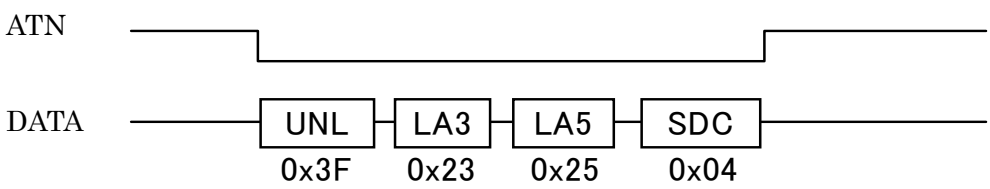
戻 値

0	正常終了
-1	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）gp_clr("");



0x14
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）gp_clr("3,5");



書 式　　INT **gp_wrt**(PCHAR adrs, PCHAR buf)

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。

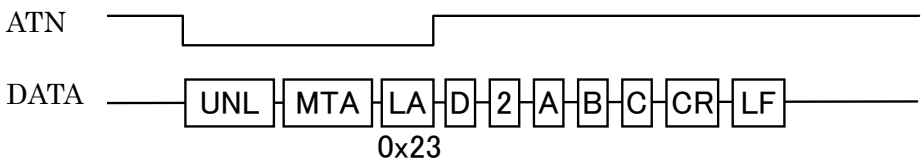
引 数

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	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	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）gp_wrt("3","D2ABC");



書 式　　INT **gp_red**(PCHAR adrs, PCHAR buf, INT bufLen)

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。

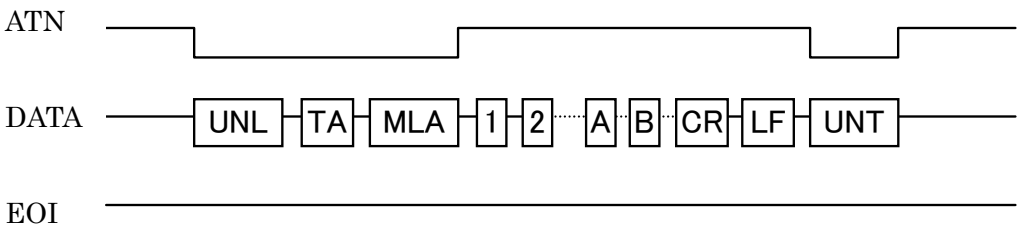
引 数

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	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	転送時エラー

GPIB バス



書 式　　INT **gp_trg**(PCHAR adrs)

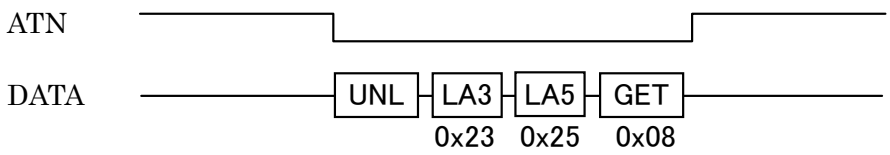
機 能　トリガコマンド（GET）を送信します。

引 数　adrs　　(IN)　GPIB 機器アドレス

戻 値

0	正常終了
-1	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	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) gp_trg("3,5");



書 式　　INT **gp_wsrq**(INT WaitSecTime)

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　　INT **gp_wsrqb**(INT WaitSecTime)

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

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。

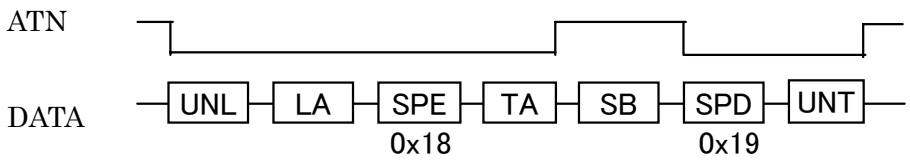
引 数

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

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。

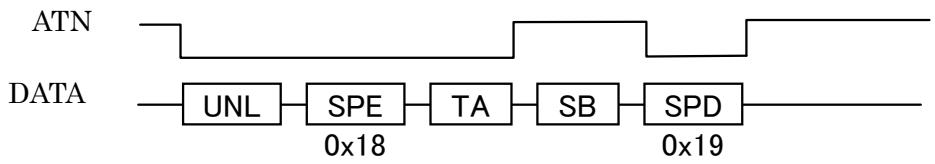
引 数

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	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	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　　INT **gp_srq**(INT SrqMode, PAPIFUNC pFunc)

機 能　SRQ 割り込みの実行および解除を行います。

引 数

SrqMode	(IN)	モードフラグ(0:解除フラグ, 1:実行フラグ)
pFunc	(IN)	コールバック関数のポインタ

戻 値

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

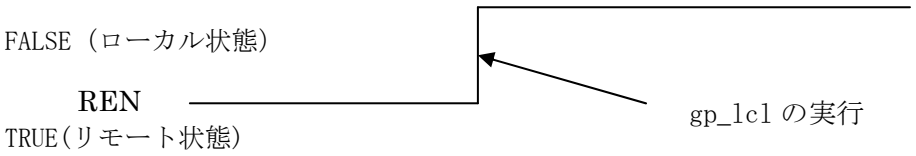
書 式　　INT gp_lcl(PCHAR adrs)

機 能　GPIB 機器をローカル状態に設定します。

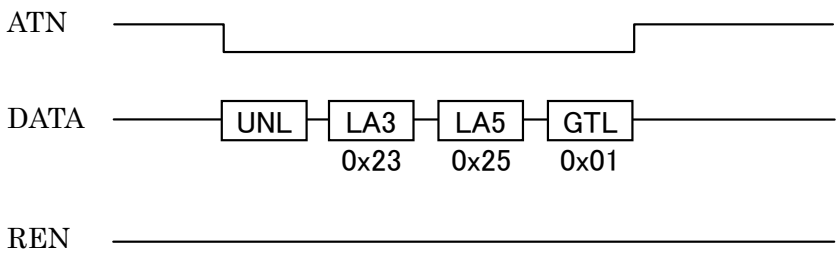
引 数　adrs　　(IN)　GPIB 機器アドレス

戻 値　0　　正常終了
　　-1　　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　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) gp_lcl("");



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) gp_lcl("3,5");



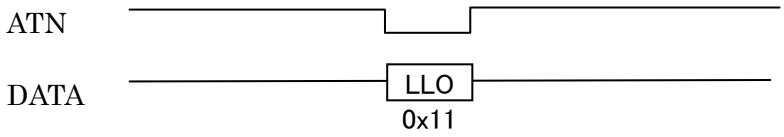
書 式　　INT **gp_llo**(void)

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　　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　転送時エラー

GPIB バス



書 式　　INT **gp_tmout**(INT SecTime)

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
-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　　INT **gp_setdelay**(INT DelayTime)

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　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　　INT **gp_count**(void)

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。

補 足　デリミタコードのカウントは行いません。

書 式　　INT **gp_delm**(PCHAR mode, UINT dlm)

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	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 に 1 を設定すると EOI が出力されます。
全ての bit を 0(dlm=0)にすると、CR+LF(0x0D＋0x0A)が設定されます。
dlm に 0x01～0x7F が設定された場合は 0x01～0x7F のデリミタコードが付加され、0x80 が設定された場合は EOI のみ出力され、0x81～0xFF が設定された場合は 0x01～0x7F のデリミタコードが付加されると同時に EOI が出力されます(送信時のデリミタコードに 0x00 は設定できません)。

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
dlm	EOI	デリミタコード（ 0x01 ～ 0x7F ）※0x00 は使用できません						

(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、即座にデータ受信を終了します。
ただし、受信時のデリミタコードに 0x80 は設定できません。

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
dlm	デリミタコード（ 0x00 ～ 0xFF ） ※0x80 は使用できません							

(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。
下記以外の値を入力してもデリミタ設定は行われません。

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 LF 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　　INT **gp_tfrout**(PCHAR adrs, INT bufLen, PCHAR buf)

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。

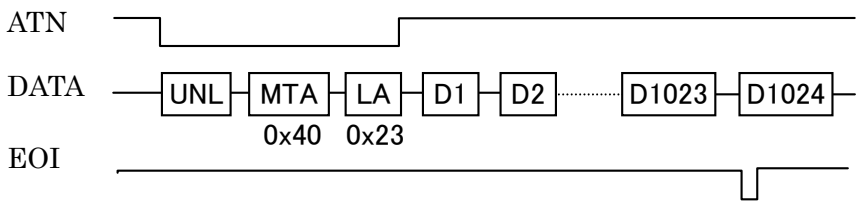
引 数

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	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	転送時エラー

GPIB バス



書 式　　INT **gp_tfrin**(PCHAR adrs, INT bufLen, PCHAR buf)

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。

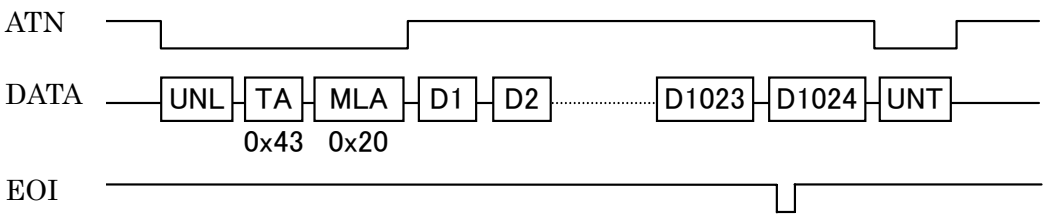
引 数

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	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	転送時エラー

GPIB バス



書 式　　INT **gp_tfrinit**(PCHAR adrs)

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, gp_tfrend 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	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	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。

```
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
```

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

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, gp_tfrend 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
-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	転送時エラー

書 式　　VOID **gp_tfrend**(void)

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, gp_tfrins と共に使用します)

引 数　なし

戻 値　なし

書 式　　INT **gp_wtb**(PCHAR buf)

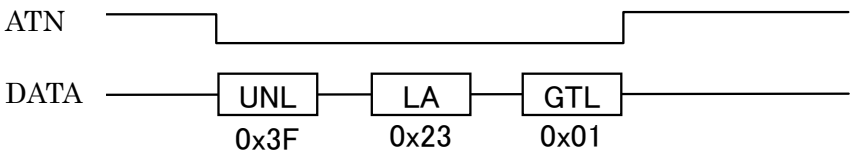
機 能　ATN ラインを TRUE にしてコマンド文字列を送信します。コマンド文字列の最後に、データ終了を示す NULL コード(0x00)を指定してください。

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
-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	転送時エラー

GPIB バス



書 式　　INT **gp_myadr**(void)

機 能　関数 gp_init で設定された REX-PCI20 の GPIB 機器アドレスを取得します。
プログラムで新たに REX-PCI20 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-PCI20 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　　VOID **gp_wait**(int WaitSecTime)

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

補助関数

書 式　　INT gp_srqCheck(void)

機 能　SRQ ラインの現在の状態を返します。

引 数　なし

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

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。

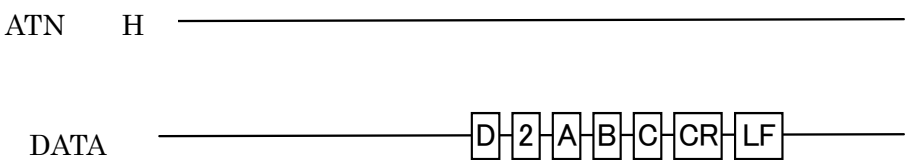
引 数

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

機 能　GPIB バス上にバイナリデータ送信します。

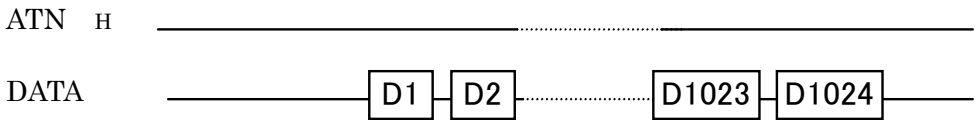
引 数

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

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。

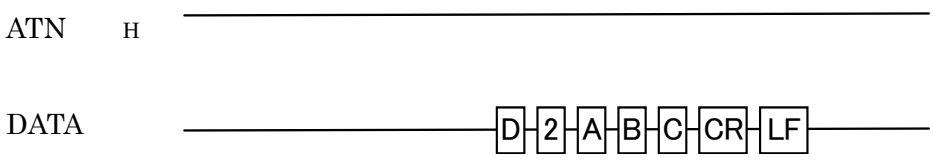
引 数

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

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。

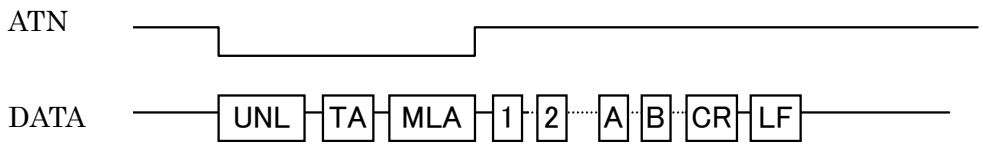
引 数

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	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　　INT **gp_redrst**(void)

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　なし

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

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

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
-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。

---

appendix. サンプルプログラム仕様

サンプルプログラムを提供する計測器について

岩通計測株式会社	ユニバーサルカウンタ（SC-7201）
ヒューレット・パッカード社	デジタルマルチメータ（HP3478A）
	ファンクションジェネレータ（33120A）
	直流電圧源（E3631A）
横河電機株式会社	デジタルパワーメータ（WT110E）
	直流標準電圧・電流発生器（YEW2553）
アドバンテスト社	デジタルマルチメータ（R6552）

各サンプルプログラムの概要

表 5. サンプルプログラム一覧

サンプルプログラム名	概要
gpibctrl	GPIB 機器検出、デリミタ等の設定、コマンドおよびバイナリデータの送受信を行います。
hp3478a_pol	SRQ をポーリングで監視した後、電圧測定を行います。
hp3478a_int	SRQ を割り込みにより検知した後、電圧測定を行います。
hp33120	「波形出力」ボタンにより、指定の波形を出力します。
r6552	「計測」ボタンにより、抵抗[測定を行います。
sc7201	指定したパラメータで周波数測定を行います。
wt110e	設定したパラメータで電圧・電流測定を行います。
e3631a	指定の電圧を出力します。
yew2553	指定の電圧を出力します。

プログラミング例

HP3478A ポーリングモード サンプルプログラム

ライブラリ関数呼び出し例（抜粋）

```
・- 初期化ボタンが押された時の処理
void Cmd_OnCmdGpInit ()
{
        // GPIBコントローラ初期化
        GpStatus = gp_init( MyGpibAdrs, 0, 0 );

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

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

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

        // SDCコマンド送出
        GpStatus = gp_clr( GpAdrsBuf );

        // LLOコマンド送出
        GpStatus = gp_llo();

        // HP3478A GPIBコマンド送信
        GpStatus = gp_wrt( GpAdrsBuf, szCommand );
}

・- 計測ボタン押されたら計測を開始します。
void Cmd_OnCmdStart ()
{
        // トリガコマンド実行
        GpStatus = gp_trg( GpAdrsBuf );

        // 指定時間SRQを待つ
        GpStatus = gp_wsrq( 10 );

        // シリアルポールを実行しステータスバイトを受信
        GpStatus = gp_rds( GpAdrsBuf, StatusByte );

        // GPIBバスからデータをリード
        GpStatus = gp_red( GpAdrsBuf, RcvData, sizeof(RcvData) );
}

・- 終了ボタンが押された時の処理
void Cmd_OnCancel ()
{
        // LCLコマンド送出
        GpStatus = gp_lcl( GpAdrsBuf );
}
```